Heterogeneous proofs —
the local and the global view

Till Mossakowski
BISS, Universität Bremen, Germany

IFIP 1.3 meeting, 24th Sept. 2002
Motivation

- Multi-logic specifications are needed, since complex problems have different aspects that are best specified in different logics.
- A combination of all the used logics would become too complex in many cases.
- Different approaches can be compared.
- Formal interoperability among languages and tools.
- We aim at heterogeneous theorem proving ⇒ able to exploit the power of specialized proof tools.
Comparison with logic combination

Compared with logic combination, heterogeneous specification (and proofs)

• has only weaker forms of feature interaction, but
• is easier, more flexible and wider applicable (one metaformalism for all logics), and
• supports better re-use of existing proof tools (no need to implement new calculi).
Institutions, logics, and (co)morphisms

- **Institutions** (Goguen, Burstall 1984) capture abstract model theory
- **Logics** (Meseguer 1987) also capture proof theory
- **Institution/logic morphisms** (Goguen, Burstall 1984) capture the intuition that one logic is built upon another one
- **Institution/logic comorphisms** (Meseguer 1987) capture the intuition that one logic is encoded into another one
Institutions

Signatures

$\Sigma \xrightarrow{\sigma} \Sigma'$

Sentences

$\text{Sen } \Sigma \xrightarrow{\text{Sen } \sigma} \text{Sen } \Sigma'$

Satisfaction

$|=_{\Sigma} \xrightarrow{} |=_{\Sigma'}$

Models

$\text{Mod } \Sigma \xrightarrow{\text{Mod } \sigma} \text{Mod } \Sigma'$
Structured specifications over an arbitrary institution

\[ SP ::= \langle \Sigma, \Gamma \rangle \mid SP \cup SP \mid \sigma(SP) \mid \sigma^{-1}(SP) \]
... and their semantics

\[ \text{Sig}(\langle \Sigma, \Gamma \rangle) = \Sigma \]
\[ \text{Mod}(\langle \Sigma, \Gamma \rangle) = \{ M \in \text{Mod}(\Sigma) \mid M \models \Gamma \} \]

\[ \text{Sig}(SP_1 \cup SP_2) = \text{Sig}(SP_1) = \text{Sig}(SP_2) \]
\[ \text{Mod}(SP_1 \cup SP_2) = \text{Mod}(SP_1) \cap \text{Mod}(SP_2) \]

\[ \text{Sig}(\sigma : \Sigma_1 \longrightarrow \Sigma_2(SP)) = \Sigma_2 \]
\[ \text{Mod}(\sigma(SP)) = \{ M \in \text{Mod}(\Sigma_2) \mid M|_{\sigma} \in \text{Mod}(SP) \} \]

\[ \text{Sig}((\sigma : \Sigma_1 \longrightarrow \Sigma_2)^{-1}(SP)) = \Sigma_1 \]
\[ \text{Mod}((\sigma : \Sigma_1 \longrightarrow \Sigma_2)^{-1}(SP)) = \{ M|_{\sigma} \mid M \in \text{Mod}(SP) \} \]
Proof calculus for entailment (Borzyszkowski)

\[(CR) \quad \frac{\{SP \vdash \varphi_i\}_{i \in I} \quad \{\varphi_i\}_{i \in I} \vdash \varphi}{SP \vdash \varphi}\]

\[(basic) \quad \frac{\varphi \in \Gamma}{\langle \Sigma, \Gamma \rangle \vdash \varphi}\]

\[(sum1) \quad \frac{SP_1 \vdash \varphi}{SP_1 \cup SP_2 \vdash \varphi}\]

\[(sum1) \quad \frac{SP_1 \vdash \varphi}{SP_1 \cup SP_2 \vdash \varphi}\]

\[(trans) \quad \frac{SP \vdash \varphi}{\sigma(SP) \vdash \sigma(\varphi)}\]

\[(derive) \quad \frac{SP \vdash \sigma(\varphi)}{\sigma^{-1}(SP) \vdash \varphi}\]
Proof calculus for refinement (Borzyszkowski)

