

「計算と論理」 Software Foundations その7

五十嵐 淳

igarashi@kuis.kyoto-u.ac.jp

京都大学

January 7, 2014

今日のメニュー

Prop.v (注意: 教科書は改訂途中なのか, 説明が前後していたり混乱してます.)

- 帰納的に定義される命題
 - ▶ 偶数性
 - ▶ 偶数に関する帰納法
 - ▶ 「美しい」数と「ゴージャスな」数
 - ▶ 導出についての場合わけ・inversion
- (帰納的に定義される) 関係
 - ▶ 不等号

偶数の集合の帰納的定義

偶数の集合 EV は、以下のふたつの規則を満たす最小の(自然数の部分)集合である。

- 0 は EV の元である
 - n が EV の元ならば $S(S(n))$ は EV の元である
-
- 規則を満たす集合はいくらでもある(例えば 1 以外の自然数の集合)
 - 最小性で「ゴミ」を取り除く
- ⇒ 帰納的定義 = 規則を満たす + 最小性

述語「偶数性」の帰納的定義

集合を述語と思うと「述語の帰納的定義」になる

自然数 n が偶数である ($ev\ n$ と書く) とは, 以下の規則から帰納的に定義される.

- $ev\ 0$ である
- 任意の自然数 n について, $ev\ n$ ならば $ev\ (S\ (S\ (n)))$ である

自然演繹流で書けば

新しい原子命題 $ev\ n$:

$$\frac{\Gamma \vdash n : \text{nat}}{\Gamma \vdash ev\ n : \text{Prop}} \quad (\text{Ev-P})$$

- 導入規則:

$$\frac{}{\Gamma \vdash ev\ 0} \quad (\text{Ev-I1})$$

$$\frac{\Gamma \vdash ev\ n}{\Gamma \vdash ev\ (S(S(n)))} \quad (\text{Ev-I2})$$

Coq での帰納的述語の定義

```
Inductive ev : nat -> Prop :=  
  | ev_0 : ev 0  
  | ev_SS : forall n:nat, ev n -> ev (S (S n)).
```

- 新しい(自然数を引数とする)命題 **ev** の定義
 - ▶ $\text{nat} \rightarrow \text{Prop} \Rightarrow$ 自然数に関する述語
- コンストラクタ 導入規則
- コンストラクタの型 規則の内容
 - ▶ $\text{forall } n:\text{nat} \Rightarrow$ 規則のパラメータ
 - ▶ $\rightarrow \Rightarrow$ 規則の前提と結論の間の水平線

例: 「4は偶数である」

カーリー・ハワード同型対応を思い出して!

```
Coq < Check ev_0.
```

```
ev_0  
      : ev 0
```

```
Coq < Check (ev_SS 0).
```

```
ev_SS 0  
       : ev 0 -> ev 2
```

```
Coq < Check (ev_SS 0 ev_0).
```

```
ev_SS 0 ev_0  
        : ev 2
```

```
Coq < Check (ev_SS 2 (ev_SS 0 ev_0)).
```

```
ev_SS 2 (ev_SS 0 ev_0)  
        : ev 4
```

例: 「4 は偶数である」

Theorem four_is_even : even 4.

Proof.

 apply ev_SS. apply ev_SS. apply ev_0.

Qed.

Print four_is_even.

例

Theorem even_plus4 :

forall n:nat, ev n -> ev (4 + n).

Proof.

Qed.

偶数に関する帰納法

$P(n)$ を自然数 n の性質について述べた命題とする

偶数に関する帰納法の原理

「任意の偶数 n について $P(n)$ 」は以下と同値

- $P(0)$ かつ
- 任意の自然数 n' について $P(n')$ ならば $P(S(S n'))$

- ふたつの場合分けは偶数性の定義に対応

自然演繹流で書くと

$$\frac{\Gamma \vdash P[0] \quad \Gamma, n : \text{nat}, IH : P[n] \vdash P[S(S(n))]}{\Gamma \vdash \forall n : \text{nat}, \text{ev } n \rightarrow P[n]} \quad (\text{Ev-E})$$

- ev が含意の条件部に来ているので，一種の除去規則
と思える

証明の例

evenb 関数 (Basics.v 参照) についての性質

定理: 自然数 n が偶数ならば, **evenb** n は **true** を返す

証明: 偶数に関する帰納法 .

