1 はじめに

通常、「計算」と「論理」は、動的/静的として対極にあるもの (相補的なもの) と考えられている。

計算 (Computation)	動的 (dynamic)、いかに変化するか
論理 (Logic)	静的 (static), いかに変わらない性質を記述するか

表 1 「計算」と「論理」に関する普通のイメージ

このような理解でも第一近似としては間違いとは言えないが、実は、計算にも静的な側面があり、論理にも動的な側面がある。すなわち、静的/動的ということと計算/論理ということは独立である。

	動的な側面	静的な側面
計算	普通の計算	?
論理	?	普通の論理

表 2 「計算」と「論理」に関する改善したイメージ

「普通の計算」とは、通常のプログラミング言語で記述されるようなプログラムの世界である。「普通の論理」とは、通常我々が目にする論理体系で記述される。たとえば、ソフトウェアの正しさを保証するための Hoare 論理[*1]などの体系はこの表でいう「普通の論理」に該当する。普通の論理はあくまで静的であり、プログラムの実行前に正しさを保証するため等に用いられる。

この講義では、この表の?のところにも登場人物が存在することを示す。すなわち、?のところを埋めることによって、実は、計算体系と論理体系は本質的に同じものであることを理解することを目的とする。

プログラム言語論は、プログラム言語に関する科学である。その中心は、特定のプログラム言語に依存した個別の議論ではなく、プログラム言語によらない一般的な仕組みについての理論の展開である。また、単に個別のプログラム言語の機能等を比較するのではなく、多数のプログラム言語に共通する機能の本質について考察する学問である。

プログラム言語によらない一般的仕組み、共通の機能について議論するためには、個別の機能をそぎ落としたシンプルなプログラム言語を用意するのがよい。ここでは、対象を明確にするための非常に単純なプログラミング言語として、型付きラムダ計算 (typed lambda calculus) の体系を取りあげて、それについて講義を行う。型付きラムダ計算は、「計算」の中心をなす機構を抽象したラムダ計算に、「型」の概念を導入したものである。現代的なプログラム言語の多くは洗練された型の概念を持つため、型付きラムダ計算の拡張ととらえることができる。したがって、本講義で述べる理論や技法を、一般のプログラム言語に拡張することが可能である。

一方、本講義で扱う論理は、普通の論理 (古典論理という) だけではなく、直観主義論理という (普通の論理とは若干異なる) 論理も扱う。直観主義論理は、構成的論理あるいはプログラムの論理とも呼ばれ、計算との

[*1] Hoare 論理は、コンピュータ科学界のノーベル賞と言われるチューリング賞受賞者である C. A. R. Hoare が創始した、プログラム検証のための論理である。詳細は、「プログラム理論」の授業か、文献 (たとえば、林晋著「プログラム検証論」共立出版) を参照すること。

相性の良い論理である。普通の論理は、直観主義論理に対する拡張と考えることができる。

本講義では、型付きラムダ計算の静的および動的な側面、直観主義論理の静的および動的な側面について解説し、それらが本質的に同じものであることを理解する。また、種々の拡張についても触れ、現代的プログラム言語や「普通の論理」をどのように扱うことが可能かについても概観する。

1.1 参考書

本講義の内容については講義ノートを参考にしてほしい。講義内容よりさらに進んだ事を勉強するためには、下記書籍が参考になる (ここでは日本語の書籍のみをあげた)。

- 型のないラムダ計算・・・高橋正子「計算論」(近代科学社)
- 型付きラムダ計算・・・大堀淳「プログラム言語の基礎理論」 (共立出版), 五十嵐淳「プログラミング言語の基礎概念」 (サイエンス社)
- 記号論理学・・・小野寛晰「情報科学における論理」(日本評論社)、萩谷昌己「ソフトウェア科学のための論理学」(岩波講座)
- 形式化・・・佐藤雅彦、桜井貴文「プログラムの基礎理論」(岩波講座)