Thin spans and their modelling of rigid intersection types

Pierre Clairambault[1] Simon Forest[2]

[1]LIS, CNRS

[2]LIS, Aix-Marseille Université

November 16, 2023

Semantic models

What is a semantic model?

$$\texttt{program} \quad \mapsto \quad \mathcal{S}ome\ \mathcal{M}athematical\ \mathcal{O}bject$$

Thin spans

What are thin spans?

- a semantic model which represents programs through **witnesses of computation**

$$A \vdash s : B \qquad \rightsquigarrow \qquad \begin{tikzcd} & S \arrow[dl] \arrow[dr] & \\ !A & & B \end{tikzcd}$$

- a (bi)categorical abstraction of concurrent game semantics
- a proof-relevant refinement of the **relational model** of linear logic

Thin spans

What are thin spans?

- a semantic model which represents programs through **witnesses of computation**

$$A \vdash s : B \quad \rightsquigarrow \quad \begin{tikzcd} & S \arrow[dl] \arrow[dr] & \\ !A & & B \end{tikzcd}$$

- a (bi)categorical abstraction of concurrent game semantics
- a proof-relevant refinement of the **relational model** of linear logic

Thin spans

What are thin spans?

- a semantic model which represents programs through **witnesses of computation**

$$A \vdash s : B \qquad \rightsquigarrow \qquad \begin{tikzcd} & S \arrow[dl] \arrow[dr] & \\ !A & & B \end{tikzcd}$$

- a (bi)categorical abstraction of concurrent game semantics
- a proof-relevant refinement of the **relational model** of linear logic

Outline

From relations to spans

Interpreting programs

A rigid intersection type system

Outline

From relations to spans

Interpreting programs

A rigid intersection type system

CBN λ-calculus with effects

Consider your favorite λ-calculus and add to it an effectful operator, like a **non-deterministic operator** $\oplus$

$$s, t, u, \ldots \quad ::= \quad x \in \mathrm{Var} \quad | \quad t\ u \quad | \quad \lambda x.t \quad | \quad n \quad | \quad \cdots \quad | \quad s \oplus t$$

so that the same program can reduce to different values:

$$3 \oplus 4 \quad \to \quad 3 \qquad\qquad 3 \oplus 4 \quad \to \quad 4$$

CBN λ-calculus with effects

A program like $\vdash \lambda x.x * x : \mathbf{Nat} \to \mathbf{Nat}$ can reduce to 9, 12 or 16 on the input $3 \oplus 4$:

$$(\lambda x.x * x)\ (3 \oplus 4) \quad \to \quad (3 \oplus 4) * (3 \oplus 4) \quad \to^* \quad 9 \text{ or } 12 \text{ or } 16$$

How to describe the semantics of a CBN program p?
Idea: use "**bags**" to represent the outcomes of arguments of programs.

For the program $\lambda x.x * x$:

$$\begin{array}{ccc} (x \leftarrow [3,3]) & \mapsto & 9 \\ (x \leftarrow [3,4]) & \mapsto & 12 \\ (x \leftarrow [4,4]) & \mapsto & 16 \end{array}$$

CBN λ-calculus with effects

A program like $\vdash \lambda x.x * x : \mathbf{Nat} \to \mathbf{Nat}$ can reduce to 9, 12 or 16 on the input $3 \oplus 4$:

$$(\lambda x.x * x)\ (3 \oplus 4) \quad \to \quad (3 \oplus 4) * (3 \oplus 4) \quad \to^* \quad 9 \text{ or } 12 \text{ or } 16$$

How to describe the semantics of a CBN program p?
Idea: use "**bags**" to represent the outcomes of arguments of programs.

For the program $\lambda x.x * x$:

$$\begin{array}{ccc} (x \leftarrow [3,3]) & \mapsto & 9 \\ (x \leftarrow [3,4]) & \mapsto & 12 \\ (x \leftarrow [4,4]) & \mapsto & 16 \end{array}$$

CBN λ-calculus with effects

A program like $\vdash \lambda x.x * x : \mathbf{Nat} \to \mathbf{Nat}$ can reduce to 9, 12 or 16 on the input $3 \otimes 4$:

$$(\lambda x.x * x)\,(3 \otimes 4) \quad \to \quad (3 \otimes 4) * (3 \otimes 4) \quad \to^* \quad 9 \text{ or } 12 \text{ or } 16$$

How to describe the semantics of a CBN program p?
Idea: use "**bags**" to represent the outcomes of arguments of programs.

For the program $\lambda x.x * x$:

$$\begin{array}{ccc} (x \leftarrow [3,3]) & \mapsto & 9 \\ (x \leftarrow [3,4]) & \mapsto & 12 \\ (x \leftarrow [4,4]) & \mapsto & 16 \end{array}$$

CBN λ-calculus with effects

More generally, the outputs of $\lambda x.x * x$ can be correctly described by a (partial) function

$$f : !\mathbb{N} \to \mathbb{N}$$

where $!\mathbb{N}$ is the set of "bags" on $\mathbb{N}$.

But more general terms of type **Nat** $\to$ **Nat** can involve non-determinism, so that their interpretation should be a function

$$f : !\mathbb{N} \to \mathcal{P}(\mathbb{N})$$

or, equivalently, a **relation** $f \subseteq !\mathbb{N} \times \mathbb{N}$ $\rightsquigarrow$ the relational model **Rel**

CBN λ-calculus with effects

More generally, the outputs of $\lambda x.x * x$ can be correctly described by a (partial) function

$$f : !\mathbb{N} \to \mathbb{N}$$

where $!\mathbb{N}$ is the set of "bags" on $\mathbb{N}$.

But more general terms of type **Nat** $\to$ **Nat** can involve non-determinism, so that their interpretation should be a function

$$f : !\mathbb{N} \to \mathcal{P}(\mathbb{N})$$

or, equivalently, a **relation** $f \subseteq !\mathbb{N} \times \mathbb{N} \rightsquigarrow$ the relational model **Rel**

The model **Rel** of **LL**

Objects: sets A, B, C, *etc*.

Morphisms $A \to B$: **relations** $R \subseteq A \times B$, i.e., sets of elements

$$a \multimap b$$

Exponential: $!A$ is $\mathcal{M}_{\mathrm{fin}}(A)$, the set of **finite multisets** on A

(co)Kleisli category $\mathbf{Rel}_!$: morphisms $A \to B$ are morphisms $!A \to B$ of **Rel**, that is, sets of elements

$$[a_1, \ldots, a_n] \multimap b$$

Interpreting programs in $\mathbf{Rel}_!$

Since $\mathbf{Rel}_!$ is **cartesian closed**, one can interpret **programs** inside it.

$$x : \mathbf{Bool} \vdash \mathbf{if}\ x\ \mathbf{then\ ff\ else\ tt} : \mathbf{Bool}$$

interpreted as

$$\{\ [\mathbf{tt}] \multimap \mathbf{ff}, \quad [\mathbf{ff}] \multimap \mathbf{tt}\ \} \quad (\subseteq \mathcal{M}_{\text{fin}}(\mathbf{Bool}) \times \mathbf{Bool})$$

Interpreting programs in $\mathbf{Rel}_!$

Since $\mathbf{Rel}_!$ is **cartesian closed**, one can interpret **programs** inside it.

