barkhausen institut

Foundations of Certified Programming Language and Compiler Design

Dr.-Ing. Sebastian Ertel

Composable Operating Systems Group, Barkhausen Institute

Outline

Lecture	Logic Coq/Lean	Formalisms	PL Haskell
1	Propositional and first-order logic		
2			Functional programming
3		Syntax and Semantics	
4			The untyped lambda calculus
5		Types	
6			The typed lambda calculus
7			Polymorphism
8		Curry-Howard	
9			Higher-order types
10			Dependent types

Goals

Let’s enter a more powerful logic system that allows us to express

- (in-)equalities and
- arithmetics.

Motivating examples

- Natural numbers with addition:

  $0 + 1 = 1$

- Generalization:

  $0 + x = x$

- And even tougher ones:

  $x + 0 = x$, $x + y = y + x$

- Verification of the correctness of a compiler transformation `trans` for a program p:

  $\texttt{exec}(\texttt{trans}\ p) = \texttt{exec}\ p$

From Propositional to First-Order Logic

Syntax A ::=		formulas/propositions:
	$\mid \quad X$	propositional variables
	$\mid \quad A \Rightarrow A$	implication
	$\mid \quad A \wedge A$	conjunction
	$\mid \quad \top$	truth
	$\mid \quad A \vee A$	disjunction
	$\mid \quad \bot$	falsity
	$\mid \quad \neg A$	negation

Examples: Propositional Logic

$$X_1 \vee X_2 \Rightarrow X_3$$

From Propositional to First-Order Logic

Syntax A ::=		formulas/propositions:
$\mid$	$P(t_1, \ldots, t_n)$	predicates on terms
$\mid$	$A \Rightarrow A$	implication
$\mid$	$A \wedge A$	conjunction
$\mid$	$\top$	truth
$\mid$	$A \vee A$	disjunction
$\mid$	$\perp$	falsity
$\mid$	$\neg A$	negation
$\mid$	$\forall x.A$	universally quantified term variables
$\mid$	$\exists x.A$	existentially quantified term variables

Examples:

Propositional Logic	First-Order Logic
$X_1 \vee X_2 \Rightarrow X_3$	$\forall x.\exists y.(x \times y = 1 \wedge y \times x = 1)$

Predicates

Syntax P	$::=$	predicates:
	$\mid \quad P \in \mathcal{P}$	predicate symbol

- ... with $a :: \mathcal{P} \to \mathbb{N}$ defining the arity of predicates.
- Let $\mathcal{P} = \mathcal{X}$ where $\mathcal{X}$ is the set of propositional variables such that $\forall X \in \mathcal{X}.a(X) = 0$,
- then

Propositional formula	First-order formula
$X \vee \neg Y$	$X() \vee \neg Y()$

Terms

Syntax

$$
\begin{array}{lcll}
t & ::= & & \text{terms:} \\
 & | & x & \text{variables} \\
 & | & f(t_1, \ldots, t_n) & \text{function application} \\
f & ::= & f \in \Sigma & \text{function symbols}
\end{array}
$$

- ... with an associated arity: $a :: \Sigma \to \mathbb{N}$
- and $\mathcal{T}$ as the smallest set of terms.

Example:

	Predicates	Terms	Formula
	$\mathcal{P} = \{= : 2\}$	$\Sigma = \{\times : 2, 1 : 0\}$	$\forall x.\exists y.(x \times y = 1 \wedge y \times x = 1)$

Bound and Free Variables

Syntax A	$::=$	$\ldots$	formulas/propositions:
	$\mid$	$\forall x.A$	universally quantified term variables
	$\mid$	$\exists x.A$	existentially quantified term variables

- We say: "Term variable x is *bound* in formula A.
- A variable that is not bound is *free*.
- A formula A is *closed* when it does not contain free variables, i.e., $FV(A) = \emptyset$.

Free Variables
Definitions and Notation

- ... of a formula A:

$$\begin{aligned} FV_A &:: A \rightarrow \{x|x \in X\} \\ FV_A(P(t_1,\ldots,t_n)) &= FV_t(t_1) \cup \ldots \cup FV_t(t_n) \\ FV_A(A_1 \Rightarrow A_2) = FV_A(A_1 \vee A_2) = FV_A(A_1 \wedge A_2) &= FV_A(A_1) \cup FV_A(A_2) \\ FV_A(\top) = FV_A(\bot) &= \emptyset \\ FV_A(\neg A_1) &= FV_A(A_1) \\ FV_A(\forall x.A_1) = FV_A(\exists x.A_1) &= FV_A(A_1) \setminus \{x\} \end{aligned}$$

