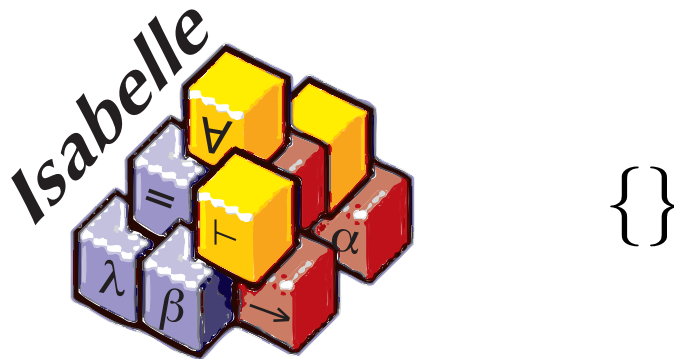


# Introduction to Isabelle

Clemens Ballarin  
Universität Innsbruck



# Contents

- ▶ Intro & motivation, getting started with Isabelle
- ▶ Foundations & Principles
  - ▶ Lambda Calculus
  - ▶ Types & Classes
  - ▶ Natural Deduction
- ▶ **Proof & Specification Techniques**
  - ▶ Isar: mathematics style proofs
  - ▶ **Inductively defined sets, rule induction**
  - ▶ Datatypes, structural induction
  - ▶ Recursive functions & code generation

# Sets

# Sets in Isabelle

Type 'a set: sets over type 'a

- ▶  $\{\}, \{e_1, \dots, e_n\}, \{x. P x\}$
- ▶  $e \in A, A \subseteq B$
- ▶  $A \cup B, A \cap B, A - B, \text{UNIV}, -A$
- ▶  $\bigcup x \in A. B x, \bigcap x \in A. B x, \bigcup A, \bigcap A$
- ▶  $\{i..j\}$
- ▶  $\text{insert} :: \alpha \Rightarrow \alpha \text{ set} \Rightarrow \alpha \text{ set}$
- ▶  $f' A \equiv \{y. \exists x \in A. y = f x\}$
- ▶ ...

# Proofs about Sets

Natural deduction proofs:

- ▶ equalityI:  $\llbracket A \subseteq B; B \subseteq A \rrbracket \Longrightarrow A = B$
- ▶ subsetI:  $(\bigwedge x. x \in A \Longrightarrow x \in B) \Longrightarrow A \subseteq B$
- ▶ IntI:  $\llbracket x \in A; x \in B \rrbracket \Longrightarrow x \in A \cap B$
- ▶ IntE:  $\llbracket x \in A \cap B; \llbracket x \in A; x \in B \rrbracket \Longrightarrow P \rrbracket \Longrightarrow P$
- ▶ ... (see LNCS 2283)

# Bounded Quantifiers

- ▶  $\forall x \in A. P x \equiv \forall x. x \in A \longrightarrow P x$
- ▶  $\exists x \in A. P x \equiv \exists x. x \in A \wedge P x$
- ▶ balll:  $(\bigwedge x. x \in A \Longrightarrow P x) \Longrightarrow \forall x \in A. P x$
- ▶ bspec:  $\llbracket \forall x \in A. P x; x \in A \rrbracket \Longrightarrow P x$
- ▶ bexl:  $\llbracket P x; x \in A \rrbracket \Longrightarrow \exists x \in A. P x$
- ▶ bexE:  $\llbracket \exists x \in A. P x; \bigwedge x. \llbracket x \in A; P x \rrbracket \Longrightarrow Q \rrbracket \Longrightarrow Q$

# Demo: Sets

# Inductive Definitions

# Example

$$\frac{}{\langle \text{skip}, \sigma \rangle \longrightarrow \sigma} \quad \frac{[[e]]\sigma = v}{\langle x := e, \sigma \rangle \longrightarrow \sigma[x \mapsto v]}$$

$$\frac{\langle c_1, \sigma \rangle \longrightarrow \sigma' \quad \langle c_2, \sigma' \rangle \longrightarrow \sigma''}{\langle c_1; c_2, \sigma \rangle \longrightarrow \sigma''}$$

$$\frac{[[b]]\sigma = \text{False}}{\langle \text{while } b \text{ do } c, \sigma \rangle \longrightarrow \sigma}$$

$$\frac{[[b]]\sigma = \text{True} \quad \langle c, \sigma \rangle \longrightarrow \sigma' \quad \langle \text{while } b \text{ do } c, \sigma' \rangle \longrightarrow \sigma''}{\langle \text{while } b \text{ do } c, \sigma \rangle \longrightarrow \sigma''}$$

# What Does This Mean?

- ▶  $\langle c, \sigma \rangle \longrightarrow \sigma'$  fancy syntax for a relation  $(c, \sigma, \sigma') \in E$
- ▶ Relations are sets:  $E :: (\text{com} \times \text{state} \times \text{state}) \text{ set}$
- ▶ The rules define a set **inductively**.