- $n = 0$ の場合. **evenb** 0 は明らかに **true** を返す
- $n = S(S(n'))$ かつ n' が偶数の場合 .
evenb ($S(S(n'))$) を計算すると, **evenb** n' と等しい . 一方, 帰納法の仮定より n' については **evenb** n' が **true** を返すので, **evenb** n も **true** を返す .

(証明終)

Coq での証明

```
Definition even (n:nat) : Prop :=  
  evenb n = true.
```

```
Theorem ev__even : forall n, ev n -> even n.
```

```
Proof.
```

```
  intros n E. induction E as [| n' E'].
```

```
  Case "E = ev_0".
```

```
    unfold even. reflexivity.
```

```
  Case "E = ev_SS n' E'".
```

```
    unfold even. apply IHE'.
```

```
Qed.
```

induction E って?

- $E : ev\ n$ ということは,
 - ▶ $E = ev_0$ かつ $n = 0$
 - ▶ $E = ev_SS\ n'\ E'$ かつ $E' : ev\ n'$ (つまり n' も偶数)

のいずれか .

- E は (枝分かれしない導出木で) ある種のリスト構造をしている!

⇒ induction E はリストに関する帰納法のようなもの!

- 「偶数性の導出に関する帰納法」ともいう

同じ証明の別の書き方

定理: 自然数 n が偶数ならば, $\text{evenb } n$ は true を返す

証明: $\text{ev } n$ の導出に関する帰納法 . 最後の規則について場合分け .

- ev_0 の場合 . この時 $n = 0$. $\text{evenb } 0$ は明らかに true を返す
- ev_{SS} の場合 , この時ある n' について $n = S(S(n'))$ かつ $\text{ev } n'$ である . $\text{evenb } (S(S(n')))$ を計算すると , $\text{evenb } n'$ と等しい . 一方 , 帰納法の仮定より n' については $\text{evenb } n'$ が true を返すので , $\text{evenb } n$ も true を返す .

(証明終)

“= true” vs. Inductive による命題定義

Q: evenb があるのに, なぜ ev を定義するの?

A1: 偶数に関する帰納法が使える

A2: 一般的に「 P 性」の(規則による)定義が書けても, 何かが「 P 性」を満たすかを判定する関数が(簡単に)書けるとは限らない

今日のメニュー

Prop.v

- 帰納的に定義される命題
 - ▶ 偶数性
 - ▶ 偶数に関する帰納法
 - ▶ 「美しい」数と「ゴージャスな」数
 - ▶ 導出についての場合わけ・inversion
- (帰納的に定義される) 関係
 - ▶ 不等号

「美しい」自然数 (深い意味は特でない)

定義: 「美しい」自然数

自然数 n が美しいとは, $n = 0, 3, 5$ のいずれかであるか, 他の美しい自然数の和で表されることをいう.

推論規則による「美しさ」の定義:

$$\frac{}{0 \text{ is beautiful}} \quad (\text{B0})$$

$$\frac{}{3 \text{ is beautiful}} \quad (\text{B3})$$

$$\frac{}{5 \text{ is beautiful}} \quad (\text{B5})$$

$$\frac{n \text{ is beautiful} \quad m \text{ is beautiful}}{n + m \text{ is beautiful}} \quad (\text{BSUM})$$

「8 は美しい」ことの導出

その1:

$$\frac{\frac{\overline{3 \text{ is beautiful}} \quad B3 \quad \overline{5 \text{ is beautiful}} \quad B5}{8 \text{ is beautiful}}}{BSUM}$$

その2:

$$\frac{\frac{\overline{5 \text{ is beautiful}} \quad B5 \quad \frac{\frac{\overline{0 \text{ is beautiful}} \quad B0 \quad \overline{3 \text{ is beautiful}} \quad B3}{3 \text{ is beautiful}}}{BSUM}}{8 \text{ is beautiful}}}{BSUM}$$

Coq での「美しさ」の定義

```
Inductive beautiful : nat -> Prop :=  
  b_0      : beautiful 0  
| b_3      : beautiful 3  
| b_5      : beautiful 5  
| b_sum    : forall n m,  
  beautiful n -> beautiful m -> beautiful (n+m).
```

- 一行目: beautiful は (自然数を添字とする) 命題
 - ▶ beautiful 0, beautiful 1, ... は (0 = 1 のような) 命題
- 残り: 「正しい」 beautiful **n** を定義
 - ▶ 推論規則を記号化

Coq での「美しさ」の証明

Theorem three_is_beautiful: beautiful 3.

