Formal Derivation of Proof Rules

Roger Bishop Jones
ICL Defence Systems
VDM-SL PROOF THEORY INVESTIGATIONS

OBJECTIVES

CONSISTENCY, TERMINATION
and
PROOF THEORETIC STRENGTH

STAGES in
FORMAL DEVELOPMENT

DETAILS of
FORMAL THEORIES

CONCLUSIONS from
WORK TO DATE
OBJECTIVES

To FACILITATE the USE OF VDM-SL in developments involving FORMAL MACHINE CHECKED PROOF of CRITICAL PROPERTIES

To INVESTIGATE the FEASIBILITY of FORMAL DERIVATION of PROOF RULES from SEMANTICS

To BROADEN APPLICABILITY and CONTAIN COMPLEXITY by DEFERRING SPECIALISATION
CONSISTENCY, TERMINATION
and
PROOF THEORETIC STRENGTH

We are concerned to ensure the
LOGICAL CONSISTENCY of:

The LOGICAL FOUNDATIONS

User APPLICATION THEORIES

Unless we are assured of their consistency then
NO VALUE
can be attached to any proofs obtained.
APPLICATION THEORIES

Since CONSISTENCY of EXTENSIONS will NOT be DECIDABLE

it will be NECESSARY to PROVIDE for their CONSISTENCY to BE PROVABLE
WITHIN the logical SYSTEM

In order that this be possible as often as possible, a logical system with high "PROOF THEORETIC STRENGTH" is needed.

Proof Theoretic Strength
Cardinality of Universe
HIGH PROOF THEORETIC STRENGTH

⇒

more SPECIFICATIONS can be SHOWN CONSISTENT

more PROGRAMS can be SHOWN TO TERMINATE

PROOFS can be SHORTER

all of these follow from Goedel’s theorems.
OBSERVATIONS on COMPLEXITY

Mike Gordon’s HOL logic has:

- 7 clause abstract syntax
- 3 primitive type constructors (bool, ind, →)
- 2 type inference rules
- 3 primitive constants (=, ⇒, μ)
- 5 axioms
- 8 primitive inference rules

It is closely related to Church’s simple theory of types.

YET, since its development in 1985, SEVERAL problems in the formulation or implementation of this logic have been found which permit the derivation of contradictions.

VDM-SL will be MORE COMPLEX
by AT LEAST a FACTOR of TEN

HOW DO WE ENSURE that VDM-SL IS SUPPORTED
by SOUND PROOF DEVELOPMENT TOOLS?
HOW DO WE ENSURE that VDM-SL IS SUPPORTED by SOUND PROOF DEVELOPMENT TOOLS?

PROVIDE FORMAL SEMANTICS in
CLASSICAL SET THEORY (ZFC)

ENSURE that
the UNIVERSE of VDM-SL is
WELL POPULATED

MECHANICALLY DERIVE PROOF RULES from
PROOF RULES for SET THEORY

FORMALISATION of SEMANTICS and
DEVELOPMENT of PROOF THEORY must be
CONCURRENT and CLOSELY COUPLED
ESTABLISHING the
CONSISTENCY of FOUNDATION SYSTEMs

To PROVE the CONSISTENCY of a system it is necessary to work with a META-LOGIC which has GREATER PROOF THEORETIC STRENGTH

Since the CONSISTENCY of the META-LOGIC may be questioned, it is desirable to CHOOSE a SYSTEM directly RELATED TO well ESTABLISHED FOUNDATIONS

