Session III

More about Isar
Overview

- Abbreviations
- Predicate Logic
- Accumulating facts
- Reasoning with chains of equations
- Locales: the module system
**Abbreviations**

\( \textit{this} \) = the previous proposition proved or assumed
\( \textit{then} \) = \textit{from this}
\( \textit{with} \ \bar{a} \) = \textit{from} \ \bar{a} \ \textit{this}

\( ?\textit{thesis} \) = the last enclosing \textit{show} formula
Mixing proof styles

from . . .

have . . .

apply -

apply(...)

:

apply(...)

done

make incoming facts assumptions
Demo: Abbreviations
Predicate Calculus
Syntax:

```latex
fix variables

Introduces new arbitrary but fixed variables (\sim parameters)
```
obtain

Syntax:

\texttt{obtain \ variables \ where \ proposition \ proof}

Introduces new variables together with property
Demo: predicate calculus
moreover/ultimately

have \( \text{formula}_1 \) . . .

have \( \text{formula}_2 \) . . .

: 

have \( \text{formula}_n \) . . .

show . . .

proof . . .
moreover/ultimately

have $formula_1$ . . .
moreover
have $formula_2$ . . .
moreover

\vdots

moreover
have $formula_n$ . . .

show . . .

proof . . .
moreover/ultimately

have $formula_1$ 
moreover
have $formula_2$ 
moreover
  ...
moreover
have $formula_n$ 
ultimately
show 

proof 

moreover/ultimately

have \( f_{\text{ormula}_1} \ldots \)
moreover
have \( f_{\text{ormula}_2} \ldots \)
moreover
:
moreover
have \( f_{\text{ormula}_n} \ldots \)
ultimately
show \ldots
— pipes facts \( f_{\text{ormula}_1} \ldots f_{\text{ormula}_n} \) into the proof
proof \ldots
Demo: moreover/ultimately
General case distinctions

show $\textit{formula}$

proof -
  have $P_1 \lor P_2 \lor P_3 \ldots$
General case distinctions

\textbf{show} \ formula

\textbf{proof} -

\begin{itemize}
  \item have $P_1 \lor P_2 \lor P_3 \ldots$
  \item moreover
    \begin{itemize}
      \item \{ assume $P_1 \ldots$ have \( \text{?thesis} \ldots \} \}
    \end{itemize}
\end{itemize}

\textbf{ultimately} show \( \text{?thesis} \) by blast

\textbf{qed}
General case distinctions

show formula

proof -

  have $P_1 \lor P_2 \lor P_3 \ldots$

  moreover

  { assume $P_1 \ldots$ have $\textit{thesis} \ldots$ }

  moreover

  { assume $P_2 \ldots$ have $\textit{thesis} \ldots$ }

ultimately show $\textit{thesis}$ by blast

qed
General case distinctions

show formula

proof -

have $P_1 \lor P_2 \lor P_3 \ldots$

moreover

{ assume $P_1 \ldots$ have $?thesis \ldots$ }

moreover

{ assume $P_2 \ldots$ have $?thesis \ldots$ }

moreover

{ assume $P_3 \ldots$ have $?thesis \ldots$ }

ultimately show $?thesis$

by blast

qed
show $\text{formula}$

proof -

have $P_1 \lor P_2 \lor P_3 \ldots$

moreover

{ assume $P_1 \ldots$ have $?thesis \ldots$ }

moreover

{ assume $P_2 \ldots$ have $?thesis \ldots$ }

moreover

{ assume $P_3 \ldots$ have $?thesis \ldots$ }

ultimately show $?thesis$ by blast

qed
Chains of equations

- Keywords also and finally.
Chains of equations

- Keywords *also* and *finally*.
- ...: predefined schematic term variable, refers to the right hand side of the last expression.
Chains of equations

- Keywords **also** and **finally**.
- ...: predefined schematic term variable, refers to the **right hand side of the last expression**.
- Uses transitivity rule.
also/finally

have "$t_0 = t_1" \ldots 
also
have "$ \ldots = t_2" \ldots 
also
\vdots
also
have "$ \ldots = t_n" \ldots
also/finally

have "\( t_0 = t_1 \)" . . .
also
have "\( \ldots = t_2 \)" . . .
also
:
also
have "\( \ldots = t_n \)" . . .

