# 定理証明支援系 Coq での<br>理論的なリーゾニング

演習問題, 集中講義 千葉大学大学院

アフェルト レナルド

産業技術総合研究所

2017 年 01 月 25–26 日

本ファイルの補題を証明する．ただし, Coq の自動タクティクを使わない (`assumption`, `trivial`, `auto`, `intuition`, `tauto`, `firstorder`, etc.)．授業で見たタクティクは十分である (まとめ: 表 1, 細かいシンタクスに関して授業に参考)．特に, 含意の除去の際, `cut` より `apply` を使う．必要なら, 古典論理を使うことができる (特に, 授業で見た `bottom_c` 補題).

| 自然演繹<br>(ルールを一つしかい表示しない) | 定義 | タクティク<br>導入 | <br>除去 |
|---|---|---|---|
| $\dfrac{\Gamma, A \vdash B}{\Gamma \vdash A \to B} \to_i$ | | `intros` | `apply` |
| $\dfrac{\Gamma \vdash \bot}{\Gamma \vdash A} \bot_e$ | `Inductive False : Prop := .` | | `destruct` |
| $\dfrac{\Gamma \vdash A \qquad \Gamma \vdash B}{\Gamma \vdash A \wedge B} \wedge_i$ | `Inductive and`<br>`(A B : Prop) : Prop :=`<br>`\| conj : A -> B -> A /\ B.` | `split` | `destruct as [...]` |
| $\dfrac{\Gamma \vdash A}{\Gamma \vdash A \vee B} \vee_i^l$<br>$\dfrac{\Gamma \vdash B}{\Gamma \vdash A \vee B} \vee_i^r$ | `Inductive or`<br>`(A B : Prop) : Prop :=`<br>`\| or_introl : A -> A \/ B`<br>`\| or_intror : B -> A \/ B` | `left`<br>`right` | `destruct as [...\|...]` |
| $\dfrac{\Gamma, A \vdash \bot}{\Gamma \vdash \neg A} \neg_i$ | `Definition not (A:Prop) :=`<br>`A -> False.` | `intros` | `apply` |
| $\dfrac{\Gamma \vdash P[x := t]}{\Gamma \vdash \exists x, P\, x} \exists_i$ | `Inductive ex (A : Type)`<br>`(P : A -> Prop) : Prop :=`<br>`\| ex_intro :`<br>`forall x : A, P x ->`<br>`exists x, P x` | `exists` | `destruct as [...]` |
| $\dfrac{\Gamma \vdash A \quad \Gamma \text{ の中に } x \text{ は自由変数ではない}}{\Gamma \vdash \forall x. A} \forall_i$ | | `intros` | `apply` |
| $\dfrac{\Gamma, \neg A \vdash \bot}{\Gamma \vdash A} \bot_c$ | `Lemma bottom_c (A : Prop) :`<br>`((~A) -> False) -> A.` | `apply bottom_c.`<br>`intros na.` | |

表 1: Coq の基本的なタクティクのまとめ

# 1 CoqIDE のインタフェースの基本的な使い方

授業で見た Hilbert の $S$ 公理を再現する．通常, スクリプトはコマンド `Proof` で開始する．コマンドはピリオドで終わる． をクリックすると, コマンドは実行される． をクリックすると, 前のステップに戻る． をクリックすると, カーソルまでのコマンドを実行する．通常, スクリプトは `Qed` で終わる．

(本資料では LaTeX のシンボルとして表示するが,) 入力基本的には ASCII で書く．$\to$ は`->`, $\wedge$ は`/\`, $\neg$ は`~`, 等と書く．

`Lemma` $\mathit{hilbertS}$ $(A\ B\ C : \texttt{Prop}) :$
$(A \to B \to C) \to (A \to B) \to A \to C.$

# 2 [Pie16] からの例

## 2.1 [Pie16, Slide 58]

`Lemma` $\mathit{exo1}$ $(P\ Q : \texttt{Prop}) :\ P \to (Q \to P).$

`Lemma` $\mathit{exo2}$ $(P\ Q : \texttt{Prop}) :\ P \to (\neg P \to Q).$

`Lemma` $\mathit{exo3}$ $(P\ Q\ R : \texttt{Prop}) :\ (P \to Q) \to ((Q \to R) \to (P \to R)).$

対偶:

`Lemma` $\mathit{exo4}$ $(P\ Q : \texttt{Prop}) :\ (P \to Q) \to (\neg Q \to \neg P).$

古典矛盾に当たる補題を用意する．まず, Coq の標準ライブラリから `Classical` を設定する:

`Require Import` *Classical*.

