Selecting Colimits for Parameterisation and Networks of Specifications

Till Mossakowski[1] Mihai Codescu[2] **Florian Rabe**[3]

[1] Otto-von-Guericke University of Magdeburg, Germany

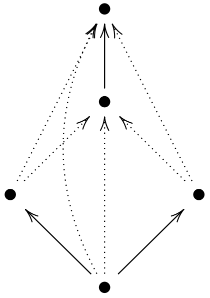
[2] Free University of Bozen-Bolzano, Italy

[3] Jacobs University Bremen, Germany

WADT 2016, September 21, Gregynog, Wales

Colimits

- Essential part of building specifications modularly
- Allow giving semantics of many specification-forming operations
  - union
  - union with sharing
  - parametrization
- Used as semantics for various specification languages
  - CASL
  - Distributed Ontology Model and Specification Language (DOL), an OMG standard
  - Meta-meta-theory (Rabe)
  - Specware (Kestrel Institute)
  - MathScheme (Carette+Farmer)



Example: CASL-style Parametrisation

Syntax:

- P declares a set of parameters
- $B[P]$ is a parametric specification
- A is a concrete specification
- $\sigma : P \to A$ provides values for the parameters
- $B[A \textbf{ fit } \phi]$ is an instance of B

$$
\begin{tikzcd}
P \arrow[r, hook] \arrow[d, "\sigma"] & B \arrow[d] \\
A \arrow[r, hook] & B[A \textbf{ fit } \sigma]
\end{tikzcd}
$$

Semantics:

- Categorical: pushout
- Intuitive: Take A and add B with P replaced according to σ

Example: Named Instantiation in MMT

Syntax:

```
theory SemiLattice {
  univ: type
  op   : univ * univ -> univ
}
theory Lattice {
  u: type
  meet: SemiLattice {univ = u}
  join: SemiLattice {univ = u}
}
```

Semantics:

```
theory Lattice {
  u: type
  meet.univ: type = u
  meet.op: meet.univ * meet.univ -> meet.univ
  join.univ: type = u
  join.op: join.univ * join.univ -> join.univ
}
```

Example: DOL networks and combinations

```
logic CASL

spec Relation =
  sort Elem; pred __R__ : s*s
end
spec Reflexive = Relation then
  forall x:Elem . x R x %(refl)%
end
spec Transitive = Relation then
  forall x,y,z:Elem . x R y /\ y R z => x R z %(tr)%
end

network N = Relation, Reflexive, Transitive end
spec PreOrder = combine N end %% colimit of network
```

Example: DOL networks and combinations

```
ontology Source =
  Class: Person
  Class: Woman SubClassOf: Person
ontology Onto1 =
  Class: Person           Class: Bank
  Class: Woman SubClassOf: Person
interpretation I1 : Source to Onto1 =
   Person |-> Person, Woman |-> Woman
ontology Onto2 =
  Class: HumanBeing       Class: Bank
  Class: Woman SubClassOf: HumanBeing
interpretation I2 : Source to Onto2 =
   Person |-> HumanBeing, Woman |-> Woman
ontology CombinedOntology =
  combine Source, Onto1, Onto2, I1, I2
```

Example: DOL networks and combinations

Resulting colimit in DOL:

$$\begin{tikzcd}
 & \{\mathit{Woman}, \mathit{Person}_\mathit{HumanBeing}, \mathit{Bank}_1, \mathit{Bank}_2\} & \\
\mathit{Onto}1 \arrow[ur, dotted] & & \mathit{Onto}2 \arrow[ul, dotted] \\
 & \{\mathit{Woman}, \mathit{Person}\} \arrow[ul] \arrow[ur] &
\end{tikzcd}$$

Problem: Selecting Colimits

Colimits only unique **up to isomorphism**

- Mostly irrelevant for categorical theory
  most constructions preserve isomorphism anyway
- Big problem in specification practice must select a specific colimit

Colimit selection for a category **C**

- a partial function from **C**-diagrams D to colimits of D
- usually no canonical selection
  may require choosing representatives of equivalence relation
- usually no well-behaved selection
  many conflicting desirable properties

Problem statement

- select colimits for categories typical in algebraic specification

Typical categories in algebraic specification

The following abstractions suffice for us:

Definition

An **inclusive category** is a category with

- a broad subcategory singling out the inclusion morphisms such that inclusion is a partial order
- non-empty products $\cap$ and finite coproducts $\cup$ such that the following is a pushout

$$\begin{tikzcd} A \cap B \arrow[r, hook] \arrow[d, hook] & A \arrow[d, hook] \\ B \arrow[r, hook] & A \cup B \end{tikzcd}$$

Definition

An inclusive category $\mathbf{C}$ has **symbols** if it also has a faithful functor $|\_| : \mathbf{C} \to \mathbb{SET}$ that preserves inclusions.