Proof.

apply b_3. (* 規則 B3 による *)

Qed.

Theorem eight_is_beautiful: beautiful 8.

Proof.

apply b_sum with (n:=3) (m:=5).

(* 規則 BSUM 中の n, m を具体化 *)

apply b_3.

apply b_5.

Qed.

導出に関する帰納法

- Inductive による型定義
 - その型のデータに関する帰納法
 - ▶ 「任意の自然数 n について...」の証明
- Inductive による命題定義
 - その命題の証明に関する帰納法 (?)
 - ▶ 「beautiful n ならば...」の証明
 - ▶ 「任意の beautiful n の証明オブジェクト E について...」の証明

「 n は美しい数である」の導出

beautiful n の導出 E について，以下のいずれかがいえる：

- $E = b_0$ かつ $n = 0$
- $E = b_3$ かつ $n = 3$
- $E = b_5$ かつ $n = 5$
- $E = b_{\text{sum } n1 \ n2 \ E1 \ E2}$ かつ
 - ▶ $n = n1+n2$ かつ
 - ▶ $E1$ は beautiful $n1$ の (E より小さい) 証明オブジェクト，かつ，
 - ▶ $E2$ は beautiful $n2$ の (E より小さい) 証明オブジェクト

beautiful n の導出に関する帰納法 , または美しい数に関する帰納法

「任意の beautiful n なる n について $P(n)$ 」つまり
「任意の n について beautiful n ならば $P(n)$ 」と以下は同値

- $P(0)$ かつ
- $P(3)$ かつ
- $P(5)$ かつ
- 任意の n_1, n_2 について $P(n_1)$ かつ $P(n_2)$ ならば $P(n_1 + n_2)$

導出の形に関する場合分けとの対応!

例: 帰納的定義される命題の同値性証明(1)

```
Inductive gorgeous : nat -> Prop :=
  g_0 : gorgeous 0
| g_plus3 :
  forall n, gorgeous n -> gorgeous (3+n)
| g_plus5 :
  forall n, gorgeous n -> gorgeous (5+n).
```

明らかに(?), 自然数がゴージャスであることと美しいことは同値

⇒ 証明してみよう!

- その前に...

ゴージャス数の導出

gorgeous n の導出 E について，以下のいずれかがいえる：

- $E = g_0$ かつ $n = 0$
- $E = g_{plus3} n_1 E_1$ かつ $n = 3 + n_1$ かつ E_1 は gorgeous n_1 の (E より小さい) 導出
- $E = g_{plus5} n_1 E_1$ かつ $n = 5 + n_1$ かつ E_1 は gorgeous n_1 の (E より小さい) 導出

ゴージャスな数についての帰納法

gorgeous n の導出に関する帰納法, またはゴージャスな数に関する帰納法

「任意の gorgeous n なる n について $P(n)$ 」つまり
「任意の n について gorgeous n ならば $P(n)$ 」と以下は同値

- $P(0)$ かつ
- 任意の n_1 について $P(n_1)$ ならば $P(3 + n_1)$
- 任意の n_1 について $P(n_1)$ ならば $P(5 + n_1)$

ゴージャスな自然数は美しい!

Theorem gorgeous_beautiful : forall n,
gorgeous n -> beautiful n.

Proof.

intros n E.

induction E as [| n' E' | n' E'].

(* 導出に関する帰納法! *)

Case "g_0". apply b_0.

Case "g_plus3".

apply b_sum. apply b_3. apply IHgorgeous.

Case "g_plus5".

apply b_sum. apply b_5. apply IHgorgeous.

Qed.

今日のメニュー

Prop.v

- 帰納的に定義される命題
 - ▶ 偶数性
 - ▶ 偶数に関する帰納法
 - ▶ 「美しい」数と「ゴージャスな」数
 - ▶ 導出についての場合わけ・inversion
- (帰納的に定義される) 関係
 - ▶ 不等号

導出についての inversion

偶数性の導出を使った推論:

- 帰納法 (induction)
- (素朴な) 場合わけ (destruct)
- 導出を「遡る」 (inversion)

例題:

Theorem ev_minus2: forall n,
 ev n ev (pred (pred n)).

