# Automated Reasoning

## Lecture 4: Propositional Reasoning in Isabelle

Jacques Fleuriot
jdf@inf.ed.ac.uk

# Recap

Last lecture:

- Completed the natural deduction system for propositional logic
- Started on proving propositions in Isabelle

Today:

- More details on proving propositions in Isabelle
- Alternative inference rules ($L$-system, a.k.a. “Sequent Calculus”)
- Why should we trust Isabelle?

# The `rule` Method

To apply an inference rule, we use `rule`.

Consider the theorem `disjI1`

$$?P \Longrightarrow ?P \vee ?Q$$

Using the command

```
apply (rule disjI1)
```

on the goal

$$[\![A; B; C]\!] \Longrightarrow (A \wedge B) \vee D$$

yields the subgoal

$$[\![A; B; C]\!] \Longrightarrow A \wedge B$$

# General definition of method `rule`

When we apply the method `rule someRule` where

$$\texttt{someRule} : [\![P_1; \ldots; P_m]\!] \Longrightarrow Q$$

to the goal

$$[\![A_1; \ldots; A_n]\!] \Longrightarrow C$$

where $Q$ and $C$ can be unified, we generate the goals

$$\begin{array}{c} [\![A'_1; \quad \ldots; \quad A'_n]\!] \Longrightarrow P'_1 \\ \vdots \\ [\![A'_1; \quad \ldots; \quad A'_n]\!] \Longrightarrow P'_m \end{array}$$

where $A'_1, A'_2, \ldots, A'_n, P'_1, P'_2, \ldots, P'_m$ are the results of applying the substitution which unifies $Q$ and $C$ to $A_1, A_2, \ldots, A_n, P_1, P_2, \ldots, P_m$.

We must now derive each of the rule's assumptions using our goal's assumptions.

## A Problem with `rule`

Consider the `disjE` rule:

$$\texttt{disjE} : \llbracket P \vee Q; P \Longrightarrow R; Q \Longrightarrow R \rrbracket \Longrightarrow R$$

If we have the goal:

$$\llbracket (A \wedge B) \vee C; D \rrbracket \Longrightarrow B \vee C$$

Then applying `rule disjE` produces three new goals:

$$\llbracket (A \wedge B) \vee C; D \rrbracket \Longrightarrow ?P \vee ?Q$$
$$\llbracket (A \wedge B) \vee C; D; ?P \rrbracket \Longrightarrow B \vee C$$
$$\llbracket (A \wedge B) \vee C; D; ?Q \rrbracket \Longrightarrow B \vee C$$

We then solve the first subgoal by applying `assumption`.

This seems pointlessly roundabout… we often want to *use* one of our assumptions in our proof.

# The `erule` Method

Used when the conclusion of theorem matches the conclusion of the current goal and the first premise of theorem matches a premise of the current goal.

Consider the theorem `disjE`

$$[\![P \vee Q; P \Longrightarrow R; Q \Longrightarrow R]\!] \Longrightarrow R$$

Applying `erule disjE` to goal

$$[\![(A \wedge B) \vee C; D]\!] \Longrightarrow B \vee C$$

yields the subgoals

$$[\![D; (A \wedge B)]\!] \Longrightarrow B \vee C \qquad [\![D; C]\!] \Longrightarrow B \vee C$$

# General definition of method `erule`

When we apply the method `erule someRule` where

$$\texttt{someRule} : [\![P_1; \ldots; P_m]\!] \Longrightarrow Q$$

to the goal

$$[\![A_1; \ldots; A_n]\!] \Longrightarrow C$$

where $P_1$ and $A_1$ are unifiable and $Q$ and $C$ are unifiable, we generate the goals:

$$\begin{array}{c} [\![A'_2; \quad \ldots; \quad A'_n]\!] \Longrightarrow P'_2 \\ \vdots \\ [\![A'_2; \quad \ldots; \quad A'_n]\!] \Longrightarrow P'_m \end{array}$$

where $A'_2, \ldots, A'_n, P'_2, \ldots, P'_m$ are the results of applying the substitution which unifies $P_1$ to $A_1$ and $Q$ to $C$ to $A_2, \ldots, A_n, P_2, \ldots, P_m$.

We **eliminate** an assumption from the rule and the goal, and must derive the rule's other assumptions using our goal's other assumptions.