`Classical` ライブラリからの二重否定 (`NNPP` 補題) または排中律 (`classic` 補題) を利用できる．

`Lemma` $\mathit{bottom\_c}$ $(A : \texttt{Prop}) :\ ((\tilde{}A) \to \mathit{False}) \to A.$

古典矛盾を用いて, 次の補題を証明する:

`Lemma` $\mathit{exo5}$ $(P\ Q : \texttt{Prop}) :\ (\neg Q \to \neg P) \to (P \to Q).$

二重否定 (古典論理の一つの定義):

`Lemma` $\mathit{exo6}$ $(P : \texttt{Prop}) :\ \neg\neg P \to P.$

`Lemma` $\mathit{exo7}$ $(P : \texttt{Prop}) :\ P \to \neg\neg P.$

`Lemma` $\mathit{exo8}$ $(P\ Q\ R : \texttt{Prop}) :\ (P \to (Q \to R)) \to (P \wedge Q \to R).$

`Lemma` $\mathit{exo9}$ $(P\ Q\ R : \texttt{Prop}) :\ (P \wedge Q \to R) \to (P \to (Q \to R)).$

次の補題を証明するために, 上記の補題を利用できる:

`Lemma` $\mathit{exo10}$ $(P : \texttt{Prop}) :\ P \wedge \neg P \to \mathit{False}.$

`Lemma` $\mathit{exo11}$ $(P : \texttt{Prop}) :\ \mathit{False} \to P \wedge \neg P.$

### 2.2 [Pie16, Slide 68]

De Morgan の法則. $\leftarrow$ の方向は古典論理による. 古典矛盾を利用する.

`Lemma` *exo12* $(P\ Q : \texttt{Prop}) :\ P \lor Q \leftrightarrow \neg\,(\neg\,P \land \neg\,Q)$.

`Lemma` *exo13* $(P : \texttt{Prop}) :\ \neg\,P \leftrightarrow (P \to \mathit{False})$.

`Lemma` *exo14* $(P\ Q : \texttt{Prop}) :\ (P \leftrightarrow Q) \leftrightarrow (P \to Q) \land (Q \to P)$.

## 3 [DNR03] からの例

[DNR03, Section 1.3.4] の証明を再現する. ただし, ビュレット`-`, `+`, `*`を用いて, 証明木の構造を明確にする.

### 3.1 [DNR03, 例 1.3.4, p. 33].

メモ: `A <-> B` は `(A -> B) /\ (B -> A)` として定義されている.

`Lemma` *exemple134* $(A\ B\ C : \texttt{Prop}) :\ (A \land B \to C) \leftrightarrow (A \to B \to C)$.

### 3.2 [DNR03, 例 1.3.5, p. 34].

`Lemma` *exemple135* $(A\ B\ C : \texttt{Prop}) :\ (C \to A) \lor (C \to B) \to (C \to A \lor B)$.

### 3.3 [DNR03, 例 1.3.6, p. 34].

`Lemma` *exemple_136* $(X : \texttt{Type})\ (A\ B : X \to \texttt{Prop}) :$
$((\forall\, x,\ A\ x) \lor (\forall\, x,\ B\ x)) \to \forall\, x,\ A\ x \lor B\ x.$

### 3.4 [DNR03, 例 1.3.7, p. 34].

`Lemma` *exemple_137* $(X : \texttt{Type})\ (A\ B : X \to \texttt{Prop}) :$
$(\exists\, x,\ A\ x \land B\ x) \to ((\exists\, x,\ A\ x) \land (\exists\, x,\ B\ x)).$

### 3.5 [DNR03, 例 1.3.8, p. 35].

次の De Morgan 法則の例を証明するために, 古典矛盾を利用する:

`Lemma` *exemple_138* $(A\ B : \texttt{Prop}) :\ \neg\,(A \land B) \to (\neg\,A \lor \neg\,B)$.

証明済みの補題を利用することもできる.

`Lemma` *exemple_138'* $(A\ B : \texttt{Prop}) :\ \neg\,(A \land B) \to (\neg\,A \lor \neg\,B)$.

### 3.6 [DNR03, 例 1.3.9, p. 35].

[DNR03] から離れないように, “lstinline!rewrite!の代わりに, 関数“lstinline!eq_ind!を利用する.