\[
\begin{align*}
\text{(Basic)} \quad SP \vdash \Gamma & \quad \frac{SP \vdash \Gamma}{\langle \Sigma, \Gamma \rangle \rightsquigarrow SP} \\
\text{(Sum)} \quad SP_1 \rightsquigarrow SP & \quad SP_2 \rightsquigarrow SP \quad \frac{SP_1 \cup SP_2 \rightsquigarrow SP}{SP_1 \cup SP_2 \rightsquigarrow SP} \\
\text{(Trans}_1 \text{)} \quad SP \rightsquigarrow \theta(SP') & \quad \theta = \sigma^{-1} \quad \frac{SP \rightsquigarrow \theta(SP')}{\sigma(SP) \rightsquigarrow SP'} \\
\text{(Trans}_2 \text{)} \quad SP \rightsquigarrow \sigma^{-1}(SP') & \quad \frac{SP \rightsquigarrow \sigma^{-1}(SP')}{\sigma(SP) \rightsquigarrow SP'} \\
\text{(Derive)} \quad SP \rightsquigarrow SP'' & \quad \frac{SP \rightsquigarrow SP''}{\sigma^{-1}(SP) \rightsquigarrow SP'} \quad \text{if } \sigma: SP' \longrightarrow SP'' \quad \text{is a conservative extension} \\
\text{(Trans-equiv)} \quad \theta(\sigma(SP)) \rightsquigarrow SP' & \quad \frac{\theta(\sigma(SP)) \rightsquigarrow SP'}{\theta \circ \sigma(SP) \rightsquigarrow SP'}
\end{align*}
\]
Soundness and Completeness

Under the assumptions that

- the institution has the Craig interpolation property,
- the institution admits weak amalgamation, and
- the logic is complete,

the calculus for structured entailment and refinement is sound and complete.

Note that for refinement, an oracle for conservative extensions is needed.

Problem: Craig interpolation often fails!
Development graphs . . .

• . . . admit easy formulation of a “global” calculus (using normal form signatures, while keeping the structure) ⇒ Craig interpolation not needed
• . . . allow easy treatment shared sub-specifications (e.g. different instances of parameterized specifications)
• . . . allow a management of change
• . . . have been used in industrial-scale applications
Development graphs $\mathcal{S} = \langle \mathcal{N}, \mathcal{L} \rangle$

Nodes in $\mathcal{N}$: $(\Sigma^N, \Gamma^N)$ with

- $\Sigma^N$ signature,
- $\Gamma^N \subseteq \text{Sen}(\Sigma^N)$ set of local axioms.

Links in $\mathcal{L}$:

- **global** $M \xrightarrow{\sigma} N$, where $\sigma : \Sigma^M \rightarrow \Sigma^N$,
- **local** $M \xrightarrow{\sigma} N$ where $\sigma : \Sigma^M \rightarrow \Sigma^N$, or
- **hiding** $M \xrightarrow{\sigma} N$ where $\sigma : \Sigma^N \rightarrow \Sigma^M$

going against the direction of the link.
Semantics of development graphs

$\text{Mod}_S(N)$ consists of those $\Sigma^N$-models $n$ for which

1. $n$ satisfies the local axioms $\Gamma^N$,

2. for each $K \xrightarrow{\sigma} N \in S$, $n|_{\sigma}$ is a $K$-model,

3. for each $K \xrightarrow{\sigma} N \in S$,
   
   $n|_{\sigma}$ satisfies the local axioms $\Gamma^K$,

4. for each $K \xrightarrow{\sigma}_{h} N \in S$,
   
   $n$ has a $\sigma$-expansion $k$ (i.e. $k|_{\sigma} = n$) that is a $K$-model.
Reachability of nodes

Global reachability

$M \xrightarrow{\sigma} N$ is defined inductively and holds iff

- either $M = N$ and $\sigma = id$, or
- $M \xrightarrow{\sigma'} K \in S$, and $K \xrightarrow{\sigma''} N$, with $\sigma = \sigma'' \circ \sigma'$.