e.g. ZFC
STAGES in FORMAL DEVELOPMENT

1 formalise ZFC in HOL

2 change PRIMITIVES to FUNCTIONS

3 introduce POLYMORPHISM

4 introduce STRUCTURING

5 TYPES?
1 ZFC is AXIOMATISED over type "::SET" which is a new PRIMITIVE

2 PURE FUNCTIONS are type "::PPF", a subtype of "::SET"

function_hereditary_DEF
\[ \vdash \text{function\_hereditary\_p} = \]
\[ (\forall f \cdot \text{function\_f} \land \]
\[ \bot_\mathcal{Z} \not\in \mathcal{Z} (\text{image\_f}) \land \]
\[ (\forall x \cdot x \in \mathcal{Z} (\text{field\_f}) \Rightarrow p \ x) \Rightarrow \]
\[ p \ f) \]
pure_function_DEF
\[ \vdash \text{pure\_function\_s} = (\forall p \cdot \text{function\_hereditary\_p} \Rightarrow p \ s) \]
REPRESENTATIONS of TYPES(2)

3 POLYMORPHIC PURE FUNCTIONS are type ":PPF", and are represented by objects of type ":PF→PF"

4 STRUCTURED POLYMORPHIC FUNCTIONS are type ":SPF", and are represented by ‘REGULAR’ functions of type ":PPF→PPF"

regular

\[ \vdash \text{regular ppfun} = (\forall pf \cdot \exists \text{pfun} \cdot \forall ppf \cdot \text{REP_PPF}(\text{ppfun} ppf)pf = \text{pfun}(\text{REP_PPF ppf pf})) \]
KEY PRIMITIVES

1  ZFC
   membership $\in_z$, separation $\Lambda_z$

2  PF
   application $f$, abstraction $\lambda_f$

3  PPF
   application $p$, abstraction $\lambda_p$, type-vars $Tv_p$, type-instantiation $\$_p$, type-env $Te_p$

4  SPF
   application $s$, abstraction $\lambda_s$, type-vars $Tv_s$, individual-vars $Iv_s$, type-instantiation $\$_s$, value-instantiation $\$$_s$, type-env $Te_s$, value-env $Ie_s$
THE AXIOMS of ZERMELO FRAENKEL

There are three main sorts of axiom:

1  LOGICAL axioms (including =)

2  axioms CHARACTERISING SETS

   Extensionality and well foundedness.

3  axioms CHARACTERISING ABSTRACTION

   The axiom of separation.

4  POPULATING axioms

   i.e. axioms which assert the existence of various sets.

This broad pattern is followed by all the foundation systems which we discuss below.
HOL-ZFC PRIMITIVES

Types  
  ":SET"

Constants:

Membership  $\in_z$  ":SET \to (SET \to \text{bool})"

Separation  $\Lambda_z$  ":SET \to ((SET \to \text{bool}) \to \text{SET})"

Empty set  $\emptyset_z$  ":SET"

Power set  $\mathbb{P}$  ":SET \to \text{SET}"

Pair constructor  pair  ":SET \to (SET \to \text{SET})"

Union  $\cup_z$  ":SET \to \text{SET}"

Natural numbers  $\mathbb{N}$  ":SET"

Choice function  $\mu$  ":SET \to \text{SET}"

HOL-ZFC DEFINED CONSTANTS

Unit set \( \text{unit} \): \( \text{SET} \rightarrow \text{SET} \)

Intersection \( \cap_z \): \( \text{SET} \rightarrow \text{SET} \)

Inclusion \( \subseteq_z \): \( \text{SET} \rightarrow (\text{SET} \rightarrow \text{bool}) \)

Intersection \( \cap_z \): \( \text{SET} \rightarrow (\text{SET} \rightarrow \text{SET}) \)

Successor \( \text{suc} \): \( \text{SET} \rightarrow \text{SET} \)

Transitive \( \text{Trans} \): \( \text{SET} \rightarrow \text{bool} \)

Connected \( \text{Con} \): \( \text{SET} \rightarrow \text{bool} \)

Ordinal \( \text{On} \): \( \text{SET} \rightarrow \text{bool} \)

Successor \( \text{Sc} \): \( \text{SET} \rightarrow \text{bool} \)