Pushout-stable Inclusions

A pushout along an inclusion is a colimit of
$$\begin{tikzcd} P \arrow[r, hook] \arrow[d, "\sigma"] & B \\ A & \end{tikzcd}$$

sel has **pushout-stable inclusions** if it selects a pushout

$$\begin{tikzcd} P \arrow[r, hook] \arrow[d, "\sigma"] & B \arrow[d, "\sigma^B"] \\ A \arrow[r, hook] & \sigma(B) \end{tikzcd}$$

whenever such a pushout exists.

- Makes pushout along $\sigma : P \to A$ a map from P-extensions to A-extensions
- Critical for intuitive semantics of parametrization

Natural names

Intuitions

- Users must be able to refer to the symbols in the colimit
- Only possible if names are predictable — avoid generated names
- Ideally reuse names already present in diagram — only possible if there are no name clashes
- Sharing condition: any two symbols with the same name can be identified

Precise statement of the sharing condition

Given a diagram $D : \mathbf{I} \to \mathbf{C}$ in an inclusive category $\mathbf{C}$

$$\bigwedge_{i \neq j \in |\mathbf{I}|} \left(D(i) \cap D(j) \subseteq \bigcup_{\exists m:k \to i, n:k \to j \in \mathbf{I}} D(k) \right)$$

sel has **natural names** if *sel*(D) consists of inclusions whenever D satisfies the sharing condition.

Natural names for pushouts

For pushouts along inclusions, we can do even better.

sel has **natural names for pushouts** if it selects a pushout

$$\begin{tikzcd} P \arrow[r, hook] \arrow[d, "\sigma"] & B \arrow[d, "\sigma^B"] \\ A \arrow[r, hook] & \sigma(B) \end{tikzcd}$$

such that

- $|B| \setminus |P| = |\sigma(B)| \setminus |A|$ and
- $|\sigma^B| \setminus |\sigma|$ is the identity map

whenever such a pushout exists.

Coherent pushouts

- Pushout-stable inclusions not enough in practice
- *sel* has **coherent pushouts** if the diagrams below commute
- Critical to avoid confusion when iterating parametrization

$$
\begin{tikzcd}
P \arrow[r, hook] \arrow[d, "\sigma_1"'] & B \arrow[d, "\sigma_1^B"'] \arrow[ddr, "(\sigma_1;\sigma_2)^B"] & \\
A \arrow[r, hook] \arrow[d, "\sigma_2"'] & \sigma_1(B) \arrow[d, "\sigma_2^{\sigma_1(B)}"'] & \\
A' \arrow[r, hook] & \sigma_2(\sigma_1(B)) \arrow[r, equal] & (\sigma_1;\sigma_2)(B)
\end{tikzcd}
$$

$$
\begin{tikzcd}
P \arrow[r, hook] \arrow[d, "\sigma"] & B_1 \arrow[r, hook] \arrow[d, "\sigma^{B_1}"] & B_2 2 \arrow[d, "\sigma^{B_2}"] \arrow[dr, "(\sigma^{B_1})^{B_2}"] & \\
A \arrow[r, hook] & \sigma(B_1) \arrow[r, hook] & \sigma(B_2) \arrow[r, equal] & \sigma^{B_1}(B2)
\end{tikzcd}
$$

Totality

Completeness

- *sel* is **complete** if it is defined for every diagram that has a colimit
- Completeness often difficult in practice

Total pushouts

- *sel* is **total for pushouts along inclusions** if it is defined for every span with one inclusion

The latter looks weak, but it is already hard to combine with coherence and natural names.

