

# 「計算と論理」 Software Foundations その0

五十嵐 淳

`cal18@fos.kuis.kyoto-u.ac.jp`

京都大学

October 2, 2018

# 担当教員について

- 名前: 五十嵐 淳 (いがらし あつし)
- 所属: 情報学研究科 通信情報システム専攻 計算機ソフトウェア分野
- オフィス: 総合研究7号館 224号室 (月曜日の17:00～18:00は在室予定)
- 講義についての質問・連絡:
  - ▶ メール: [cal18@fos.kuis.kyoto-u.ac.jp](mailto:cal18@fos.kuis.kyoto-u.ac.jp)
  - ▶ Twitter ハッシュタグ: #kuisca18
- 講義 WWW ページ: <http://www.fos.kuis.kyoto-u.ac.jp/~igarashi/class/cal/>

# TA

- 西川 剛史 (にしかわ たけし)
- 所属: 情報学研究科 通信情報システム専攻 計算機ソフトウェア分野
- オフィス: 総合研究7号館 227号室

# 講義内容

## シラバスより

数理論理学の基礎と、数理論理学を用いた計算機プログラムの検証について講述する。また、講義を補完するため、証明支援系(計算機上で数学的証明を行うシステム)である Coq を用いた演習を行う。

# 数理論理学

判断 (judgment) について (数理的手法で) 考える学問

判断 (命題ということもある)

≡ 真偽を考えることが可能な文

- 命題論理: 単純な判断を組み合わせて複合的な判断を構成する「接続詞」の理論
  - ▶ 「かつ」「または」「ならば」「～ではない」
- 述語論理: 量化を伴う判断の理論
  - ▶ 「任意の○○について～である」「ある○○が存在して～である」
- (様相論理: 真偽を修飾する副詞の理論)
  - ▶ 「必然的に～である」「～である可能性がある」「未来永劫～である」

# 数理論理学: 意味論と証明論

- 意味論…与えられた判断が「真である」とはどういうことかを考える
  - ▶ 真理値表 (論理関数) は命題論理の意味論のひとつ
- 証明論…与えられた命題の「証明」とは何か, 「証明できること」と「真であること」との関係 (健全性と完全性), 「証明が同じ・違う」とはどういうことかを考える
  - ▶ 様々な証明 (記述) 体系: 自然演繹, シーケント計算, ヒルベルト流公理

# 数理論理学: 意味論と証明論

- 意味論…与えられた判断が「真である」とはどういうことかを考える
  - ▶ 真理値表 (論理関数) は命題論理の意味論のひとつ
- 証明論…与えられた命題の「証明」とは何か, 「証明できること」と「真であること」との関係 (健全性と完全性), 「証明が同じ・違う」とはどういうことかを考える
  - ▶ 様々な証明 (記述) 体系: 自然演繹, シーケント計算, ヒルベルト流公理

# 計算機プログラムの検証

「計算機プログラム」の正しさの証明を与える

- 「正しさ」の基準  $\Rightarrow$  判断として書かれた仕様 (specification)
- 例:

リストを反転させる OCaml 関数 `rev` の仕様  
任意のリスト `xs` について  $\text{rev} (\text{rev } xs) = xs$

Q. これだけで仕様として十分といえるだろうか？  
(他にも `rev` が満たすべき仕様はないだろうか？)

- c.f. 単体 (unit) テスト



# 証明支援系 Coq を用いた演習

- 証明支援系: 計算機で数学をするためのソフトウェア
- 数学的対象 (数, リスト, 木などのデータ) 定義とその対象を操作するプログラムの記述言語
    - ▶ OCaml, Scheme, Haskell のような関数型プログラミング
    - ▶ ただし静的に型がついている
    - ▶ 文法は OCaml に近い (が微妙に違うので困る ;-)
  - (対象の性質を述べる) 判断の記述言語
  - 判断の証明の記述言語
  - 証明の検査機能
  - (自動証明機能)
- を使って, 色々なプログラムや, それが正しいことの証明を書く

# Coq について

- フランスの INRIA (国立の情報学研究所) で開発されている証明支援系
- OCaml (これも INRIA 製) で実装されている
- 2013 年に ACM SIGPLAN Programming Languages Software Award と ACM Software System Award を受賞
- 大規模な応用例も:
  - ▶ ソフトウェア安全性・正しさの保証
    - ★ レピダム社による OpenSSL のバグ発見
    - ★ C コンパイラの検証 (CompCert プロジェクト)
  - ▶ 数学の証明の正しさのチェック
    - ⇒ 例) 四色問題, ケプラー予想

# 講義内容

## シラバスより

数理論理学の基礎と、数理論理学を用いた計算機プログラムの検証について講述する。また、講義を補完するため、証明支援系(計算機上で数学的証明を行うシステム)である Coq を用いた演習を行う。

## 講義の(裏)テーマ

証明 = プログラム

(「Curry-Howard 同型対応」としても知られる論理と計算の関係)

# 教科書

Benjamin C. Pierce, et al. Logical Foundations. Vol.1 of The Software Foundations Series.