Natural number \( \text{N} \): \( \text{SET} \rightarrow \text{bool} \)
HOL-ZFC AXIOMS

EXT  |\(\forall x\ y\cdot (\forall z\cdot z \in_z x \iff z \in_z y) \Rightarrow (x = y)\)

ZF2  |\(\forall A\ z\ x\cdot x \in_z (\Lambda z\ z A) \iff x \in_z z \land A x\)

ZF3  |\(\emptyset_z = \Lambda z\emptyset_z (\lambda x^1\cdot F)\)

ZF4  |\(\forall y\ x\cdot x \in_z (\mathbb{P} y) \iff x \subseteq_z y\)

ZF5  |\(\forall y\ z\ x\cdot x \in_z (\text{pair } y\ z) \iff (x = y) \lor (x = z)\)

ZF6  |\(\forall y\ x\cdot x \in_z (\bigcup_z y) \iff (\exists z\cdot z \in_z y \land x \in_z z)\)

ZF7  |\(\forall x\cdot x \neq \emptyset_z \Rightarrow (\exists y\cdot y \in_z x \land (y \cap_z x = \emptyset_z))\)

ZF8  |\(\forall x^1\cdot x^1 \in_z \mathbb{N} \iff \mathbb{N} x^1\)

ZF9  |\(\forall f\ r\cdot (\forall x\ y\ z\cdot f x\ y \land f x\ z \Rightarrow (z = y)) \Rightarrow (\exists w\cdot \forall y\ y \in_z w \iff (\exists x\cdot x \in_z r \land f x\ y))\)

ZF10 |\(\forall x^1\cdot x^1 \neq \emptyset_z \Rightarrow (\mu x^1) \in_z x^1\)
HOL-ZFC THEOREMS

ZF_thm19 $\vdash N \varnothing$

ZF_thm20 $\vdash \forall x^1 \cdot \varnothing \neq \text{suc } x^1$

ZF_thm22 $\vdash \forall x^1 \cdot N x^1 \Rightarrow N(\text{suc } x^1)$

ZF_thm23 $\vdash \forall x^1 x^2 \cdot (\text{suc } x^1 = \text{suc } x^2) \Rightarrow (x^1 = x^2)$

ZF_thm25

$\vdash \forall A \cdot$

$A \varnothing \land (\forall x \cdot N x \land A x \Rightarrow A(\text{suc } x)) \Rightarrow$

$(\forall x \cdot N x \Rightarrow A x)$
THE THEORY of "PURE FUNCTIONS"

Types -- ":PF"

Constants --

function_hereditary ":(SET → bool) → bool"
pure_function ":SET → bool"
Ω_f ":PF" ⊥_f ":PF" ∪_f ":PF → PF"
λ_f ":PF → ((PF → PF) → PF)" Π_f ":PF → PF"

Curried Infixes --

f ":PF → (PF → PF)"
_→_f ":PF → (PF → PF)"
⊕_f ":PF → (PF → PF)"

Definitions --

function_hereditary_DEF
|– function_hereditary p =
   (∀f• function f ∧
    ⊥_z ∈_z (image f) ∧
    (∀x• x ∈_z (field_z f) ⇒ p x) ⇒
    p f)
pure_function_DEF
|– pure_function s = (∀p• function_hereditary p ⇒ p s)
AXIOMS of PURE FUNCTION THEORY

Of the three main sorts of axiom:

1 LOGICAL axioms (including =)

    these remain UNCHANGED

2 axioms CHARACTERISING SETS

    These are replaced by comparable axioms for PURE
    FUNCTIONS (extensionality, well foundedness)

3 axioms CHARACTERISING ABSTRACTION

    The axiom of separation is replaced by an axiom of
    BETA REDUCTION

4 POPULATING axioms

    These are changed in detail but play a similar role.
THEOREMS concerning PURE FUNCTIONS