\[ t_0 = t_1 \]
also/finally

have "\( t_0 = t_1 \)" ... 
also have "... = t_2" ...
also ...
also have "... = t_n" ...
also/finally

have 
\[ t_0 = t_1 \]
also
have 
\[ \ldots = t_2 \]
also
\[ \vdots \]
also
have 
\[ \ldots = t_n \]

\[ t_0 = t_1 \]
\[ t_0 = t_2 \]
\[ \vdots \]
\[ t_0 = t_{n-1} \]
also

have "\(t_0 = t_1\)" . . .
also
have "\(\ldots = t_2\)" . . .
also
:
also
have "\(\ldots = t_n\)" . . .

finally show . . .
— pipes fact \(t_0 = t_n\) into the proof

proof
:

\[
\begin{align*}
t_0 &= t_1 \\
t_0 &= t_2 \\
\vdots \\
t_0 &= t_{n-1}
\end{align*}
\]
More about also

- Works for all combinations of $=\,$, $\leq\,$ and $<\,$.
More about also

- Works for all combinations of $=$, $\leq$ and $<$.  
- Uses rules declared as $[\text{trans}]$. 
More about also

- Works for all combinations of $=$, $\leq$ and $<$.  
- Uses rules declared as $[\text{trans}]$.  
- To view all combinations in Proof General:  

  Isabelle/Isar $\rightarrow$ Show me $\rightarrow$ Transitivity rules
Demo: also/finally
Locales

Isabelle’s Module System
Isar is based on contexts

\[ \forall x.\ A \Rightarrow C \]

proof -
  
  fix \ x

  assume Ass: \ A

  : 

  from Ass show \ C \ldots

qed
Isar is based on contexts

\[ \forall x. A \implies C \]

proof -

fix \( x \)

assume \( \text{Ass}: A \)

\( \vdash \)

from \( \text{Ass} \) show \( C \ldots \)

qed

*\( x \) and \( \text{Ass} \) are visible inside this context*
Beyond Isar contexts

Locales are extended contexts
Locales are extended contexts

- Locales are named
Beyond Isar contexts

Locales are extended contexts

- Locales are named
- Fixed variables may have syntax
Locales are extended contexts

- Locales are **named**
- Fixed variables may have **syntax**
- It is possible to **add** and **export** theorems
Locales are extended contexts

- Locales are named
- Fixed variables may have syntax
- It is possible to add and export theorems
- Locale expression: combine and modify locales
Locales consist of context elements.
Locales consist of context elements.

fixes Parameter, with syntax
Locales consist of context elements.

- **fixes** Parameter, with syntax
- **assumes** Assumption
Locales consist of context elements.

- fixes: Parameter, with syntax
- assumes: Assumption
- defines: Definition
Locales consist of context elements.

- **fixes**  Parameter, with syntax
- **assumes**  Assumption
- **defines**  Definition
- **notes**  Record a theorem
locale \textit{loc} =
\begin{verbatim}
loc1 +
\end{verbatim}
\begin{verbatim}
fixes . . .
\end{verbatim}
\begin{verbatim}
assumes . . .
\end{verbatim}
Declaring locales

locale \( \text{loc} = \)

\( \text{loc1} + \)

fixes . . .

assumes . . .

Declares named locale \( \text{loc} \).
locale \textit{loc} = \textit{loc1} + \text{Import fixes} \ldots \text{assumes} \ldots

Declares named locale \textit{loc}. 
Declaring locales

\[
\text{locale } \textit{loc} =
\]
\[
\textit{loc1} + \\
\textit{fixes} \ldots \hspace{1cm} \text{Context elements} \\
\textit{assumes} \ldots
\]

Declares named locale \textit{loc}.
Declaring locales

Theorems may be stated relative to a named locale.

\textbf{lemma} (in \textit{loc}) \( P \) [simp]: \textit{proposition}

\textit{proof}
Declaring locales

Theorems may be stated relative to a named locale.

\textbf{lemma} \hspace{5pt} \textbf{(in} \hspace{5pt} \textit{loc}) \hspace{5pt} P \hspace{5pt} \textbf{[simp]}: \hspace{5pt} \textit{proposition}

\textit{proof}

- Adds theorem \textit{P} to context \textit{loc}. 
Declaring locales

Theorems may be stated relative to a named locale.

