软件科学基础

Induction: Proof by Induction

熊英飞
北京大学

多文件程序开发

- `From LF Require Export Basics.`
    - 从LF目录读取编译后的Basics.vo文件并导入
- LF目录
    - _CoqProject文件指明当前目录为LF
        - `-Q . LF`
- 编译得到Basics.vo文件
    - Proof General和RocqIDE会自动编译
        - VSCode不行
    - Linux下编译
        - Make Basics.vo或make
    - Windows下编译
        - rocq compile –Q . LF Basics.v

复习

- 是否能通过simpl和reflexivity证明n+0=n

```
Theorem add_0_r_firsttry : forall n:nat,
  n + 0 = n.
Proof. intros n.
(*   n : nat
 *   =========
 *   n + 0 = n
 *)
  simpl.(** [Rocq Proof View]
 *   n : nat
 *   =========
 *   n + 0 = n
 *)
 (* Does nothing! *)
Abort.
```

Destruct也不行

```
Theorem add_0_r_secondtry : forall n:nat,
  n + 0 = n.
Proof.
  intros n. destruct n as [| n'] eqn:E.
  - (* n = 0 *)
    reflexivity. (* so far so good... *)
  - (* n = S n’ *)
    (* S n' + 0 = S n’ *)
    simpl.
    (* S (n' + 0) = S n’ *)
Abort.
```

结构归纳法 Structural Induction

- 数学归纳法：
  - 首先证明性质P对0成立
  - 然后证明P(k)->P(k+1)
- 结构归纳法：
  - 对于任意递归定义的类型
  - 证明P对该类型的每一个构造子都成立
    - 如果构造子接受该类型的参数，则假设P对参数成立
- 数学归纳法是结构归纳法在自然数上的特例

采用结构归纳法证明

```
Theorem add_0_r : forall n:nat, n + 0 = n.
(** [Rocq Proof View]
 * 1 subgoal
 *
 *   =========================
 *   forall n : nat, n + 0 = n
 *)
Proof. intros n.
(** [Rocq Proof View]
 * 1 subgoal
 *
 *   n : nat
 *   =========
 *   n + 0 = n
 *)
```

采用结构归纳法证明

```
 induction n as [| n' IHn'].
(** [Rocq Proof View]
 * 2 subgoals
 *
 *   ==============
 *   0 + 0 = 0
 *
 * subgoal 2 is:
 *  S n' + 0 = S n'
 *)
```

为变量命名，可省略

采用结构归纳法证明

```
  - reflexivity.
  - (** [Rocq Proof View]
 * 1 subgoal
 *
 *   n' : nat
 *   IHn' : n' + 0 = n'
 *   ==================
 *   S n' + 0 = S n'
 *)
    simpl. rewrite -> IHn'. reflexivity.
Qed.
```

归纳假设

Induction同时作用于一个变量的多个实例

```
Theorem minus_n_n : forall n,
  minus n n = 0.
Proof.
  (* WORKED IN CLASS *)
  intros n. induction n as [| n' IHn'].
  - (* n = 0 *)
    simpl. reflexivity.
  - (* n = S n' *)
    simpl. rewrite -> IHn'. reflexivity.  Qed.
```

Replace：引入等价子定理

```
Theorem mult_0_plus' : forall n m : nat,
  (n + 0 + 0) * m = n * m.
Proof.
  intros n m.
  replace (n + 0 + 0) with n.
  - reflexivity.
  - rewrite add_comm. simpl.
    rewrite add_comm. reflexivity.
Qed.
```

采用replace帮助rewrite定位

```
Theorem plus_rearrange_firsttry : forall n m p q : nat,
  (n + m) + (p + q) = (m + n) + (p + q).
Proof. intros n m p q.
(** [Rocq Proof View]
 * 1 subgoal
 *
 *   n, m, p, q : nat
 *   =================================
 *   n + m + (p + q) = m + n + (p + q)
 *)
```

采用replace帮助rewrite定位

```
  rewrite add_comm.
(** [Rocq Proof View]
 * 1 subgoal
 *
 *   n, m, p, q : nat
 *   =================================
 *   p + q + (n + m) = m + n + (p + q)
 *)
```

- Rewrite只重写了最外层的加法

采用replace帮助rewrite定位

```
Theorem plus_rearrange : forall n m p q : nat,
  (n + m) + (p + q) = (m + n) + (p + q).
Proof.
  intros n m p q.
  replace (n + m) with (m + n).
  - reflexivity.
  - rewrite add_comm. reflexivity.
Qed.
```

形式化证明 vs 非形式化证明

- Rocq写的是形式化证明
- 数学课上/论文里写的是非形式化证明
- 非形式化证明的作用
  - 交流——非形式化证明可以写得更简洁，更抽象，帮助交流
  - 理解——采用Rocq自动证明策略很容易试出来证明,但更重要的是确保自己能理解证明

作业

- 完成Induction.v中standard非optional的6道习题
  - 请使用最新英文版教材