Computer Supported Modeling and Reasoning

David Basin, Achim D. Brucker, Jan-Georg Smaus, and Burkhart Wolff

April 2005

http://www.infsec.ethz.ch/education/permanent/csmr/

Metatheory II: Rules

David Basin

ETH Zurich
15.11.04

Overview — Proof Rules in the LF

- Last lecture: simple types classify syntax
- Dependent types classify rules: $pr : o \longrightarrow Type$

$$\frac{A \wedge B}{A} \quad \rightsquigarrow \quad andel : \Pi x^o.\, \Pi y^o.\, pr(and\ x\ y) \longrightarrow pr(x)$$

- Judgements as Types:

$$\begin{array}{l} \vdots\ P \\ \vdash \phi \end{array} \quad \rightsquigarrow \quad \ulcorner P \urcorner \in pr(\ulcorner \phi \urcorner)$$

  - Models syntax: $\phi \in Prop$ *iff* $\ulcorner \phi \urcorner \in o$
  - Models provability: $\vdash_L \phi$ *iff* $\vdash_{LF} pr(\ulcorner \phi \urcorner)$
  - Models proofs: P *iff* $\ulcorner P \urcorner$

Representing Proofs (cont.)

- Additional levels in the theory: Term $\in$ Type $\in$ Kind

$$pr \quad \in \quad o \to Type \quad \in \quad Kind$$

- Add dependent Π-types:

$$\lambda x^N.\, x + 1 : \Pi x^N.\, \{y : N | x < y\}$$

- Captures dependencies in proof rules

$andel : pr \to pr$ becomes $andel : \Pi a^o b^o.\, pr(and\, a\, b) \to pr(a)$

- Resulting type theory: minimal logic over $\forall/\Rightarrow$ with ω-order quantification

LF Presentation — Pseudo-Terms

- Let *Var* and *Const* be sets of variables and constants.

$$\{Type, Kind\} \in \mathit{Const}$$

- Pseudo-terms $\mathcal{T}$

$$\mathcal{T} ::= \mathit{Var} \mid \mathit{Const} \mid \mathcal{T}\mathcal{T} \mid \lambda x^{\mathcal{T}}.\mathcal{T} \mid \Pi x^{\mathcal{T}}.\mathcal{T}$$

  - In $\Pi x^{\tau_1}.\tau_2$, x is bound in τ_2
  - Write $\Pi x^{\tau_1}.\tau_2$ as $\tau_1 \to \tau_2$ when x not in τ_2

- Rules prescribe which pseudo-terms are sensible, e.g., in $\lambda^{\to}$
  - Some are terms: $\lambda x^{N}.x$
  - Some are types: $\Pi x^{N}.N \equiv N \to N$
  - Some are nonsense: $\Pi x^{\lambda y^{N}.y}.x$

LF — Signatures, $\vdash_{sig}$

- Let s range over sorts $\{Type, Kind\}$
- Define valid signature $\vdash_{sig}$, context $\vdash_{con}$, and type assignment $\vdash_{\Sigma}$
- A signature Σ is a sequence ($c \in \mathit{Const}$)

$$\Sigma ::= \langle\,\rangle \mid \Sigma, c : A$$

- Valid signature Σ satisfies $\vdash_{sig}$ (all constants are well-typed or well-kinded)

$$\vdash_{sig} \langle\,\rangle \qquad \frac{\vdash_{sig} \Sigma \quad \vdash_{\Sigma} A : s \quad c \notin dom(\Sigma)}{\vdash_{sig} \Sigma, c : A}$$

- Example (type environment for constants):

$$\Sigma = \langle o : Type, not : o \to o, \ldots \rangle$$

LF Context, $\vdash_{con}$

- A context Γ is a sequence (for $x \in \mathit{Var}$)

$$\Gamma ::= \langle\,\rangle \mid \Gamma, x : A$$

- Γ valid w.r.t. Σ if it satisfies $\vdash_{con}$ (all variables are well-typed or well-kinded)

$$\vdash_{con} \langle\,\rangle \qquad \frac{\vdash_{con} \Gamma \quad \Gamma \vdash_{\Sigma} A : s \quad x \notin dom(\Gamma)}{\vdash_{con} \Gamma, x : A}$$