$$x : \mathbf{Bool} \vdash \mathbf{if}\ x\ \mathbf{then}\ (\mathbf{if}\ x\ \mathbf{then\ ff\ else\ tt})\ \mathbf{else}\ (\mathbf{if}\ x\ \mathbf{then\ tt\ else\ ff}) : \mathbf{Bool}$$

interpreted as

$$\{\ [\mathbf{tt}, \mathbf{tt}] \multimap \mathbf{ff}, \quad [\mathbf{tt}, \mathbf{ff}] \multimap \mathbf{tt}, \quad [\mathbf{ff}, \mathbf{ff}] \multimap \mathbf{ff}\ \}$$

Interpreting programs in $\mathbf{Rel}_!$

Since $\mathbf{Rel}_!$ is **cartesian closed**, one can interpret **programs** inside it.

$$x : \mathbf{Bool} \vdash \mathbf{if}\ x\ \mathbf{then}\ (\mathbf{if}\ x\ \mathbf{then}\ \mathbf{ff}\ \mathbf{else}\ \mathbf{tt})\ \mathbf{else}\ (\mathbf{if}\ x\ \mathbf{then}\ \mathbf{tt}\ \mathbf{else}\ \mathbf{ff}) : \mathbf{Bool}$$

interpreted as

$$\{\ [\mathbf{tt}, \mathbf{tt}] \multimap \mathbf{ff}, \quad [\mathbf{tt}, \mathbf{ff}] \multimap \mathbf{tt}, \quad [\mathbf{ff}, \mathbf{ff}] \multimap \mathbf{ff}\ \}$$

Here, two different executions get identified in the interpretation.

Interpreting programs in $\mathbf{Rel}_!$

Since $\mathbf{Rel}_!$ is **cartesian closed**, one can interpret **programs** inside it.

$$x : \mathbf{Bool} \vdash \mathbf{if}\ x\ \mathbf{then}\ (\mathbf{if}\ x\ \mathbf{then}\ \mathbf{ff}\ \mathbf{else}\ \mathbf{tt})\ \mathbf{else}\ (\mathbf{if}\ x\ \mathbf{then}\ \mathbf{tt}\ \mathbf{else}\ \mathbf{ff}) : \mathbf{Bool}$$

interpreted as

$$\{\ [\mathbf{tt}, \mathbf{tt}] \multimap \mathbf{ff}, \quad [\mathbf{tt}, \mathbf{ff}] \multimap \mathbf{tt}, \quad [\mathbf{ff}, \mathbf{ff}] \multimap \mathbf{ff}\ \}$$

Here, two different executions get identified in the interpretation.

Hence, $\mathbf{Rel}_!$ aggregates different executions.

Witnesses of executions

How do we represent the different possible executions of a program?

Example of a program with non-determinism:

$$x : \mathbf{Nat}, y : \mathbf{Nat} \vdash x \circledvee y : \mathbf{Nat}$$

has the executions

$$\begin{array}{lcl} x \leftarrow 40, \quad y \leftarrow 2 & \rightsquigarrow & \texttt{output} = 40 \\ & \rightsquigarrow & \texttt{output} = 2 \end{array}$$

Executions are described by **witnesses**: triples (inputs,outputs,reason).

Example: $([x \leftarrow 40, y \leftarrow 2], 40, \circledvee$ chose left$)$ for the first execution.

Witnesses of executions

How do we represent the different possible executions of a program?

Example of a program with non-determinism:

$$x : \mathbf{Nat}, y : \mathbf{Nat} \vdash x \varovee y : \mathbf{Nat}$$

has the executions

$$x \leftarrow 40, \quad y \leftarrow 2 \qquad \leadsto \qquad \mathtt{output} = 40$$
$$\leadsto \qquad \mathtt{output} = 2$$

Executions are described by **witnesses**: triples (inputs,outputs,reason).

Example: $([x \leftarrow 40, y \leftarrow 2], 40, \varovee$ chose left) for the first execution.

Witnesses of executions

How do we represent the different possible executions of a program?

Example of a program with non-determinism:

$$x : \mathbf{Nat}, y : \mathbf{Nat} \vdash x \varovee y : \mathbf{Nat}$$

has the executions

$$\begin{array}{lcl} x \leftarrow 40, \quad y \leftarrow 2 & \rightsquigarrow & \texttt{output} = 40 \text{ because } \varovee \text{ chose left} \\ & \rightsquigarrow & \texttt{output} = 2 \text{ because } \varovee \text{ chose right} \end{array}$$

Executions are described by **witnesses**: triples (inputs,outputs,reason).

Example: $([x \leftarrow 40, y \leftarrow 2], 40, \varovee$ chose left) for the first execution.

Witnesses as spans

Witnesses of executions of a program $A \to B$: set of triples

$$S = \{ (a_i, b_i, r_i) \mid i \in \mathcal{I} \}.$$

There are canonical projections to A and B, so that S is in fact a **span**

$$\begin{tikzcd} & S \arrow[dl, "\partial^S_A"'] \arrow[dr, "\partial^S_B"] & \\ A & & B \end{tikzcd}$$

Spans can be seen as **generalized relations**: 0, 1 or several "proofs" that $a \in A$ and $b \in B$ are related.

Spans and composition

What about composition?

program s:	program t:
$x : \mathbf{Nat} \vdash x \ovee_1 (x+2) : \mathbf{Nat}$	$y : \mathbf{Nat} \vdash y \ovee_2 (2y) : \mathbf{Nat}$

Spans and composition

What about composition?

$$\text{program } s: \qquad\qquad \text{program } t:$$

$$x : \mathbf{Nat} \vdash x \varovee_1 (x+2) : \mathbf{Nat} \qquad y : \mathbf{Nat} \vdash y \varovee_2 (2y) : \mathbf{Nat}$$

Witnesses between 0 and 2 of "$t \circ s$":

- $\varovee_1$ chose right ($s[x \leftarrow 0] \rightsquigarrow 2$) and $\varovee_2$ chose left ($t[y \leftarrow 2] \rightsquigarrow 2$).

Spans and composition

What about composition?

$$\text{program } s: \qquad\qquad \text{program } t:$$

$$x : \mathbf{Nat} \vdash x \varovee_1 (x+2) : \mathbf{Nat} \qquad y : \mathbf{Nat} \vdash y \varovee_2 (2y) : \mathbf{Nat}$$

Witnesses between 0 and 0 of "$t \circ s$":

- $\varovee_1$ chose left ($s[x \leftarrow 0] \rightsquigarrow 0$) and $\varovee_2$ chose left ($t[y \leftarrow 0] \rightsquigarrow 0$); or
- $\varovee_1$ chose left ($s[x \leftarrow 0] \rightsquigarrow 0$) and $\varovee_2$ chose right ($t[y \leftarrow 0] \rightsquigarrow 0$).