Local reachability

$M \xrightarrow{\sigma} N$ iff $M \xrightarrow{\sigma} N$ or there is a node $K$ with

$M \xrightarrow{\sigma'} K \in S$ and $K \xrightarrow{\sigma''} N$, such that $\sigma = \sigma'' \circ \sigma'$. 
Theorem links

Theorem links come, like definition links, in three different versions:

- **global** theorem links \( M \xrightarrow{\sigma} N \), where \( \sigma: \Sigma^M \longrightarrow \Sigma^N \),
- **local** theorem links \( M \xrightarrow{\sigma} N \), where \( \sigma: \Sigma^M \longrightarrow \Sigma^N \),
- **hiding** theorem links \( M \xrightarrow{h \theta} N \), where for some \( \Sigma \),

\[
\begin{array}{c}
\Sigma^M \\
\theta \\
\Sigma \\
\end{array} \xrightarrow{\sigma} 
\begin{array}{c}
\Sigma \\
\Sigma^N \\
\end{array}
\]
Semantics of theorem links

- $\mathcal{S} \models M \overset{\sigma}{\longrightarrow} N$ iff for all $n \in \text{Mod}_S(N)$, $n|_\sigma \in \text{Mod}_S(M)$.

- $\mathcal{S} \models M \overset{\sigma}{\longrightarrow} N$ iff for all $n \in \text{Mod}_S(N)$, $n|_\sigma \models \Gamma^M$.

- $\mathcal{S} \models M \overset{h \theta}{\longrightarrow} N$ iff for all $n \in \text{Mod}_S(N)$, $n|_\sigma$ has a $\theta$-expansion to some $M$-model.
Conservativity annotations

A global definition link $\mathcal{M} \xrightarrow{\sigma} \mathcal{N}$ can be marked to be conservative:

$$\mathcal{M} \xrightarrow{\sigma} \mathcal{N}$$

This is a proof obligation expressing that

each $\mathcal{M}$-model can be $\sigma$-expanded to an $\mathcal{N}$-model.

Similarly: Annotation $d$ for definitional extension, i.e. unique model expansion.
CASL Extended static semantics

\[ N, \Gamma_s, (S, Th) \vdash \text{SPEC} \gg N', (S', Th) \]

\( \Gamma_s, (S, Th) \) is an extended static global environment. 
\( (S', Th') \) is a development graphs extending \( (S, Th) \).
\( N \) and \( N' \) are nodes in \( S \) and \( S' \), resp.
\( \Sigma^N \) is an extension of \( \Sigma^{N'} \).
Rule for translations

\[
N, \Gamma_s, (S, Th) \vdash \text{SPEC} \triangleright \triangleright N', (S', Th') \\
\Sigma^{N'} \vdash \text{RENAMEING} \triangleright \triangleright \sigma: \Sigma^{N'} \longrightarrow \Sigma'' \\
|\sigma| \text{ is the identity on } |\Sigma^N| \\
S'' = S' \uplus \{N'' = (\Sigma'', \emptyset); \quad N' \xrightarrow{\sigma} N'' \}
\]

\[
N, \Gamma_s, (S, Th) \vdash \text{translation SPEC RENAMEING} \triangleright \triangleright N'', (S'', Th')
\]
Rule for instantiation

\[ (\text{SN} \rightarrow GS_s) \in \mathcal{G}_s \]

\[ GS_s = (N_I, (N_1, \ldots, N_n), N_B) \]

\[ N_i, N_i, \Gamma_s, (S, Th) \vdash FA_i \bowtie \sigma_i, N_i^A, (S_i, Th_i) \]

\[ (\Sigma', \sigma_f) = GS_s((\Sigma_1^A, \sigma_1), \ldots, (\Sigma_n^A, \sigma_n)) \]

is defined

\[ \Sigma^N \cup \Sigma' \text{ is defined } \]