- Example ('symbol table for variables'):

$$\Gamma = \langle x : o, y : o, z : o \to o \rangle$$

LF Type Assignment Relation, $\vdash_\Sigma$

axiom $$\Gamma \vdash_\Sigma Type : Kind$$

assumption $$\frac{c : A \in \Sigma}{\Gamma \vdash_\Sigma c : A}$$

hypothesis $$\frac{x : A \in \Gamma}{\Gamma \vdash_\Sigma x : A}$$

formation $$\frac{\Gamma \vdash_\Sigma A : Type \quad \Gamma, x : A \vdash_\Sigma B : s}{\Gamma \vdash_\Sigma \Pi x^A . B : s}$$

abstraction $$\frac{\Gamma, x : A \vdash_\Sigma b : B \quad \Gamma \vdash_\Sigma \Pi x^A . B : s}{\Gamma \vdash_\Sigma \lambda x^A . b : \Pi x^A . B}$$

application $$\frac{\Gamma \vdash_\Sigma F : \Pi x^A . B \quad \Gamma \vdash_\Sigma a : A}{\Gamma \vdash_\Sigma F(a) : B[x \leftarrow a]}$$

Restriction of LF rules are $\lambda^{\rightarrow}$ rules

- Restrict typing assertion $e : B$

  where e is $\lambda^{\rightarrow}$ term and B is simple-type.

- LF rules formalize well-formed signature, context, and type.

$\tau ::= T \mid \tau \rightarrow \tau$ (Type Grammar) $\quad \Gamma, x : \tau, \Delta \vdash x : \tau$ hyp

$$\frac{\Gamma \vdash e : \sigma \rightarrow \tau \quad \Gamma \vdash e' : \sigma}{\Gamma \vdash e e' : \tau}\ \text{app} \qquad \frac{\Gamma, x : \sigma \vdash e : \tau}{\Gamma \vdash \lambda x^{\sigma}.\, e\ : \sigma \rightarrow \tau}\ \text{abs}$$

LF and $\lambda^{\rightarrow}$ — Example

- Signature for logic over arrow (base type explicit):

$$\Sigma = \langle o : \mathit{Type}, not : o \rightarrow o, imp : o \rightarrow o \rightarrow o \rangle \,,$$

- Signature well-formed

$$\vdash_{\langle\,\rangle} Type : Kind$$

and

$$\frac{\dfrac{o : Type \in \langle o : Type \rangle}{\vdash_{\langle o:Type \rangle} o : Type} \qquad \dfrac{o : Type \in \langle o : Type \rangle}{x : o \vdash_{\langle o:Type \rangle} o : Type}}{\vdash_{\langle o:Type \rangle} o \rightarrow o : Type}$$

similar for type of imp

(Full) LF Type Theory

- Additional rules add power, e.g., formation (here $s = Kind$)

$$\frac{\Gamma \vdash_\Sigma A : Type \quad \Gamma, x : A \vdash_\Sigma B : Kind}{\Gamma \vdash_\Sigma \Pi x^A . B : Kind}$$

$$\begin{aligned} \Sigma_{prop} \quad = \quad & \langle o : Type, pr : o \to Type, not : o \to o, \\ & imp : o \to o \to o, and : o \to o \to o \rangle \end{aligned}$$

- Type-valued functions in context are now well-typed

$$\frac{o : Type \vdash_\Sigma o : Type \quad o : Type, x : o \vdash_\Sigma Type : Kind}{o : Type \vdash_\Sigma o \to Type : Kind}$$

- Moreover, dependent types not possible in $\lambda^{\to}$

Example of Dependent Types — Rule Formalization

- Consider $andel : \Pi x^o.\, \Pi y^o.\, pr(and\ x\ y) \to pr(x)$
- Part I

$$\dfrac{\dfrac{\dfrac{}{x:o, y:o \vdash_\Sigma pr : o \to Type} \quad \dfrac{\dfrac{\dfrac{}{x:o, y:o \vdash_\Sigma and : o \to o \to o} \quad \dfrac{}{x:o, y:o \vdash_\Sigma x : o}}{x:o, y:o \vdash_\Sigma and\, x : o \to o}\, app \quad \dfrac{}{x:o, y:o \vdash_\Sigma y : o}}{x:o, y:o \vdash_\Sigma and\, x\, y : o}\, app}{x:o, y:o \vdash_\Sigma pr(and\, x\, y) : Type}\, app$$

