CAS 734 Winter 2006

02 Review of Mathematical Logic

William M. Farmer

Department of Computing and Software
McMaster University

21 September 2006

Outline

- What is mathematical logic?
- Syntax and semantics of logical languages
- Proof systems
- Axiomatic theories
- Example: simple type theory

What is Mathematical Logic?

- Study of the principles underlying mathematical reasoning.
  - Central idea: logical consequence.
- Branch of mathematics.
- Makes explicit several fundamental distinctions:
  - Syntax vs. semantics.
  - Language vs. metalanguage.
  - Theory vs. model.
  - Truth vs. proof.
- Principal tools: formal systems called logics.

Syntax vs. Semantics

- The syntax of a language is concerned with how the expressions of the language are constructed.
  - For example, "the numeral 144 has three digits" is a statement about syntax.
- The semantics of a language is concerned with what the expressions of the language mean.
  - For example, "the number 144 is a perfect square" is a statement about semantics.
- This distinction is crucial in mathematics and computing.
  - Confusion between syntax and semantics is the source of many errors.
- Logic carefully disentangles the roles of syntax and semantics in reasoning.

What is a Logic?

- Informally, a logic is a system of reasoning.
- Formally, a logic is a family of formal languages with:
  1. A common syntax.
  2. A common semantics.
  3. A notion of logical consequence.
- A logic may include a proof system for proving that a given formula is a logical consequence of a given set of formulas.
- Examples:
  - Propositional logic.
  - First-order logic.
  - Simple type theory (higher-order logic).
  - Zermelo-Fraenkel (ZF) set theory.

Language Syntax

- A language defines a collection of expressions formed from:
  - Variables.
  - Constants (nonlogical constants).
  - Constructors (logical constants).
- Two kinds of expressions:
  - Terms: Denote objects or values.
  - Formulas: Make assertions about objects or values.
- Some languages have constructors that bind variables (e.g., $\forall$, $\exists$, λ, I, ϵ, $\{ \mid \}$).

Language Semantics

- A model M for a language L is a pair (D, V) where:
  1. D is a set of values called the domain that includes the truth values T and F.
  2. V is a function from the expressions of L to D called the valuation function.
- M satisfies a formula A of L, written $M \models A$, if $V(A) = \mathrm{T}$.
- M satisfies a set Σ of formulas of L, written $M \models \Sigma$, if M satisfies each $A \in \Sigma$.
- A is a semantic consequence of Σ, written $\Sigma \models A$, if every model for L that satisfies Σ also satisfies A.
- A is valid, written $\models A$, if every model for L satisfies A.
- Σ is satisfiable if there exists some model for L that satisfies Σ.

Language vs. Metalanguage

- A language is for talking about a certain subject.
- A metalanguage for a language L is a language for talking about L itself.
- A natural language, such as English, usually serves as its own metalanguage.
  - As a result, the distinction is not explicit in English.
- A formal language, such as a logical or programming language, usually is not expressive enough to serve as its own metalanguage.
  - A metalanguage of a formal language may be a formal language, but usually it is only informal.

Proof

- Mathematical proof is an essential component of the mathematics process which is unique to mathematics.
- It is a method of communication, certification, and discovery.
- An informal proof is a convincing argument that a statement about a mathematical model is true.
- A formal proof is a logical deduction from a set of premises to a conclusion.
  - Can be mechanically checked.
- A formal proof can be presented in two ways:
  - As a description of the actual deduction.
  - As a prescription for creating the deduction.

Hilbert-Style Proof Systems

- A Hilbert-style proof system **P** for a language L consists of:
  1. A set of formulas of L called logical axioms.
  2. A set of rules of inference.
- A proof of A from Σ in **P** is a finite sequence $B_1, \ldots, B_n$ of formulas of L with $B_n = A$ such that each B_i is either a logical axiom, a member of Σ, or follows from earlier B_j by one of the rules of inference.
- A is syntactic consequence of Σ in **P**, written $\Sigma \vdash_{\mathbf{P}} A$, if there is a proof of A from Σ in **P**.
- A is a theorem in **P**, written $\vdash_{\mathbf{P}} A$, if there is a proof of A from $\emptyset$ in **P**.
- Σ is consistent in **P** if not every formula is a syntactic consequence of Σ in **P**.

Kinds of Proof Systems

- Hilbert style.
- Symmetric sequent (Gentzen).
- Asymmetric sequent.
- Natural deduction (Gentzen, Quine, Fitch, Berry).
- Semantic tableaux (Beth, Hintikka).
- Resolution (J. Robinson).