the \( S_i \) disjointly extend \( S \)

\[ S' = \bigcup_{i=1}^{n} S_i \uplus \{ N' = (\Sigma^N \cup \Sigma', \emptyset); \quad \Sigma^N \rightarrow \Sigma^N \cup \Sigma' \} \]

\[ \uplus \{ N_B \stackrel{\sigma_f}{\longrightarrow} N' \} \uplus \{ N_i^A \stackrel{\sigma_i}{\longrightarrow} N' \mid i = 1, \ldots, n \} \]

\[ Th' = \bigcup_{i=1}^{n} Th_i \]

\[ N, \Gamma_s, (S, Th) \vdash \text{spec-inst SN FA}_1 \ldots \text{FA}_n \bowtie N', (S', Th') \]
Proof rules: Structural rules

\[ K \xrightarrow{\sigma \circ \tau} M \quad \text{for each} \quad K \xrightarrow{\tau} N \]

\[ L \xrightarrow{\sigma \circ \tau} M \quad \text{for each} \quad L \xrightarrow{\theta} K \quad \text{and} \quad K \xrightarrow{\tau} N \]

\[ N \xrightarrow{\sigma} M \]

Glob-Decomposition

\[ M \xrightarrow{\sigma} N \]

\[ M \xrightarrow{\sigma} N \]

Subsumption

\[ K \xrightarrow{\sigma} L \quad L \xrightarrow{\theta} M \]

Composition

\[ K \xrightarrow{\theta \circ \sigma} M \]
Proof rules: Basic Inference

\[ \text{Th}_S(N) \vdash \sum_N \sigma(\varphi) \quad \text{for each } \varphi \in \Gamma^M \]

\[ M \overset{\sigma}{\longrightarrow} N \]

Basic Inference

For \( N \in \mathcal{N} \), the theory \( \text{Th}_S(N) \) of \( N \) is defined by

\[ \Gamma^N \cup \bigcup_{K} \sigma(\Gamma^K) \]

\[ K \overset{\sigma}{\longrightarrow} N \]
**Proof rules: Hiding**

\[ G(i) \]
\[ \mu_{\langle N \rangle} \circ \sigma \]
\[ \mu_i \]
\[ i \in |J| \]
\[ M \longrightarrow C \]

\[ \sigma \]
\[ M \longrightarrow N \]

\[ (\mu_i) \]

A weakly amalgamable cocone for “zig-zag path diagram” \( D \)

\[ \sigma' \circ \theta = \theta' \circ \sigma \]

**Theorem-Hide-Shift**

Till Mossakowski  BISS, Universität Bremen, Germany: Heterogeneous Proofs. IFIP 1.3 meeting, 24th Sept. 2002
Example: fields and groups
Example: fields and groups
# Comparison of completeness results

<table>
<thead>
<tr>
<th>Completeness theorem needs:</th>
<th>Borzyszkowski’s calculus</th>
<th>development graphs</th>
</tr>
</thead>
<tbody>
<tr>
<td>completeness of logic</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>oracle for conservativness</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>(for hiding in refinement sources)</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>weak amalgamation</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>Craig interpolation</td>
<td>yes</td>
<td>no</td>
</tr>
</tbody>
</table>

**Treatment of hiding in refinement targets**

- **local**
- **global**
A sample logic graph

IndexedPropModal \rightarrow \text{make worlds explicit} \rightarrow CASL

IndexedPropModal \downarrow \text{forget indexing} \rightarrow \text{PropModal}

HOL \downarrow \text{code out partiality, subsorting and sort generation}
Morphisms (projections)
"rich" Comorphisms (inclusions) "poor"

Signatures

\[ \begin{align*}
\mu(\Sigma) & \rightarrow \mu \\
\Sigma & \rightarrow \Sigma
\end{align*} \]

Entailment

\[ \begin{align*}
\mu(\Gamma) & \models \mu(\phi) \\
\Gamma & \models \phi
\end{align*} \]