# General definition of method `drule`

When we apply the method `drule someRule` where

$$\texttt{someRule} : [\![P_1; \ldots ; P_m]\!] \Longrightarrow Q$$

to the goal

$$[\![A_1; \ldots ; A_n]\!] \Longrightarrow C$$

where $P_1$ and $A_1$ are unifiable, we generate the goals:

$$\begin{array}{r}
[\![A'_2; \quad \ldots; \quad A'_n]\!] \Longrightarrow P'_2 \\
\vdots \\
[\![A'_2; \quad \ldots; \quad A'_n]\!] \Longrightarrow P'_m \\
[\![Q'; A'_2; \quad \ldots; \quad A'_n]\!] \Longrightarrow C'
\end{array}$$

where $A'_2, A'_3, \ldots, A'_n, P'_2, P'_3 \ldots, P'_m, Q', C'$ are the results of applying the substitution which unifies $P_1$ and $A_1$ to $A_2, A_3, \ldots, A_n, P_2, P_3, \ldots, P_m, Q, C$.

We **delete** an assumption, replacing it with the conclusion of the rule.

# General definition of method `frule`

When we apply the method `frule someRule` where

$$\texttt{someRule} : [\![P_1; \ldots; P_m]\!] \Longrightarrow Q$$

to the goal

$$[\![A_1; \ldots; A_n]\!] \Longrightarrow C$$

where $P_1$ and $A_1$ are unifiable, we generate the goals:

$$\begin{aligned}
[\![A_1'; A_2'; \quad \ldots; \quad A_n']\!] &\Longrightarrow P_2' \\
&\vdots \\
[\![A_1'; A_2'; \quad \ldots; \quad A_n']\!] &\Longrightarrow P_m' \\
[\![Q'; A_1'; A_2'; \quad \ldots; \quad A_n']\!] &\Longrightarrow C'
\end{aligned}$$

where $A_1', A_2', \ldots, A_n', P_2', \ldots, P_m', Q', C'$ are the results of applying the substitution which unifies $P_1$ and $A_1$ to $A_1, A_2, \ldots, A_n, P_2, \ldots, P_m, Q, C$.

This is like `drule` except the assumption in our goal is kept.

# More Methods

- `rule_tac`, `erule_tac`, `drule_tac` and `frule_tac` are like their counterparts, but you can give substitutions for variables in the rule before they are applied.

**Example**

```
apply (erule_tac Q="B ∧ D" in conjE)
```

applied to the subgoal

$$[\![A \wedge B; C \wedge B \wedge D]\!] \Longrightarrow B \wedge D$$

generates the new goal

$$[\![A \wedge B; C; B \wedge D]\!] \Longrightarrow B \wedge D$$

- Isabelle also provides advanced tactics, like `simp` and `auto` which perform some **automatic deduction**.

# $L$-systems/Sequent Calculus

The `erule` tactic points to another way of phrasing a system of inference rules in a system with sequents $\Gamma \vdash A$.

Instead of *elimination* rules:

$$\frac{\Gamma \vdash P \vee Q \qquad \Gamma, P \vdash R \qquad \Gamma, Q \vdash R}{\Gamma \vdash R} \text{ (disjE)}$$

Have *left introduction rules* (all the introduction rules in natural deduction introduce connectives on the right-hand side of the $\vdash$):

$$\frac{\Gamma, P \vdash R \qquad \Gamma, Q \vdash R}{\Gamma, P \vee Q \vdash R}$$

This corresponds to applying rules using `erule` in Isabelle.

The *left introduction rules* are often much easier to use in a backwards, goal-directed style.

# $L$-systems/Sequent Calculus

The following $L$-System (a.k.a. Sequent Calculus) rules are an alternative sound and complete proof system for propositional logic:

$$\frac{}{\Gamma, P \vdash P} \text{ (assumption)}$$

$$\frac{\Gamma \vdash P \qquad \Gamma \vdash Q}{\Gamma \vdash P \wedge Q} \text{ (conjI)} \qquad \frac{\Gamma, P, Q \vdash R}{\Gamma, P \wedge Q \vdash R} \text{ (e conjE)}$$

$$\frac{\Gamma \vdash P}{\Gamma \vdash P \vee Q} \text{ (disjI1)} \qquad \frac{\Gamma \vdash Q}{\Gamma \vdash P \vee Q} \text{ (disjI2)} \qquad \frac{\Gamma, P \vdash R \qquad \Gamma, Q \vdash R}{\Gamma, P \vee Q \vdash R} \text{ (e disjE)}$$

$$\frac{\Gamma, A \vdash B}{\Gamma \vdash A \longrightarrow B} \text{ (impI)} \qquad \frac{\Gamma \vdash P \qquad \Gamma, Q \vdash R}{\Gamma, P \longrightarrow Q \vdash R} \text{ (e impE)}$$

$$\text{no right-intro rule for} \bot \qquad \frac{}{\Gamma, \bot \vdash P} \text{ (e FalseE)}$$

$$\frac{\Gamma, P \vdash \bot}{\Gamma \vdash \neg P} \text{ (notI)} \qquad \frac{}{\Gamma, P, \neg P \vdash R} \text{ (e notE)} \qquad \frac{}{\Gamma \vdash P \vee \neg P} \text{ (excluded\_middle)}$$

Note: `e someRule` is short for `erule someRule`.

