

# The MMT Perspective on Conservativity

Florian Rabe

Jacobs University Bremen

September 2017

## Basic Definitions

For the purposes of this talk, a logic  $L$  consists of

- ▶ theories  $T$
- ▶  $T$ -formulas  $F : prop$
- ▶  $T$ -proofs and provability judgment  $\vdash_T F$
- ▶  $T$ -models  $M$  and satisfaction  $M \models_T F$

Theories are lists of symbol declarations.

types, functions, predicates, axioms, proof rules, rewrite rules, ...

$S \hookrightarrow T$  is a theory **extension** if  $S$ -declarations  $\subseteq T$ -declarations

## Two Conflicting Definitions of Conservativity

Intuition:  $S \hookrightarrow T$  is **conservative** if  $T$ -semantics does not substantially differ from  $S$ -semantics.

e.g.,  $T$  adds only definitions, theorems, admissible rules, ...

Problem: How to define that rigorously?

Two answers for “When is  $S \hookrightarrow T$  conservative?”:

- ▶ **proof theorist**: for any  $T$ -proof of  $S$ -formula  $F$ , there is an  $S$ -proof of  $F$ 
  - proof retraction
  - mentions only proofs, no models
- ▶ **model theorist**: for any  $S$ -model  $M$ , there is a  $T$ -model  $M'$  that agrees with  $M$  on  $S$ -symbols
  - model extension
  - mentions only models, no proofs

## Tension between the Definitions

Are proofs or models primary for semantics?

practical, social, and philosophical difference

Not so unusual — compare “When is  $F$  a theorem?”

- ▶ proof theorist: if there is a proof of  $F$
- ▶ model theorist: if  $F$  holds in all models
- ▶ both are equivalent via soundness/completeness  
achieved by fine-tuning the definitions

Ideally, model and proof-theoretical conservativity also equivalent.

(they aren't)

## Relating the Definitions

Theorem:

If  $L$  is sound and complete, then  
model-conservative implies proof-conservative.

But not the other way around.

causes confusion at best, conflict at worst

## Motivation

### Vision: UniFormal

a universal framework for the formal representation of knowledge

- ▶ integrate all domains
  - model theory, proof theory, computation, mathematics, ...
- ▶ be independent of foundational languages
  - logics, programming languages, foundations of mathematics, ...
- ▶ build generic, reusable implementations
  - type checker, module system, library manager, IDE, ...

### My (evolving) solution: MMT

- ▶ a uniformal knowledge representation framework
  - developed since 2006, ~ 100,000 loc, ~ 500 pages of publications
- ▶ allows foundation-independent solutions
  - module system, type reconstruction, theorem proving, ...
  - IDE, search, build system, library, ...

<http://uniformal.github.io/>

## Foundation-Independent Development

### Foundation-specific workflow (almost all systems)

1. choose foundation  
type theories, set theories, first-order logics, higher-order logics, ...
2. implement kernel
3. develop support algorithms, tools reconstruction, proving, IDE, ...
4. build library

### Foundation-independent workflow (MMT)

1. MMT provides generic kernel  
no built-in bias towards any foundation
2. develop generic support on top of MMT
3. flexibly customize MMT for desired foundation(s)
4. build multi-foundation universal library

## Advantages of Foundation-Independence

- ▶ Avoids segregation into mutually incompatible systems
- ▶ Allows maximally general results  
meta-theorems, algorithms, formalizations
- ▶ Separation of concerns between
  - ▶ foundation developers
  - ▶ support service developers: search, axiom selection, . . .
  - ▶ application developers: IDE, proof assistant, . . .
- ▶ Rapid prototyping for logic systems
- ▶ Allows evolving and experimenting with foundations

But how much can be done foundation-independently?

surprisingly much — this talk: conservativity



## Logical Frameworks and Syntax

Logical framework LF in MMT

```

theory LF {
  type
  Pi      #  $\Pi V1 . 2$                                 name[: type][#notation]
  arrow   #  $1 \rightarrow 2$ 
  lambda  #  $\lambda V1 . 2$ 
  apply   #  $1\ 2$ 
}

```

Logics in MMT/LF

```

theory Logic : LF {
  prop : type
  ded   : prop  $\rightarrow$  type #  $\vdash 1$                                 judgments-as-types
}
theory FOLSyn : LF {
  include Logic
  term      : type                                higher-order abstract syntax
  forall    : (term  $\rightarrow$  prop)  $\rightarrow$  prop #  $\forall V1 . 2$ 
}