Spans and composition

What about composition?

program s:	program t:
$x : \mathbf{Nat} \vdash x \mathbin{\varovee_1} (x+2) : \mathbf{Nat}$	$y : \mathbf{Nat} \vdash y \mathbin{\varovee_2} (2y) : \mathbf{Nat}$

Witnesses between 2 and 4 of "$t \circ s$":

- $\varovee_1$ chose right ($s[x \leftarrow 2] \rightsquigarrow 4$) and $\varovee_2$ chose left ($t[y \leftarrow 4] \rightsquigarrow 4$); or
- $\varovee_1$ chose left ($s[x \leftarrow 2] \rightsquigarrow 2$) and $\varovee_2$ chose right ($t[y \leftarrow 2] \rightsquigarrow 4$).

Spans and composition

What about composition?

$$x : A \vdash s : B \qquad\qquad y : B \vdash t : C$$

$$S = \{ (a_i, b_i, r_i) \mid i \in \mathcal{I} \} \qquad T = \{ (b'_j, c_j, s_j) \mid j \in \mathcal{J} \}.$$

Spans and composition

What about composition?

$$x : A \vdash s : B \qquad\qquad y : B \vdash t : C$$

$$S = \{\, (a_i, b_i, r_i) \mid i \in \mathcal{I} \,\} \qquad T = \{\, (b'_j, c_j, s_j) \mid j \in \mathcal{J} \,\}.$$

Witnesses of "$t \circ s$":

$$T \odot S = \{\, (a_i, c_j, (r_i, s_j)) \quad \mid \quad (a_i, b_i, r_i) \in S, \quad (b'_j, c_j, s_j) \in T, \quad b_i = b'_j \,\}$$

Spans and composition

What about composition?

$$x : A \vdash s : B \qquad\qquad y : B \vdash t : C$$

$$S = \{\, (a_i, b_i, r_i) \mid i \in \mathcal{I} \,\} \qquad T = \{\, (b'_j, c_j, s_j) \mid j \in \mathcal{J} \,\}.$$

Witnesses of "$t \circ s$":

$$\begin{tikzcd}
 & & T \odot S \arrow[dl, dashed] \arrow[dr, dashed] \arrow[dd, phantom, "\lrcorner" very near start, rotate=-45] & & \\
 & S \arrow[dl, "\partial^S_A"'] \arrow[dr, "\partial^S_B"] & & T \arrow[dl, "\partial^T_B"'] \arrow[dr, "\partial^T_C"] & \\
A & & B & & C
\end{tikzcd}$$

Towards a semantic model of spans

Spans can be used to describe the semantics of toy examples and compose them.

But what about more complex examples?

- CBN and effects, lambda abstractions, higher-order functions...

Idea: follow the constructions on **Rel**

- define a model of linear logic based on spans
- derive a cartesian closed (bi)category, in which we can interpret programs

A first bicategory of spans

Before defining a model of **LL**, we must start with some categorical structure.

Pullbacks are defined up to isomorphism
$\rightsquigarrow$ associativity of composition $\odot$ is expressed by a 2-dimensional structure

Given two spans $S, T \colon A \to B$, a **morphism** between S and T is $m \colon S \to T$ such that

$$\begin{tikzcd} S \arrow[rr, "m"] \arrow[dr, "\partial_A^S"'] \arrow[rr, phantom, shift right=5ex, "="] & & T \arrow[dl, "\partial_A^T"] \\ & A & \end{tikzcd} \quad \text{and} \quad \begin{tikzcd} S \arrow[rr, "m"] \arrow[dr, "\partial_B^S"'] \arrow[rr, phantom, shift right=5ex, "="] & & T \arrow[dl, "\partial_B^T"] \\ & B & \end{tikzcd} .$$

One gets a bicategory **Span** $=$ **Span**(**Set**) of sets, spans and morphisms of spans.

Some structure on **Span**

The cocartesian structure of **Set** translates to a cartesian structure on **Span**.

Some structure on **Span**

The cocartesian structure of **Set** translates to a cartesian structure on **Span**.

$\top \triangleq \emptyset$ is the terminal object of **Span**.

$$
\begin{tikzcd}
 & \emptyset \arrow[dl, "{[]}"'] \arrow[dr, "{[]}"] & \\
A & & \emptyset
\end{tikzcd}
$$

Some structure on **Span**

The cocartesian structure of **Set** translates to a cartesian structure on **Span**.

$A \mathbin{\&} B \triangleq A \sqcup B$ is the cartesian product on **Span**.

$$
\begin{tikzcd}
 & S \arrow[dl, "\partial^S_X"'] \arrow[dr, "\partial^S_A"] & \\
X & & A
\end{tikzcd}
\qquad
\begin{tikzcd}
 & T \arrow[dl, "\partial^T_X"'] \arrow[dr, "\partial^T_B"] & \\
X & & B
\end{tikzcd}
\qquad \rightsquigarrow \qquad
\begin{tikzcd}
 & S \sqcup T \arrow[dl, "{[\partial^S_X, \partial^T_X]}"'] \arrow[dr, "\partial^S_A \sqcup \partial^S_B"] & \\
X & & A \sqcup B
\end{tikzcd}
$$

$$S \colon X \nrightarrow A \qquad T \colon X \nrightarrow B \qquad \langle S, T \rangle \colon X \nrightarrow A \mathbin{\&} B$$

Some structure on **Span**

The cartesian structure of **Set** translates to a monoidal structure on **Span**.

$A \otimes B \triangleq A \times B$ gives a tensor product on **Span**.

$$\begin{tikzcd} & S \arrow[dl, "\partial^S_A"'] \arrow[dr, "\partial^S_{A'}"] & \\ A & & A' \end{tikzcd} \qquad \begin{tikzcd} & T \arrow[dl, "\partial^T_B"'] \arrow[dr, "\partial^T_{B'}"] & \\ B & & B' \end{tikzcd} \quad \rightsquigarrow \quad \begin{tikzcd} & S \times T \arrow[dl, "\partial^S_A \times \partial^T_B"'] \arrow[dr, "\partial^S_{A'} \times \partial^S_{B'}"] & \\ A \times B & & A' \times B' \end{tikzcd}$$

$$S \qquad\qquad T \qquad\qquad S \otimes T$$

The exponential issue

An ingredient of an **LL** model: the exponential modality

Do we still have an exponential for **Span**?

The exponential issue

An ingredient of an **LL** model: the exponential modality

Do we still have an exponential for **Span**?

- First try: can we use $\mathcal{M}_{\mathrm{fin}}(-)$ of **Rel** as exponential for **Span**?

  Given $S \in$ **Span**, define

$$\mathcal{M}_{\mathrm{fin}}\left(\begin{tikzcd} & S \arrow[dl, "\partial_A^S"'] \arrow[dr, "\partial_B^S"] & \\ A & & B \end{tikzcd}\right) = \begin{tikzcd} & \mathcal{M}_{\mathrm{fin}}(S) \arrow[dl, "\mathcal{M}_{\mathrm{fin}}(\partial_A^S)"'] \arrow[dr, "\mathcal{M}_{\mathrm{fin}}(\partial_B^S)"] & \\ \mathcal{M}_{\mathrm{fin}}(A) & & \mathcal{M}_{\mathrm{fin}}(B) \end{tikzcd}$$

  Problem: $\mathcal{M}_{\mathrm{fin}}$ **does not respect composition**, because pullbacks are not preserved. Thus, not a functor **Span** $\to$ **Span**.

The exponential issue

An ingredient of an **LL** model: the exponential modality

Do we still have an exponential for **Span**?

