Course 2D1453, 2006-07

Advanced Formal Methods

Lecture 7: Isabelle – Sets

Mads Dam
KTH/CSC

Material from L. Paulson

Basic Constructions

types $'\alpha$ set = $'\alpha \to$ bool

Note that HOL sets are always *typed*

Easy to define basic set constructions:
$\emptyset :: '\alpha$ set = $\lambda x.$ false
$\{t\} = \lambda x.\ x = t$
$t \in A = A\ t$
$A \subseteq B = \forall x.\ A\ x \to B\ x$
$A \cup B = \lambda x.\ \neg(A\ x) \to B\ x$
$\forall x \in A.\ P\ x = \forall x.\ x \in A \to P\ x$
$\exists\ x \in A.\ P\ x = \exists x.\ x \in A \wedge P\ x$

Exercises

Exercise 1:
Represent in Isabelle the following set operations:
1. $A \cap B$
2. $\cap_{x \in A} B\ x$, $\cup_{x \in A} B\ x$
3. insert :: $'\alpha \Rightarrow '\alpha$ set $\Rightarrow '\alpha$ set

Proof Rules

Prove lemmas following natural deduction-style introduction and elimination rules:

subsetI: $(\wedge\ x.\ x \in A \Rightarrow x \in B) \Rightarrow A \subseteq B$
subsetE: $[\![\ A \subseteq B;\ x \in A\ ]\!] \Rightarrow x \in B$
ballI: $(\wedge\ x.\ x \in A \Rightarrow P\ x) \Rightarrow \forall x \in A.\ P\ x$
ballE: $[\![\ \forall x \in A.\ P\ x\ ;\ x \in A\ ]\!] \Rightarrow P\ x$

etc. etc.

Finite Sets

Inductive definition:
- The empty set is finite
- Adding an element to a finite set produces a finite set
- These are the only finite sets

HOL encoding:
consts Fin :: $'\alpha$ set set
inductive Fin
Intros
  $\emptyset \in$ Fin
  $A \in$ Fin $\Rightarrow$ insert a $A \in$ Fin

Example: Even Numbers

Inductively:
- 0 is even
- If n is even then so is n + 2
- These are the only even numbers

In Isabelle HOL:
consts Ev :: nat set
inductive Ev
intros
  $0 \in$ Ev
  $n \in$ Ev $\Rightarrow n + 2 \in$ Ev

Inductively Defined Sets

Definition mechanism:

- Define carrier set
- Declare set to be inductively defined
- Declare introduction methods

Declaration **inductive** tells Isabelle to produce a number of proof rules:

- Introduction rules
- Induction rules
- Elimination rules (case construction)
- Rule inversion rules

Example: Proof by Induction

Definition of set Ev produces induction principle

Goal: $m \in Ev \Rightarrow m + m \in Ev$

Proof:

- $m = 0 \rightarrow 0 + 0 \in Ev$
- $m = n + 2$ and $n \in Ev$ and $m \in Ev$ and $n + n \in Ev$ $\Rightarrow m + m = (n + 2) + (n + 2) = ((n + n) + 2) + 2 \in Ev$

Rule Induction for Ev

To prove $n \in Ev \Rightarrow P\ n$

by rule induction on $n \in Ev$ we must prove

- $P\ 0$
- $P\ n \Rightarrow P\ (n + 2)$

Isabelle-generated induction principle:

$[\![n \in Ev ;\ P\ 0 ;\ \wedge n.\ P\ n \Rightarrow P\ (n+2)]\!] \Rightarrow P\ n$

An elimination rule for Ev!

Rule Inversion

Isabelle proves this by induction:

ev.cases:

$[\![n \in Ev;$

$n = 0 \Rightarrow P ;$

$\wedge m.\ [\![n = Suc(Suc\ m);\ m \in Ev]\!] \Rightarrow P]\!] \Rightarrow P$

inductive_cases <Name>: Suc(Suc n) $\in$ Ev

Instantiates even.cases to produce:

<Name>: $[\![Suc(Suc\ n) \in Ev ;\ n \in Ev \Rightarrow P]\!] \Rightarrow P$

Equivalently:

$Suc(Suc\ n) \in Ev \Rightarrow n \in Ev$

Mutual Induction

Even and odd numbers:

consts Ev :: nat set

Odd :: nat set

inductive Ev Odd

intros

zero: $0 \in Ev$

evenI: $n \in Odd \Rightarrow Suc\ n \in Ev$

oddI: $n \in Ev \Rightarrow Suc\ n \in Odd$

Exercise

Exercise 2: Define a recursive datatype of implicational formulas containing constants tt and ff and binary constructor -> (infixed, probably, for readability). Define inductively the set of provable formulas as the least set containing, for all formulas F, G, H, the formulas

F -> F

F -> (G -> F)

(F -> (G -> H)) -> (F -> G) -> (F -> H)

and closed under modus ponens (F and F -> G in the set implies G in the set).

Formulate a predicate "valid" on formulas, and show, in Isabelle, that all provable formulas a valid.