```

## Proof Theory

FOLSyn from previous slide:

```

theory FOLSyn: LF {
  include Logic
  term      : type
  forall    : (term → prop) → prop #  ∀ V1 . 2
}
  
```

Proof-theory = syntax + calculus

```

theory FOL: LF {
  include FOLSyn
  forallIntro : ΠF:term→prop .
                (Πx:term.⊢ (F x)) → ⊢ ∀(λx:term.F x)
  forallElim  : ΠF:term→prop .
                ⊢ ∀(λx:term.F x) → Πx:term.⊢ (F x)
}
  
```

rules are constants

## Domain Theories

FOLSyn from previous slide:

```

theory FOLSyn: LF {
  include Logic
  term      : type
  forall    : (term → prop) → prop #  ∀ V1 . 2
}

```

Algebraic theories in MMT/LF/FOL:

```

theory Magma: FOL {
  comp : term → term → term # 1 o 2
}
theory SemiGroup: FO {
  include Magma
  associative : ⊢ ∀ x,y,z. (x o y) o z = x o (y o z)
}

```

## MMT Theory Morphisms (highly simplified)

An MMT **theory** is a list of declarations  $c[: E]$ , where  $E$  is an expression using the previous symbols.

An MMT theory **morphism**  $m : S \rightarrow T$  maps every  $S$ -symbol to a  $T$ -expression such that  
if  $\vdash_S A : B$  then  $\vdash_T m(A) : m(B)$       preservation of typing/truth

## Model Theory

Universe = set theory, category theory, programming languages, ...

```

theory ZFC: LF {
  set      : type
  prop     : type
  in       : set → set → prop # 1 ∈ 2
  equal    : set → set → prop # 1 = 2
  ded      : prop → type # ⊢ 1
  ...
  bool    : set = {0,1}
  ...
}

```

Interpretation = theory morphism from syntax+calculus to semantics

```

morphism FOLMod : FOL → ZFC {
  prop ↦ bool
  ded  ↦ λx∈bool.x=1
  ...
}

```

(proof rules mapped to their soundness proofs)

## Individual Models

FOLMod from previous slide:

```
morphism FOLMod : FOL → ZFC {  
  prop  ↦ bool  
  ded   ↦ λx∈bool . x=1  
  ...  
}
```

Integer addition as a model of SemiGroup:

```
morphism IntegerAddition : SemiGroup → ZFC {  
  include FOLMod  
  term   ↦ ℤ  
  comp   ↦ +  
  assoc  ↦ ... (proof that + is associative)  
}
```

## Derivable and Admissible Rules

Consider an extension  $S \hookrightarrow T$  in MMT.

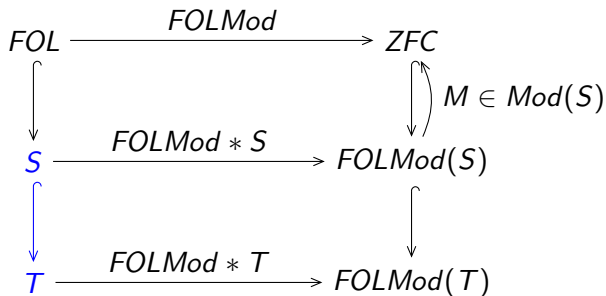
**Example:**  $T = S$ ,  $cut : R$

$S$  is a cut-free sequent calculus and  $R$  is the cut rule.

We say that  $S \hookrightarrow T$  is

- ▶ **derivable** if
  - ▶ example case: there is a term  $r : R$  over  $S$
  - ▶ general case: there is a retraction morphism  $r : T \rightarrow S$
- ▶  **$\vdash$ -admissible** if  $\vdash F$  is inhabited over  $S$  whenever it is inhabited over  $T$

# Conservative as a Special Case of Derivable/Admissible

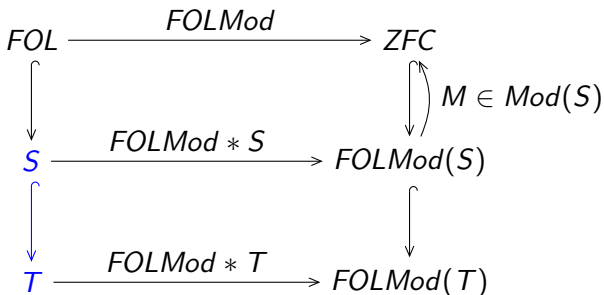


$FOLMod(S)$

- ▶ pushout of  $L \hookrightarrow S$  along  $FOLMod$
- ▶ obtained by homomorphic translation of  $S$ -declarations