Soundness and Completeness

- Let **P** be a proof system for a language L.
- **P** is sound if

$$\Sigma \vdash_{\mathbf{P}} A \text{ implies } \Sigma \models A.$$

- **P** is complete if

$$\Sigma \models A \text{ implies } \Sigma \vdash_{\mathbf{P}} A.$$

- A unsound proof system is not usually very useful, while a sound but incomplete proof system can be quite useful.

Axiomatic Theories

- A axiomatic theory is a pair $T = (L, \Gamma)$ where:
  1. L is a language (the language of T).
  2. Γ is a set of formulas of L (the axioms of T).
- M is a model of T, written $M \models T$, if $M \models \Gamma$.
- A is valid in T, written $T \models A$, if $\Gamma \models A$.
- A is a theorem of T in $\mathbf{P}$, written $T \vdash_{\mathbf{P}} A$, if $\Gamma \vdash_{\mathbf{P}} A$.
- T is satisfiable if Γ is satisfiable.
- T is consistent in $\mathbf{P}$ if Γ is consistent in $\mathbf{P}$.

Theory vs. Model

- A model for a language is a concrete mathematical model.
- A axiomatic theory is an abstract mathematical model.
- An axiomatic theory can be viewed as a specification of its models.
  - A theory is to a model as a specification is to an implementation.
- Axiomatic theories fall into two categories:
  - Those that are intended to describe a single model (e.g., a theory of natural number arithmetic).
  - Those that are described a collection of models (e.g., a theory of monoids).

Truth vs. Proof

Semantics	**Syntax**
truth	proof
semantic consequence	syntactic consequence
A is valid $\models A$	A is a theorem in $\mathbf{P}$ $\vdash_{\mathbf{P}} A$
A is valid in T $T \models A$	A is a theorem of T in $\mathbf{P}$ $T \vdash_{\mathbf{P}} A$
T is satisfiable	T is consistent in $\mathbf{P}$

- Semantic consequence and syntactic consequence are different forms of logical consequence.
- The semantic and syntactic notions are equivalent in the most common logics:
  - Propositional logic.
  - First-order logic (Gödel, 1930).
  - Simple type theory (Henkin, 1950).

Mathematical Problems: Fundamental Form

- Most mathematical problems can be expressed as statements of the form

  $$T \models A$$

  where T is an axiomatic theory and A is a formula.
- There are three basic ways of deciding whether or not $T \models A$:
  1. Model checking:
     Show that $M \models A$ for each model M of T.
  2. Proof:
     Show $T \vdash_{\mathbf{P}} A$ for some sound proof system $\mathbf{P}$.
  3. Counterexample:
     Show $M \models \neg A$ for some model M of T.

What is Simple Type Theory?

- A simple, elegant, highly expressive, and practical logic.
  - Familiar to some computer scientists but not to many mathematicians, engineers, and other scientists.
- Most popular form of type theory.
  - Types are used to classify expressions by value and control the formation of expressions.
  - Classical: nonconstructive, 2-valued.
  - Higher order: quantification over functions.
  - Can be viewed as a "function theory".
- Natural extension of first-order logic.
  - Based on the same principles as first-order logic.
  - Includes nth-order logic for all $n \geq 1$.
- We will present a simple, pure form of simple type theory called STT.

History of Simple Type Theory

1908	Russell
	Ramified theory of types.
1910	Russell, Whitehead
	Principia Mathematica.
1920s	Chwistek, Ramsey
	Simple theory of types (simple type theory).
1920–30s	Carnap, Gödel, Tarski, Quine
	Detailed formulations of simple type theory.
1940	Church
	Simple type theory with lambda-notation.
1950	Henkin
	General models and completeness theorem.
1963	Henkin, Andrews
	Concise formulation based on equality.
1980-90s	HOL, IMPS, Isabelle, ProofPower, PVS, TPS
	Higher-order theorem proving systems.

References: Key Historical Publications

1. B. Russell, "Mathematical Logic as Based on the Theory of types", *American Journal of Mathematics*, 30:222–262, 1908.
2. A. N. Whitehead and B. Russell, *Principia Mathematica*, Cambridge University Press, 1910–13.
3. A. Church, "A Formulation of the Simple Theory of Types", *Journal of Symbolic Logic*, 5:56–68, 1940.
4. L. Henkin, "Completeness in the Theory of Types", *Journal of Symbolic Logic*, 15:81–91, 1950.
5. L. Henkin, "A Theory of Propositional Types", *Fundamenta Mathematicae*, 52:323–344, 1963.
6. P. B. Andrews, "A Reduction of the Axioms for the Theory of Propositional Types", *Fundamenta Mathematicae* 52:345–350, 1963.