Theorem SSev_ev : forall n,
 ev (S (S n)) -> ev n.

Theorem `ev_minus2`: forall n,
 ev n ev (pred (pred n)).

Proof.

intros n E.

inversion E as [| n' E'].

Case "E = ev_0". simpl. apply ev_0.

Case "E = ev_SS n' E'". simpl. apply E'.

Qed.

destruct だと失敗する

```
Theorem SSev_ev_firsttry : forall n,  
  ev (S (S n)) -> ev n.
```

Proof.

```
  intros n E.
```

```
  destruct E as [| n'].
```

```
    (* The goal is still "ev n" *)
```


矛盾する場合は取り除かれる!

```
Theorem SSev__even : forall n,  
  ev (S (S n)) -> ev n.
```

Proof.

```
  intros n E.
```

```
  inversion E as [| n' E'].
```

```
    (* E = ev_0 なわけがない! *)
```

```
    apply E'.
```

Qed.

任意の自然数について $ev(S(S(n)))$ ならば $ev\ n$

(証明) $ev(S(S(n)))$ の導出について場合分け

- ev_0 の場合: ありえない.
- ev_SS の場合: 導出の前提は $ev\ n$ のはずなので題意が示せる.

(証明終)

導出に対する inversion

文脈で $H : I$ (I は帰納的に定義された命題) とする時, inversion H は:

- コンストラクタ (導出規則) 毎に場合わけ
 - ▶ ev_0, ev_SS の場合
- 各場合での前提条件...
 - ▶ ...を文脈に追加
 - ★ ev_SS の場合の前提 $ev\ n$ が追加
 - ▶ ...が矛盾している場合は場合そのものの除去
 - ★ ev_0 の場合 ($S\ (S\ n) = 0$ はありえない)

を一気に行う

今日のメニュー

Prop.v

- 帰納的に定義される命題
 - ▶ 偶数性
 - ▶ 偶数に関する帰納法
 - ▶ 「美しい」数と「ゴージャスな」数
 - ▶ 導出についての場合わけ・inversion
- (帰納的に定義される) 関係
 - ▶ 不等号

命題としての関係

- 一引数の命題 (一引数述語): 「もの」の性質を表す
 - ▶ beautiful, even など
- 二引数の命題 (二引数述語): 「もの」と「もの」の関係を表す
 - ▶ eq

関係「以下」の帰納的定義

```
Inductive le : nat -> nat -> Prop :=  
  | le_n : forall n, le n n  
  | le_S : forall n m, (le n m) -> (le n (S m))  
Notation "m <= n" := (le m n).
```

導出規則:

$$\frac{}{n \leq n} \quad (\text{LE-N})$$

$$\frac{n \leq m}{n \leq S m} \quad (\text{LE-S})$$

「以下」に関する証明

基本的には `ev`, `beautiful` などと同じ:

- ゴールにあるならコンストラクタを `apply`
- 文脈にあるなら `inversion`

```
Theorem test_le1 : 3 <= 3.
```

```
Proof. apply le_n. Qed.
```

```
Theorem test_le2 : 3 <= 6.
```

```
Proof.
```

```
  apply le_S. apply le_S.
```

```
  apply le_S. apply le_n. Qed.
```

Theorem test_le3 : ~ (2 <= 1).

Proof.

intros H.

inversion H. inversion H1. Qed.

「未満」の定義

Definition $lt (n m : nat) := le (S n) m$.

Notation " $m < n$ " $:= (lt m n)$.

- le を使わずに直接帰納的な定義をしたら？

宿題：1/14 午前10:30 締切

- Exercise: `double_even` (1), `ev_sum` (2), `b_times2` (2), `gorgeous_plus13` (2), `gorgeous_sum` (2), `inversion_practice` (1)
- 解答を書き込んだ **Prop.v までのファイルを全てを**まるごとオンライン提出システムを通じて提出
- 以下をコメント欄に明記:
 - ▶ 講義・演習に関する質問，わかりにくいと感じたこと，その他気になること．（「特になし」はダメです．）
 - ▶ 友達に教えてもらったなら、その人の名前，他の資料（web など）を参考にした場合，その情報源（URL など）．