Lemma *exemple_139* $(X : \texttt{Type}) : \forall\ (x1\ x2 : X),\ x1 = x2 \to x2 = x1.$

### 3.7 [DNR03, 例 1.3.10, p. 36].

上記と同様.

Lemma *exemple_140* $(X : \texttt{Type}) : \forall\ (x1\ x2\ x3 : X),\ x1 = x2 \land x2 = x3 \to x1 = x3.$

## 4 帰納的型を利用せず, 論理結合子を形式化

含意を表現するために, 型理論による関数の型を使った. 論理結合子を表現するために, 帰納的型を使った. 実は, 帰納的型を使わない形式化もある.

今回, 型理論の product を用いて論理結合子を表現し, 標準ライブラリの定義と比較する. 定義を展開するために, タクティク `unfold` を利用できる.

### 4.1 矛盾の例

矛盾は何でもの命題の証明を返すとして定義できる:

Definition *FALSE* $: \texttt{Prop} := \forall\ (P : \texttt{Prop}),\ P.$

Goal *FALSE*.
unfold *FALSE*.
intros *p*.
Abort.

Lemma *FALSE_False* $: \mathit{FALSE} \leftrightarrow \mathit{False}.$

### 4.2 帰納的型を使わない定義

論理積の二階形式化 (任意の命題に対する全称記号のせい). 一階形式化 $A \land B = \neg(A \to \neg B)$ もあるが, 論理が古典になる.

Definition *AND* $(A\ B : \texttt{Prop}) := \forall\ (P : \texttt{Prop}),\ (A \to B \to P) \to P.$

Definition *OR* $(A\ B : \texttt{Prop}) := \forall\ (P : \texttt{Prop}),\ ((A \to P) \to (B \to P) \to P).$

Definition *EX* $(A : \texttt{Type})\ (P : A \to \texttt{Prop}) := \forall\ (Q : \texttt{Prop}),\ (\forall\ a,\ P\ a \to Q) \to Q.$

Definition *EQ* $(A : \texttt{Type})\ (a\ a' : A) := \forall\ (P : A \to \texttt{Prop}),\ P\ a \to P\ a'.$

### 4.3 標準ライブラリとの比較

導入と除去ルールは補題として再現できる.

`Lemma` $\mathit{SPLIT}$ $(A\ B : \texttt{Prop}) :\ A \to B \to \mathit{AND}\ A\ B.$

`Lemma` $\mathit{PROJ1}$ $(A\ B : \texttt{Prop}) :\ \mathit{AND}\ A\ B \to A.$

`Lemma` $\mathit{PROJ2}$ $(A\ B : \texttt{Prop}) :\ \mathit{AND}\ A\ B \to B.$

`Lemma` $\mathit{ORINTROL}$ $(A\ B : \texttt{Prop}) :\ A \to \mathit{OR}\ A\ B.$

`Lemma` $\mathit{ORINTROR}$ $(A\ B : \texttt{Prop}) :\ B \to \mathit{OR}\ A\ B.$

帰納的型を用いた定義と同値であることを証明できる.

`Lemma` $\mathit{AND\_and}$ $(A\ B : \texttt{Prop}) :\ \mathit{AND}\ A\ B \leftrightarrow A \land B.$

`Lemma` $\mathit{OR\_or}$ $(A\ B : \texttt{Prop}) :\ \mathit{OR}\ A\ B \leftrightarrow A \lor B.$

`Lemma` $\mathit{EX\_exists}$ $(A : \texttt{Type})\ (P : A \to \texttt{Prop}) :\ \mathit{EX}\ A\ P \leftrightarrow \exists\ a,\ P\ a.$

メモ: 書き換えをタクティク `rewrite` で行う.

`Lemma` $\mathit{EQ\_eq}$ $(A : \texttt{Type})\ (a\ a' : A) : \mathit{EQ}\ \_\ a\ a' \leftrightarrow a = a'.$

## 5 補足

`Lemma` $\mathit{subsidiaire}$ $(A : \texttt{Prop}) :\ \neg\ \neg\ (A \lor \neg\ A).$

## 参考文献


[DNR03] René David, Karim Nour, and Christophe Raffalli, *Introduction à la logique*, 2ème ed., Dunod, 2003.

[Pie16] Thomas Pietrzak, *Logique—logique propositionnelle—*, Licence Informatique, Université de Lille 1 Sciences et Technologies, 2016, `http://www.thomaspietrzak.com/download.php?f=CoursLogique0.pdf`.