- Part II

$$\dfrac{\dfrac{}{\vdash_\Sigma o : Type} \quad \dfrac{\dfrac{}{x : o \vdash_\Sigma \Pi y^o.\, pr(and\, x\, y) \to pr(x) : Type}}{\vdash_\Sigma \Pi x^o.\, \Pi y^o.\, pr(and\, x\, y) \to pr(x) : Type}\, form}{x : o \vdash_\Sigma \Pi y^o.\, pr(and\, x\, y) \to pr(x) : Type}\, form$$

$$\dfrac{\dfrac{}{x:o, y:o \vdash_\Sigma pr(and\, x\, y) \to pr(x) : Type}}{x:o, y:o, z:pr(x) \vdash_\Sigma pr(x) : Type}\, form$$

(Part I)

$$\dfrac{x:o, y:o, z:pr(x) \vdash_\Sigma pr : o \to Type \quad x:o, y:o, z:pr(x) \vdash_\Sigma x : o}{x:o, y:o, z:pr(x) \vdash_\Sigma pr(x) : Type}\, app$$

Metatheoretic Properties of LF

- $\forall/\Rightarrow$ fragment of intuitionistic ω-order logic

$$\Pi x^o.\, \Pi y^o.\, pr(and\ x\ y) \rightarrow pr(x)$$
$$\Downarrow$$
$$\forall x^o.\, \forall y^o.\, pr(and\ x\ y) \Rightarrow pr(x)$$

- Unique normal forms and subject reduction

  Faithfulness/adequacy by analysis of normal forms

- $\Gamma \vdash_\Sigma t : \tau$ is decidable.
  - Well-formedness is decidable: $\tau : Type$ or $\tau : Kind$
  - Proof checking is decidable: $t : \tau$

Idea of LF Encodings

Rules organized around basic assertions (judgements)

- $pr(p)$: p is provable
- $valid(p)$: p is true in all models
- $true(p)$: p is true in some model
- $acc(w_1, w_2)$: '$world$' w_1 accesses w_2
- Multiple judgements, e.g. type theory and modal logics

\+ higher-order judgements: (J_i judgement, C syntactic category)

- Schematic: $\bigwedge_{x \in C} J(x)$
- Hypothetical: $J_1 \vdash_\Sigma J_2$ expresses consequence
  and allows hybrid notions of consequence

$$valid(p) \vdash_\Sigma true(p)$$

Idea of LF Encodings: *Judgements as Types*

- Basic judgements associated with type families,
  e.g. $pr : o \to Type$

$$\overline{p \text{ provable}} \rightsquigarrow pr(p)$$

- Higher-order forms available

$$\overline{\bigwedge_{x \in C} J(x)} \rightsquigarrow \Pi x^{\overline{C}}.\, \overline{J(x)}$$

$$\overline{J_1 \vdash J_2} \rightsquigarrow \overline{J_1} \to \overline{J_2}$$

- Truth $=$ inhabitation

$t : pr(p)$ *iff* t represents proof that p is provable

Example: Propositional Logic

- Formalize language ....

$$P ::= x \mid P \wedge P \mid P \Rightarrow P \ldots$$

- ... and proof rules

$$\frac{A \wedge B}{A}\ \wedge\text{-}EL \qquad \frac{A \wedge B}{B}\ \wedge\text{-}ER \qquad \frac{A \quad B}{A \wedge B}\ \wedge\text{-}I$$

$$\frac{A \Rightarrow B \quad A}{B}\ \Rightarrow\text{-}E \qquad \frac{\begin{array}{c}[A]\\ \vdots \\ B\end{array}}{A \Rightarrow B}\ \Rightarrow\text{-}I$$

Propositional Logic in LF

- Σ contains syntactic categories

$$o : Type$$

- and judgements

$$pr : o \rightarrow Type$$

- and operators (declaring syntax)

$$imp : o \rightarrow o \rightarrow o, and : o \rightarrow o \rightarrow o, \ldots$$