Interchange

sel has **interchange** if

- for a bifunctor $D : \mathbf{I} \times \mathcal{J} \to \mathbf{C}$
- that has only inclusions and satisfies the sharing condition,
- the following holds

$$\mathbf{colim}_{i \in \mathbf{I}}(\mathbf{colim}_{j \in \mathcal{J}} D(i,j)) = \mathbf{colim}_{j \in \mathcal{J}}(\mathbf{colim}_{i \in \mathbf{I}} D(i,j))$$

Interchange and pushout-coherence are special cases of

General coherence

- iteratively taking colimits of increasingly large subdiagrams ...
- should ultimately yield the same colimit ...
- no matter which subdiagrams are used

Example: Selecting Colimits in Set

- All properties are reasonable and can be realized individually.
- But they cannot be realized together.

Counter-example

Consider the category $\mathbb{SET}$ with standard inclusions and $|S| = S$.
There is no selection of pushouts that has

- total pushouts
- natural names for pushouts
- coherent pushouts

Problem:

- Natural names not always possible ...
- so that we have to choose non-naturally ...
- which breaks coherence

Example: Selecting Colimits in Set

Using the sharing condition, we can do better:

Selection in $\mathbb{SET}$

$\mathbb{SET}$ has a selection of colimits that has

- natural names
- pushout-stable inclusions
- pushouts with natural names whenever the sharing condition holds
- coherent pushouts
- interchange
- completeness (and thus total pushouts)

But the sharing condition is very strong and excludes practically important use cases.

More on Natural Symbol Names

The standard construction of the colimit of $D : I \to \mathbb{SET}$ builds

- the disjoint union of all nodes of D
- quotiented by identifying all x with $f(x)$ for all edges f of D

Accordingly, we define for a diagram $D : \mathbf{I} \to \mathbf{C}$

- $Sym(D) := \{(i, x) | i \in |I|, x \in |D(i)|\}$
- $\sim_D$ is the equivalence relation generated by

$$(i, x) \;\; \leq_D \;\; (j, |D(m)|(x)) \text{ for any } m : i \to j \in \mathbf{I}$$

Then $|\mathbf{colim}(D)| \cong Sym(D)/ \sim_D$.

This effectively reduces colimit selection to selecting a system of representatives for $Sym(D)/ \sim_D$.

Selecting Names

We define the following properties for selections of symbol names:

- **Natural symbol names**: The representative of a class K is some x with $(i, x) \in K$.
  - minimal condition that avoids generating names
  - but may be impossible due to name clashes
- **Origin**: If (i, x) is a $\leq_D$-least element in class K, the representative of K is x.
  - intuitive for pushouts of non-inclusions
  - but awkward for pushouts of inclusions
- **Majority**: If some x occurs most often in K, then the representative of K is x.
  - good for large diagrams
  - but also awkward for pushouts of inclusions

Example: Selecting Names in $\mathbb{SET}$

We consider $\mathbb{SET}$ with standard inclusions and $|S| = S$ again.
Then:

- Origin and majority may contradict each other
- The previous example selection for $\mathbb{SET}$ can be modified to satisfy
  - natural symbol names
  - origin
  - majority in those cases where origin does not apply

Generalisation to Product Categories

We can lift colimit selections to product categories:

- For $j \in J$, let sel_j be a selection for category $\mathbf{C}_j$
- Define a selection for the product $\Pi_{j \in J} C_j$ by

$$sel(D)_j = sel_j(\pi_j(D))$$

  where π_j is the projection into $\mathbf{C}_j$
- Define a symbol functor by

$$|(A_j)_{j \in J}| = \{(j, x) \mid j \in J, x \in |A_j|\}$$

- Then: If every sel_j has property P, then so does sel, where P is any of
  - natural names
  - pushout-stable inclusions
  - coherent pushouts
  - natural names for pushouts if the sharing condition holds
  - total pushouts
  - interchange
  - completeness

Example: Selection of colimits in OWL

OWL ontologies consists of

- a set of classes
- a set of properties
- a set of individuals

So OWL can be seen as $\mathbb{SET} \times \mathbb{SET} \times \mathbb{SET}$.

