A Foundational View on Integration Problems

Florian Rabe[1], Michael Kohlhase[1], Claudio Sacerdoti Coen[2]

[1] Computer Science, Jacobs University, Bremen (DE)
[2] Department of Computer Science, University of Bologna (IT)

Motivation

- Computer algebra systems, deduction systems, MKM systems are becoming more and more powerful

  How can we make them work together?
- Avoid duplication of efforts
- Let systems and developers specialize
- Overall gain for developers and users

A Basic System Integration Work Flow

1. We have a problem in System 1
2. We send it to System 2 (e.g., via Content MathML)
3. System 2 finds a solution
4. We send the solution back to System 1

For example,

Problem	Solution
proof goal	proof (in practice often only: "yes")
expression	simplified/decomposed expression
formula with free variables	(set of) substitution(s)

A Basic System Integration Work Flow

1. We have a problem in System 1
2. We send it to System 2 (e.g., via Content MathML)
3. System 2 finds a solution
4. We send the solution back to System 1

For example,

Problem	Solution
proof goal	proof (in practice often only: "yes")
expression	simplified/decomposed expression
formula with free variables	(set of) substitution(s)

Key challenge: make sure that System 1 and System 2 agree on the semantics of problem and solution

The Formality Spectrum of System Integration

1) The pragmatic approach

- Slogan: “send problem/solution and hope for the best”
  - works well if the semantics is clear: literals, finite collections, first-order formulas, . . .
  - gets unreliable fast: partial functions, side conditions in analysis, any other logic, . . .

  ambiguity already with $0 \in N$ or with x/x

- Key method: semi-formal specification of the System 1-System 2 interface
- Standardized through content dictionaries
  symbol N in OpenMath CD *setname1* is natural numbers with 0

The Formality Spectrum of System Integration

2) The fundamentalist approach — our work

- Slogan: "prove everything and hope you'll ever have the time to get a running system"
- expensive but then works perfectly
- requires formalizing semantics of systems and their relation

Classifying Fundamentalist Approaches (1)

When does integration happen?

- a priori: translate a whole library to a different system
  forward translation run once by developer
- on-demand: translate individual problems — our work
  forward and backward translation run automatically

Examples:

- a priori
  - using HOL in Nuprl, Schürmann, Stehr, 2004
  - using Isabelle/HOL in HOL Light, McLaughlin, 2006
- on-demand
  - using first-order logic in Isabelle, Meng, Paulson, 2008
  - using first-order logic in SUMO, Trac, Sutcliffe, Pease, 2008

Classifying Fundamentalist Approaches (2)

When is the integration verified?

- dynamically
  - solution-providing system is unconstrained
  - solution-requesting system verifies the solution
  - key advantage: no trust in the providing system of the communication needed
- statically — our work
  - define both systems in a meta-language
  - formalize systems and translations between them
  - prove correctness
  - key advantage: no communication of proofs needed

Examples:

- dynamically: using Maple in HOL Light, Harrison, Thery, 1998
- statically: using first-order logic in modal logic, Hustadt, Schmidt, 2000

Classifying Fundamentalist Approaches (3)

How is the static integration verified?

- on paper using semi-formal mathematics, using
  - an ad hoc argument
  - an argument within a (usually categorical) framework such as institutions, fibrations
- mechanically in a deduction system — our work
  typically, based on type theory as in LF, Coq, Isabelle

Examples:

- on paper, ad hoc: using Isabelle/HOL in Isabelle/ZF, Krauss, Schropp, 2010
- on paper, with framework: integrating logics in the Hets system, Mossakowski et al., 2007
- mechanized: using HOL in Nuprl
- mechanized: LATIN logic integrator, recall this morning's talk

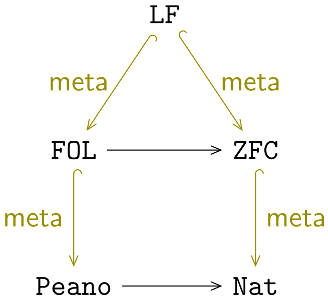
Our Frameworks of Choice: MMT + LF/Twelf

- MMT: module system for mathematical theories, Rabe, Kohlhase 2008
  generic declarative language based on OMDoc/OpenMath
- LF: Harper, Honsell, Plotkin, 1993
  logical framework based on dependent type theory
- Twelf: Pfenning, Schürmann, 1999
  mechanization of LF

Division of labor:

- MMT provides the global semantics: theory graphs, module system, scalable MKM framework
- LF/Twelf provide the local semantics: type reconstruction, proof checking, adequate encodings

Our Frameworks of Choice: MMT + LF/Twelf




```
form: type
proof: form → type
impl: form → form → form
modus_ponens:
  proof (A impl B) →
      proof A → proof B
```

Division of labor:

- MMT provides the global semantics: theory graphs, module system, scalable MKM framework
- LF/Twelf provide the local semantics: type reconstruction, proof checking, adequate encodings

Static Verification in MMT (ideally)

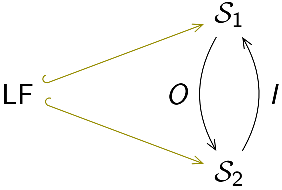
1. Define an MMT theory M for the meta-language M (e.g., LF)

   M provides semantics, e.g., type- and proof-checking

2. Represent System 1 and System 2 as MMT-theories $\mathcal{S}_1$, $\mathcal{S}_2$ with meta-theory M

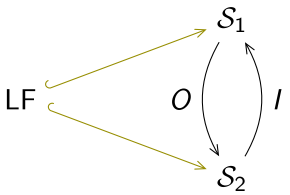
   $\mathcal{S}_i$ contains, e.g., symbol $\vdash_i$ for truth judgment

3. Give mutually inverse M-theory morphisms $I : \mathcal{S}_2 \to \mathcal{S}_1$ and $O : \mathcal{S}_1 \to \mathcal{S}_2$



Static Verification in MMT (ideally)

- Given a proof goal $\vdash_2 F$ in System 2
  1. translate it to $\vdash_1 I(F)$ in System 1,
  2. find a proof $\vdash_1 p : I(F)$ in System 1
  3. translate it back yielding $\vdash_2 O(p) : O(I(F)) = F$
- Static verification: valid theory morphism O preserves judgment $\vdash_1 p : I(F)$
- Mechanical verification: validity of O is verified by MMT+Twelf



Problem: This is really difficult

1. Representing systems in M is hard
   - need to represent syntax and semantics
   - need to show adequacy of representation

     assuming the semantics is documented
   - good progress in LATIN
2. Giving theory morphisms I and O is even harder
   - need to translate syntax and semantics
   - ongoing work in LATIN

Problem: This is really difficult

1. Representing systems in M is hard
   - need to represent syntax and semantics
   - need to show adequacy of representation

     assuming the semantics is documented
   - good progress in LATIN
2. Giving theory morphisms I and O is even harder
   - need to translate syntax and semantics
   - ongoing work in LATIN
3. But even then: mismatch of libraries

Classifying Fundamentalist Approaches (4)

- Integration is most interesting if there are big libraries
- But: system libraries use different concrete formalizations of the same abstract concept
  e.g., natural numbers N_i in $\mathcal{S}_i$, and $O(N_1) \neq N_2$
- How does the integration relate, e.g., $O(N_1)$ and N_2?
  - not at all
  - isomorphism theorems established individually: e.g., $O(N_1) \cong N_2$
  - ad hoc correspondence of symbols, e.g., $N_1 \sim N_2$
    translation can yield (only) proof sketches
  - formal framework
    our work

Filtering in MMT

- theory morphisms may be partial

theory A	theory B	morphism $\mu : A \to B$
s : *type*	t : *type*	$s \mapsto t$
c : s		**filter** c

Filtering in MMT

- theory morphisms may be partial
- partiality is strict, i.e., propagates along the dependency relation

theory A	theory B	morphism $\mu : A \to B$
$s : type$	$t : type$	$s \mapsto t$
$c : s$		**filter** c
$c' := c$		necessarily: **filter** c'

Filtering in MMT

- theory morphisms may be partial
- partiality is strict, i.e., propagates along the dependency relation
- key new idea: controlled relaxation of propagation

theory A	theory B	morphism $\mu : A \to B$
$s : type$	$t : type$	$s \mapsto t$
$c : s$		**filter** c
$c' := c$		necessarily: **filter** c'

Filtering in MMT

- theory morphisms may be partial
- partiality is strict, i.e., propagates along the dependency relation
- key new idea: controlled relaxation of propagation

theory A	theory B	morphism $\mu : A \to B$
$s : type$	$t : type$	$s \mapsto t$
$c : s$		**filter** c
$c' := c$		necessarily: **filter** c'
	$d : t$	

