Logic in Coq

Prof. Clarkson
Fall 2017

Today’s music: *Autologic* by Rage Against The Machine
Review

Previously in 3110:
• Functional programming in Coq
• Proofs about simple programs

Today:
• Logic in Coq, at the CS 2800 level
TYPES
Propositions

Check list.
list : Type -> Type

Check 42 = 42.
42 = 42 : Prop

Check 42 = 3110.
42 = 3110 : Prop
Propositions

Theorem one_plus_one_is_two : 1 + 1 = 2.
Proof. trivial. Qed.

Check one_plus_one_is_two.
one_plus_one_is_two : 1 + 1 = 2

Print one_plus_one_is_two.
one_plus_one_is_two = eq_refl : 1 + 1 = 2

Check 1 + 1 = 2.
1 + 1 = 2 : Prop
Propositions

Check Prop.

\( \text{Prop} : \text{Type} \)

Check Type.

\( \text{Type} : \text{Type} \)

Logical formulas. Things that can be proved.
Sets

Let $x := 42$.
Check $x$.

$x : \text{nat}$

Check $\text{nat}$.

$\text{nat} : \text{Set}$

Check $\text{bool}$.

$\text{bool} : \text{Set}$
Sets

Check Set.

Set : Type

Check Type.

Type : Type

Program data types. Things that can be computed.
Type hierarchy

Type

Prop

Set
Logical connectives

- Implication: $p \rightarrow p$
- Conjunction: $p \land p$
- Disjunction: $p \lor p$
- Negation: $\neg p$
Implication

Theorem p_implies_p : forall P:Prop, P -> P.
Proof.
  intros P. intros P_assumed. assumption.
Qed.

Check p_implies_p.
p_implies_p : forall P:Prop, P -> P
Implication

Print \( p \_\text{implies} \_p \).

\[
p \_\text{implies} \_p = \text{fun} \ (P : \text{Prop}) \ (P\_\text{assumed} : P) \Rightarrow P\_\text{assumed} \\
: \forall P : \text{Prop}, \ P \Rightarrow P
\]
Coq proofs are functional programs
**Conjunction**

Theorem and_fst : forall P Q, P \( \land \) Q -> P.
Proof.

intros P Q PandQ.
destruct PandQ as [P_holds Q_holds].
assumption.
Qed.

- case analysis on why hypothesis holds
- names for pieces of evidence for hypothesis
Conjunction

Check and_fst.

\( \text{and}_f\text{st} : \forall P \ Q : \text{Prop}, \ P \land Q \rightarrow P \)

Print and_fst.

\( \text{and}_f\text{st} = \)

\( \text{fun} \ (P \ Q : \text{Prop}) \ (\text{PandQ} : P \land Q) \Rightarrow \text{match PandQ with} \)

\( | \text{conj} \ P\text{\_holds} \_ \Rightarrow P\text{\_holds} \)

\( \text{end} \)

\( : \forall P \ Q : \text{Prop}, \ P \land Q \rightarrow P \)
Conjunction

and\_fst is a function

takes two propositions as input

and proof of their conjunction

pattern matches to extract first proof

returns that proof

Print and\_fst.

\texttt{and\_fst = fun (P Q : Prop) (PandQ : P /\ Q) => match PandQ with} \texttt{| conj P\_holds _ => P\_holds}

\texttt{end : forall Prop \rightarrow Prop, P /\ Q \rightarrow P}
Conjunction

Theorem and_ex : 42=42 \(\land\) 43=43.
Proof. split. trivial. trivial. Qed.

Print and_ex.

\texttt{and\_ex} = \texttt{conj eq\_refl eq\_refl} : 42 = 42 \(\land\) 43 = 43
Theorem and_comm: for all $P, Q$,
\[ P \land Q \rightarrow Q \land P. \]
Proof.
  intros $P, Q, P \land Q$.
  destruct $P \land Q$ as $[P_{holds}, Q_{holds}]$.
  split.
  all: assumption.
Qed.

**Case analysis on why**
$P \land Q$ holds
Conjunction

Print and_comm.

\[
\text{and\_comm} = \text{fun } (P \text{ Q : Prop}) \text{ (PandQ : } P \text{ } /\text{\_ } Q) \Rightarrow \\
\text{match PandQ with} \\
| \text{conj P\_holds Q\_holds } \Rightarrow \\
\text{conj Q\_holds P\_holds} \\
\text{end} \\
\text{: forall P Q : Prop, } P \text{ } /\text{\_ } Q \Rightarrow Q \text{ } /\text{\_ } P
\]
Disjunction

Theorem or_left :
  forall (P Q : Prop), P -> P \ / Q.
Proof.
  intros P Q P_holds. left. assumption.
Qed.

Print or_left.

or_left =
fun (P Q : Prop) (P_holds : P) =>
  or_introl P_holds
  : forall P Q : Prop, P -> P \ / Q
Disjunction

Check or_introl.

\[\text{or_introl} : \forall A B : \text{Prop}, A \to A \vee B\]

Print or_introl.

\[\text{Inductive or } (A B : \text{Prop}) : \text{Prop} := \]
\[\text{or_introl} : A \to A \vee B\]
\[\text{or_intror} : B \to A \vee B\]

proves left side

two constructors, one to prove each side
Disjunction

Theorem or_comm :
   \forall P Q, P \lor Q \rightarrow Q \lor P.
Proof.
intros P Q PorQ. destruct PorQ.
- right. assumption.
- left. assumption.
Qed.

choose to prove left side
choose to prove right side

case analysis on why P \lor Q holds
Disjunction

Print or_comm.

\[
\text{or\_comm} = \\
\text{fun (P Q : Prop) (PorQ : P \lor Q) \Rightarrow} \\
\text{match PorQ with} \\
| \text{or\_introl H \Rightarrow or\_intror H} \\
| \text{or\_intror H \Rightarrow or\_introl H} \\
\text{end} \\
: \forall P Q : \text{Prop}, P \lor Q \Rightarrow Q \lor P
\]
Negation

Theorem explosion : forall P, False → P.
Proof.
   intros P false_holds.
   contradiction.
Qed.
Negation

Print False.

Inductive False : Prop :=

Print not.

not = fun A : Prop => A -> False :

not is a function: not P is P -> False

False has no constructors
Negation

Theorem contra_implies_anything :
  forall P Q, P \land \neg P \rightarrow Q.
Proof.
  intros P Q PandnotP.
  destruct PandnotP as [P_holds notP_holds].
  contradiction.
Qed.

give names to the two pieces of evidence for why P \land \neg P holds
Upcoming events

• N/A

This is logical.

THIS IS 3110