**But which set?**

# Simpler Example

$$\frac{}{0 \in N} \qquad \frac{n \in N}{n + 1 \in N}$$

- ▶  $N$  is the set of natural numbers  $\mathbb{N}$ .
- ▶ But why not the set of real numbers?  
 $0 \in \mathbb{R}, n \in \mathbb{R} \implies n + 1 \in \mathbb{R}$
- ▶  $\mathbb{N}$  is the **smallest** set that is **consistent** with the rules.

## Why the smallest set?

- ▶ **No junk**. Only what must be in  $N$  shall be in  $N$ .
- ▶ Gives rise to a nice proof principle: **rule induction**.
- ▶ Greatest set occasionally also useful: coinduction.

## Formally

Set of rules  $R \subseteq \mathcal{P}(A) \times A$

$$\frac{a_1 \in X \quad \dots \quad a_n \in X}{a \in X}$$

with  $a_1, \dots, a_n, a \in A$  defines set  $X \subseteq A$ .

## Applying rules $R$

$$\hat{R} B \equiv \{x. \exists H. (H, x) \in R \wedge H \subseteq B\}$$

## Example

$$\begin{aligned} R &\equiv \{(\{\}, 0)\} \cup \{(\{n\}, n+1). n \in \mathbb{R}\} \\ \hat{R} \{3, 6, 10\} &= \{0, 4, 7, 11\} \end{aligned}$$

# The Set

**Definition**  $B$  is  $R$ -closed iff  $\hat{R} B \subseteq B$

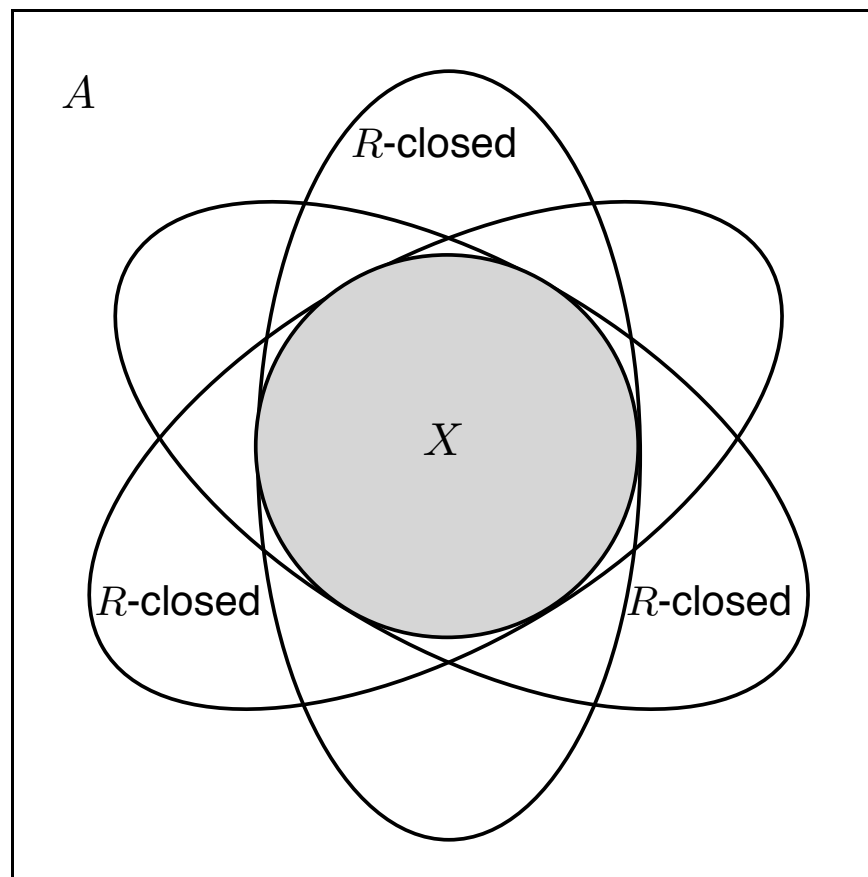
**Definition**  $X$  is the least  $R$ -closed subset of  $A$

This does always exist:

**Fact**  $B_1$   $R$ -closed  $\wedge$   $B_2$   $R$ -closed  $\implies B_1 \cap B_2$   $R$ -closed

**Hence**  $X = \bigcap \{B \subseteq A. B \text{ } R\text{-closed}\}$

# Generation from Above



# Rule Induction

The rules

$$\frac{}{0 \in N} \quad \frac{n \in N}{n + 1 \in N}$$

induce the induction principle

$$\llbracket P \ 0; \bigwedge n. P \ n \implies P \ (n + 1) \rrbracket \implies \forall x \in X. P \ x$$

