Representing Knowledge and Querying Data using Double-Functorial Semantics

ACT 2024

Michael Lambert and Evan Patterson

17 June 2024

Question: What is “knowledge representation?”

Partial answer: OLOGs “ontology logs” [KS12]

OLOGs are schematic representations of concepts, their attributes, and facts about them. Example: arginine

$$\begin{tikzcd}
\boxed{\text{arginine}} \arrow[r, "\text{has}"] \arrow[d, "\text{is}"'] & \boxed{\text{an electrically-charged side chain}} \arrow[d, "\text{is}"] \\
\boxed{\text{an amino acid}} \arrow[r, "\text{has}"'] & \boxed{\text{a side chain}}
\end{tikzcd}$$

Identify:

1. objects = concepts
2. attributes = arrows
3. facts = commutative diagrams

OLOG $\approx$ small category (finitely-presented)

Certain type of database scheme [Spi12]

Functors $\mathscr{C} \to \mathbf{Set}$ thought of as populating a scheme $\mathscr{C}$ with actual data in the form of tables

Example: let $\mathscr{C}$ denote the scheme

$$\begin{tikzcd}
\boxed{\text{Employee}} \arrow[loop left, "\text{manager}"] \arrow[r, "\text{isIn}"] \arrow[d, "\text{first}"'] \arrow[dr, "\text{last}"] & \boxed{\text{Department}} \\
\boxed{\text{String1}} & \boxed{\text{String2}}
\end{tikzcd}$$

Example: data table on $\mathscr{C}$ given by

Employee				
ID	**First**	**Last**	**Mgr**	**isIn**
101	David	Hilbert	103	q10
102	Bertrand	Russell	102	x02
103	Alan	Turing	103	q10

Each arrow out of $\boxed{\text{Employee}}$ is a column; $\boxed{\text{Employee}}$ names the table; ends up being functional: David Hilbert has one manager and works in one department etc...

“Bicategorical or Relational OLOGs” [Pat17]

1. a small (finitely-presented) ‘bicategory of relations’ [CW87]; in particular a cartesian bicategory
2. arrows encode relations rather than functions
3. more expressive: “friend of,” “enemy of”
4. functional relationships encoded as *maps*
5. Drawback: cartesian bicategories not so intuitive; more equations; have to make choices when setting up an OLOG; equational reasoning hard

If only there were a way to combine or synthesize the theories of functional and relational OLOGs.

A **double category** $\mathbb{D}$ is a pseudo-category internal to categories.

1. $\mathbb{D}_0 =$ category of objects and arrows
2. $\mathbb{D}_1 =$ category of proarrows and cells
3. cells look like

$$\begin{tikzcd} A \arrow[r, "m", "|"{marking}] \arrow[d, "f"'] \arrow[dr, phantom, "\Downarrow \theta"] & B \arrow[d, "g"] \\ C \arrow[r, "n"', "|"{marking}] & D \end{tikzcd}$$

4. $\otimes$ external composition associative up to coherent isomorphism; external identity functor $\mathrm{y} \colon \mathbb{D}_0 \to \mathbb{D}_1$

Many examples: sets and spans, sets and relations, categories and profunctors, rings and modules, metric spaces, posets

Any bicategory is a double category without ordinary arrows and cells with trivial external source and target.

Any 2-category is a double category without proarrows and only cells with trivial internal domain and codomain.

Likewise any double category has an underlying (2-)category (functional part) and an underlying bicategory (relational part).

Thus, just as a cartesian bicategory is the basis of a relational OLOG, we expect that a cartesian double category is a possible generaization incorporating the missing "functional aspects."

A double category $\mathbb{D}$ is **cartesian** [Ale18] if $\mathbb{D} \to \mathbb{D} \times \mathbb{D}$ and $\mathbb{D} \to 1$ have right adjoints in the 2-category of double categories, pseudo double functors and (vertical) transformations.

A double category is an **equipment** if $\mathbb{D}_1 \to \mathbb{D}_0 \times \mathbb{D}_0$ is a (bi)fibration (equivalently if every ordinary arrow has a proarrow companion and proarrow conjoint).

Theorem ([Lam])
The bicategory underlying any locally posetal cartesian equipment is a locally posetal cartesian bicategory.

Define: a 'double category of relations' is a locally posetal cartesian equipment satisfying Carboni and Walter's Frobenius Axiom. A **double-categorical OLOG** is then a small 'double category of relations'.

