COMP 4161

NICTA Advanced Course

Advanced Topics in Software Verification

Gerwin Klein, June Andronick, Toby Murray

$$\{\}$$

Content

	Rough timeline
➜ Intro & motivation, getting started	[1]
➜ Foundations & Principles	
• Lambda Calculus, natural deduction	[2,3,4[a]]
• Higher Order Logic	[5,6[b],7]
• Term rewriting	[8,9,10[c]]
➜ Proof & Specification Techniques	
• Isar	[11,12[d]]
• Inductively defined sets, rule induction	[13[e],15]
• Datatypes, recursion, induction	[16,17[f],18,19]
• Calculational reasoning, mathematics style proofs	[20]
• Hoare logic, proofs about programs	[21[g],22,23]

[a] a1 out; [b] a1 due; [c] a2 out; [d] a2 due; [e] session break; [f] a3 out; [g] a3 due

Last Time

- ➜ Sets
- ➜ Type Definitions
- ➜ Inductive Definitions

NICTA

How Inductive Definitions Work

The Nat Example

NICTA

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

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

Why the smallest set?

- ➜ Objective: **no junk**. Only what must be in X shall be in X.
- ➜ Gives rise to a nice proof principle (rule induction)

Formally

$$\text{Rules } \frac{a_1 \in X \quad \dots \quad a_n \in X}{a \in X} \text{ with } a_1, \dots, a_n, a \in A$$

define set $X \subseteq A$

Formally: set of rules $R \subseteq A$ set $\times A$ (R, X possibly infinite)

Applying rules R to a set B: $\hat{R}\ B \equiv \{x.\ \exists H.\ (H, x) \in R \wedge H \subseteq B\}$

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

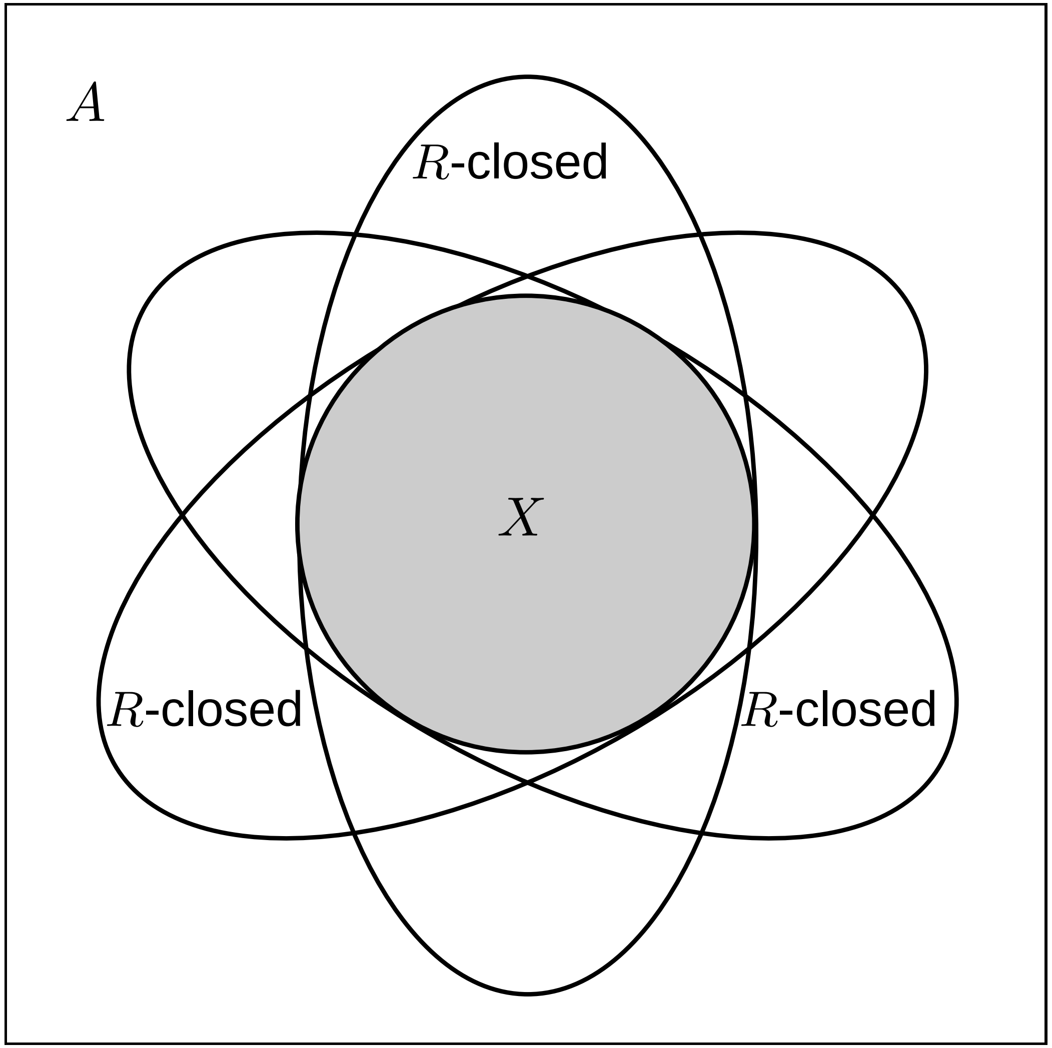
Definition: B is R-closed iff $\hat{R}\ B \subseteq B$