\textbf{lemma} (in }\textit{loc} \textbf{)} \( P \) [simp]: \textit{proposition}
\begin{proof}
\item Adds theorem \( P \) to context \( \textit{loc} \).
\item Theorem \( P \) is in the simpset in context \( \textit{loc} \).
\end{proof}
Theorems may be stated relative to a named locale.

`lemma (in loc) P [simp]: proposition
proof`

- Adds theorem \( P \) to context `loc`.
- Theorem \( P \) is in the simpset in context `loc`.
- Exported theorem `loc.P` visible in the entire theory.
Demo: locales 1
Parameters must be consistent!

- Parameters in **fixes** are distinct.
Parameters must be consistent!

- Parameters in `fixes` are distinct.
- Free variables in `assumes` and `defines` occur in preceding `fixes`. 
Parameters must be consistent!

- Parameters in \textit{fixes} are distinct.
- Free variables in \textit{assumes} and \textit{defines} occur in preceding \textit{fixes}.
- Defined parameters must neither occur in preceding \textit{assumes} nor \textit{defines}.
Locale expressions

Locale name: $n$
Locale expressions

Locale name: $n$
Rename: $e \ q_1 \ldots \ q_n$
Change names of parameters in $e$. 

IJCAR 2004, Tutorial T4 -- p.25
Locale expressions

Locale name: \( n \)
Rename: \( e \ q_1 \ldots \ q_n \)
Change names of parameters in \( e \).
Merge: \( e_1 + e_2 \)
Context elements of \( e_1 \), then \( e_2 \).
Locale expressions

Locale name: $n$
Rename: $e \ q_1 \ldots \ q_n$
Change names of parameters in $e$.
Merge: $e_1 + e_2$
Context elements of $e_1$, then $e_2$.

- Syntax is lost after rename (currently).
Demo: locales 2
Normal form of locale expressions

Locale expressions are converted to flattened lists of locale names.
Normal form of locale expressions

Locale expressions are converted to flattened lists of locale names.

- With full parameter lists
Normal form of locale expressions

Locale expressions are converted to flattened lists of locale names.

- With full parameter lists
- Duplicates removed
Normal form of locale expressions

Locale expressions are converted to flattened lists of locale names.

- With full parameter lists
- Duplicates removed

Allows for multiple inheritance!
Instantiation

Move from abstract to concrete.
Instantiation

Move from \textit{abstract} to \textit{concrete}.

\texttt{instantiate \textit{label} : \textit{loc}}
Instantiation

Move from abstract to concrete.

\textbf{instantiate} \textit{label} : \textit{loc}

- From chained fact \textit{loc} \textit{t}_1 \ldots \textit{t}_n instantiate locale \textit{loc}. 
Instantiation

Move from abstract to concrete.

\textbf{instantiate} \textit{label} : \textit{loc}

\begin{itemize}
\item From chained fact \textit{loc} \( t_1 \ldots t_n \) instantiate locale \textit{loc}.
\item Imports all theorems of \textit{loc} into current context.
\end{itemize}
Instantiation

Move from abstract to concrete.

\textbf{instantiate} \textit{label} : \textit{loc}

- From chained fact \textit{loc} \textit{t}_1 \ldots \textit{t}_n instantiate locale \textit{loc}.
- Imports all theorems of \textit{loc} into current context.
  - Instantiates the parameters with \textit{t}_1 \ldots \textit{t}_n.
  - Interprets attributes of theorems.
  - Prefixes theorem names with \textit{label}.

IJCAR 2004, Tutorial T4 – p.28
Instantiation

Move from **abstract** to **concrete**.

\[ \text{instantiate label : loc} \]

- From chained fact \( \text{loc } t_1 \ldots t_n \) instantiate locale \( \text{loc} \).
- Imports all theorems of \( \text{loc} \) into current context.
  - Instantiates the parameters with \( t_1 \ldots t_n \).
  - Interprets attributes of theorems.
  - Prefixes theorem names with \( \text{label} \).
- Currently only works inside Isar contexts.
Demo: locales 3
Practical Session III

The sun spills darkness
A dog howls after midnight
Goals remain unsolved.

— Chris Owens