10　応用とまとめ

10.1　応用

本講義で述べた「計算の論理」は，型理論と論理体系の対応付けを考えるものであった．型理論 (プログラム言語における型システムに関する理論) は，現代的プログラム言語に必須の重要な基礎理論であり，その応用は多方面にわたっている．

そのうち最も重要な応用は，型を利用したプログラムの解析，検証である．型システムの健全性定理は，「プログラムが実行前に型を整合的に持てば，実行中，実行後に常に型を整合的に持つ」ということを意味していた．従って，型システムを適切に設計して，健全性定理を証明すれば，「プログラムの実行中に，良い状態を保つ (悪い状態にならない)」ことを保証することができる．

このような考え方を応用して，ある種のプログラムの検証を行うことができる．たとえば，「型エラーを起こさない」「スタックを無限に消費しない」など基本的な性質のほか，「確保した範囲のメモリ以外にアクセスしない」「機密情報が漏れない (機密情報を勝手に読みださない)」といったセキュリティ上重要な性質も含まれる．「確保した範囲のメモリ以外にアクセスしない」という性質 (メモリの安全性) は，地味な性質に見えるかもしれないが，現代のセキュリティ破りの半分以上は buffer overflow 攻撃 (確保した範囲を越えて配列にアクセスする) というものであるので，実用上は極めて重要である．この性質を保証するためには，「拡張」で述べた「依存型」が活躍する (「長さのわからない配列」ではなく「長さ n の配列」を表す型が表現できるので，メモリの範囲を型のレベルで計算することができる．)

このほかに，型システムの応用は広く，コンパイラの最適化に利用したり，証明からのプログラム抽出に応用することができる．

10.2　まとめ

この講義では，構文的な (機械的な) 操作に着目し，演習を通じて理解を深めてきた．また，これらの背景となる技術・理論について述べてきた．具体的な定理や結果は別として，これらの過程の基本的な考え方，立場を以下にまとめる．

- 導出ゲーム
  導出ゲームごとに，具体的な規則は異なるが (しかも，同じ目的のゲームであってもゲーム設計者ごとに異なるが)，ゲーム (導出規則) を定めれば，「その有限回の操作によって帰納的に生成されるもの」として導出が定まる，ということは共通している．
- 「導出」という形式的操作により，いろいろな概念を形式的に定めることができること
  プログラムの構文，論理における証明などは，全て，「導出」によって形式的に定めることができる; さらに，プログラムの計算，論理における証明の計算，など，もともと「意味論」に属する概念も，「導出」によって形式的に定めることができる．これにより，プログラムや論理に対する操作を計算機上に実装することが可能になる．(プログラムの型推論・計算・検証，定理の自動証明など)
- 型システムと論理体系の関係
  型付きラムダ計算の体系と (直観主義) 論理の体系は本質的に同じである．この対応関係を利用して，各種の拡張など，片方での知見を他方へ移転することができる．プログラム言語を理解することは，そ

の論理を理解することと等価である.