- ... of a term t:

$$\begin{aligned} FV_t &:: t \rightarrow \{x|x \in X\} \\ FV_t(x) &= \{x\} \\ FV_t(f(t_1,\ldots,t_n)) &= FV_t(t_1) \cup \ldots \cup FV_t(t_n) \end{aligned}$$

- ... of a context Γ:

$$\begin{aligned} FV_\Gamma &:: \Gamma \rightarrow \{x|x \in X\} \\ FV_\Gamma(x : A_1,\ldots,x : A_n) &= FV_A(A_1) \cup \ldots \cup FV_A(A_n) \end{aligned}$$

- Notation: $A(x_1,\ldots,x_n)$ where $x_1,\ldots,x_n$ are free variable of A

Substitution

Definition (Substitution)

A *substitution* is a function $\sigma :: \mathcal{X} \to \mathcal{T}$ such that the set $\{x \in \mathcal{X} | \sigma(x) \neq x\}$ is finite.

- We write $t[\sigma(x)]$ for the term where every occurence of x in t is replaced by $\sigma(x)$.
- The replacement function is trivial:

$$\begin{aligned} x[\sigma] &= \sigma(x) \\ f(t_1, \ldots, t_n)[\sigma] &= f(t_1[\sigma], \ldots, t_n[\sigma]) \end{aligned}$$

- Notation:

  $A(t_1, \ldots t_n)$ instead of $A[t_1/x_1, \ldots, t_n/x_n]$

Natural Deduction Rules

- We extend the natural deduction rules of propositional logic with the following rules:

$$\frac{\Gamma \vdash \forall x.A}{\Gamma \vdash A[t/x]}\ (\forall_E) \qquad \frac{\Gamma \vdash A}{\Gamma \vdash \forall x.A}\ (\forall_I) \text{ if } x \notin FV_\Gamma(\Gamma)$$

$$\frac{\Gamma \vdash \exists x.A_1 \quad \Gamma, A_1 \vdash A_2}{\Gamma \vdash A_2}\ (\exists_E) \text{ if } x \notin FV_\Gamma(\Gamma) \cup FV_A(A_2) \qquad \frac{\Gamma \vdash A[t/x]}{\Gamma \vdash \exists x.A}\ (\exists_I)$$

Theories

Definition (First-order Theory)

A first-order theory $\Theta(\mathcal{P}, \Sigma)$ on a set of predicates $\mathcal{P}$ and a signature of terms Σ is a (possibly infinite) set of closed formulas also called *axioms*.

Definition (Provability)

A formula A is *provable* if there is $\Gamma \subseteq \Theta$ such that $\Gamma \vdash A$ is provable.

- Properties:

consistent	when $\perp$ is not provable in the theory
complete	when for every formula A, **either** A **or** $\neg A$ is provable in the theory
decidable	when there is an algorithm that decides whether a given formula is provable or not

The decidable theory of equality with uninterpreted functions

$$\Theta := (\mathcal{P}, \Sigma) \;\; \text{where} \;\; \begin{aligned} \mathcal{P} &= \{=: 2\} \\ \Sigma &= \{f : n \mid f \in \text{Strings}, n \in \mathbb{N}\} \end{aligned}$$

- Axioms:

$\boxed{t = t'}$

$$\frac{}{t = t} \text{ Reflexivity} \qquad \frac{t_2 = t_1}{t_1 = t_2} \text{ Symmetry} \qquad \frac{t_1 = t_{12} \quad t_{12} = t_2}{t_1 = t_2} \text{ Transitivity}$$

$$\frac{f = f' \quad t_1 = t_1' \quad \dots \quad t_n = t_n'}{f(t_1, \dots, t_n) = f'(t_1', \dots, t_n')} \text{ Function Congruence}$$

$$\frac{P(A_1, \dots, A_m) \quad A_1 = A_1' \quad \dots \quad A_m = A_m'}{P(A_1', \dots, A_m')} \text{ Predicate Congruence}$$

Presburger Arithmetic

- Axiomatizes the addition over natural numbers.
- Theory of equality over the signature $\Sigma = \{0 : 0, S : 1, + : 2\}$
- with the following axioms:

$$\frac{0 = S(x)}{\perp}\ (\mathsf{ax}_{=\perp}) \qquad \frac{S(x) = S(y)}{x = y}\ (S_{\mathsf{inj}}) \qquad \frac{}{0 + x = x}\ (\mathsf{ax}_{+_1}) \qquad \frac{}{S(x) + y = S(x + y)}\ (\mathsf{ax}_{+_2})$$