That $\mathbb{D}$ is an equipment means that $\mathbb{D}$ has all *restrictions* and *extensions*:

$$\begin{tikzcd} A \arrow[d, "f"'] & B \arrow[d, "g"] \\ C \arrow[r, "n"', "|" marking] & D \end{tikzcd} \quad \rightsquigarrow \quad \begin{tikzcd} A \arrow[r, "f_! \otimes n \otimes g^*", "|" marking] \arrow[d, "f"'] \arrow[dr, phantom, "\text{restr}"] & B \arrow[d, "g"] \\ C \arrow[r, "n"', "|" marking] & D \end{tikzcd}$$

and

$$\begin{tikzcd} A \arrow[r, "m", "|" marking] \arrow[d, "f"'] & B \arrow[d, "g"] \\ C & D \end{tikzcd} \quad \rightsquigarrow \quad \begin{tikzcd} A \arrow[r, "m", "|" marking] \arrow[d, "f"'] \arrow[dr, phantom, "\text{ext}"] & B \arrow[d, "g"] \\ C \arrow[r, "f^* \otimes n \otimes g_!"', "|" marking] & D \end{tikzcd}$$

Restrictions are cartesian and extensions are opcartesian with respect to the source-target projection $\mathbb{D}_1 \to \mathbb{D}_0 \times \mathbb{D}_0$.

In the case of relations, restrictions are computed by pullback and extensions are images:

$$\begin{tikzcd}
P \arrow[r, dashed] \arrow[d, dashed] \arrow[dr, phantom, very near start, "\lrcorner"] & S \arrow[d, "\langle s,t \rangle"] \\
A \times B \arrow[r, "f \times g"'] & C \times D
\end{tikzcd}
\qquad
\begin{tikzcd}
R \arrow[r, dashed, two heads] \arrow[d, "\langle s,t \rangle"'] & I \arrow[d, dashed, tail] \\
A \times B \arrow[r, "f \times g"'] & C \times D
\end{tikzcd}$$

In general, restrictions and extensions allow us to formulate complex propositions in double-categorical OLOGs and manipulate instance data.

That $\mathbb{D}$ is cartesian and an equipment means that

1. $\mathbb{D}_0$ and $\mathbb{D}_1$ each have finite products
2. external structure functors preserve them
3. consequently $\mathbb{D}$ has *local products*:

$$\begin{tikzcd}
A \arrow[r, "\top", "|"{marking}] \arrow[d, "!"'] \arrow[dr, phantom, "\text{restr}"] & A \arrow[d, "!"] \\
1 \arrow[r, "\mathrm{y}_1"', "|"{marking}] & 1
\end{tikzcd}
\qquad
\begin{tikzcd}
A \arrow[r, "m \wedge n", "|"{marking}] \arrow[d, "\Delta"'] \arrow[dr, phantom, "\text{restr}"] & B \arrow[d, "\Delta"] \\
A \times A \arrow[r, "m \times n"', "|"{marking}] & B \times B
\end{tikzcd}$$

Example time: Stargate SG1

Figure 1: SG1

Summary: MacGyver leads USAF team through a wormhole gate to various planets in search of tech and allies to defend against the threat of the Goa'uld (basically ancient aliens).

Posit two individuals

$$\begin{tikzcd} 1 \arrow[d, "\text{Apophis}"'] & & 1 \arrow[d, "\text{Teal'c}"] \\ \boxed{\text{system lord}} & & \boxed{\text{jaffa}} \end{tikzcd}$$

constants of type "system lord" (rulers of the Goa'uld) and type "jaffa" (warrior-slaves of the system lords).

Figure 2: Apophis

Figure 3: Teal'c

Functional OLOG style of a certain fact: “Teal’c is (or rather was) the first prime of Apophis”

$$\begin{tikzcd}
 & 1 \arrow[dl, "\text{Apophis}"'] \arrow[dr, "\text{Teal'c}"] & \\
\boxed{\text{system lord}} \arrow[rr, "\text{first prime}"'] & & \boxed{\text{jaffa}}
\end{tikzcd}$$

Just ask that the diagram commutes.

But perhaps we’d like to view “first prime” as a relation.