- and rules (next page)

Propositional Logic in LF

- and-rules (and impe) are 1st-order judgements

$$\frac{A \wedge B}{A} \quad \rightsquigarrow \quad \Pi a^o b^o.\, pr(and\ a\ b) \rightarrow pr(a)$$

- $\Rightarrow$-I is 2nd-order judgement

$$\frac{\begin{array}{c}[A]\\ \vdots \\ B\end{array}}{A \Rightarrow B}\ \Rightarrow\text{-}I \quad \rightsquigarrow \quad \begin{array}{l}\Pi a^o b^o.\, (pr(a) \rightarrow pr(b)) \\ \qquad \rightarrow pr(imp\ a\ b)\end{array}$$

  - Consequence over *hypothetical proofs*: $(A \vdash B) \vdash A \Rightarrow B$
  - Shifts assumption discharging to LF

$$\dfrac{\dfrac{x : pr(a) \vdash_\Sigma \ldots : pr(b)}{\vdash_\Sigma \ldots : pr(a) \rightarrow pr(b)}\, abs_{T,T}}{\vdash_\Sigma \ldots : pr(imp\ a\ b)}\, impi^*$$

Propositional Logic — More Rules

$$\frac{A}{A \vee B}\ \vee\text{-}IL \qquad \frac{B}{A \vee B}\ \vee\text{-}IR \qquad \frac{A \vee B \qquad \begin{array}{c}[A]\\ \vdots \\ C\end{array} \qquad \begin{array}{c}[B]\\ \vdots \\ C\end{array}}{C}\ \vee\text{-}E$$

- Augment operators: $or : o \to o \to o$
- Augment rules:

$$\begin{aligned} &oril : \Pi a^o b^o.\, pr(a) \to pr(or\, a\, b)\\ &orir : \Pi a^o b^o.\, pr(b) \to pr(or\, a\, b)\\ &ore : \Pi a^o b^o c^o.\, pr(or\, a\, b) \to (pr(a) \to pr(c))\\ &\qquad \to (pr(b) \to pr(c)) \to pr(c) \end{aligned}$$

Propositional Proofs: $a \Rightarrow a$

Part I:

$$\dfrac{\dfrac{a : o \vdash_\Sigma impi : \Pi a^o b^o.\, (pr(a) \to pr(b)) \to pr(imp\, a\, b) \quad a : o \vdash_\Sigma a : o}{a : o \vdash_\Sigma impi\, a : \Pi b^o.\, (pr(a) \to pr(b)) \to pr(imp\, a\, b) \qquad a : o \vdash_\Sigma a : o}}{a : o \vdash_\Sigma impi\, a\, a : (pr(a) \to pr(a)) \to pr(imp\, a\, a)}$$

Part II:

$$\dfrac{a : o, y : pr(a) \vdash_\Sigma y : pr(a)}{a : o \vdash_\Sigma \lambda y^{pr(a)}.\, y : pr(a) \to pr(a)}$$

Part III:

$$\dfrac{\dfrac{\text{(Part I)} \qquad \text{(Part II)}}{a : o \vdash_\Sigma impi\, a\, a\, (\lambda y^{pr(a)}.\, y) : pr(imp\, a\, a)}}{\vdash_\Sigma \lambda a^o.\, impi\, a\, a\, (\lambda y^{pr(a)}.\, y) : \Pi a^o.\, pr(imp\, a\, a)}$$

LF — Aside on System Support

Proof detail impractical. Implementations support:

- Implicit quantification/type reconstruction

$$\lambda a^{o}.\, impi\, a\, a\, (\lambda y^{pr(a)}.\, y) : \Pi a^{o}.\, pr(imp\, a\, a)$$

- Automatic type checking
- "Resolution" and refinement style proof.

$$\frac{\Gamma \vdash_{\Sigma} \ldots : pr(imp\, a\, a)}{\Gamma, x : pr(a) \vdash_{\Sigma} \ldots : pr(a)}\, impi^{*}$$