References: Introductions

1. P. B. Andrews, *An Introduction to Mathematical Logic and Type Theory: To Truth through Proof*, Second Edition, Kluwer, 2002.
2. W. M. Farmer, "A basic extended simple type theory", SQRL Report No. 14, 12 pp., McMaster University, 2003 (revised 2004).
3. W. M. Farmer, "The Seven Virtues of Simple Type Theory", SQRL Report No. 18, 30 pp., McMaster University, 2003 (revised 2006).

Syntax of STT: Types

- A type of STT is defined by the following rules:

$$\textbf{T1}\ \frac{}{\textbf{type}[\iota]}\ \text{(Type of individuals)}$$

$$\textbf{T2}\ \frac{}{\textbf{type}[*]}\ \text{(Type of truth values)}$$

$$\textbf{T3}\ \frac{\textbf{type}[\alpha],\ \textbf{type}[\beta]}{\textbf{type}[(\alpha \to \beta)]}\ \text{(Function type)}$$

- Let $\mathcal{T}$ denote the set of types of STT.

Syntax of STT: Symbols

- The logical symbols of STT are:
  - Function application: @ (hidden).
  - Function abstraction: λ.
  - Equality: $=$.
  - Definite description: I (capital iota).
  - An infinite set $\mathcal{V}$ of symbols called variables.
- A language of STT is a pair $L = (\mathcal{C}, \tau)$ where:
  - $\mathcal{C}$ is a set of symbols called constants.
  - $\tau : \mathcal{C} \to \mathcal{T}$ is a total function (which assigns a type to each constant).

Syntax of STT: Expressions (1)

An expression E of type α of a STT language $L = (\mathcal{C}, \tau)$ is defined by the following rules:

E1 $$\frac{x \in \mathcal{V},\ \mathbf{type}[\alpha]}{\mathbf{expr}_L[(x : \alpha), \alpha]}$$ (Variable)

E2 $$\frac{c \in \mathcal{C}}{\mathbf{expr}_L[c, \tau(c)]}$$ (Constant)

E3 $$\frac{\mathbf{expr}_L[A, \alpha],\ \mathbf{expr}_L[F, (\alpha \to \beta)]}{\mathbf{expr}_L[F(A), \beta]}$$ (Application)

E4 $$\frac{x \in \mathcal{V},\ \mathbf{type}[\alpha],\ \mathbf{expr}_L[B, \beta]}{\mathbf{expr}_L[(\lambda\, x : \alpha\ .\ B), (\alpha \to \beta)]}$$ (Abstraction)

Syntax of STT: Expressions (2)

E5 $$\frac{\mathbf{expr}_L[E_1, \alpha],\ \mathbf{expr}_L[E_2, \alpha]}{\mathbf{expr}_L[(E_1 = E_2), *]}$$ (Equality)

E6 $$\frac{x \in \mathcal{V},\ \mathbf{type}[\alpha],\ \mathbf{expr}_L[A, *]}{\mathbf{expr}_L[(\mathrm{I}\, x : \alpha\ .\ A), \alpha]}$$ (Definite description)

Syntax of STT: Conventions

- E_α denotes an expression E of type α.
- Parentheses and the types of variables may be dropped when meaning is not lost.

Semantics of STT: Standard Models

- A standard model for a language $L = (\mathcal{C}, \tau)$ of STT is a triple $M = (\mathcal{D}, I, e)$ where:
  - $\mathcal{D} = \{D_\alpha : \alpha \in \mathcal{T}\}$ is a set of nonempty domains (sets).
  - $D_* = \{\mathrm{T}, \mathrm{F}\}$, the domain of truth values.
  - $D_{\alpha\to\beta}$ is the set of all functions from D_α to D_β.
  - I maps each $c \in \mathcal{C}$ to an element of $D_{\tau(c)}$.
  - e maps each $\alpha \in \mathcal{T}$ to a member of D_α.
- A variable assignment into M is a function that maps each expression $(x : \alpha)$ to an element of D_α.
- Given a variable assignment φ into M, an expression $(x : \alpha)$, and $d \in D_\alpha$, let $\varphi[(x : \alpha) \mapsto d]$ be the variable assignment φ' into M such that $\varphi'((x : \alpha)) = d$ and $\varphi'(v) = \varphi(v)$ for all $v \neq (x : \alpha)$.

