Logic Foundations

Induction: Proof by Induction

熊英飞 胡振江
信息学院计算机系

2021年3月16日

An Example

```
Theorem plus_n_O_firsttry : forall n:nat,
  n = n + o.

Proof.
  intros n.
  simpl. (* Does nothing! *)
Abort.
```

An Example

```
Theorem plus_n_O_secondtry : forall n:nat,
 n = n + o.

Proof.
 intros n. destruct n as [| n'] eqn:E.
  - (* n = o *)
   reflexivity. (* so far so good... *)
  - (* n = S n' *)
   simpl.      (* ...but here we are stuck again *)
Abort.
```

An Example

```
Theorem plus_n_O : forall n:nat, n = n + o.

Proof.
 intros n. induction n as [| n' IHn'].
  - (* n = o *)    reflexivity.
  - (* n = S n' *) simpl. rewrite <- IHn'.
reflexivity.  Qed.
```

Another Example

```
Theorem minus_n_n : forall n,
  minus n n = 0.

Proof.
intros n. induction n as [| n' IHn'].
  - (* n = 0 *)
    simpl. reflexivity.
  - (* n = S n' *)
    simpl. rewrite -> IHn'. reflexivity.
Qed.
```

Proofs Within Proofs

```
Theorem mult_0_plus' : forall n m : nat,
  (0 + n) * m = n * m.

Proof.
  intros n m.
  assert (H: 0 + n = n). { reflexivity. }
  rewrite -> H.
  reflexivity.
Qed.
```

Proofs Within Proofs

```
Theorem plus_rearrange_firsttry : forall n m p q : nat,
  (n + m) + (p + q) = (m + n) + (p + q).

Proof.
  intros n m p q.
  (* We just need to swap (n + m) for (m + n)... seems
    like plus_comm should do the trick! *)
  rewrite -> plus_comm.
  (* Doesn't work... Coq rewrites the wrong plus! :-( *)
Abort.
```

Proofs Within Proofs

```
Theorem plus_rearrange : forall n m p q : nat,
  (n + m) + (p + q) = (m + n) + (p + q).

Proof.
  intros n m p q.
  assert (H: n + m = m + n).
  { rewrite -> plus_comm. reflexivity. }
  rewrite -> H. reflexivity.
Qed.
```

Formal vs. Informal Proof

- Informal proofs are algorithms; formal proofs are code.
- A proof is an act of communication.
  - A "valid" proof is one that makes the reader believe P.
  - There is no universal standard.
- The formal proof is much more explicit in some ways (e.g., the use of reflexivity) but much less explicit in others (e.g., the proof state).

⬇

Write formal proof more algorithmically!

作业

- 完成 Induction.v中的至少10个练习题。