

「計算と論理」 Software Foundations その0

五十嵐 淳

cal23@fos.kuis.kyoto-u.ac.jp

京都大学

October 3, 2023

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 コンピュータソフトウェア分野
- オフィス: 総合研究7号館 224号室 (火曜日の16:00~18:00は在室予定)
- 講義についての質問・連絡:
 - ▶ メール: `cal23@fos.kuis.kyoto-u.ac.jp`
 - ▶ Twitter ハッシュタグ: `#kuisca123`
- 講義 WWW ページ: `http://www.fos.kuis.kyoto-u.ac.jp/~igarashi/class/cal/`

TA

- 村瀬 唯斗 (むらせ ゆいと)
- 所属: 情報学研究科 通信情報システム専攻 コンピュータソフトウェア分野 (D2)
- 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, ヒルベルト流公理系

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, ヒルベルト流公理系

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, Haskell のような関数型プログラミング
 - ▶ 静的に型がついている
 - ▶ 止まるプログラムしか書けない
 - ▶ 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 GitHub Classroom を使って宿題提出管理をするので, 本家のものは**使わないでください**
- かなり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 (PandA で伝えます) にアクセスすると, `https://github.com/ComputationAndLogicAtKUEng/hw2023-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

成績評価

- Coq 演習 35%
 - ▶ 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 60%
- 教室での演習 5%
- 随意課題を提出した場合、さらに加点する。

Coq 環境の構築

- 1 Coq のインストール
- 2 編集環境の整備 (以下どれでも)
 - 1 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
 - 2 Visual Studio Code 拡張の vscoq
 - ★ 最新版は不安定なので旧版を推奨 (後述)
 - 3 Emacs 限定だが proofgeneral も定評がある
 - 4 その他 coquille (vim), Coqoon (Eclipse) など

参考: <https://staff.aist.go.jp/reynald.affeldt/ssrcoq/install.html>

Coq 環境構築 (Ubuntu 編)

- opam は入ってますよね？
 - ▶ <https://opam.ocaml.org/doc/Install.html>
- Coq (と CoqIDE) のインストール
 - ▶ `opam install coq`
 - ▶ `opam install coqide`
 - ★ 依存する Ubuntu パッケージを apt で入れる必要あり?:
`pkg-config, libgtksourceview3.0-dev`
- Proof General をインストール
 - ▶ <https://proofgeneral.github.io/> を見よ
 - ▶ Company-coq も入れると記号がかっこよく表示される
 - ★ <https://github.com/cpitclaudel/company-coq>
- apt で入れる opam, coq は古い可能性が高いので使わない

Coq 環境構築 (MacOS X 編)

- Coq (と CoqIDE) のインストール
 - ▶ Ubuntu と同じく opam がおすすめ
- 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:
<https://proofgeneral.github.io/>

Coq 環境構築 (Windows 編)

- Coq Platform のインストーラを使おう
 - ▶ <https://github.com/coq/platform/releases/tag/2023.03.0>
 - ▶ CoqIDE もメニューに追加されるらしい
- WSL2 内で opam からインストールという手もある
- Emacs, Proof General のインストール
 - ▶ Windows の Emacs 環境わかりません ;-)

vscoq のインストール

- vscoq は 2.0 と旧版 (0.40) があるが、2.0 はリリースされたばかりで (かなり) 不安定なので旧版推奨
- Extension パネルから "VsCoq Legacy" をインストール ("VsCoq" ではないので注意！)

Coq の動作確認 (vscoq 編)

- Basics.v を開く
- Ctrl + Alt + Down で上から順に式を一つ評価す
 - ▶ 評価された部分は緑色で表示される
 - ▶ 証明の状態を表示する Proof View というタブが右側に出る
- Ctrl + Alt + Up で式を一つ undo する
- Ctrl + Alt + Right で今のカーソルまでの式を全て評価
- 他のコマンドはコマンドパレット (Ctrl + Shift + P) で "Coq" と入力すると一覧が見られる

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

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」¹ 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

¹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

受講上の注意

- 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