- First try: can we use $\mathcal{M}_{\mathrm{fin}}(-)$ of **Rel** as exponential for **Span**? No.
- Second try: can we use lists as exponential?

$$a_1, \ldots, a_n \in A \quad \rightsquigarrow \quad [a_1; \cdots ; a_n] \in \mathbf{List}(A)$$

The exponential issue

An ingredient of an **LL** model: the exponential modality

Do we still have an exponential for **Span**?

- First try: can we use $\mathcal{M}_{\mathrm{fin}}(-)$ of **Rel** as exponential for **Span**? No.
- Second try: can we use lists as exponential?

$$a_1, \ldots, a_n \in A \quad \rightsquigarrow \quad [a_1; \cdots; a_n] \in \mathbf{List}(A)$$

$$\mathbf{List}\left(\begin{tikzcd} & S \arrow[dl, "\partial_A^S"'] \arrow[dr, "\partial_B^S"] & \\ A & & B \end{tikzcd}\right) = \begin{tikzcd} & \mathbf{List}(S) \arrow[dl, "\mathbf{List}(\partial_A^S)"'] \arrow[dr, "\mathbf{List}(\partial_B^S)"] & \\ \mathbf{List}(A) & & \mathbf{List}(B) \end{tikzcd}$$

The exponential issue

An ingredient of an **LL** model: the exponential modality

Do we still have an exponential for **Span**?

- First try: can we use $\mathcal{M}_{\mathrm{fin}}(-)$ of **Rel** as exponential for **Span**? No.
- Second try: can we use lists as exponential?

$$a_1, \ldots, a_n \in A \quad \rightsquigarrow \quad [a_1; \cdots; a_n] \in \mathbf{List}(A)$$

$$\mathbf{List}\left(\begin{tikzcd} & S \arrow[dl, "\partial_A^S"'] \arrow[dr, "\partial_B^S"] & \\ A & & B \end{tikzcd}\right) = \begin{tikzcd} & \mathbf{List}(S) \arrow[dl, "\mathbf{List}(\partial_A^S)"'] \arrow[dr, "\mathbf{List}(\partial_B^S)"] & \\ \mathbf{List}(A) & & \mathbf{List}(B) \end{tikzcd}$$

We now have a (pseudo)functor, but no **Seely equivalence**

$$\mathrm{see}_{A,B} : \ \mathbf{List}\,A \otimes \mathbf{List}\,B \underset{\longleftarrow}{\overset{\longrightarrow}{\simeq}} \mathbf{List}(A \mathbin{\&} B) : \overline{\mathrm{see}}_{A,B} \qquad \in \mathbf{Span}$$

because $[b_1; a_1; a_2; b_2] \neq [a_1; a_2; b_1; b_2]$: **lack of symmetries**

The exponential issue

An ingredient of an **LL** model: the exponential modality

Do we still have an exponential for **Span**?

- First try: can we use $\mathcal{M}_{\text{fin}}(-)$ of **Rel** as exponential for **Span**? No.
- Second try: can we use lists as exponential? Probably no.

$$a_1, \ldots, a_n \in A \quad \rightsquigarrow \quad [a_1; \cdots ; a_n] \in \mathbf{List}(A)$$

Span is dead, long live **Span**!

Problem:

- our spans are **set-based**
- there is **no adequate Seely equivalence** in this setting

We must change the kind of spans that we use.

Span is dead, long live **Span**!

Let **Gpd** be the 2-category of groupoids, functors and **natural transformations**.
$\rightsquigarrow$ within groupoids, there are symmetries between objects:

$$[b_1; a_1; a_2; b_2] \quad \cong \quad [a_1; a_2; b_1; b_2] \qquad \in \mathbf{List}^\star(A \sqcup B)$$

We now have a Seely equivalence

$$\mathrm{see}_{A,B} : \mathbf{List}^\star A \times \mathbf{List}^\star B \underset{\longleftarrow}{\overset{\longrightarrow}{\simeq}} \mathbf{List}^\star(A \sqcup B) : \overline{\mathrm{see}}_{A,B} \qquad \in \mathbf{Gpd}$$

because the symmetries allow us to reindex:

$$\begin{aligned} \overline{\mathrm{see}}_{A,B} \circ \mathrm{see}_{A,B} \quad &= \quad \mathrm{id}_{\mathbf{List}^\star A \times \mathbf{List}^\star B} \\ \mathrm{see}_{A,B} \circ \overline{\mathrm{see}}_{A,B} \quad &\cong \quad \mathrm{id}_{\mathbf{List}^\star(A \sqcup B)} \end{aligned}$$

Span is dead, long live **Span**!

Let **Gpd** be the 2-category of groupoids, functors and **natural transformations**.
$\rightsquigarrow$ within groupoids, there are symmetries between objects:

$$[b_1; a_1; a_2; b_2] \quad \cong \quad [a_1; a_2; b_1; b_2] \qquad \in \mathbf{List}^\star(A \sqcup B)$$

We now have a Seely equivalence

$$\mathsf{see}_{A,B} : \mathbf{List}^\star A \times \mathbf{List}^\star B \underset{\longleftarrow}{\overset{\longrightarrow}{\simeq}} \mathbf{List}^\star(A \sqcup B) : \overline{\mathsf{see}}_{A,B} \qquad \in \mathbf{Gpd}$$

because the symmetries allow us to reindex:

$$\begin{array}{ccc} \overline{\mathsf{see}}_{A,B} \circ \mathsf{see}_{A,B} & = & \mathrm{id}_{\mathbf{List}^\star A \times \mathbf{List}^\star B} \\ \mathsf{see}_{A,B} \circ \overline{\mathsf{see}}_{A,B} & \cong & \mathrm{id}_{\mathbf{List}^\star(A \sqcup B)} \end{array}$$

Span is dead, long live **Span**!

We (re)define **Span** as **Span**(**Gpd**)

Span is dead, long live **Span**!

We (re)define **Span** as **Span**(**Gpd**)

- objects: groupoids $A, B, \ldots$

Span is dead, long live **Span**!

We (re)define **Span** as **Span**(**Gpd**)

- objects: groupoids $A, B, \ldots$
- 1-morphisms: spans $S, T, \ldots$

$$S \quad \rightsquigarrow \quad \begin{tikzcd} & S \arrow[dl, "\partial_A^S"'] \arrow[dr, "\partial_B^S"] & \\ A & & B \end{tikzcd}$$

Span is dead, long live **Span**!

We (re)define **Span** as **Span**(**Gpd**)

- objects: groupoids $A, B, \ldots$
- 1-morphisms: spans $S, T, \ldots$
- 2-morphisms: **pseudo-commutative** triangles $(F, \phi), (G, \psi), \ldots$

$$(F,\phi)\colon S \Rightarrow T \quad \rightsquigarrow \quad \begin{tikzcd} S \arrow[rr, "F"] \arrow[dr, "\partial^S_A"'] & \arrow[d, phantom, "\overset{\phi^A}{\Longrightarrow}"] & T \arrow[dl, "\partial^T_A"] \\ & A & \end{tikzcd} \quad \text{and} \quad \begin{tikzcd} S \arrow[rr, "F"] \arrow[dr, "\partial^S_B"'] & \arrow[d, phantom, "\overset{\phi^B}{\Longrightarrow}"] & T \arrow[dl, "\partial^T_B"] \\ & B & \end{tikzcd}$$

Span is dead, long live **Span**!