Sentences

\[ \begin{align*}
\text{Sen } \mu(\Sigma) & \rightarrow \mu \\
\text{Sen } \Sigma & \rightarrow \text{Sen } \Sigma
\end{align*} \]

Satisfaction

\[ \begin{align*}
M & \models \mu(\phi) \\
\mu(M) & \models \phi
\end{align*} \]

Models

\[ \begin{align*}
\text{Mod } \mu(\Sigma) & \rightarrow \mu \\
\text{Mod } \Sigma & \rightarrow \text{Mod } \Sigma
\end{align*} \]
“rich”  Comorphisms (encodings)  “even richer”

Signatures  \[ \mu \Sigma \]  \[ \Sigma \]

Entailment  \[ \mu(\Gamma) \models \mu(\phi) \]  \[ \Gamma \models \phi \]

Sentences  \[ \text{Sen } \mu \Sigma \]  \[ \text{Sen } \Sigma \]

Satisfaction  \[ M \models \mu(\phi) \]  \[ \mu(M) \models \phi \]

Models  \[ \text{Mod } \mu \Sigma \]  \[ \text{Mod } \Sigma \]
A sample heterogeneous specification

```
spec Marriage =
{   logic IndexedPropModal
 sort Person
 props isMarried, immortal, dead : Person;
     isMarriedWith : Person × Person
 var x, y : Person
     isMarriedWith(x, y) ⇒ ¬dead(x)
     immortal(x) ⇔ ¬◊ dead(x)  } with logic → CASL
then var x : Person; w : World
     isMarried(x, w) ⇒ ∃y : Person • isMarriedWith(x, y, w)
then %implies
 logic IndexedPropModal
 var x : Person
     (□isMarried(x)) ⇒ immortal(x) } implicitly
                                 coerced to CASL
```
The corresponding development graph
Problem

How to provide a formal basis for heterogeneous specification?
Solution

- Indexed coinstitution = diagram of coinstitutions and morphisms = diagram of institutions and comorphisms
- For this, we have a Grothendieck construction
- Why do we need just comorphisms?
  ⇒ See WADT talk on Thursday

Slogan:
Heterogeneous specification is structured specification over the Grothendieck institution
Signatures are pairs consisting of a logic $L$ and a signature $\Sigma$ in the logic $L$. 

$(\Sigma, L)$ $(\Sigma', L')$
Signatures are pairs consisting of a logic $L$ and a signature $\Sigma$ in the logic $L$.

Signatures morphisms are pairs consisting of a logic morphism $\mu$ and a signature morphism $\sigma$ in the logic $L$. 

\[(\Sigma, L) \xrightarrow{(\sigma, \mu)} (\Sigma', L')\]

$\sigma : \Sigma \rightarrow \mu(\Sigma')$
$\mu : L' \rightarrow L$
The Grothendieck logic

Signatures

$(\Sigma, L) \xrightarrow{(\sigma, \mu)} (\Sigma', L')$

$\sigma: \Sigma \rightarrow \mu(\Sigma')$
$\mu: L' \rightarrow L$

Sentences

$\text{Sen}^L \Sigma$

$\text{Sen}^{L'} \Sigma'$
The Grothendieck logic

Signatures

$(\Sigma, L) \xrightarrow{(\sigma, \mu)} (\Sigma', L')$

$\sigma: \Sigma \rightarrow \mu(\Sigma')$
$\mu: L' \rightarrow L$

Sentences ... and their translation

$\text{Sen}^L \Sigma$

$\sigma$

within the logic $L$

$\text{Sen}^L \mu(\Sigma')$

$\mu$

from logic $L$ to logic $L'$

$\text{Sen}^{L'} \Sigma'$
The Grothendieck logic

Signatures

\[(\Sigma, L) \xrightarrow{(\sigma, \mu)} (\Sigma', L')\]

\[\sigma : \Sigma \rightarrow \mu(\Sigma')\]

\[\mu : L' \rightarrow L\]

