Exercises from last time

- Construct a type derivation tree for the term \( \lambda x \, y \, z \, . \, z \, x \) \((y \, x)\)
- Find a unifier (substitution) such that \( \lambda x \, y \, z \, . \, \lambda x \, y \, z \, . \, (\lambda x \, y \, y) \)
Proofs in Isabelle

General schema:

lemma name: "<goal>"
apply <method>
apply <method>
done

→ Sequential application of methods until all subgoals are solved.

Isabelle Theories

Syntax:

theory MyTh
imports ImpTh1 ... ImpThn
begin
  (declarations, definitions, theorems, proofs, ...)
end

→ MyTh: name of theory. Must live in file MyTh.thy
→ ImpThi: name of imported theories. Import transitive.

Unless you need something special:

theory MyTh imports Main begin ... end

Natural Deduction Rules

For each connective (\land, \lor, etc):

introduction and elimination rules
Proof by assumption

apply assumption

proves

1. \([B_1; \ldots; B_m] \implies C\)

by unifying \(C\) with one of the \(B_i\).

There may be more than one matching \(B_i\), and multiple unifiers.

Backtracking!

Explicit backtracking command: back

Intro rules

Intro rules decompose formulae to the right of \(\implies\).

apply (rule <intro-rule>)

Intro rule \([A_1; \ldots; A_n] \implies A\) means

\(\implies\) To prove \(A\) it suffices to show \(A_1 \ldots A_n\)

Applying rule \([A_1; \ldots; A_n] \implies A\) to subgoal \(C\):

\(\implies\) unify \(A\) and \(C\)

\(\implies\) replace \(C\) with \(n\) new subgoals \(A_1 \ldots A_n\)

Elim rules

Elim rules decompose formulae on the left of \(\implies\).

apply (rule <elim-rule>)

Elim rule \([A_1; \ldots; A_n] \implies A\) means

\(\implies\) If I know \(A\) and want to prove \(A\) it suffices to show \(A_1 \ldots A_n\)

Applying rule \([A_1; \ldots; A_n] \implies A\) to subgoal \(C\):

Like rule but also

\(\implies\) unifies first premise of rule with an assumption

\(\implies\) eliminates that assumption
MORE PROOF RULES

Iff, Negation, True and False

\[
\begin{align*}
A \implies B & \quad B \implies A \\
A & \implies B \\
A \implies \text{False} & \implies \neg A \\
\text{True} & \implies \text{False}
\end{align*}
\]

Equality

\[
\begin{align*}
\text{refl} & \quad t = t \\
\text{sym} & \quad t = s \implies s = t \\
\text{trans} & \quad t = s \implies s = f \\
\text{subst} & \quad s = t \implies P \substitute{t}{s} \\
\end{align*}
\]

Rarely needed explicitly — used implicitly by term rewriting

Classical

\[
\begin{align*}
P = \text{True} \lor P = \text{False} \\
\text{excluded-middle} \\
\neg A & \implies \text{False} \\
\neg A & \implies A \\
\text{contr} & \quad \text{classical}
\end{align*}
\]

They make the logic “classical”, “non-constructive”
Cases

\[ P \lor \neg P \]

is a case distinction on type \( \text{bool} \)

Isabelle can do case distinctions on arbitrary terms:

\[
\text{apply\ (case_tac \ term)}
\]

Safe and not so safe

**Safe rules** preserve provability
- \( \text{conjI, impI, notI, iffI, refl, ccontr, classical, conjE, disjE} \)

**Unsafe rules** can turn a provable goal into an unprovable one
- \( \text{disjI1, disjI2, impE, iffD1, iffD2, notE} \)

**Apply safe rules before unsafe ones**

What we have learned so far...

- natural deduction rules for \( \land, \lor, \rightarrow, \neg, \iff \)
- proof by assumption, by intro rule, elim rule
- safe and unsafe rules
Exercises

- Redo the demo alone + exercises
- Assignment 1 is out today!

- Reminder: DO NOT CHEAT
  - Assignments and exams are take-home. This does NOT mean you can work in groups. Each submission is personal.
  - For more info, see Plagiarism Policy