- Term synthesis via unification

$$\Gamma \vdash_{\Sigma} ?t : pr(imp\, a\, a)$$

- Pretty printing, implicit contexts, and implicit judgement coercions

$$\frac{\vdash a \Rightarrow a}{a \vdash a}$$

- Tactics

LF — Terms Encode Proofs

- $a \Rightarrow a \quad \rightsquigarrow \quad \Pi a^{o}.\, pr(imp\, a\, a)$

$$\lambda a^{o}.\, impi\, a\, a\, (\lambda y^{pr(a)}.\, y)$$

$$\frac{[A]^{y}}{A \Rightarrow A}\, \Rightarrow\text{-}I^{y}$$

- $a \Rightarrow b \Rightarrow a \quad \rightsquigarrow \quad \Pi a^{o} b^{o}.\, pr(imp\, a\, (imp\, b\, a))$

$$\lambda a^{o}.\, \lambda b^{o}.\, impi\, a\, (imp\, b\, a)\, (\lambda x^{pr(a)}.\, impi\, b\, a\, (\lambda y^{pr(b)}.\, x))$$

$$\frac{\dfrac{[A]^{x}}{B \Rightarrow A}\, \Rightarrow\text{-}I^{y}}{A \Rightarrow (B \Rightarrow A)}\, \Rightarrow\text{-}I^{x}$$

- Adequacy & faithfulness: There exists a bijection between proofs of $\phi \in Prop$ and canonical members of $pr(\phi)$

Schematic Judgements — F.O. Arithmetic Example

- Review: syntax

$$\begin{array}{lll} \textit{Terms } T & ::= & x \mid 0 \mid sT \mid T+T \mid T \times T \\ \textit{Formulae } F & ::= & t = t \mid \neg F \mid F \wedge F \mid \ldots \\ & & \forall x.\, A \mid \exists x.\, A \end{array}$$

- Type declarations for context

$$B = i : \textit{Type}, o : \textit{Type}$$

- Context $\Gamma = \langle B, \Gamma_T, \Gamma_P, \Gamma_Q \rangle$:

$$\begin{array}{lll} \Gamma_T & = & 0 : i,\ s : i \to i,\ plus : i \to i \to i,\ times : i \to i \to i \\ \Gamma_P & = & eq : i \to i \to o,\ not : o \to o,\ and : o \to o \to o,\ \ldots \\ \Gamma_Q & = & all : (i \to o) \to o,\ exists : (i \to o) \to o \end{array}$$

FO Arithmetic — ∀-Quantifier Rules

$$\frac{A}{\forall x.\, A}\ \forall\text{-}I^{*} \qquad \frac{\forall x.\, A}{A[x \leftarrow t]}\ \forall\text{-}E$$

$*$: x not free in any undischarged assumptions on which A depends

$$alli : \Pi A^{i \to o}.\, (\Pi x^{i}.\, pr(A(x))) \to pr(all(A))$$
$$alle : \Pi A^{i \to o} x^{i}.\, pr(all(A)) \to pr(A(x))$$

Side Conditions Captured via Π-binding

Prove $(\forall x.\, A \Rightarrow B(x)) \Rightarrow (A \Rightarrow \forall x.\, B(x))$ in context $\Gamma, A : o, B : i \to o, x : i$

Part I:

$$\frac{pr(all(\lambda x^{i}.\, A \Rightarrow B(x))), pr(A) \vdash_{\Sigma} pr(all(\lambda x^{i}.\, A \Rightarrow B(x)))}{pr(all(\lambda x^{i}.\, A \Rightarrow B(x))), pr(A) \vdash_{\Sigma} pr(A \Rightarrow B(x))}\, alle$$

Part II:

$$\cfrac{\cfrac{\cfrac{\cfrac{\cfrac{\text{(Part I)}\quad pr(all(\lambda x^{i}.\, A \Rightarrow B(x))), pr(A) \vdash_{\Sigma} pr(A)}{pr(all(\lambda x^{i}.\, A \Rightarrow B(x))), pr(A) \vdash_{\Sigma} pr(B(x))}\, impe}{pr(all(\lambda x^{i}.\, (A \Rightarrow B(x)))), pr(A) \vdash_{\Sigma} \Pi x^{i}.\, pr(B(x))}\, abs \text{ (discharge } x : i)}{pr(all(\lambda x^{i}.\, (A \Rightarrow B(x)))), pr(A) \vdash_{\Sigma} pr(all(\lambda x^{i}.\, B(x)))}\, alli}{pr(all(\lambda x^{i}.\, (A \Rightarrow B(x)))) \vdash_{\Sigma} pr(A \Rightarrow all(\lambda x^{i}.\, B(x)))}\, impi}{pr(all(\lambda x^{i}.\, (A \Rightarrow B(x)))) \Rightarrow (A \Rightarrow all(\lambda x^{i}.\, B(x))))}\, impi$$

