CAS 701 Fall 2004

09 Simple Type Theory

Instructor: W. M. Farmer

Revised: 2 December 2004

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 $n \geq 1$.

Who needs Simple Type Theory?

An understanding of simple type would be beneficial to anyone who needs to work with or apply mathematical logic. This is particularly true for:

- Engineers who need to write (and read) precise specifications.

- Computer scientists who employ functional programming languages such as Lisp, ML, and Haskell.

- Software engineers who use higher-order theorem proving systems to model and analyze software systems.

- Mathematics students who are studying the foundations of mathematics or model theory.

Purpose of this Presentation

- Present a pure form of simple type theory named STT.
- Show the virtues of simple type theory using STT.
- Argue that simple type theory is an attractive alternative to first-order logic for practical-minded scientists, engineers, and mathematicians.

History

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.

Syntax of STT: Types

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

T1 $\dfrac{}{\textbf{type}[\iota]}$ **(Type of individuals)**

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

T3 $\dfrac{\textbf{type}[\alpha],\ \textbf{type}[\beta]}{\textbf{type}[(\alpha \rightarrow \beta)]}$ **(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**.
  - $\tau : \mathcal{C} \rightarrow \mathcal{T}$ is a total function.

Syntax of STT: Expressions

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

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

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

E3 $$\frac{\textbf{expr}_L[A, \alpha],\ \textbf{expr}_L[F, (\alpha \rightarrow \beta)]}{\textbf{expr}_L[F(A), \beta]}$$ **(Application)**

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

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

E6 $$\frac{x \in \mathcal{V},\ \textbf{type}[\alpha],\ \textbf{expr}_L[A, *]}{\textbf{expr}_L[(\mathrm{I}\, x : \alpha \,.\, A), \alpha]}$$ **(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).
  - $D_* = \{\mathsf{t}, \mathsf{f}\}$, the domain of truth values.
  - $D_{\alpha \rightarrow \beta}$ 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 $f : D_\alpha \rightarrow D_\beta$ 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 $V^M_\varphi(E) =$ t; otherwise $V^M_\varphi(E) =$ f.

6. Let E be $(\mathrm{I}\, x : \alpha \,.\, A)$. If there is a unique $d \in D_\alpha$ such that $V^M_{\varphi[(x:\alpha)\mapsto d]}(A) =$ 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	$(\lambda\, f : * \rightarrow (* \rightarrow *) \,.\, f(\mathsf{T})(\mathsf{T})) =$
		$(\lambda\, f : * \rightarrow (* \rightarrow *) \,.\, f(A_*)(B_*))$.
$(A_* \vee B_*)$	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	$\mathrm{I}\, x : \alpha \,.\, x \neq x$.
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:

$$\begin{array}{l} \textsf{equiv-rel} = \lambda\, p : (\iota \rightarrow (\iota \rightarrow *))\ . \\ \quad \forall\, x : \iota\ .\ p(x)(x) \wedge \\ \quad \forall\, x, y : \iota\ .\ p(x)(y) \Rightarrow p(y)(x) \wedge \\ \quad \forall\, x, y, z : \iota\ .\ (p(x)(y) \wedge p(y)(z)) \Rightarrow p(x)(z) \end{array}$$

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

$$\begin{array}{l} \textsf{inv-image} = \lambda\, f : (\iota \rightarrow \iota)\ .\ \lambda\, s : (\iota \rightarrow *)\ . \\ \quad \mathrm{I}\, s' : (\iota \rightarrow *)\ .\ \forall\, x : \iota\ .\ s'(x) \Leftrightarrow s(f(x)) \end{array}$$

Peano Arithmetic

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

  $L = (\{0, S\}, \tau)$ where $\tau(0) = \iota$ and $\tau(S) = \iota \rightarrow \iota$.

  Γ is the set of the following three formulas:

  1. **0 has no predecessor**: $\forall\, x : \iota \,.\, 0 \neq S(x)$.

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

  3. **Induction principle**:
     $\forall\, P : \iota \rightarrow * \,.$
     $\quad P(0) \wedge (\forall\, x : \iota \,.\, P(x) \Rightarrow P(S(x))) \Rightarrow \forall\, x : \iota \,.\, P(x)$.

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

Complete Ordered Field (1)

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

  $L = (\{+, 0, -, \cdot, 1, {}^{-1}, \mathsf{pos}, <, \leq, \mathsf{ub}, \mathsf{lub}\}, \tau)$ where

Constant c	**Type** $\tau(c)$
0,1	ι
$-, {}^{-1}$	$\iota \rightarrow \iota$
pos	$\iota \rightarrow *$
$+, \cdot$	$\iota \rightarrow (\iota \rightarrow \iota)$
$<, \leq$	$\iota \rightarrow (\iota \rightarrow *)$
ub, lub	$(\iota \rightarrow *) \rightarrow (\iota \rightarrow *)$

Complete Ordered Field (2)

Γ is the set of the following eighteen formulas:

1. $\forall\, x, y, z : \iota \,.\, (x + y) + z = x + (y + z)$.
2. $\forall\, x, y : \iota \,.\, x + y = y + x$.
3. $\forall\, x : \iota \,.\, x + 0 = x$.
4. $\forall\, x : \iota \,.\, x + (-x) = 0$.
5. $\forall\, x, y, z : \iota \,.\, (x \cdot y) \cdot z = x \cdot (y \cdot z)$.
6. $\forall\, x, y : \iota \,.\, x \cdot y = y \cdot x$.
7. $\forall\, x : \iota \,.\, x \cdot 1 = x$.
8. $\forall\, x : \iota \,.\, x \neq 0 \Rightarrow x \cdot x^{-1} = 1$.
9. $0 \neq 1$.
10. $\forall\, x, y, z : \iota \,.\, x \cdot (y + z) = (x \cdot y) + (y \cdot z)$.
11. $\forall\, x : \iota \,.\, (x = 0 \wedge \neg \mathsf{pos}(x) \wedge \neg \mathsf{pos}(-x)) \vee$
    $(x \neq 0 \wedge \mathsf{pos}(x) \wedge \neg \mathsf{pos}(-x)) \vee$
    $(x \neq 0 \wedge \neg \mathsf{pos}(x) \wedge \mathsf{pos}(-x))$.

Complete Ordered Field (3)

12. $\forall\, x, y : \iota\, .\, (\mathsf{pos}(x) \wedge \mathsf{pos}(y)) \Rightarrow \mathsf{pos}(x + y)$.
13. $\forall\, x, y : \iota\, .\, (\mathsf{pos}(x) \wedge \mathsf{pos}(y)) \Rightarrow \mathsf{pos}(x \cdot y)$.
14. $\forall\, x, y : \iota\, .\, x < y \Leftrightarrow \mathsf{pos}(y - x)$.
15. $\forall\, x, y : \iota\, .\, x \leq y \Leftrightarrow (x < y \vee x = y)$.
16. $\forall\, s : \iota \to *\, .\, \forall\, x : \iota\, .\, \mathsf{ub}(s)(x) = \forall\, y : \iota\, .\, s(y) \Rightarrow y \leq x$.
17. $\forall\, s : \iota \to *\, .\, \forall\, x : \iota\, .$
    $\mathsf{lub}(s)(x) = (\mathsf{ub}(s)(x) \wedge (\forall\, y : \iota\, .\, \mathsf{ub}(s)(y) \Rightarrow x \leq y))$.
18. $\forall\, s : \iota \to *\, .$
    $\exists\, x : \iota\, .\, s(x) \wedge \exists\, x : \iota\, .\, \mathsf{ub}(s)(x) \Rightarrow \exists\, x : \iota\, .\, \mathsf{lub}(s)(x)$.

- **Theorem. COF** has (up to isomorphism) a unique standard model $M = (\mathcal{D}, I, e)$ where $D_\iota = \mathbf{R}$, the set of real numbers.

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) **PA** $\not\vdash_{\mathbf{P}} A$. By the completeness of **P**, (2) implies **PA** $\not\models A$ and hence $M \not\models A$ since M is the only standard model of PA, which contradicts (1). $\Box$

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.

- Let 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).
  - $D_* = \{\mathsf{t}, \mathsf{f}\}$, the domain of truth values.
  - $D_{\alpha \rightarrow \beta}$ 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). STT is complete with respect to general models.

Corollary. 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 STT

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

Conclusion

- Simple type theory is a logic that is effective for practice as well as theory—unlike first-order logic.
  - More expressive and more convenient.
  - Closer to mathematical practice.
  - Includes the full machinery of first-order logic.

- We recommend that simple type theory be incorporated into logic courses offered by mathematics departments by replacing the two-logic sequence with a three-logic sequence—propositional logic, first-order logic, simple type theory.
  - Not much harder to learn than first-order logic.
  - Integrates the study of predicate logic, functions, and types.

The Seven Virtues

Virtue 1: STT has a simple and highly uniform syntax.

Virtue 2: The semantics of STT is based on a small collection of well-established ideas.

Virtue 3: STT is a highly expressive logic.

Virtue 4: STT admits categorical theories of infinite structures.

Virtue 5: There is a proof system for STT that is simple, elegant, and powerful.

Virtue 6: Henkin's general models semantics enables the techniques of first-order model theory to be applied to STT and illuminates the distinction between standard and nonstandard models.

Virtue 7: There are practical variants of STT that can be effectively implemented.

References

B. Russell, "Mathematical logic as based on the theory of types", *American Journal of Mathematics* 30:222–262, 1908.

A. N. Whitehead and B. Russell, *Principia Mathematica*, Cambridge University Press, 1910.

A. Church, "A formulation of the simple theory of types", *Journal of Symbolic Logic* 5: 56–68, 1940.

L. Henkin, "Completeness in the theory of types", *Journal of Symbolic Logic* 15:81–91, 1950.

L. Henkin, "A theory of propositional types", *Fundamenta Mathematicae* 52:323–344, 1963.

P. B. Andrews, "A reduction of the axioms for the theory of propositional types", *Fundamenta Mathematicae* 52:345–350, 1963.

P. B. Andrews, *An Introduction to Mathematical Logic and Type Theory: To Truth through Proof*, Second Edition, Kluwer, 2002.