We (re)define **Span** as **Span**(**Gpd**)

- objects: groupoids $A, B, \dots$
- 1-morphisms: spans $S, T, \dots$
- 2-morphisms: **pseudo-commutative** triangles $(F, \phi), (G, \psi), \dots$

$$(F,\phi)\colon S \Rightarrow T \quad \rightsquigarrow \quad \begin{tikzcd} S \arrow[rr, "F"] \arrow[dr, "\partial^S_A"'] & \arrow[d, phantom, "\overset{\phi^A}{\Longrightarrow}"] & T \arrow[dl, "\partial^T_A"] \\ & A & \end{tikzcd} \quad \text{and} \quad \begin{tikzcd} S \arrow[rr, "F"] \arrow[dr, "\partial^S_B"'] & \arrow[d, phantom, "\overset{\phi^B}{\Longrightarrow}"] & T \arrow[dl, "\partial^T_B"] \\ & B & \end{tikzcd}$$

We can now hope that the Seely equivalence of **Gpd** lifts in **Span**(**Gpd**).

The horizontal composition issue

But is $\textbf{Span} = \textbf{Span}(\textbf{Gpd})$ a bicategory?

In particular, are we able to define a **horizontal composition**?

$$
\begin{tikzcd}
& & S \arrow[dll, "\partial^S_A"'] \arrow[drr, "\partial^S_B"] \arrow[dd, "F" description] & & \\
A \arrow[rr, phantom, "\Downarrow\phi^A"] & & & & B \arrow[ll, phantom, "\Downarrow\phi^B"'] \\
& & S' \arrow[ull, "\partial^{S'}_A"] \arrow[urr, "\partial^{S'}_B"'] & &
\end{tikzcd}
\qquad
\begin{tikzcd}
& & T \arrow[dll, "\partial^T_B"'] \arrow[drr, "\partial^T_C"] \arrow[dd, "G" description] & & \\
B \arrow[rr, phantom, "\Downarrow\psi^B"] & & & & C \arrow[ll, phantom, "\Downarrow\psi^C"'] \\
& & T' \arrow[ull, "\partial^{T'}_B"] \arrow[urr, "\partial^{T'}_C"'] & &
\end{tikzcd}
$$

The horizontal composition issue

But is **Span** = **Span**(**Gpd**) a bicategory?

In particular, are we able to define a **horizontal composition**?

$$
\begin{tikzcd}
 & & S \arrow[dll, "\partial_A^S"'] \arrow[drr, "\partial_B^S"] \arrow[dd, "F" description] & & & & T \arrow[dll, "\partial_B^T"'] \arrow[drr, "\partial_C^T"] \arrow[dd, "G" description] & & \\
A & \Downarrow\phi^A & & \Downarrow\phi^B & B & \Downarrow\psi^B & & \Downarrow\psi^C & C \\
 & & S' \arrow[ull, "\partial_A^{S'}"] \arrow[urr, "\partial_B^{S'}"'] & & & & T' \arrow[ull, "\partial_B^{T'}"] \arrow[urr, "\partial_C^{T'}"'] & &
\end{tikzcd}
$$

The horizontal composition issue

But is **Span** = **Span**(**Gpd**) a bicategory?

In particular, are we able to define a **horizontal composition**?

$$\begin{tikzcd}
 & & & & T \odot S \arrow[dll, "l"'] \arrow[drr, "r"] & & & & \\
 & & S \arrow[dll, "\partial^S_A"'] \arrow[drr, "\partial^S_B"] \arrow[dd, "F"] & & & & T \arrow[dll, "\partial^T_B"'] \arrow[drr, "\partial^T_C"] \arrow[dd, "G"] & & \\
A \arrow[rr, phantom, "\Downarrow \phi^A"] & & & & B \arrow[ll, phantom, "\Downarrow \phi^B"] \arrow[rr, phantom, "\Downarrow \psi^B"] & & & & C \arrow[ll, phantom, "\Downarrow \psi^C"] \\
 & & S' \arrow[ull, "\partial^{S'}_A"] \arrow[urr, "\partial^{S'}_B"'] & & & & T' \arrow[ull, "\partial^{T'}_B"] \arrow[urr, "\partial^{T'}_C"'] & & \\
 & & & & T' \odot S' \arrow[ull, "l"] \arrow[urr, "r"'] & & & &
\end{tikzcd}$$

The horizontal composition issue

But is **Span** $=$ **Span**(**Gpd**) a bicategory?

In particular, are we able to define a **horizontal composition**?

$$\begin{tikzcd}
 & & T \odot S \arrow[dl, "l"'] \arrow[dr, "r"] \arrow[dddd, "H"] & & \\
 & S \arrow[dl, "\partial_A^S"'] \arrow[dd, "F"] \arrow[dr, gray, "\partial_B^S"] & & T \arrow[dr, "\partial_C^T"] \arrow[dd, "G"] \arrow[dl, gray, "\partial_B^T"'] & \\
A \arrow[r, phantom, "\Downarrow\phi^A"] & {} & {} \arrow[l, phantom, "\textcolor{gray}{\Downarrow\phi^B}\Downarrow\chi^A"] \arrow[r, phantom, "\Downarrow\chi^C\textcolor{gray}{\Downarrow\psi^B}"] & {} & C \arrow[l, phantom, "\Downarrow\psi^C"'] \\
 & S' \arrow[ul, "\partial_A^{S'}"] \arrow[ur, gray, "\partial_B^{S'}"'] & & T' \arrow[ur, "\partial_C^{T'}"'] \arrow[ul, gray, "\partial_B^{T'}"] & \\
 & & T' \odot S' \arrow[ul, "l"] \arrow[ur, "r"'] & &
\end{tikzcd}$$

The horizontal composition issue

But is $\mathbf{Span} = \mathbf{Span}(\mathbf{Gpd})$ a bicategory?

In particular, are we able to define a **horizontal composition**?

$$\begin{tikzcd}
 & & T \odot S \arrow[dl, "l"'] \arrow[dr, "r"] \arrow[dddd, "H"] & & \\
 & S \arrow[dl, "\partial_A^S"'] \arrow[dd, "F"] \arrow[dr, gray, "\partial_B^S"] & & T \arrow[dr, "\partial_C^T"] \arrow[dd, "G"] \arrow[dl, gray, "\partial_B^T"'] & \\
A \arrow[r, phantom, "\Downarrow\phi^A"] & {} \arrow[r, phantom, "\Downarrow\phi^B \Downarrow\chi^A"] & {} \arrow[r, phantom, "\Downarrow\chi^C \Downarrow\psi^B"] & {} \arrow[r, phantom, "\Downarrow\psi^C"] & C \\
 & S' \arrow[ul, "\partial_A^{S'}"] \arrow[ur, gray, "\partial_B^{S'}"'] & & T' \arrow[ur, "\partial_C^{T'}"'] \arrow[ul, gray, "\partial_B^{T'}"] & \\
 & & T' \odot S' \arrow[ul, "l"] \arrow[ur, "r"'] & &
\end{tikzcd}$$

$\rightsquigarrow$ not always possible to find such H, χ^A, χ^C! So **Span is not a bicategory**!