Start with a niche:

$$\begin{tikzcd}
1 \arrow[d, "\text{Apophis}"'] & 1 \arrow[d, "\text{Teal'c}"] \\
\boxed{\text{system lord}} \arrow[r, "\shortmid"{marking}, "\text{first prime}"'] & \boxed{\text{jaffa}}
\end{tikzcd}$$

complete using a restriction:

$$\begin{tikzcd}
1 \arrow[r, "\shortmid"{marking}, "\text{Teal'c is First Prime of Apophis}"] \arrow[d, "\text{Apophis}"'] \arrow[dr, phantom, "\text{restr}"] & 1 \arrow[d, "\text{Teal'c}"] \\
\boxed{\text{system lord}} \arrow[r, "\shortmid"{marking}, "\text{first prime}"'] & \boxed{\text{jaffa}}
\end{tikzcd}$$

Then impose the requirement that the top proarrow is the terminal object in $\mathbb{D}(1,1)$ (or just admits an arrow from the terminal).

So, how do we get from OLOGs to data manipulation?

Innovation: model theory

Recall: a structure-preserving functor on a small category $M\colon \mathscr{C} \to \mathscr{D}$ is a model of $\mathscr{C}$, viewed as a theory (finite products, finite limits, a topos etc) in suitably structured $\mathscr{D}$

A **data instance** is a cartesian double functor $\mathbb{D} \to \mathbb{R}\mathbf{el}$, that is, a "model" of $\mathbb{D}$ in relations. (Note: any double functor preserves restrictions and extensions.)

Upshot: this reduces data manipulation to functorial semantics and computations of pullbacks and images in $\mathbb{R}\mathbf{el}$.

In [Spi12] get data manipulation via adjoint functors. Given database scehmes $\mathscr{C}$ and $\mathscr{D}$, together with a translation $F\colon \mathscr{C} \to \mathscr{D}$, there are adjoint functors, namely,

$$\begin{tikzcd}[\mathscr{D}, \mathbf{Set}] \arrow[r, "F^*" description] & {[\mathscr{C}, \mathbf{Set}]} \arrow[l, bend right, "\Pi_F"'] \arrow[l, bend left, "\Sigma_F"]\end{tikzcd} \qquad \Sigma_F \dashv F^* \dashv \Pi_F$$

Substitution behaves like a select query; the right adjoint provides an inner join; the left adjoint is a skolemized outer join.

Using double categories we can do these operations without translations of schemes, hence without adjoint functors, by simply setting up the OLOGs carefully.

Select queries, filtering, joins end up being computations of images and pullbacks in $\mathbb{R}\mathbf{el}$.

Simple OLOG describing a “mission concept”

$$\boxed{\text{date}} \times \boxed{\text{location}} \xrightarrow{\text{mission}} \boxed{\text{purpose}} \times \boxed{\text{team}}$$

instanced by a table

mission			
date	**location**	**purpose**	**team**
2-6-98	P41-771	search & rescue	SG3
7-31-98	Cimmeria	assist Cimmerians	SG1
1-2-99	P3R-272	investigate inscriptions	SG1
10-22-99	Ne’tu	search & rescue	SG1
8-6-2004	Tegalus	negotiation	SG9

Perhaps we only care about when our teams were off-world but not about either the date or purpose. Extend along projections:

$$\begin{tikzcd}
\boxed{\text{date}} \times \boxed{\text{location}} \arrow[r, "\text{mission}", "\shortmid"{marking}] \arrow[d, "\pi_1"'] \arrow[dr, phantom, "\text{ext}"] & \boxed{\text{purpose}} \times \boxed{\text{team}} \arrow[d, "\pi_2"] \\
\boxed{\text{date}} \arrow[r, "\text{date and team}"', "\shortmid"{marking}] & \boxed{\text{team}}
\end{tikzcd}$$

The instance $M\colon \mathbb{D} \to \mathbb{R}\mathbf{el}$ preserves the extension.

Compute the image in relations.

date and team	
date	**team**
2-6-98	SG3
7-31-98	SG1
1-2-99	SG1
10-22-99	SG1
8-6-2004	SG9

This would be a “select” operation in SQL.

We might also filter by those missions of a certain team, say, SG1.

Start by asking for an individual constant:

$$1 \xrightarrow{\text{SG1}} \boxed{\text{team}}$$

and form the restriction cell

$$\begin{tikzcd}
\boxed{\text{date}} \times \boxed{\text{location}} \arrow[r, "\text{SG1 missions}", "\shortmid"{marking}] \arrow[d, "1"'] \arrow[dr, phantom, "\text{restr}"] & \boxed{\text{purpose}} \times 1 \arrow[d, "1\times\text{SG1}"] \\
\boxed{\text{date}} \times \boxed{\text{location}} \arrow[r, "\text{mission}"', "\shortmid"{marking}] & \boxed{\text{purpose}} \times \boxed{\text{team}}
\end{tikzcd}$$

In this case, the returned data would be a subtable consisting only of certain rows of the original **mission** table, that is,

SG1 missions		
date	**location**	**purpose**
7-31-98	Cimmeria	assist Cimmerians
1-2-99	P3R-272	investigate inscriptions
10-22-99	Ne'tu	search & rescue

that is, "filter" for rows with a certain value in SQL.