Recurrence Principle

$$\dfrac{\pi_0 \qquad \pi_S}{\forall x. x + 0 = x} \text{ (cases)}$$

$$\dfrac{x = 0 \vdash x + 0 = x \qquad x = S(y) \vdash S(y) + 0 = S(y)}{x + 0 = x} \text{ (cases)}$$

Recurrence Principle

$$\pi_S \quad :=$$

$$\pi_0 \quad :=$$

$$\pi_{\text{Left Identity}} \quad := \quad \frac{\pi_0 \qquad \pi_S}{\forall x. x + 0 = x} \text{ (cases)}$$

```
fn pi_left_identity(x:Nat) { }
```

Recurrence Principle

$$\pi_S \quad :=$$

$$\pi_0 \quad := \quad 0 + 0 = 0 \qquad\qquad =: \quad (x + 0 = x)[0/x]$$

$$\pi_{\text{Left Identity}} \quad := \quad \frac{\pi_0 \qquad \pi_S}{\forall x. x + 0 = x} \text{ (cases)}$$

Recurrence Principle

$$\pi_S \quad := \quad \dfrac{\dfrac{\dfrac{}{\forall x. S(x) + 0 = S(x+0)}\ (\text{ax}_{+_2}) \qquad \dfrac{\dfrac{}{S = S}\ (\text{ax}_{S-\text{cong}}) \qquad \dfrac{\forall x. f(x)}{\forall x. x + 0 = x}}{\forall x. S(x+0) = S(x)}\ (\text{fun cong})}{\forall x. S(x) + 0 = S(x)}\ (=_{\text{transitivity}})}{S(x) + 0 = S(x)}\ (\forall_E)$$

$$\pi_0 \quad := \quad \dfrac{}{0 + 0 = 0}\ (\text{ax}_{+_1})$$

$$f(n) := \begin{cases} \pi_0 & \text{if } n \text{ is } 0 \\ \pi_S[y/x] & \text{if } n \text{ is } S(y) \end{cases}$$

$$\pi_{\text{Left Identity}} \quad := \quad \dfrac{\pi_0 \qquad \pi_S}{\forall x. x + 0 = x}\ (\text{cases})$$

Recurrence Principle

$$\pi_S := \dfrac{\dfrac{\dfrac{}{\forall x. S(x) + 0 = S(x+0)}\,(\mathsf{ax}_{+_2}) \quad \dfrac{\dfrac{}{S = S}\,(\mathsf{ax}_{S-\mathsf{cong}}) \quad \dfrac{\forall x. f(x)}{\forall x. x + 0 = x}}{\forall x. S(x+0) = S(x)}\,(\text{fun cong})}{\dfrac{\forall x. S(x) + 0 = S(x)}{S(x) + 0 = S(x)}\,(\forall_E)}\,(=_{\text{transitivity}})$$

$$\pi_0 := \dfrac{}{0 + 0 = 0}\,(\mathsf{ax}_{+_1})$$

$$f(n) := \begin{cases} \pi_0 & \text{if } n \text{ is } 0 \\ \pi_S[y/x] & \text{if } n \text{ is } S(y) \end{cases}$$

$$\pi_{\text{Left Identity}} := \dfrac{\pi_0 \qquad \pi_S}{\forall x. x + 0 = x}\,(\text{cases})$$

```
fn f(x:Nat) {
  match x {
    0    => pi_0,
    S(y) => pi_S(y)
  }
}
```

Recurrence Principle

$$\pi_S \quad := \quad \dfrac{\dfrac{\dfrac{}{\forall x. S(x) + 0 = S(x+0)}\,(\mathsf{ax}_{+_2}) \qquad \dfrac{\dfrac{}{S = S}\,(\mathsf{ax}_{S-\mathsf{cong}}) \qquad \dfrac{\forall x. f(x)}{\forall x. x + 0 = x}}{\forall x. S(x+0) = S(x)}\,(\text{fun cong})}{\forall x. S(x) + 0 = S(x)}\,(=_{\text{transitivity}})}{S(x) + 0 = S(x)}\,(\forall_E)$$

$$\pi_0 \quad := \quad \dfrac{}{0 + 0 = 0}\,(\mathsf{ax}_{+_1})$$

$$f(n) := \begin{cases} \pi_0 & \text{if } n \text{ is } 0 \\ \pi_S[y/x] & \text{if } n \text{ is } S(y) \end{cases}$$

$$\pi_{\text{Left Identity}} \quad := \quad \dfrac{\pi_0 \qquad \pi_S}{\forall x. x + 0 = x}\,(\text{cases})$$

axiom scheme: $\boxed{\dfrac{A(0) \qquad (\forall x. A(x) \rightarrow A(S(x)))}{\forall x. A\ x}\,(\text{Rec})}$