Taking into account symmetries

This problem actually arises in other proof-relevant bicategorical models and needs to be addressed.

- generalized species of structures [**fiore2008cartesian**]: quotient of the witnesses through a **coend**
- template games [**mellies2019template**]: use of **deformations** to correctly align the witnesses

Span is dead (again), long live **Thin**!

Our solution: we add structures to constrain the spans and the morphisms of spans, so that horizontal composition exists.

A **thin span** is a tuple

$$\mathcal{A} \quad = \quad (A, A_-, A_+, \mathcal{U}_{\mathcal{A}}, T_{\mathcal{A}})$$

where A is a groupoid and the remainder is the data associated with two orthogonality relations.

Span is dead (again), long live **Thin**!

Our solution: we add structures to constrain the spans and the morphisms of spans, so that horizontal composition exists.

A **thin span** is a tuple

$$\mathcal{A} \quad = \quad (A, \ldots)$$

where A is a groupoid and the remainder is ~~the data associated with two orthogonality relations~~ **black box magic**.

Span is dead (again), long live **Thin**!

Our solution: we add structures to constrain the spans and the morphisms of spans, so that horizontal composition exists.

Theorem (C., F.)

We get a bicategory **Thin***:*

- *objects: thin groupoids* $\mathcal{A}, \mathcal{B}, \ldots$
- 1*-morphisms:* ***thin spans*** *(i.e. spans compatible with the black box magic)*
- 2*-morphisms:* ***positive morphisms*** *(i.e. morphisms of spans compatible with the black box magic)*
- 1*-identity on* $\mathcal{A}$*:*

$$\begin{tikzcd} & A \arrow[dl] \arrow[dr] & \\ A & & A \end{tikzcd}$$

- 1*-composition: pullbacks*

The exponential modality

Recall that the exponential $! \colon \mathbf{Rel} \to \mathbf{Rel}$ is derived from the monad $\mathcal{M}_{\mathrm{fin}} \colon \mathbf{Set} \to \mathbf{Set}$.

We derive an exponential $! \colon \mathbf{Thin} \to \mathbf{Thin}$ from the monad $\mathbf{List}^{\star} \colon \mathbf{Gpd} \to \mathbf{Gpd}$.

The exponential modality

The monad $\mathbf{List}^\star \colon \mathbf{Gpd} \to \mathbf{Gpd}$? The "free strict symmetric monoidal construction".

To $A \in \mathbf{Gpd}$, associates $\mathbf{List}^\star(A) \in \mathbf{Gpd}$:

- objects: lists $[a_1; \ldots; a_n] \in \mathbf{List}(\mathrm{Ob}(A))$;
- morphisms $[a_1; \ldots; a_n] \to [a'_1; \ldots; a'_m]$: pairs $(\pi, (f_i)_{i \in I})$ where
  - π is a bijection $\{ 1, \ldots, n \} \to \{ 1, \ldots, m \}$;
  - f_i is a morphism $a_i \to a'_{\pi(i)} \in A$.

The unit $\eta_A \colon A \to \mathbf{List}^\star(A)$: maps $a \in A$ to $[a]$;

The multiplication $\mu_A \colon \mathbf{List}^\star(\mathbf{List}^\star(A)) \to \mathbf{List}^\star(A)$: merges lists of lists into lists.

The exponential modality

The monad $\mathbf{List}^\star\colon \mathbf{Gpd} \to \mathbf{Gpd}$? The "free strict symmetric monoidal construction".

To $A \in \mathbf{Gpd}$, associates $\mathbf{List}^\star(A) \in \mathbf{Gpd}$:

- objects: lists $[a_1; \ldots; a_n] \in \mathbf{List}(\mathrm{Ob}(A))$;
- morphisms $[a_1; \ldots; a_n] \to [a'_1; \ldots; a'_m]$: pairs $(\pi, (f_i)_{i \in I})$ where
  - π is a bijection $\{1, \ldots, n\} \to \{1, \ldots, m\}$;
  - f_i is a morphism $a_i \to a'_{\pi(i)} \in A$.

The unit $\eta_A\colon A \to \mathbf{List}^\star(A)$: maps $a \in A$ to $[a]$;

The multiplication $\mu_A\colon \mathbf{List}^\star(\mathbf{List}^\star(A)) \to \mathbf{List}^\star(A)$: merges lists of lists into lists.

The exponential modality

The monad $\mathbf{List}^\star\colon \mathbf{Gpd} \to \mathbf{Gpd}$? The "free strict symmetric monoidal construction".

To $A \in \mathbf{Gpd}$, associates $\mathbf{List}^\star(A) \in \mathbf{Gpd}$:

- objects: lists $[a_1; \ldots; a_n] \in \mathbf{List}(\mathrm{Ob}(A))$;
- morphisms $[a_1; \ldots; a_n] \to [a'_1; \ldots; a'_m]$: pairs $(\pi, (f_i)_{i \in I})$ where
  - π is a bijection $\{ 1, \ldots, n \} \to \{ 1, \ldots, m \}$;
  - f_i is a morphism $a_i \to a'_{\pi(i)} \in A$.

The unit $\eta_A\colon A \to \mathbf{List}^\star(A)$: maps $a \in A$ to $[a]$;

The multiplication $\mu_A\colon \mathbf{List}^\star(\mathbf{List}^\star(A)) \to \mathbf{List}^\star(A)$: merges lists of lists into lists.

The exponential modality

The monad $\mathbf{List}^\star \colon \mathbf{Gpd} \to \mathbf{Gpd}$? The "free strict symmetric monoidal construction".

To $A \in \mathbf{Gpd}$, associates $\mathbf{List}^\star(A) \in \mathbf{Gpd}$:

- objects: lists $[a_1; \ldots; a_n] \in \mathbf{List}(\mathrm{Ob}(A))$;
- morphisms $[a_1; \ldots; a_n] \to [a'_1; \ldots; a'_m]$: pairs $(\pi, (f_i)_{i \in I})$ where
  - π is a bijection $\{ 1, \ldots, n \} \to \{ 1, \ldots, m \}$;
  - f_i is a morphism $a_i \to a'_{\pi(i)} \in A$.

The unit $\eta_A \colon A \to \mathbf{List}^\star(A)$: maps $a \in A$ to $[a]$;

The multiplication $\mu_A \colon \mathbf{List}^\star(\mathbf{List}^\star(A)) \to \mathbf{List}^\star(A)$: merges lists of lists into lists.

The exponential modality

The monad $\mathbf{List}^\star \colon \mathbf{Gpd} \to \mathbf{Gpd}$? The "free strict symmetric monoidal construction".

To $A \in \mathbf{Gpd}$, associates $\mathbf{List}^\star(A) \in \mathbf{Gpd}$:

- objects: lists $[a_1; \ldots; a_n] \in \mathbf{List}(\mathrm{Ob}(A))$;
- morphisms $[a_1; \ldots; a_n] \to [a'_1; \ldots; a'_m]$: pairs $(\pi, (f_i)_{i \in I})$ where
  - π is a bijection $\{ 1, \ldots, n \} \to \{ 1, \ldots, m \}$;
  - f_i is a morphism $a_i \to a'_{\pi(i)} \in A$.