# Conservative as a Special Case of Derivable/Admissible



Theorem:  $S \leftrightarrow T$  is

- ▶ proof-conservative iff  $S \leftrightarrow T$  is  $\vdash$ -admissible
- ▶ model-conservative iff  $\text{FOLMod}(S) \leftrightarrow \text{FOLMod}(T)$  is derivable

## Different Kinds of Conservativity

General case: 4 notions of conservativity

$$\begin{array}{ccc}
 S & \xrightarrow{FOLMod * S} & FOLMod(S) \\
 \downarrow & & \downarrow \\
 T & \xrightarrow{FOLMod * T} & FOLMod(T)
 \end{array}$$

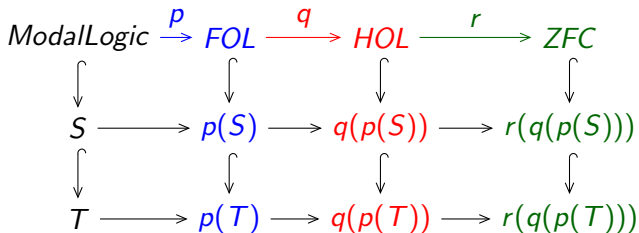
- ▶  $S \hookrightarrow T$  is  $\vdash$ -admissible proof-conservative
- ▶  $S \hookrightarrow T$  is derivable
- ▶  $FOLMod(S) \hookrightarrow FOLMod(T)$  is derivable model-conservative
- ▶  $FOLMod(S) \hookrightarrow FOLMod(T)$  is  $\vdash$ -admissible

## Relating the Different Kinds of Conservativity

- ▶  $S \leftrightarrow T$  is derivable
  - ▶ syntax has witness for conservativity
  - ▶ minimal/strongest reasonable definition
- ▶  $S \leftrightarrow T$  is  $\vdash$ -admissible proof-conservative
  - ▶ syntax has no counter-example for conservativity
  - ▶ maximal/weakest reasonable definition
- ▶  $FOLMod(S) \leftrightarrow FOLMod(T)$  is derivable model-conservative
  - ▶ semantics has witness for conservativity
  - ▶ in between the above
- ▶  $FOLMod(S) \leftrightarrow FOLMod(T)$  is  $\vdash$ -admissible
  - ▶ equivalent to proof-conservative for sound and complete logics

## Conservativity under Refinement of Semantics

Refinement chain of multiple interpretations, e.g.,



At each step, 2 notions of conservativity of  $S \hookrightarrow T$ :

- ▶ using  $\vdash$ -admissibility:
  - ▶ all notions equivalent for sound+complete interpretations
  - ▶ strongest possible notion                      **proof-conservativity (absolute)**
- ▶ Using derivability:    **model-conservativity relative to semantics**
  - ▶ notions grow weaker as semantics is more refined
  - ▶ converges to proof-conservativity for increasing refinements

## Summary

- ▶ MMT: foundation-independent framework for formal systems  
maximally general conceptualizations, theorems, implementations
- ▶ Allows resolving conflict between notions of conservativity  
results apply to arbitrary logic defined in arbitrary logical framework
- ▶ Proof-conservativity
  - ▶ corresponds to  $\vdash$ -admissibility of rules
  - ▶ weakest possible notion
- ▶ Model-conservativity
  - ▶ corresponds to derivability of rules
  - ▶ relative to chosen model theory
  - ▶ strongest possible notion if applied to initial semantics
  - ▶ grows weaker as semantics is more refined
  - ▶ converges against proof-conservativity

# Terms

Abstract syntax

contexts	$\Gamma$	$::=$	$(x[: E][= E])^*$
terms	$E$	$::=$	
constants			$c$
variables			$x$
complex terms			$c(\Gamma; E^*)$

Complex term examples

- ▶ typical operators:  $\Gamma$  empty      e.g., `apply(·; f, a)` for  $(f\ a)$
- ▶ typical binders:  $\Gamma$  and  $\vec{E}$  have length 1  
e.g., `lambda(x:A; t)` for  $\lambda x:A.t$

Judgments relative to a theory  $T$  that declares the constants  $c$

$\Gamma \vdash_T t : E$	$t$ has type $E$
$\Gamma \vdash_T E = E'$	$E$ and $E'$ are equal
$\Gamma \vdash_T \_ : E$	$E$ is inhabitable