In general

$$\frac{\forall (\{a_1, \dots, a_n\}, a) \in R. P \ a_1 \wedge \dots \wedge P \ a_n \implies P \ a}{\forall x \in X. P \ x}$$

# Why Does This Work?

$$\frac{\forall(\{a_1, \dots, a_n\}, a) \in R. P a_1 \wedge \dots \wedge P a_n \implies P a}{\forall x \in X. P x}$$

Proof

$$\forall(\{a_1, \dots, a_n\}, a) \in R. P a_1 \wedge \dots \wedge P a_n \implies P a$$

says  $\{x. P x\}$  is  $R$ -closed

but  $X$  is the least  $R$ -closed set

hence  $X \subseteq \{x. P x\}$

which means  $\forall x \in X. P x$  qed.

# Rules with Side Conditions

$$\frac{a_1 \in X \quad \dots \quad a_n \in X \quad C_1 \quad \dots \quad C_m}{a \in X}$$

Induction scheme

$$\begin{aligned} & (\forall (\{a_1, \dots, a_n\}, a) \in R. P a_1 \wedge \dots \wedge P a_n \wedge \\ & \quad C_1 \wedge \dots \wedge C_m \wedge \\ & \quad \{a_1, \dots, a_n\} \subseteq X \implies P a) \\ & \implies \\ & \forall x \in X. P x \end{aligned}$$

# $X$ as Fixed Point

How to compute  $X$ ?

$X = \bigcap \{B \subseteq A. B \text{ } R\text{-closed}\}$  hard to work with.

Instead

View  $X$  as least fixed point,  $X$  least set with  $\hat{R} X = X$ .

Fixed points can be approximated by iteration

$$X_0 = \hat{R}^0 \{\} = \{\}$$

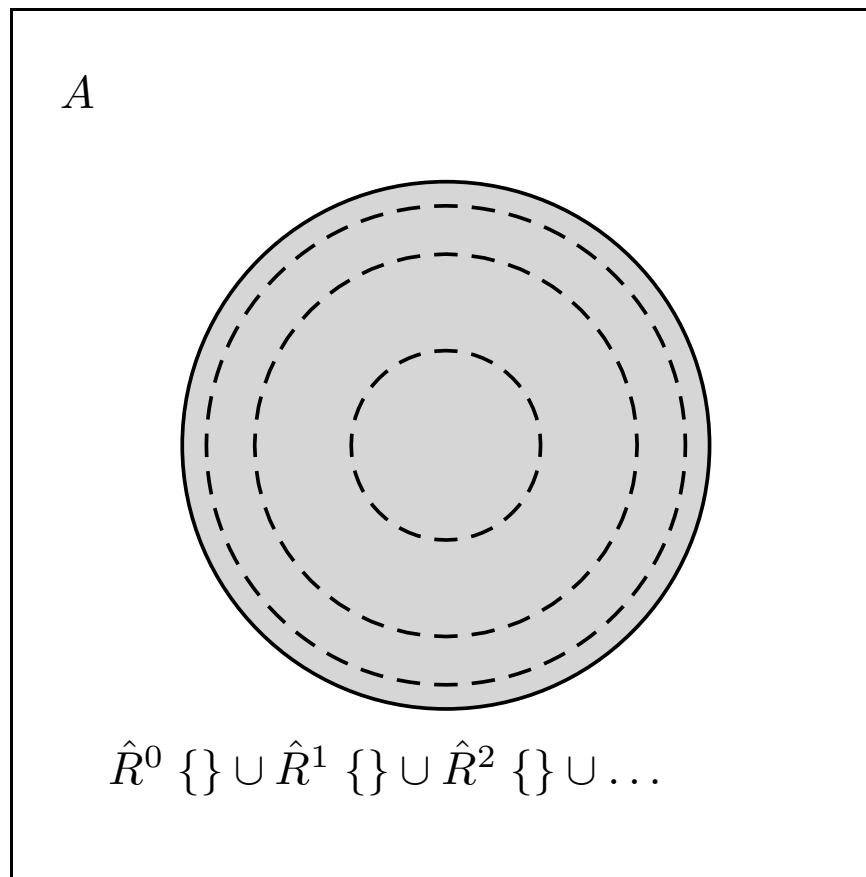
$$X_1 = \hat{R}^1 \{\} = \text{rules without hypotheses}$$

$\vdots$

$$X_n = \hat{R}^n \{\}$$

$$X_\omega = \bigcup_{n \in \mathbb{N}} (\hat{R}^n \{\}) = X$$

# Generation from Below



# Demo: Inductive Definitions

# Inductive Definitions in Isabelle

## Inductive Set

**inductive\_set**  $S$  [ :: " $\tau$ " ]

**where**

rule<sub>1</sub>: " $\llbracket x_1 \in S; \dots; x_k \in S; C_1; \dots; C_m \rrbracket \implies t \in S$ "  
| rule<sub>2</sub>: ...  
|  $\vdots$   
| rule<sub>n</sub>: ...