The unit $\eta_A \colon A \to \mathbf{List}^\star(A)$: maps $a \in A$ to $[a]$;

The multiplication $\mu_A \colon \mathbf{List}^\star(\mathbf{List}^\star(A)) \to \mathbf{List}^\star(A)$: merges lists of lists into lists.

The exponential modality

We get an exponential

$$!\colon \mathbf{Thin} \to \mathbf{Thin}$$

where

$$!\mathcal{A} \quad \hat{=} \quad (\mathbf{List}^{\star}(A), \ldots)$$

for every thin groupoid $\mathcal{A}$ and

$$! \left(\begin{tikzcd} & S \arrow[dl, "\partial_A^S"'] \arrow[dr, "\partial_B^S"] & \\ A & & B \end{tikzcd} \right) \quad \hat{=} \quad \begin{tikzcd} & \mathbf{List}^{\star}(S) \arrow[dl, "\mathbf{List}^{\star}(\partial_A^S)"'] \arrow[dr, "\mathbf{List}^{\star}(\partial_B^S)"] & \\ \mathbf{List}^{\star}(A) & & \mathbf{List}^{\star}(B) \end{tikzcd}$$

for every thin span $S\colon \mathcal{A} \to \mathcal{B}$.

The exponential modality

The structure of comonad of ! is derived from the monad structure of $\mathbf{List}^\star$.

Given a thin groupoid $\mathcal{A}$,

$$\check{\eta}_{\mathcal{A}} = \begin{tikzcd} & A \arrow[dl, "\eta_A"'] \arrow[dr, "\mathrm{id}_A"] & \\ \mathbf{List}^\star(A) & & A \end{tikzcd} \qquad \check{\mu}_{\mathcal{A}} = \begin{tikzcd} & \mathbf{List}^\star(\mathbf{List}^\star(A)) \arrow[dl, "\mu_A"'] \arrow[dr, "\mathrm{id}_{!!A}"] & \\ \mathbf{List}^\star(A) & & \mathbf{List}^\star(\mathbf{List}^\star(A)) \end{tikzcd} .$$

The Kleisli bicategory

We thus get a Kleisli bicategory $\mathbf{Thin}_!$ with $! = \mathbf{List}^{\star}$, whose 1-cells $\mathcal{A} \to \mathcal{B}$ are of the form

$$\begin{tikzcd}
 & S \arrow[dl, "\partial^S_{!A}"'] \arrow[dr, "\partial^S_B"] & \\
!A & & B
\end{tikzcd} .$$

The Kleisli bicategory

We thus get a Kleisli bicategory $\mathbf{Thin}_!$ with $! = \mathbf{List}^\star$, whose 1-cells $\mathcal{A} \to \mathcal{B}$ are of the form

$$\begin{tikzcd} & S \arrow[dl, "\partial^S_{!A}"'] \arrow[dr, "\partial^S_B"] & \\ !A & & B \end{tikzcd} .$$

In categorical models of **LL**, the Kleisli category is cartesian closed.

The Kleisli bicategory

We thus get a Kleisli bicategory $\mathbf{Thin}_!$ with $! = \mathbf{List}^\star$, whose 1-cells $\mathcal{A} \to \mathcal{B}$ are of the form

$$\begin{tikzcd} & S \arrow[dl, "\partial^S_{!A}"'] \arrow[dr, "\partial^S_B"] & \\ !A & & B \end{tikzcd} .$$

In categorical models of **LL**, the Kleisli category is cartesian closed.

Theorem (C., F.)

The bicategory $\mathbf{Thin}_!$ *is cartesian closed.*

Outline

From relations to spans

Interpreting programs

A rigid intersection type system

Examples of interpretations

Example 1:

$$x : \mathbf{Bool} \vdash \mathbf{if}\ x\ \mathbf{then\ ff\ else\ tt} : \mathbf{Bool}$$

interpreted as the span (which happens to be a relation)

$$\begin{tikzcd}
& \{\ [\mathbf{tt}] \multimap \mathbf{ff}, \quad [\mathbf{ff}] \multimap \mathbf{tt}\ \} \arrow[dl] \arrow[dr] & \\
!\mathbf{Bool} & & \mathbf{Bool}
\end{tikzcd}.$$

Examples of interpretations

Example 2:

$$x : \mathbf{Bool} \vdash \mathbf{if}\ x\ \mathbf{then}\ (\mathbf{if}\ x\ \mathbf{then}\ \mathbf{ff}\ \mathbf{else}\ \mathbf{tt})\ \mathbf{else}\ (\mathbf{if}\ x\ \mathbf{then}\ \mathbf{tt}\ \mathbf{else}\ \mathbf{ff}) : \mathbf{Bool}$$

interpreted as the span (which happens to be a relation)

$$\begin{tikzcd}
& \{\ [\mathbf{tt};\mathbf{tt}] \multimap \mathbf{ff},\quad [\mathbf{tt};\mathbf{ff}] \multimap \mathbf{tt},\quad [\mathbf{ff};\mathbf{tt}] \multimap \mathbf{tt},\quad [\mathbf{ff};\mathbf{ff}] \multimap \mathbf{ff}\ \} \arrow[dl] \arrow[dr] & \\
!\mathbf{Bool} & & \mathbf{Bool}
\end{tikzcd}$$

Examples of interpretations

Example 2:

$$x : \mathbf{Bool} \vdash \mathbf{if}\ x\ \mathbf{then}\ (\mathbf{if}\ x\ \mathbf{then\ ff\ else\ tt})\ \mathbf{else}\ (\mathbf{if}\ x\ \mathbf{then\ tt\ else\ ff}) : \mathbf{Bool}$$

interpreted as the span (which happens to be a relation)

$$\begin{tikzcd}[column sep=small] & \{\ [\mathbf{tt};\mathbf{tt}] \multimap \mathbf{ff},\quad [\mathbf{tt};\mathbf{ff}] \multimap \mathbf{tt},\quad [\mathbf{ff};\mathbf{tt}] \multimap \mathbf{tt},\quad [\mathbf{ff};\mathbf{ff}] \multimap \mathbf{ff}\ \} \arrow[dl] \arrow[dr] & \\ !\mathbf{Bool} & & \mathbf{Bool} \end{tikzcd}$$

to compare with the interpretation in $\mathbf{Rel}_!$:

$$\{\ [\mathbf{tt},\mathbf{tt}] \multimap \mathbf{ff},\quad {\color{red}[\mathbf{tt},\mathbf{ff}] \multimap \mathbf{tt}},\quad [\mathbf{ff},\mathbf{ff}] \multimap \mathbf{tt}\ \}.$$

Categorical interpretation

Since $\mathbf{Thin}_!$ is cartesian closed, we can interpret simply-typed λ-calculus. What would it look like?

Considered types:

$$A, B, \ldots \quad ::= \quad \mathbf{Bool} \quad | \quad A \to B$$

Interpretation $(\!|A|\!)$ of a type A:

$$(\!|\mathbf{Bool}|\!) \quad = \quad 1 \sqcup 1 \qquad\qquad (\!|A \to B|\!) \quad = \quad !(\!|A|\!) \times (\!|B|\!)$$

Interpretation $(\!|\Gamma|\!)$ of a context $\Gamma = (x_1 : A_1, \ldots, x_n : A_n)$:

$$(\!|\Gamma|\!) \quad = \quad (\!|A_1|\!) \sqcup \cdots \sqcup (\!|A_n|\!)$$

Categorical interpretation

Since $\mathbf{Thin}_!$ is cartesian closed, we can interpret simply-typed λ-calculus. What would it look like?