Induction Principle

$$\pi_S \quad := \quad \cfrac{\cfrac{}{S(x)+0 = S(x+0)}\;(\mathsf{ax}_{+_2}) \qquad \cfrac{\cfrac{}{S = S}\;(\mathsf{ax}_{S-\mathsf{cong}}) \qquad x+0=x \vdash x+0=x}{x+0=x \vdash S(x+0) = S(x)}\;(\text{fun cong})}{\cfrac{x+0=x \vdash S(x)+0 = S(x)}{x+0=x \Rightarrow S(x)+0=S(x)}\;(\Rightarrow_I)}\;(=_{\text{transitivity}})$$

$$\pi_0 \quad := \quad \cfrac{}{0+0=0}\;(\mathsf{ax}_{+_0})$$

$$\pi_{\text{Left Identity}} \quad := \quad \cfrac{\pi_0 \qquad \pi_S}{\forall x. x+0 = x}\;(\textbf{Ind})$$

$$\boxed{\cfrac{A(0) \qquad (\forall x. A(x) \Rightarrow A(S(x)))}{\forall x. A\ x}\;(\text{Ind})}$$

Induction Principle

$$\pi_S \quad := \quad \dfrac{\dfrac{\dfrac{}{S(x)+0 = S(x+0)}\,(\text{ax}_{+_2}) \qquad \dfrac{\dfrac{}{S = S}\,(\text{ax}_{S-\text{cong}}) \qquad \dfrac{}{x+0=x \vdash x+0=x}\,(\text{ax})}{x+0=x \vdash S(x+0) = S(x)}\,(\text{fun cong})}{x+0=x \vdash S(x)+0 = S(x)}\,(=_{\text{transitivity}})}{x+0=x \Rightarrow S(x)+0 = S(x)}\,(\Rightarrow_I)$$

$$\pi_0 \quad := \quad \dfrac{}{0+0=0}\,(\text{ax}_{+_0})$$

$$\pi_{\text{Left Identity}} \quad := \quad \dfrac{\pi_0 \qquad \pi_S}{\forall x. x+0 = x}\,(\textbf{Ind})$$

$$\boxed{\dfrac{A(0) \qquad (\forall x. A(x) \Rightarrow A(S(x)))}{\forall x. A\ x}\,(\text{Ind})}$$

Induction Principle

Induction Hypothesis

$$\pi_S \quad := \quad \dfrac{\dfrac{\dfrac{}{S(x)+0 = S(x+0)}\,(\mathsf{ax}_{+_2}) \qquad \dfrac{\dfrac{}{S = S}\,(\mathsf{ax}_{S-\mathsf{cong}}) \qquad \dfrac{}{x+0=x \vdash x+0=x}\,(\mathsf{ax})}{x+0=x \vdash S(x+0) = S(x)}\,(\mathsf{fun\ cong})}{x+0=x \vdash S(x)+0 = S(x)}\,(=_{\mathsf{transitivity}})}{x+0=x \Rightarrow S(x)+0 = S(x)}\,(\Rightarrow_I)$$

$$\pi_0 \quad := \quad \dfrac{}{0+0=0}\,(\mathsf{ax}_{+_0})$$

$$\pi_{\text{Left Identity}} \quad := \quad \dfrac{\pi_0 \qquad \pi_S}{\forall x. x+0 = x}\,(\mathbf{Ind})$$

$$\boxed{\dfrac{A(0) \qquad (\forall x. A(x) \Rightarrow A(S(x)))}{\forall x. A\ x}\,(\mathsf{Ind})}$$

Induction vs. Cases

$$\frac{x = 0 \vdash A(x) \qquad (\forall x. A(x) \Rightarrow A(S(x)))}{\forall x. A\ x} \text{ (Ind)} \qquad \frac{x = 0 \vdash A(x) \quad x = S(y) \vdash A(S(y))}{\forall x. A(x)} \text{ (cases)}$$

- Important point to remember:
  - Cases does not provide a induction hypothesis, i.e.,
  - it does not cover recursion!

Induction vs. Recurrence

$$\frac{A(0) \qquad (\forall x. A(x) \Rightarrow A(S(x))}{\forall x. A\ x}\ (\text{Ind}) \qquad \frac{A(0) \qquad (\forall x. A(x) \to A(S(x))}{\forall x. A\ x}\ (\text{Rec})$$

- Look what just happened:
  - We used a proof just like a function!
  - There seems to be a deeper connection between implication and a function (type)?
- Let's explore this connection in this lecture!