## Inductive Predicate

**inductive**  $P$  [ :: " $\tau$ " ]

**where**

rule<sub>1</sub>: " $\llbracket P x_1; \dots; P x_k; C_1; \dots; C_m \rrbracket \implies P t$ " ...

# Inductive Definitions in Isabelle

```
inductive_set  $S$   
where rule1: ... | rule $n$ : ...
```

## Proved Theorems

- ▶ Introduction rules  
S.rule<sub>1</sub> ... S.rule <sub>$n$</sub>  S.intros
- ▶ Elimination rule (case analysis)  
S.cases
- ▶ Induction rules  
S.induct (default) S.inducts (for mutual induction)

# Case Analysis

**inductive\_set**  $S$

**where** emptyl: " $\{\} \in S$ " | insertl: " $A \in S \implies \text{insert } a A \in S$ "

## Proof Schema

**fix**  $x$  **assume** " $x \in S$ " **then show** " $P x$ "

**proof cases**

**assume** " $x = \{\}$ "

**show** " $P x$ "  $\langle \text{proof} \rangle$

**next**

**fix**  $a$  and  $A$

**assume** " $x = \text{insert } a A$ "

**and** " $A \in S$ "

**show** " $P x$ "  $\langle \text{proof} \rangle$

**qed**

**proof cases**

**case** emptyl

**show** ?thesis  $\langle \text{proof} \rangle$

**next**

**case** (insertl  $A a$ )

**show** ?thesis  $\langle \text{proof} \rangle$

**qed**

# Rule Induction

**inductive\_set**  $S$

**where** emptyl: " $\{\} \in S$ " | insertl: " $A \in S \implies \text{insert } a A \in S$ "

## Proof Schema

**fix**  $x$  **assume** " $x \in S$ " **then show** " $P x$ "

**proof induct**

**show** " $P \{\}$ "  $\langle proof \rangle$

**next**

**fix**  $a$  and  $A$

**assume** " $A \in S$ "

**and** " $P A$ "

**show** " $P (\text{insert } a A)$ "  $\langle proof \rangle$

**qed**

**proof induct**

**case** emptyl

**show** ?case  $\langle proof \rangle$

**next**

**case** (insertl  $A a$ )

**show** ?case  $\langle proof \rangle$

**qed**

State variables  $A a$  in order of occurrence in the rule.

# How the Rule is Selected

## Cases

- ▶ no information — classical case split  $P \vee \neg P$
- ▶ cases “ $t$ ” — datatype exhaustion (type of  $t$ )
- ▶ chained fact  $t \in S$  — elimination of inductive set  $S$
- ▶ cases rule:  $R$  — explicit selection of rule

## Induct

- ▶ cases  $x$  — datatype induction (type of induction variable)  
Multiple induction variables for mutual induction.
- ▶ chained fact  $x \in S$  — rule induction for  $S$
- ▶ cases rule:  $R$  — explicit selection of rule

# A Remark on Style

- ▶ **case** ( $\text{rule}_i \ x \ y$ ) ... **show** ?case  
is easy to write and maintain
- ▶ **fix**  $x \ y$  **assume**  $\langle \text{formula} \rangle$  ... **show**  $\langle \text{formula} \rangle$   
is easier to read:
  - ▶ All information is shown locally
  - ▶ No contextual references (e.g. ?case)

# Demo: Rule Induction in Isar

# Induction on Structured Statements

## Generalising Variables

To generalise variables  $x_1, \dots, x_n$  in the induction hypothesis:

(induct **arbitrary**:  $x_1, \dots, x_n$ )

## Pushing Assumptions

To push assumptions  $A_1, \dots, A_m$  into the induction use chaining:

**from** R **and**  $A_1 \dots A_m$  **show**  $P$  **proof** induct  $\langle proof \rangle$

or

**show**  $P$  **using** R **and**  $A_1 \dots A_m$  **proof** induct  $\langle proof \rangle$

# We Have Seen so far . . .

- ▶ Sets in Isabelle
- ▶ Inductive Definitions
- ▶ Rule induction
- ▶ Fixed points
- ▶ Case analysis and induction in Isar