Filtering in MMT

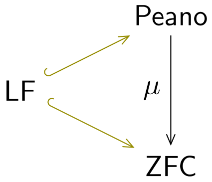
- theory morphisms may be partial
- partiality is strict, i.e., propagates along the dependency relation
- key new idea: controlled relaxation of propagation

theory A	theory B	morphism $\mu : A \to B$
$s : type$	$t : type$	$s \mapsto t$
$c : s$		**filter** c
$c' := c$		necessarily: **filter** c'
	$d : t$	possibly: $c' \mapsto d$

Filtering: Example

- Peano: MMT theory with axiomatic presentation of natural numbers
- ZFC: MMT theory with a concrete definition for them
- μ: (total) theory morphism that proves ZFC realizes Peano

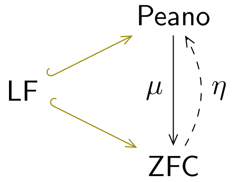
Peano	ZFC	μ
	$\varnothing$, $\cup$, etc.	
0	$0 := \varnothing$	$0 \mapsto 0$
succ	$\mathtt{succ}(n) := n \cup \{n\}$	$\mathit{succ} \mapsto \mathtt{succ}$
$\mathit{nocycle} : 0 \neq \mathit{succ}(X)$	$\mathtt{nocycle} :=$ [PROOF]	$\mathit{nocycle} \mapsto \mathtt{nocycle}$



Filtering: Example

- Peano: MMT theory with axiomatic presentation of natural numbers
- ZFC: MMT theory with a concrete definition for them
- μ: (total) theory morphism that proves ZFC realizes Peano

Peano	ZFC	μ
	$\emptyset$, $\cup$, etc.	
0	$0 := \emptyset$	$0 \mapsto 0$
succ	$\texttt{succ}(n) := n \cup \{n\}$	$\mathit{succ} \mapsto \texttt{succ}$
$\mathit{nocycle} : 0 \neq \mathit{succ}(X)$	$\texttt{nocycle} :=$ [PROOF]	$\mathit{nocycle} \mapsto \texttt{nocycle}$



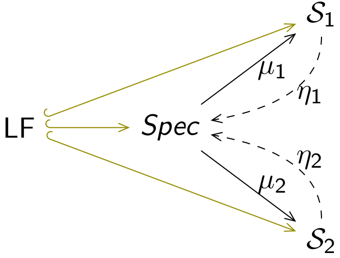
η: partial theory morphism that inverts μ

filter $\emptyset$, **filter** $\cup$,

$0 \mapsto 0$, $\texttt{succ} \mapsto \mathit{succ}$, $\texttt{nocycle} \mapsto \mathit{nocycle}$

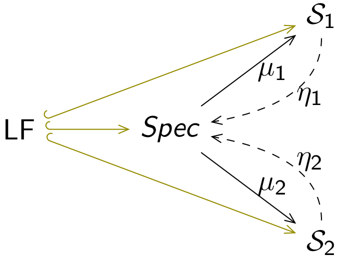
Integration by Filtering

- *Spec*: specification of the abstract concepts
  e.g., axiomatic presentation of the natural numbers
- $\mathcal{S}_i$: two concrete definitions of *Spec*
  e.g., natural numbers in ZFC and in Coq
- μ_i: theory morphism that proves $\mathcal{S}_i$ realizes *Spec*
- η_i: partial theory morphism that inverts μ_i



Integration by Filtering

- *Spec*: specification of the abstract concepts
  e.g., axiomatic presentation of the natural numbers
- $\mathcal{S}_i$: two concrete definitions of *Spec*
  e.g., natural numbers in ZFC and in Coq
- μ_i: theory morphism that proves $\mathcal{S}_i$ realizes *Spec*
- η_i: partial theory morphism that inverts μ_i



mediating morphisms now definable:
$I : \mathcal{S}_2 \to \mathcal{S}_1 \;=\; \mu_2 \circ \eta_1$
$O : \mathcal{S}_1 \to \mathcal{S}_2 \;=\; \mu_1 \circ \eta_2$
MMT guarantees truth-preservation along I, O whenever defined

Conclusion

- Filtering with relaxed propagation
  - technically, a minor change in MMT
  - pragmatically, a major step forward for applications in LATIN
- Does not cover all integration challenges, but a lot
  e.g., we can now finish our Mizar $\rightarrow$ ZFC translation in LF
- Implementation
  - adaptation in MMT finished
  - integration with Twelf pending