Semantics of STT: Valuation Function

The valuation function for a standard model $M = (\mathcal{D}, I, e)$ for a language $L = (\mathcal{C}, \tau)$ of STT is the binary function V^M that satisfies the following conditions for all variable assignments φ into M and all expressions E of L:

1. Let E is $(x : \alpha)$. Then $V^M_\varphi(E) = \varphi((x : \alpha))$.
2. Let $E \in \mathcal{C}$. Then $V^M_\varphi(E) = I(E)$.
3. Let E be $F(A)$. Then $V^M_\varphi(E) = V^M_\varphi(F)(V^M_\varphi(A))$.
4. Let E be $(\lambda\, x : \alpha\ .\ B_\beta)$. Then $V^M_\varphi(E)$ is the $f : D_\alpha \longrightarrow D_\beta$ such that, for each $d \in D_\alpha$, $f(d) = V^M_{\varphi[(x:\alpha)\mapsto d]}(B_\beta)$.
5. Let E be $(E_1 = E_2)$. If $V^M_\varphi(E_1) = V^M_\varphi(E_2)$, then $V^M_\varphi(E) = \mathrm{T}$; otherwise $V^M_\varphi(E) = \mathrm{F}$.
6. Let E be $(\mathrm{I}\, x : \alpha\ .\ A)$. If there is a unique $d \in D_\alpha$ such that $V^M_{\varphi[(x:\alpha)\mapsto d]}(A) = \mathrm{T}$, then $V^M_\varphi(E) = d$; otherwise $V^M_\varphi(E) = e(\alpha)$.

Abbreviations

T	means	$(\lambda\, x : * \,.\, x) = (\lambda\, x : * \,.\, x)$.
F	means	$(\lambda\, x : * \,.\, \mathsf{T}) = (\lambda\, x : * \,.\, x)$.
$(\neg A_*)$	means	$A_* = \mathsf{F}$.
$(A_\alpha \neq B_\alpha)$	means	$\neg(A_\alpha = B_\alpha)$.
$(A_* \wedge B_*)$	means	$(\lambda\, f : * \to (* \to *) \,.\, f(\mathsf{T})(\mathsf{T})) =$ $(\lambda\, f : * \to (* \to *) \,.\, f(A_*)(B_*))$.
$(A_* \vee B_*)$	means	$\neg(\neg A_* \wedge \neg B_*)$.
$(A_* \Rightarrow B_*)$	means	$\neg A_* \vee B_*$.
$(A_* \Leftrightarrow B_*)$	means	$A_* = B_*$.
$(\forall\, x : \alpha \,.\, A_*)$	means	$(\lambda\, x : \alpha \,.\, A_*) = (\lambda\, x : \alpha \,.\, \mathsf{T})$.
$(\exists\, x : \alpha \,.\, A_*)$	means	$\neg(\forall\, x : \alpha \,.\, \neg A_*)$.
$\bot_\alpha$	means	$\mathrm{I}\, x : \alpha \,.\, x \neq x$.
$\mathsf{if}(A_*, B_\alpha, C_\alpha)$	means	$\mathrm{I}\, x : \alpha \,.\, (A_* \Rightarrow x = B_\alpha) \wedge (\neg A_* \Rightarrow x = C_\alpha)$ where x does not occur in A_*, B_α, or C_α.

Expressivity

- **Theorem.** *There is a faithful interpretation of nth-order logic in* STT *for all* $n \geq 1$.
- Most mathematical notions can be directly and naturally expressed in STT.
- Examples:

$$\text{equiv-rel} = \lambda\, p : (\iota \to (\iota \to *))\ .$$
$$\forall\, x : \iota\ .\ p(x)(x) \wedge$$
$$\forall\, x, y : \iota\ .\ p(x)(y) \Rightarrow p(y)(x) \wedge$$
$$\forall\, x, y, z : \iota\ .\ (p(x)(y) \wedge p(y)(z)) \Rightarrow p(x)(z)$$

$$\text{compose} = \lambda\, f : (\iota \to \iota)\ .\ \lambda\, g : (\iota \to \iota)\ .\ \lambda\, x : \iota\ .\ f(g(x))$$

$$\text{inv-image} = \lambda\, f : (\iota \to \iota)\ .\ \lambda\, s : (\iota \to *)\ .$$
$$\mathrm{I}\, s' : (\iota \to *)\ .\ \forall\, x : \iota\ .\ s'(x) \Leftrightarrow s(f(x))$$

Peano Arithmetic

- Let **PA** $= (L, \Gamma)$ be the theory of STT such that:

  $L = (\{0, S\}, \tau)$ where $\tau(0) = \iota$ and $\tau(S) = \iota \to \iota$.
  Γ is the set of the following three formulas:

  0 has no predecessor : $\forall\, x : \iota \,.\, 0 \neq S(x)$.

  S is injective : $\forall\, x, y : \iota \,.\, S(x) = S(y) \Rightarrow x = y$.