Indeed there are other examples of joins using a construction like local products.

We can also do (inner) joins of tables with a column in common. In general, this looks like:

$$\begin{tikzcd}
A \times B \arrow[r, "p \bowtie q", "\shortmid"{marking}] \arrow[d, "1 \times \Delta"'] \arrow[dr, phantom, "\text{restr}"] & C \times D \arrow[d, "1"] \\
A \times B \times B \arrow[r, "p \times q"', "\shortmid"{marking}] & C \times D
\end{tikzcd}$$

Think of this as giving two tables, namely, p and q with entries from sets A, B, C and D. The tables have entries in one column in common, namely, those from the set B.

A simple OLOG:

$$\begin{tikzcd} \boxed{\text{person}} \arrow[r, "\text{expertise}"'] & \boxed{\text{skill}} \end{tikzcd}$$

$$\begin{tikzcd} \boxed{\text{person}} \arrow[r, "\text{membership}"'] & \boxed{\text{team}} \end{tikzcd}$$

Data:

expertise	
person	**skill**
Hammond	command
Kovacek	law
Maybourne	chicanery
Morrison	combat
O'Neill	command
Rothman	archaeology
Simmons	sociopathy
Warren	combat

membership	
person	**team**
Kovacek	SG9
Morrison	SG3
O'Neill	SG1
Rothman	SG11
Warren	SG3

Form the restriction cell:

$$\begin{tikzcd}
\boxed{\text{person}} \arrow[r, "\text{expertise} \bowtie \text{membership}", "|" marking] \arrow[d, "\Delta"'] \arrow[dr, phantom, "\text{restr}"] & \boxed{\text{skill}} \times \boxed{\text{team}} \arrow[d, "1"] \\
\boxed{\text{person}} \times \boxed{\text{person}} \arrow[r, "\text{expertise} \times \text{membership}"', "|" marking] & \boxed{\text{skill}} \times \boxed{\text{team}}
\end{tikzcd}$$

In relations this is just computed as a pullback of the product relation along the diagonal crossed with the identity.

That is, form the cartesian product of all pairs from the two relations and match the $\boxed{\text{person}}$ argument:

expertise ⋈ membership		
person	**skill**	**team**
Kovacek	law	SG9
Morrison	combat	SG3
O'Neill	command	SG1
Rothman	archaeology	SG11
Warren	combat	SG3

which is the join of the two along the common column.

So, it appears we can replicate SQL-type data operations without recourse to adjoint functors. This is “data munging.”

Figure 4: Ologging Cimmerians

Julia Language (2009 Bezanson et. al. https://julialang.org/)

1. high-level, fast, used in scientific computing, numerical analysis, alternative to R and Python
2. macros like LISP
3. key feature: multiple dispatch (can overload a function symbol as long as you're clear what types arguments have)
4. Algebraic Julia and CatLab (https://github.com/AlgebraicJulia/Catlab.jl): category theory enhancing capabilities of scientific computing
5. machinery of GATs (generalized algebraic theories): categories, monoidal categories, bicategories, (now) double categories, cartesian double categories, and equipments

Remaining questions:

1. outer joins?
2. program relation-valued double functors
3. query language?
4. translations of double-categorical schemes for full data migration; adjoints between virtual double categories of (lax) double functors?

For more you can see

1. paper draft: https://arxiv.org/abs/2403.19884
2. blog post: https://topos.site/blog/2022/09/data-operations-are-functorial-semantics/

THANK YOU!

Figure 5: Golf time

References

Evangelia Aleiferi.
Cartesian Double Categories with an Emphasis on Characterizing Spans.
PhD thesis, Dalhousie University, 2018.

A. Carboni and R.C.F. Walters.
Cartesian bicategories I.
Journal of Pure and Applied Algebra, 49:11–32, 1987.

Robert E. Kent and David I. Spivak.
Ologs: A categorical framework for knowledge representation.
PLoS ONE, 7(1), 2012.

Michael Lambert.
'Double Categories of Relations'.
Theory and Applications of Categories (to appear).

Evan Patterson.
Knowledge representation in bicategories of relations.
2017.
https://arxiv.org/abs/1706.00526.

David I. Spivak.
Functorial data migration.
Information and Computation, 217:31–51, 2012.