<https://softwarefoundations.cis.upenn.edu/>

- 注意: オンライン・テキストで本家のものは予告なく内容が変わる可能性あり
- 本講義では 2018/09 末時点でのスナップショットを使うので, 本家のものは使わないでください
- 古い版の和訳もネットに転がっている

# 入手方法

- 1 GitHub アカウントを作る
- 2 ブラウザで秘密の URL(問い合わせてください) にアクセスすると, `https://github.com/ComputationAndLogicAtKUEng/hw-XXXX` に自分だけの教科書レポジトリができる (XXXX は GitHub アカウント名)
- 3 このレポジトリを clone する

## 入手方法その2

GitHub の使い方が (今は) わからないという人のための避難手段:

- <https://www.fos.kuis.kyoto-u.ac.jp/~igarashi/class/cal/lf-201810.zip>
- ユーザ名: cal2018
- パスワード: coockadoodledoo

宿題提出に GitHub が必要なのであくまでも一時しのぎと考えてください.

# 成績評価

- 宿題 30%
  - ▶ 宿題は、教科書のファイルを編集して GitHub 経由で提出します
  - ▶ git, GitHub の使い方がわからない人は補講します
- 期末試験 70%
- 随意課題や教室での演習によりさらに加点

# 講義スケジュール

- 通常スケジュール: 10/2, 9, 16, 23, 30, 11/6, 13, 20, 27, 12/4, 12/11, 12/18, 12/25(!?), 1/8, 22



# 宿題：10/9 午前10:30まで

- テキスト Preface.v, Basics.v の予習
- Coq 環境の構築
- (提出物はないです)

# Coq 環境の構築

- ① Coq 8.8.1 のインストール
  - ▶ 先週 8.8.2 が出てしまったのですがテストできていません
- ② Emacs 使いの人は proofgeneral (と company-coq) のインストール
  - ▶ Emacs から証明支援系を使うための Emacs Lisp ソフトウェア
- ③ そうでない人は CoqIDE のインストール
  - ▶ GTK を使った Coq 専用の証明統合開発環境

# Coq 環境構築 (Ubuntu 編)

- opam (OCaml パッケージマネージャ) のインストール
  - ▶ <https://opam.ocaml.org/doc/Install.html>
- Coq 8.8.1 (と CoqIDE) のインストール
  - ▶ `opam install coq`
  - ▶ `opam install coqide`
- Proof General をインストール
  - ▶ <https://proofgeneral.github.io/> を見てください

- さらに `Company-coq` を入れると記号がカッコよく表示される
  - ▶ `https://github.com/cpitclaudel/company-coq`

# Coq 環境構築 (MacOS X 編)

- Coq 8.8.1 (と CoqIDE) のインストール
  - ▶ `https://github.com/coq/coq/releases/tag/V8.8.1` からダウンロード・インストール
  - ▶ CoqIDE もいっしょに入る
  - ▶ Ubuntu と同じく `opam` で入れてもよい
- Emacs と Proof General のインストール
  - ▶ Emacs は `homebrew` で入れるのがいいかな
  - ▶ Proof General は `https://proofgeneral.github.io/` を見てください

# Coq 環境構築 (Windows 編)

- Coq 8.8.1 (と CoqIDE) のインストール
  - ▶ `https://github.com/coq/coq/releases/tag/V8.8.1` からダウンロード・インストール
  - ▶ CoqIDE もいっしょに入る
- Emacs, Proof General のインストール
  - ▶ Windows の Emacs 環境わかりません ;-)
  - ▶ Proof General は `https://proofgeneral.github.io/` を見てください

# Coq の動作確認 (Proof General 編)

## Proof General 起動方法

Emacs で教科書の Basics.v を読み込む

「じえねらるたん」<sup>1</sup> が現れた後、ファイルの内容が表示される

- C-c C-n で、ファイルの内容が少しずつ (決まった単位で) Coq に送られ、処理済部分の背景が青くなる
- C-c C-u は逆 (undo)
  - ▶ ツールバーの左右矢印でも操作可能
- C-c RET で現在のカーソル位置まで一気に読み込まれる・巻き戻される

<sup>1</sup>とある日本人の活躍(?)で、以前はもっといかつい軍人さん (See <http://proofgeneral.inf.ed.ac.uk/gallery>) だったのがかわいくなった

# Coq の動作確認 (CoqIDE 編)

- Basics.v を ファイル → 開く, で開く
- ツールバーの下矢印で, ファイルの内容が少しずつ (決まった単位で) coq に送られ, 処理済部分の背景が緑になる
- 上矢印は逆で undo する.
  - ▶ ショートカットキーもあります



# 受講上の注意

- Twitter でのつぶやき (#kuisca18) を (講義中でも) 歓迎します
- 来週から, **スライドの紙配布はありません**
- ノートPC 持込受講を歓迎します
- 実際に証明を書いてみないと身につきません
- 書かれている記号の意味をよくよく考えましょう
  - ▶ とにかくコマンドを連打していたらいつの間にか証明ができる, というのは (じきにわからなくなる一歩前の) 危険な徴候