  Induction principle :

  $\forall\, P : \iota \to * \,.$

  $P(0) \wedge (\forall\, x : \iota \,.\, P(x) \Rightarrow P(S(x))) \Rightarrow$

  $\forall\, x : \iota \,.\, P(x)$.

- **Theorem** (Dedekind, 1888). **PA** *has (up to isomorphism) a unique standard model* $M = (\mathcal{D}, I, e)$ *where* $D_\iota = \{0, 1, 2, \ldots\}$.

Incompleteness of STT

Theorem. *There is no sound and complete proof system for* STT.

Proof. Suppose **P** is a sound and complete proof system for STT. By the soundness of **P** and Gödel's Incompleteness Theorem, there is a sentence A such that (1) $M \models A$, where M is the unique standard model for **PA** (up to isomorphism), and (2) $\mathbf{PA} \nvdash_{\mathbf{P}} A$. By the completeness of **P**, (2) implies $\mathbf{PA} \not\models A$ and hence $M \not\models A$ since M is the only standard model of PA, which contradicts (1). □

A Proof System for STT (1)

- **Axioms**:

  A1 (Truth Values)

  $\forall f : * \to * \,.\, (f(\mathsf{T}_*) \wedge f(\mathsf{F}_*)) \Leftrightarrow (\forall x : * \,.\, f(x))$.

  A2 (Leibniz' Law)

  $\forall x, y : \alpha \,.\, (x = y) \Rightarrow (\forall p : \alpha \to * \,.\, p(x) \Leftrightarrow p(y))$.

  A3 (Extensionality)

  $\forall f, g : \alpha \to \beta \,.\, (f = g) = (\forall x : \alpha \,.\, f(x) = g(x))$.

  A4 (Beta-Reduction)

  $(\lambda x : \alpha \,.\, B_\beta)(A_\alpha) = B_\beta[x \mapsto A_\alpha]$

  provided A_α is free for x in B_β.

  A5 (Proper Definite Description)

  $(\exists ! x : \alpha \,.\, A) \Rightarrow A[(x : \alpha) \mapsto (\mathrm{I} x : \alpha \,.\, A)]$.

  A6 (Improper Definite Description)

  $\neg(\exists ! x : \alpha \,.\, A) \Rightarrow (\mathrm{I} x : \alpha \,.\, A) = \bot_\alpha$.

A Proof System for STT (2)

- **Rule of inference**:

  R (Equality Substitution)

  From $A_\alpha = B_\alpha$ and C_* infer the result of replacing one occurrence of A_α in C_* by an occurrence of B_α.

- Call this proof system **A**.
  - Due to Andrews, 1963.
- **Theorem** (Jensen, 1969). **A** *plus an axiom of infinity is equiconsistent with bounded Zermelo set theory.*

General Models

- A general structure for a language $L = (\mathcal{C}, \tau)$ of STT is a triple $M = (\mathcal{D}, I, e)$ where:
  - $\mathcal{D} = \{D_\alpha : \alpha \in \mathcal{T}\}$ is a set of nonempty domains (sets).
  - $D_* = \{\mathrm{T}, \mathrm{F}\}$, the domain of truth values.
  - $D_{\alpha\to\beta}$ is some set of functions from D_α to D_β.
  - I maps each $c \in \mathcal{C}$ to an element of $D_{\tau(c)}$.
  - e maps each $\alpha \in \mathcal{T}$ to a member of D_α.
- M is a general model for L if there is a binary function V^M that satisfies the same conditions as the valuation function for a standard model.
- A general model is a nonstandard model if it is not a standard model.

Completeness of STT

Theorem (Henkin, 1950). *Church's type theory (and hence* STT*) is complete with respect to general models.*

Corollary. *Church's type theory (and hence* STT*) is compact with respect to general models.*

Theorem (Andrews, 1963). **A** *is a sound and complete proof system for* STT *with respect to general models.*

Ways of Making STT More Practical

- Make the logic many-sorted by allowing several types of individuals, e.g., $\iota_1, \ldots, \iota_n$.
- Add machinery for basic mathematical objects such as sets, tuples, and lists.
- Admit polymorphic operators like $(\lambda\, x : t\ .\ x)$ by introducing type variables.
- Enrich the type system of STT with new machinery such as subtypes, dependent types, and user-defined type constructors.
- Modify the semantics of STT to include partial functions and undefined expressions.

Theorem Proving Systems Based On Variants of Simple Type Theory

- HOL (Gordon).
- IMPS (Farmer, Guttman, Thayer).
- Isabelle/HOL (Paulson).
- ProofPower (Lemma 1).
- PVS (Owre, Rushby, Shankar).
- TPS (Andrews).