Sentences

\[\text{Sen}^L \Sigma \xrightarrow{\sigma} \text{Sen}^L \mu(\Sigma') \xrightarrow{\mu} \text{Sen}^{L'} \Sigma'\]

Models

\[\text{Mod}^L \Sigma \xleftarrow{\sigma} \text{Mod}^L \mu(\Sigma') \xleftarrow{\mu} \text{Mod}^{L'} \Sigma'\]
The Grothendieck logic

Signatures

\[(\Sigma, L) \xrightarrow{(\sigma, \mu)} (\Sigma', L')\]

\[\sigma : \Sigma \rightarrow \mu(\Sigma')\]
\[\mu : L' \rightarrow L\]

Entailment

\[\models \Sigma\]
\[\models \Sigma'\]

Sentences

\[\sigma\]
\[\mu\]

\[\text{Sen}^L \Sigma \rightarrow \text{Sen}^L \mu(\Sigma') \rightarrow \text{Sen}^L' \Sigma'\]

Satisfaction

\[\models \Sigma\]
\[\models \Sigma'\]

Models

\[\sigma\]
\[\mu\]

\[\text{Mod}^L \Sigma \leftarrow \text{Mod}^L \mu(\Sigma') \leftarrow \text{Mod}^L' \Sigma'\]

Entailment

\[\models \Sigma\]
\[\models \Sigma'\]

Sentences

\[\sigma\]
\[\mu\]

\[\text{Sen}^L \Sigma \rightarrow \text{Sen}^L \mu(\Sigma') \rightarrow \text{Sen}^L' \Sigma'\]

Satisfaction

\[\models \Sigma\]
\[\models \Sigma'\]

Models

\[\sigma\]
\[\mu\]

\[\text{Mod}^L \Sigma \leftarrow \text{Mod}^L \mu(\Sigma') \leftarrow \text{Mod}^L' \Sigma'\]
Heterogeneous proofs

- Either via a **global** encoding of the Grothendieck institution into a “universal” logic $U$
  $\Rightarrow$ **Heterogeneous borrowing**
  (needs structured proving in $U$)

- Or via **local** encodings of the individual institutions into logics
  $\Rightarrow$ **truly heterogeneous proofs**
  (needs structured proving in the Grothendieck institution)
Borrowing

Local borrowing
If an institution comporphy ism $\rho: I \rightarrow J$ admits model expansion, $J$-entailment in basic specifications can be re-used for $I$.

Global borrowing
If an institution comorphism $\rho: I \rightarrow J$ admits weak amalgamation, $J$-entailment in structured specifications can be re-used for $I$. 
The global view: Heterogeneous borrowing

Given an indexed comorphism $C: \text{Ind} \rightarrow \text{Comorph}(U)$ over an indexed institution $I: \text{Ind} \rightarrow \text{Ins}$, we can form its Grothendieck representation $C^#: I^# \rightarrow U$.

$C^#$ admits model expansion and weak amalgamation if $C$ does so (pointwise) ⇒ local and global borrowing.

Needed assumptions for heterogeneous proofs:
$U$ is a complete logic, has a conservativity checker and admits weak amalgamation.
The local view: Grothendieck proofs

Alternatively, we can directly prove in the Grothendieck institution.

Needed assumptions for heterogeneous proofs:

- $I^#$ is a complete logic,
- has a conservativity checker and
- admits weak amalgamation.
Results for the Grothendieck institution

- $I^\#$ is complete whenever all the individual logics are complete (or embedded into a complete logic).
- $I^\#$ has a conservativity checker if all the individual logics have and all comorphisms admit model expansion.
- Craig interpolation: no idea.
- What about weak amalgamation?
Theorem (Diaconescu)

The morphism-based Grothendieck institution admits amalgamation if

1. all individual institutions admit amalgamation

2. all institution morphisms admit amalgamation

3. the institution indexing admits amalgamation

4. all signature translations have left adjoints

(This also holds for weak amalgamation.)
However . . .