PF1  \[ \forall x \ y \cdot (x = y) \iff (\forall z \cdot x_f z = y_f z) \]
PF2  \[ \forall d \ f \ z \cdot (\lambda \cdot d \ f) f z = ((d_f z = \bot_f) \Rightarrow \bot_f | f z) \]
PF3  \[ \forall x \cdot \Omega_f x = \bot_f \]
PF4  \[ \forall f \ z \cdot (\Pi f) f z = \]
   \[ (\forall g \cdot ((f_f g = \bot_f) \Rightarrow (z_f g = \bot_f) | (f_f g)_f (z_f g) \neq \bot_f) \Rightarrow T_f | \bot_f) \]
PF5  \[ \forall x y z \cdot (x |_f y)_f z = ((z = x) \Rightarrow y | \bot_f) \]
PF6  \[ \forall x y z \cdot (x \oplus_f y)_f z = \]
   \[ ((y_f z = \bot_f) \Rightarrow x_f z | y_f z) \]
PF7  \[ \forall p \cdot \]
   \[ (\forall q \cdot (\forall r \cdot ((f_e q)_f r) \neq \bot_f \Rightarrow p r) \Rightarrow p q) \Rightarrow (\forall q \cdot p q) \]
PF11 \[ \forall f g \cdot \]
   \[ (((\cup_f f) f g) \neq \bot_f) \Rightarrow \]
   \[ (\exists i \cdot \]
   \[ (f_f i) \neq \bot_f \land (i_f g = (\cup_f f)_f g)) | \]
   \[ (\forall i \cdot (f_f i) \neq \bot_f \Rightarrow (i_f g = \bot_f)) \]
PF13 \[ \neg (\bot_f = T_f) \]
The Theory ppf136
Types -- ":PPF"

Constants --

regular ":(PPF \rightarrow PPF) \rightarrow \text{bool}"
\lambda_p ":PPF \rightarrow ((PPF \rightarrow PPF) \rightarrow PPF)"
\Omega_p ":PPF" \perp_p ":PPF"
T_p ":PPF" F_p ":PPF"
if_p ":PPF \rightarrow (PPF \rightarrow (PPF \rightarrow PPF))"
\cup_p ":PPF \rightarrow PPF" \Pi_p ":PPF \rightarrow PPF"
Tv_p ":PPF \rightarrow PPF" Te_p ":PPF"

Curried Infixes --

p ":PPF \rightarrow (PPF \rightarrow PPF)"
\rightarrow_p ":PPF \rightarrow (PPF \rightarrow PPF)"
\oplus_p ":PPF \rightarrow (PPF \rightarrow PPF)"
==_p ":PPF \rightarrow (PPF \rightarrow PPF)"
§_p ":PPF \rightarrow (PPF \rightarrow PPF)"
the AXIOMATISATION of POLYMORPHIC PURE FUNCTIONS

Of the three main sorts of axiom:

1 LOGICAL axioms (including =)

The host logic (HOL) no longer supplies adequate machinery. Equality and conditionals need to be redefined (if_p, ==_p).

2 axioms CHARACTERISING FUNCTIONS

These are replaced by comparable axioms for POLYMORPHIC PURE FUNCTIONS (extensionality, well foundedness)

3 the axiom of BETA REDUCTION

A variant of BETA REDUCTION is introduced dependent on REGULARITY, TYPE INSTANTIATION is introduced, supported by an analogous axiom.

