CS720

Logical Foundations of Computer Science

Lecture 8: Logical connectives in Coq

Tiago Cogumbreiro

Today we will learn...

- more logic connectives
- constructive logic (and its relation to classical logic)
- building propositions with functions
- building propositions with inductive definitions

UMass Boston

Logic connectives

Truth

Truth

Truth can be encoded in Coq as a proposition that always holds, which can be described as a proposition type with a single constructor with 0-arity.

UMass Boston

Truth example

(Done in class.)

UMass Boston

Equivalence

Logical equivalence

Split equivalence in goal

- When induction is required, prove each side by induction independently.
  Split, and prove each side in its own theorem by induction.

UMass Boston

Apply equivalence to assumption

Interpret equivalence as equality

The library lets you treat an equivalence as an equals:

Tactics , , and all handle equivalence as well.

Existential quantification

Existential quantification

Notation:

- To conclude a goal we can use tactics which yields .
- To use a hypothesis of type , you can use

Use exist for existential in goal

To conclude a goal we can use tactics which yields .

- Give the value that satisfies the equality.
- You can play around with exists to figure out what makes sense.

UMass Boston

Destruct existential in assumption

Constructive logic

is not classical logic

Constructive logic is not classical logic

- Coq implements a constructive logic
- Every proof consists of evidence that is constructed
- You cannot assume the law of the excluded middle
  (proofs that appear out of thin air)
- Truth tables may fail you!
  Especially if there are negations involved.

The following are **unprovable** in constructive logic (and therefore in Coq):

UMass Boston

Building propositions with functions

Building propositions with functions

List membership example

- Computation cannot match on propositions
- Computations destruct types, not propositions

UMass Boston

Building propositions with data structures (inductively)

Enumerated propositions

Recall enumerated types?

You can think of true as an enumerated type.

Many equivalent proofs

Many equivalent proofs

Yet, same as having one

- We can prove with or with , we still just have
- What happens when we do a case analysis on ? Show when holds, then show when holds.

UMass Boston

Falsehood

Falsehood in Coq is represented by an **empty** type.

This explains why case analysis proves the following goal:

UMass Boston

Composite inductive propositions

Disjunction

Conjunction

Adding parameters to predicates

Adding parameters to predicates

Alternative definition of Bar

Existential

Recursive inductive propositions

Defining inductively

Defining inductively

Fixed parameters in inductive propositions

Defining even numbers