1. some institutions don’t admit weak amalgamation ($\mathsf{Casl}$)

2. not all institution (co)morphisms admit weak amalgamation (e.g. $\mathsf{Casl} \rightarrow \mathit{HOL}$)

3. the institution indexing admits weak amalgamation only if we have a “big” projection-minimal institution combining all the institutions involved (not existent in practice)

4. not all signature translations have left adjoints (e.g. $\mathsf{Casl} \dashv \mathit{LTL} \rightarrow \mathsf{Casl}$)
... hence ...

- ... in practice, the Grothendieck institution does not admit weak amalgamation!
- Therefore, an important prerequisite for the structured proof calculus is missing!
Our solution to the problem

Use a comorphism-based Grothendieck institution and . . .

1. work with cocones instead of colimits

2. require weak amalgamation only for a subset $\mathcal{L}$ of the institutions (and each institution must be encoded into such an institution)

3. require comorphisms only to admit weak amalgamation

4. require only weak amalgamation for the indexing (this means that there is some “rich, but not that big” encoding-maximal institution like HOL)
Theorem

Under the above assumptions, if the institutions in $\mathcal{L}$ also are \textit{complete}, then the proof calculus for devel. graphs is \textit{sound and complete} for the comorphism-based Grothendieck institution relative to oracles for conservativeness for $\mathcal{L}$.

This means we can do \textit{heterogeneous proofs}!
Proof idea for the theorem

Given a diagram for the rule Theorem-Hide-Shift, how to find a weakly amalgamable cocone?

- successively construct a weakly amalgamable cocone at the level of the institution indexing
- from the tip of the cocone, find a comorphism into a logic in $\mathcal{L}$
- translate everything into this logic and find a weakly amalgamable cocone there
- note that amalgamable cocone (as opposed to colimits) translate and compose nicely
## Comparison with Diaconescu’s Theorem

<table>
<thead>
<tr>
<th></th>
<th>Diaconescu needs</th>
<th>we need</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>arrow type</strong></td>
<td>morphism</td>
<td>comorphism</td>
</tr>
<tr>
<td>individual logics</td>
<td>all admit amalgamation</td>
<td>some admit weak amalgamation</td>
</tr>
<tr>
<td>logic morphisms</td>
<td>all admit amalgamation</td>
<td>some admit weak amalgamation</td>
</tr>
<tr>
<td>logic indexing</td>
<td>admits amalgamation</td>
<td>admits weak amalgamation</td>
</tr>
<tr>
<td>signature translations</td>
<td>all have left adjoints</td>
<td>-</td>
</tr>
</tbody>
</table>
New rules for heterogeneous proofs

\[
\begin{align*}
\sigma' \circ \theta &= \theta' \circ \sigma \\
\text{Global Borrowing} & & \text{Inverse Borrowing}
\end{align*}
\]
\[ M \quad N \]

\[ \theta' \quad c \]

\[ M' \quad \sigma' \quad N' \]

\[ \sigma \]

\[ M \quad \sigma \quad N \]

\[ \theta' \quad c \]

\[ M' \quad N' \]

\[ \theta: \Sigma^M \rightarrow \Sigma^{M'}, \sigma' \circ \theta = \theta' \circ \sigma, \theta(\Gamma^M) \subseteq \Gamma^{M'} \]

**Local Borrowing**
if \( d \) is marked as being a faithful comorphism and \( N \) is isolated.

**Faithfulness**

if \( d \) is marked as being a model-bijective comorphism and \( N \) is isolated.