4 POPULATING axioms

These are changed in detail but play a similar role.
Definitions --

regular

\[ \text{regular ppfun} = (\forall pf \cdot \exists pfun \cdot \forall ppf \cdot \ REP_{PPF}(ppfun ppf)pf = pfun(\text{REP}_{PPF} ppf pf)) \]

Theorems --

PPF1  \[ \vdash \forall x y \cdot (x = y) \iff (\forall z \cdot x \ p \ z = y \ p \ z) \]

PPF2  \[ \vdash \forall d \ m \cdot \text{regular m} \Rightarrow \\
(\forall z \cdot (\lambda p \ d m) \ p \ z = \\
\text{if}_p ((d \ p \ z) == \ p \bot) \linebreak \ p \bot (m \ z)) \]

PPF3  \[ \vdash \forall p \cdot \Omega p \ p \ p = \bot \]

PPF5  \[ \vdash \forall x y z \cdot (x \to y) \ p \ z = \text{if}_p ((z == \ p \ x) \ y) \ \bot \]

PPF6  \[ \vdash \forall x y z \cdot (x \oplus y) \ p \ z = \\
\text{if}_p ((y \ p \ z) == \ p \bot) (x \ p \ z)(y \ p \ z) \]

PPF13 \[ \vdash \neg (\bot = T) \]
the AXIOMATISATION of
STRUCTURED POLYMORPHIC FUNCTIONS

Of the three main sorts of axiom:

1 LOGICAL axioms (including =)

The host logic (HOL) no longer supplies adequate machinery. Equality and conditionals need to be redefined (if$_s$, ==$_s$), host language abstraction is now displaced.

2 axioms CHARACTERISING POLYMORPHIC FUNCTIONS

These are replaced by comparable axioms for STRUCTURED POLYMORPHIC FUNCTIONS (extensionality, well foundedness)

3 the axiom of BETA REDUCTION

The REGULARITY CLAUSE IN the axiom of BETA REDUCTION is ELIMINATED, VALUE INSTANTIATION is introduced, supported by an analogous axiom.

4 POPULATING axioms

These are changed in detail but play a similar role.
The Theory ppf137

Types -- ":SPF"

Constants --

\[ \lambda_s \quad ":SPF \rightarrow (SPF \rightarrow (SPF \rightarrow SPF))" \]
\[ \text{Iv}_s \quad ":SPF \rightarrow SPF" \]
\[ \text{Ie}_s \quad ":SPF" \quad \text{Tv}_s \quad ":SPF \rightarrow SPF" \quad \text{Te}_s \quad ":SPF" \]
\[ \Omega_s \quad ":SPF" \quad \bot_s \quad ":SPF" \quad T_s \quad ":SPF" \]
\[ \text{if}_s \quad ":SPF \rightarrow (SPF \rightarrow (SPF \rightarrow SPF))" \]
\[ \cup_s \quad ":SPF \rightarrow SPF" \]
\[ \Pi_s \quad ":SPF \rightarrow SPF" \]

Curried Infixes --

\[ s \quad ":SPF \rightarrow (SPF \rightarrow SPF)" \]
\[ §§_s \quad ":SPF \rightarrow (SPF \rightarrow SPF)" \]
\[ §_s \quad ":SPF \rightarrow (SPF \rightarrow SPF)" \]
\[ |\rightarrow_s \quad ":SPF \rightarrow (SPF \rightarrow SPF)" \]
\[ \oplus_s \quad ":SPF \rightarrow (SPF \rightarrow SPF)" \]
\[ ==_s \quad ":SPF \rightarrow (SPF \rightarrow SPF)" \]

Theorems --

\[ \text{SPF1} \quad \vdash \forall x \ y \bullet (x = y) \leftrightarrow (\forall z \bullet x \ s \ z = y \ s \ z) \]
\[ \text{SPF13} \quad \vdash \neg (\bot_s = T_s) \]
CONCLUSIONS

FORMAL DERIVATION of PROOF RULES

is

DESIRABLE

and

FEASIBLE

but

EXPENSIVE

(c300 terminal hours so far)
WHAT I WOULD DO DIFFERENT IF I STARTED AGAIN FROM SCRATCH

DIFFERENT formulation of SET THEORY

THEN

NOT PURE FUNCTIONS

but

PURE FUNCTIONS and ‘TYPES’
without ⊥

this is mathematically nicer
and provides a staging post towards
PURE FUNCTORS and CATEGORIES