Definition: X is the least R-closed subset of A

This does always exist:

Fact: $X = \bigcap\{B \subseteq A.\ B\ R\text{−closed}\}$

Generation from Above



Rule Induction

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

induces induction principle

$$[\![P\ 0;\ \bigwedge n.\ P\ n \Longrightarrow P\ (n+1)]\!] \Longrightarrow \forall x \in X.\ P\ x$$

In general:

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

Why does this work?

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

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

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

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

induction scheme:

$$(\forall (\{a_1, \ldots a_n\}, a) \in R.\ P\ a_1 \wedge \ldots \wedge P\ a_n \wedge$$
$$C_1 \wedge \ldots \wedge C_m \wedge$$
$$\{a_1, \ldots, a_n\} \subseteq X \Longrightarrow P\ a)$$

$$\Longrightarrow$$

$$\forall x \in X.\ P\ x$$

X as Fixpoint

How to compute X?

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

Instead: view X as least fixpoint, X least set with $\hat{R}\ X = X$.

Fixpoints can be approximated by iteration:

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

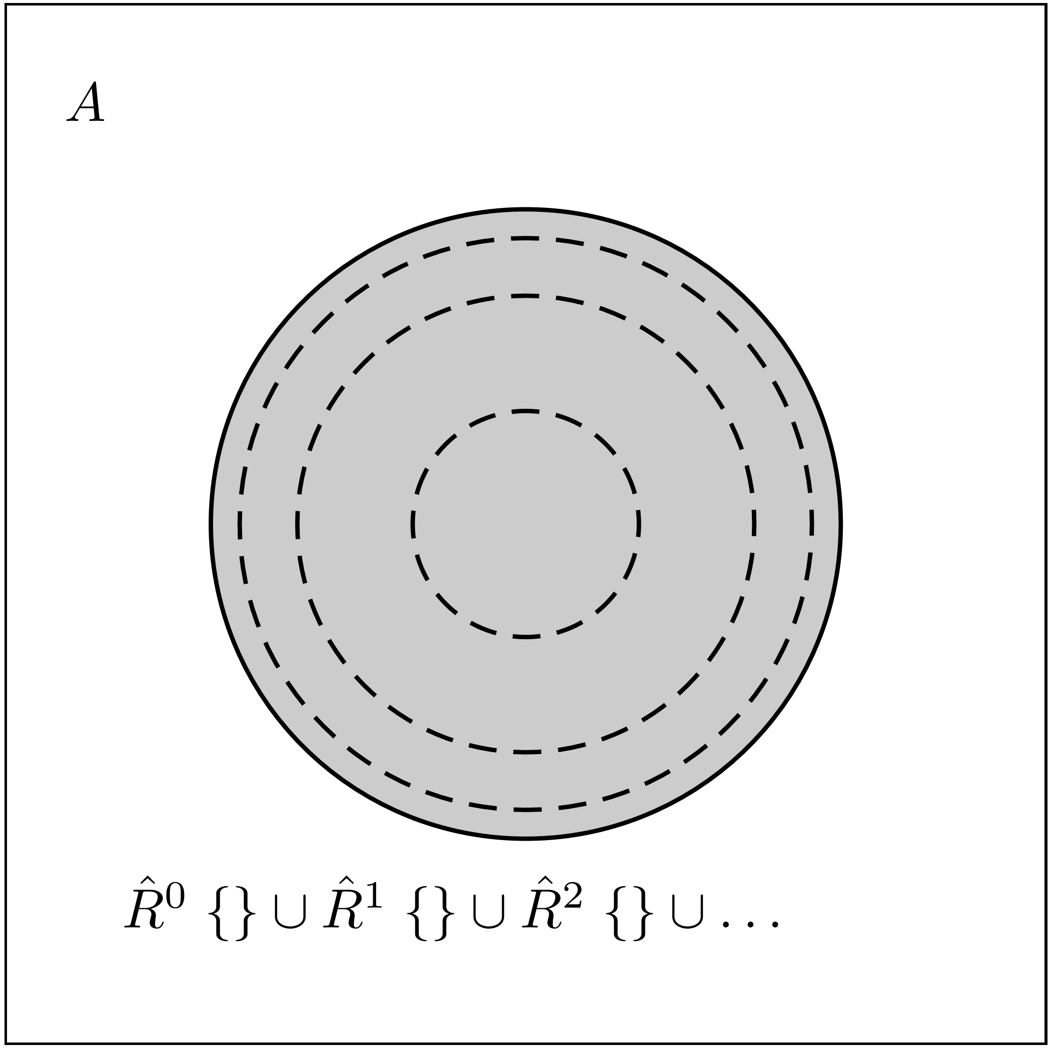
$$X_1 = \hat{R}^1\ \{\} = \text{rules without hypotheses}$$