**Model-bijection**
A sample heterogeneous specification

```
spec Marriage =
{  logic IndexedPropModal
sort  Person
props isMarried, immortal, dead : Person;
   isMarriedWith : Person × Person
var   x, y : Person
   •  isMarriedWith(x, y) ⇒ ¬dead(x)
   •  immortal(x) ⇔ ¬◊ dead(x)  } with logic → CASL
then var x : Person; w : World
   •  isMarried(x, w) ⇒ ∃y : Person • isMarriedWith(x, y, w)
then  %implies
   logic IndexedPropModal                          implicitly
   var x : Person                                      coerced
   •  (□isMarried(x)) ⇒ immortal(x) } to CASL
```
A sample heterogeneous proof

\[ \text{CASL}_1 \rightarrow \text{CASL}_2 \]

\[ \text{IPM}_1 \leftarrow \text{IPM}_2 \]
Apply Glob-Decomposition

\[ \text{CASL}_1 \rightarrow \text{CASL}_2 \]

\[ \text{IPM}_1 \rightarrow \text{IPM}_2 \]
Apply Subsumption

$$\text{CASL}_1 \rightarrow \text{CASL}_2$$

$$\text{IPM}_1 \rightarrow \text{IPM}_2$$

Till Mossakowski, BISS, Universität Bremen, Germany: Heterogeneous Proofs. IFIP 1.3 meeting, 24th Sept. 2002
Apply Basic Inference trivially

\[ \text{CASL}_1 \rightarrow \text{CASL}_2 \]

\[ \text{IPM}_1 \rightarrow \text{IPM}_2 \]

Till Mossakowski
BISS, Universität Bremen, Germany: Heterogeneous Proofs. IFIP 1.3 meeting, 24th Sept. 2002
Apply Composition, introduce Lemma

\[ \text{IPM}_3: \text{local axiom } is\text{Married}(x) \Rightarrow \neg\text{dead}(x) \]
Apply Glob-Decomposition

\[ CASL_1 \rightarrow IPM_3 \rightarrow CASL_2 \]

\[ IPM_1 \rightarrow IPM_3 \rightarrow IPM_2 \]

IPM\(_3\): local axiom \( isMarried(x) \Rightarrow \neg dead(x) \)
Apply Subsumption

\[ \text{CASL}_1 \xrightarrow{	ext{IPM}_3} \text{CASL}_2 \]

\[ \text{IPM}_1 \xrightarrow{	ext{IPM}_3} \text{IPM}_2 \]

\( \text{IPM}_3 \): local axiom \( \text{isMarried}(x) \Rightarrow \lnot \text{dead}(x) \)
Apply Basic Inference in $\text{CASL}$

\[
\begin{align*}
\text{CASL}_1 \quad & \quad \text{CASL}_2 \\
\text{IPM}_1 \quad & \quad \text{IPM}_2 \\
\text{IPM}_3
\end{align*}
\]

\[
\{ \text{isMarried}(x, w) \Rightarrow \exists y \cdot \text{isMarriedWith}(x, y, w); \\
\text{isMarriedWith}(x, y, w) \Rightarrow \neg \text{dead}(x, w) \} \\
\vdash \text{isMarried}(x, w) \Rightarrow \neg \text{dead}(x, w)
\]
Apply local borrowing

\[ \text{CASL}_1 \rightarrow \text{CASL}_2 \]

\[ \text{IPM}_1 \rightarrow \text{IPM}_3 \]

\[ \text{IPM}_3 \rightarrow \text{PM}_3 \]

\[ \text{PM}_3 \leftarrow \text{PM}_2 \]
Apply Basic Inference in PM

\[
\begin{array}{c}
\text{CASL}_1 \rightarrow \text{CASL}_2 \\
\text{IPM}_1 \rightarrow \text{IPM}_3 \\
\text{IPM}_3 \rightarrow \text{PM}_3 \\
\end{array}
\]

\(\square \text{isMarried} \vdash \square \neg \text{dead} \vdash \neg \Diamond \text{dead} \vdash \text{immortal}\)
Conclusion

- Global treatment of hiding is better than local treatment
  ⇒ no Craig interpolation needed
- Local heterogeneous proofs are better than global heterogeneous borrowing
  ⇒ chance to exploit specialized tool support
- Homogeneous development graphs (and tools for them) have been used for industrial-scale applications (hundreds of specifications)
  ― we are planning to make the tools heterogeneous
Future work

- **Taxonomy** of (co)morphisms and morphism-comorphism-based Grothendieck institution: see my WADT talk
- more examples
- tool development