FO Arithmetic — Substitution

$$\frac{t = u \quad A[t]}{A[u]}\, sub \qquad \frac{a = b \quad P(a, a, a)}{P(a, b, a)}$$

$A[t]$ means some distinguished (not necessarily all) occurrences of t in A. In above, $A = \lambda x.\, P(a, x, a)$

$$sub : \Pi t^{i} u^{i} A^{i \to o}.\, pr(eq\, t\, u) \to pr(A(t)) \to pr(A(u))$$

Aside on Derived Rules

- Encoding of FOA adequate and faithful

  But more then FOA sentences are expressible/derivable!

- Rules are derivable

$$\dfrac{\forall x.\, A \qquad \begin{array}{c}[A[x \leftarrow t]] \\ \vdots \\ B\end{array}}{B}\, alle_{SH}$$

$$alle_{SH} : \Pi A^{i\to o} B^{o}.\, pr(all(A)) \to ((\Pi x^{i}.\, pr(A(x))) \to pr(B)) \to pr(B)$$

- Inhabitants are proof-construction functions. $alle_{SH} =$

$$\lambda A^{i\to o} B^{o} p^{pr(all(A))} q^{\Pi x^{i}.\, pr(A(x))\to pr(B)}.\, q(\lambda x^{i}.\, alle(\lambda x^{i}.\, B(x)(x)(p)))$$

- Can verify all *derivable* rules.

  Judgements are open: no induction over encoded theory

Summary of Proof/Rule Encoding

Object Language	Meta Language
Judgement $\vdash_\Sigma p$	Type Family $pr : o \to Type$
Inference Rule $\dfrac{A \wedge B}{A}\ \wedge\text{-}EL$	Constant Declaration $andel : \Pi a^o b^o.\, pr(and\, a\, b) \to pr(a)$
Deduction	Typing Proof
Deductive System	Context Declaration

Advantages/Disadvantages of LF

\+ High-level representation: HOAS and assumption management for ND

\+ Judgements as Types: Proof construction and adequacy arguments

\+ Good metatheoretic properties

  \+ Decidable type checking $\Gamma \vdash_\Sigma t : \tau$

+/− Supports some metatheoretic extensibility

  − No rules by induction over theory
  \+ Derived rules hold in theory extensions

+/− Biased towards certain logics: $\vdash_L$ shares properties with $\vdash_{LF}$

Relation to Type Theories based on HOL (Isabelle)

- $\lambda^{\rightarrow}$ used to declare syntax

$$\begin{aligned}\Gamma_{prop} \;=\; & \langle o : Type,\, pr : o \rightarrow Prop,\, not : o \rightarrow o, \\ & imp : o \rightarrow o \rightarrow o, and : o \rightarrow o \rightarrow o\rangle\end{aligned}$$

- $\forall/\Rightarrow$ fragment of HOL used to declare axioms

$$\begin{array}{c} andel : \Pi x^{o}.\, \Pi y^{o}.\, pr(and\ x\ y) \rightarrow pr(x) \\ \downarrow \\ andel : \forall x^{o}.\, \forall y^{o}.\, pr(and\ x\ y) \Rightarrow pr(x) \end{array}$$

- Formalizations similar, but no Judgements as Types

- Adequacy/faithfulness by proof (not term) normalization

Further Reading

- Barendregt, "Introduction to Generalized Type Systems", Journal of Functional Programming, April 1991.
- Harper, Honsell, and Plotkin, "A Framework for Defining Logics", JACM, January 1993.
- Paulson, "Isabelle: A Generic Theorem Prover", Springer LNCS 828.