Note: in the above presentation left-hand-sides are *sets* of formulas.

# An Old Friend Revisited

$$
\dfrac{\dfrac{\dfrac{\dfrac{}{S, \neg S \vdash R}\ (\text{e notE}) \qquad \dfrac{}{R, \neg S \vdash R}\ (\text{assumption})}{(S \vee R), \neg S \vdash R}\ (\text{e disjE})}{(S \vee R) \wedge \neg S \vdash R}\ (\text{e ConjE})}{\vdash (S \vee R) \wedge \neg S \longrightarrow R}\ (\text{impI})
$$

## Re-using proofs: The Cut rule

So far, all proofs have been self-contained; they have only used the pre-existing rules of inference.

By the completeness theorem, this suffices to prove everything that is true, but can lead to extremely repetitive proofs.

# Re-using proofs: The Cut rule

So far, all proofs have been self-contained; they have only used the pre-existing rules of inference.

By the completeness theorem, this suffices to prove everything that is true, but can lead to extremely repetitive proofs.

The cut rule: (we "cut" $P$ into the proof)

$$\frac{\Gamma \vdash P \qquad \Gamma, P \vdash Q}{\Gamma \vdash Q}$$

allows the use of a *lemma* $P$ in a proof of $Q$. We can now reuse $P$ multiple times in the proof of $Q$.

# Re-using proofs: The Cut rule

So far, all proofs have been self-contained; they have only used the pre-existing rules of inference.

By the completeness theorem, this suffices to prove everything that is true, but can lead to extremely repetitive proofs.

The cut rule: (we "cut" $P$ into the proof)

$$\frac{\Gamma \vdash P \qquad \Gamma, P \vdash Q}{\Gamma \vdash Q}$$

allows the use of a *lemma* $P$ in a proof of $Q$. We can now reuse $P$ multiple times in the proof of $Q$.

In Isabelle:

`cut_tac lemmaName` — adds the conclusion of `lemmaName` as a new assumption, and its assumptions as new subgoals

`subgoal_tac` $P$ — adds $P$ as a new assumption, and introduces $P$ as a new subgoal.

# Why should you believe Isabelle?

When Isabelle says "`No subgoals!`" why should we believe that we have *really* proved something? Is Isabelle sound?

It is doing non-trivial work behind the scenes: unification, rewriting, maintaining a database of theorems+assumptions, automatic proof.

# Why should you believe Isabelle?

When Isabelle says "`No subgoals!`" why should we believe that we have *really* proved something? Is Isabelle sound?

It is doing non-trivial work behind the scenes: unification, rewriting, maintaining a database of theorems+assumptions, automatic proof.

Isabelle uses two strategies to maintain soundness:

- A small trusted kernel: internally, every proof is broken down into primitive rule applications which are checked by a small piece of hand-verified code. This is the "LCF" model. So new *proof procedures* cannot introduce unsoundness.
- Encourages *definitional* extension of the logic: new concepts are introduced by definition rather than axiomatisation (more on this in Lecture 6). So new definitions cannot introduce unsoundness.

# Why should you believe Isabelle?

When Isabelle says "`No subgoals!`" why should we believe that we have *really* proved something? Is Isabelle sound?

It is doing non-trivial work behind the scenes: unification, rewriting, maintaining a database of theorems+assumptions, automatic proof.

Isabelle uses two strategies to maintain soundness:

- A small trusted kernel: internally, every proof is broken down into primitive rule applications which are checked by a small piece of hand-verified code. This is the "LCF" model. So new *proof procedures* cannot introduce unsoundness.
- Encourages *definitional* extension of the logic: new concepts are introduced by definition rather than axiomatisation (more on this in Lecture 6). So new definitions cannot introduce unsoundness.

Threats to (practical) soundness still exist, including: Have we proved what we thought we proved? Are the formulas displayed on screen correctly? …

See: Pollack, R. *How to Believe a Machine-Checked Proof*, 1997 (non-examinable).

# Summary

- More tools for proving propositions in Isabelle
  - The `erule`, `drule`, `frule` methods
  - Their —`_tac` variants
  - $L$-systems, and Cut rules (`cut_tac`, `subgoal_tac`).
  - See the propositional logic exercises and examples:
    - Tutorial 1 and Additional Exercise on the AR webpage;
    - The Isabelle theory file `Prop.thy`;
    - Start using Isabelle (if you haven't done so already).
- How Isabelle maintains soundness
  - Small trusted kernel
  - Definitional extension instead of axiomatic extension
- Next time:
  - First-Order Logic: $\forall x.P$ and $\exists x.P$