Thus, we obtain a colimit selection immediately by

- using $J = \{\mathtt{class}, \mathtt{property}, \mathtt{individual}\}$
- using the colimit selection for $\mathbb{SET}$ for all three components

Selection of colimits in many-sorted equational logic

Fibred representation of many-sorted equational logic signatures

- Given a set of sorts S:
  $B(S) = \mathbf{Sign}_S^{MSEQL}$ = category of multi-sorted algebraic signatures with sort set S:

  $$\mathbf{Sign}_S^{MSEQL} = \Pi_{w \in S^*, s \in S}\ \mathbb{SET}.$$

  Objects = sets of operation symbols $F_{w,s}$ for each string of argument sorts w and result sort s.
- Given a function $u : S \to S'$, we have a functor

  $$B_u : \mathbf{Sign}_{S'}^{MSEQL} \to \mathbf{Sign}_S^{MSEQL}$$

  defined as $B_u(F') = F$, where $F_{w,s} = F'_{u(w),u(s)}$.
- B_u has a left adjoint

  $$L_u : \mathbf{Sign}_S^{MSEQL} \to \mathbf{Sign}_{S'}^{MSEQL}$$

  defined as $L_u(F) = F'$, where
  $F'_{w',s'} = \cup_{w \in S^*, s \in S, u(w)=w', u(s)=s'} F_{w,s}$.

Index categories and the Grothendieck construction

An indexed inclusive category is just a functor $B : \mathbb{SET}^{op} \to ICat$.

The Grothendieck category $B^{\#}$ has

Objects pairs (i, A) with $i \in |B|$, $A \in B_i$ where $B_i = B(i)$

Morphisms $(u, \sigma){:}(i, A) \to (j, B)$ with $u{:}j \to i$, $\sigma{:}B_u(A) \to B$

Example

$\mathbf{Sign}^{MSEQL}$ is the Grothendieck category $B^{\#}$ of B from previous slide. (Left adjoints not needed here yet.)

Questions:

- how to obtain colimits in $B^{\#}$
- how to obtain **selected** colimits in $B^{\#}$

Colimits in the Grothendieck category

Theorem (Tarlecki, Goguen, Burstall 1991, inclusive version)

Let $B : \mathbf{Ind}^{op} \to \mathbb{IC}at$ *be an indexed inclusive category with* **Ind** *inclusive such that*

- B *is locally reversible, i.e. for each* $u : i \to j$ *in* **Ind**, $B_u : B_j \to B_i$ *has an left adjoint* $F_u : B_i \to B_j$
- **Ind** *has colimits,*
- *each category* B_i *has colimits, for* $i \in |\mathbf{Ind}|$.

Then $B^{\#}$ *is an inclusive category and it has colimits.*

Selected colimits in the Grothendieck category

Theorem (Tarlecki, Goguen, Burstall 1991, selection version)

Let $B : \mathbf{Ind}^{op} \to \mathbb{IC}at$ be an indexed inclusive category with **Ind** *inclusive such that*

- *B is locally reversible, i.e. for each $u : i \to j$ in* **Ind**, *$B_u : B_j \to B_i$ has a selected left adjoint $F_u : B_i \to B_j$*
- **Ind** *has a selection of colimits $sel_{\mathbf{Ind}}$,*
- *each category B_i has a selection of colimits sel^i, for $i \in |\mathbf{Ind}|$.*

Then $B^{\#}$ is an inclusive category and it has a selection of colimits.

Transfer of properties

Theorem

Under the same assumptions as above extended by:

- F_u *preserves inclusions,*
- *the unit and counit of the adjunction are inclusions.*

If **Ind** *and each* B_i *enjoy some property taken from of*

- *natural names*
- *pushout-stable inclusions*
- *coherent pushouts and natural names for pushouts for those spans satisfying the sharing condition*
- *total pushouts*
- *interchange*
- *completeness*

then $B^{\#}$ *enjoys the same property.*

Conclusions and Future Work

- Good selection of colimits
  - is crucial for their acceptance
  - has no optimal solution
- We formulate desirable properties of selections
  - set of requirements for colimit selection
  - cannot be realized at once
- We give concrete compromise selections
  - for $\mathbb{SET}$
  - carries over to more complex categories
- Future work
  - can we select even better selection of colimits?
  - implementation in Heterogeneous tool set Hets and Meta-Framework MMT