$$\vdots$$

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

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

Generation from Below



Does this always work?

Knaster-Tarski Fixpoint Theorem:

Let $(A, \leq)$ be a complete lattice, and $f :: A \Rightarrow A$ a monotone function.
Then the fixpoints of f again form a complete lattice.

Lattice:

Finite subsets have a greatest lower bound (meet) and least upper bound (join).

Complete Lattice:

All subsets have a greatest lower bound and least upper bound.

Implications:

- ➜ least and greatest fixpoints exist (complete lattice always non-empty).
- ➜ can be reached by (possibly infinite) iteration. (Why?)

Exercise

Formalize the this lecture in Isabelle:

- ➜ Define **closed** $f\ A$ :: $(\alpha$ set $\Rightarrow \alpha$ set$) \Rightarrow \alpha$ set $\Rightarrow$ bool
- ➜ Show closed $f\ A \wedge$ closed $f\ B \Longrightarrow$ closed $f\ (A \cap B)$ if f is monotone (**mono** is predefined)
- ➜ Define **lfpt** f as the intersection of all f-closed sets
- ➜ Show that lfpt f is a fixpoint of f if f is monotone
- ➜ Show that lfpt f is the least fixpoint of f
- ➜ Declare a constant R :: $(\alpha$ set $\times\ \alpha)$ set
- ➜ Define $\hat{R}$ :: α set $\Rightarrow \alpha$ set in terms of R
- ➜ Show soundness of rule induction using R and lfpt $\hat{R}$

Rule Induction in Isar

Inductive definition in Isabelle

inductive $X :: \alpha \Rightarrow$ bool

where

rule$_1$: "$[\![X\ s; A]\!] \Longrightarrow X\ s'$"

$\vdots$

$|$ rule$_n$: $\ldots$

Rule induction

show "$X\ x \Longrightarrow P\ x$"

proof (induct rule: X.induct)

  fix s and s' **assume** "$X\ s$" and "A" and "$P\ s$"

  ...

  show "$P\ s'$"

next

⋮

qed

Abbreviations

show "$X\ x \Longrightarrow P\ x$"
proof (induct rule: X.induct)
  case rule$_1$
  . . .
  show ?case
next
⋮
next
  case rule$_n$
  . . .
  show ?case
qed

Implicit selection of induction rule

assume A: "$X\ x$"

⋮

show "$P\ x$"

using A **proof** induct

⋮

qed

lemma assumes A: "$X\ x$" **shows** "$P\ x$"

using A **proof** induct

⋮

qed

Renaming free variables in rule

case (rule$_i$ $x_1 \ldots x_k$)

Renames first k variables in rule$_i$ to $x_1 \ldots x_k$.

A remark on style

- **case** (rule$_i$ x y) . . . **show** ?case
  is easy to write and maintain

- **fix** x y **assume** $formula$ . . . **show** $formula'$
  is easier to read:
  - all information is shown locally
  - no contextual references (e.g. ?case)

Demo: Rule Induction in Isar

We have learned today ...

- Formal background of inductive definitions
- Definition by intersection
- Computation by iteration
- Formalisation in Isabelle
- Rule Induction in Isar