Considered types:

$$A, B, \ldots \quad ::= \quad \mathbf{Bool} \quad | \quad A \to B$$

Interpretation $(\!|A|\!)$ **of a type** A:

$$(\!|\mathbf{Bool}|\!) \quad = \quad 1 \sqcup 1 \qquad\qquad (\!|A \to B|\!) \quad = \quad !(\!|A|\!) \times (\!|B|\!)$$

Interpretation $(\!|\Gamma|\!)$ **of a context** $\Gamma = (x_1 : A_1, \ldots, x_n : A_n)$:

$$(\!|\Gamma|\!) \quad = \quad (\!|A_1|\!) \sqcup \cdots \sqcup (\!|A_n|\!)$$

Categorical interpretation

Since $\mathbf{Thin}_!$ is cartesian closed, we can interpret simply-typed λ-calculus. What would it look like?

Considered types:

$$A, B, \ldots \quad ::= \quad \mathbf{Bool} \quad | \quad A \to B$$

Interpretation $(\!|A|\!)$ **of a type** A:

$$(\!|\mathbf{Bool}|\!) \quad = \quad 1 \sqcup 1 \qquad\qquad (\!|A \to B|\!) \quad = \quad !(\!|A|\!) \times (\!|B|\!)$$

Interpretation $(\!|\Gamma|\!)$ **of a context** $\Gamma = (x_1 : A_1, \ldots, x_n : A_n)$:

$$(\!|\Gamma|\!) \quad = \quad (\!|A_1|\!) \sqcup \cdots \sqcup (\!|A_n|\!)$$

Categorical interpretation

Since $\mathbf{Thin}_!$ is cartesian closed, we can interpret simply-typed λ-calculus. What would it look like?

Considered types:

$$A, B, \ldots \qquad ::= \qquad \mathbf{Bool} \quad | \quad A \to B$$

Interpretation $(\!|A|\!)$ **of a type** A:

$$(\!|\mathbf{Bool}|\!) \quad = \quad 1 \sqcup 1 \qquad\qquad (\!|A \to B|\!) \quad = \quad !(\!|A|\!) \times (\!|B|\!)$$

Interpretation $(\!|\Gamma|\!)$ **of a context** $\Gamma = (x_1 : A_1, \ldots, x_n : A_n)$:

$$(\!|\Gamma|\!) \qquad = \qquad (\!|A_1|\!) \sqcup \cdots \sqcup (\!|A_n|\!)$$

Categorical interpretation

Given a derivation of $\Gamma \vdash t : A$ in STLC, its **categorical interpretation** is a span

$$
\begin{tikzcd}
& (\!|t|\!) \arrow[dl, "\partial_l^{(\!|t|\!)}"'] \arrow[dr, "\partial_r^{(\!|t|\!)}"] & \\
!(\!|\Gamma|\!) & & (\!|A|\!)
\end{tikzcd}
$$

by induction on the derivation.

- this is automatically derived from the cartesian closed structure
- but uneasy to describe syntactically (notably the projection on $!(\!|\Gamma|\!)$)

The bagged interpretation

We introduce a more relevant interpretation of contexts.

Bagged interpretation of a context $\Gamma = (x_1 : A_1, \dots, x_n : A_n)$:

$$[\![\Gamma]\!] \qquad = \qquad ![\![A_1]\!] \times \cdots \times ![\![A_n]\!] \qquad \in \qquad \mathbf{Thin}$$

(Recall that $!(\!|\Gamma|\!) = !((\!|A_1|\!) \sqcup \cdots \sqcup (\!|A_n|\!))$)

Proposition

The underlying groupoid of $[\![\Gamma]\!]$ is equipped with a structure of monoid.

- multiplication: given $\gamma = (\gamma_1, \dots, \gamma_n)$ and $\delta = (\delta_1, \dots, \delta_n)$ in $[\![\Gamma]\!]$,

$$\gamma \mathbin{\vec{\oplus}} \delta \qquad = \qquad (\gamma_1 +\!\!+ \delta_1, \dots, \gamma_n +\!\!+ \delta_n)$$

  where $\gamma_i +\!\!+ \delta_i$ is the concatenation of lists in $![\![A_i]\!]$

The bagged interpretation

We introduce a more relevant interpretation of contexts.

Bagged interpretation of a context $\Gamma = (x_1 : A_1, \ldots, x_n : A_n)$:

$$[\![\Gamma]\!] \qquad = \qquad ![\![A_1]\!] \times \cdots \times ![\![A_n]\!] \qquad \in \qquad \mathbf{Thin}$$

(Recall that $!(\!|\Gamma|\!) = !((\!|A_1|\!) \sqcup \cdots \sqcup (\!|A_n|\!))$)

Proposition

The underlying groupoid of $[\![\Gamma]\!]$ is equipped with a structure of monoid.

- multiplication: given $\gamma = (\gamma_1, \ldots, \gamma_n)$ and $\delta = (\delta_1, \ldots, \delta_n)$ in $[\![\Gamma]\!]$,

$$\gamma \mathbin{\vec{\oplus}} \delta \qquad = \qquad (\gamma_1 +\!\!+ \delta_1, \ldots, \gamma_n +\!\!+ \delta_n)$$

  where $\gamma_i +\!\!+ \delta_i$ is the concatenation of lists in $![\![A_i]\!]$

The bagged interpretation

We introduce a more relevant interpretation of contexts.

Bagged interpretation of a context $\Gamma = (x_1 : A_1, \ldots, x_n : A_n)$:

$$[\![\Gamma]\!] \qquad = \qquad ![\![A_1]\!] \times \cdots \times ![\![A_n]\!] \qquad \in \qquad \mathbf{Thin}$$

(Recall that $!(\!|\Gamma|\!) = !((\!|A_1|\!) \sqcup \cdots \sqcup (\!|A_n|\!))$)

Proposition

The underlying groupoid of $[\![\Gamma]\!]$ is equipped with a structure of monoid.

- multiplication: given $\gamma = (\gamma_1, \ldots, \gamma_n)$ and $\delta = (\delta_1, \ldots, \delta_n)$ in $[\![\Gamma]\!]$,

$$\gamma \mathbin{\vec{\oplus}} \delta \qquad = \qquad (\gamma_1 +\!\!+\, \delta_1, \ldots, \gamma_n +\!\!+\, \delta_n)$$

  where $\gamma_i +\!\!+\, \delta_i$ is the concatenation of lists in $![\![A_i]\!]$

The bagged interpretation

Given $\Gamma = x_1 : A_1, \ldots, x_n : A_n$ and a derivation $\Gamma \vdash t : A$, we define a thin span

$$
\begin{tikzcd}
 & \llbracket t \rrbracket \arrow[dl, "\partial_l^{\llbracket t \rrbracket}"'] \arrow[dr, "\partial_r^{\llbracket t \rrbracket}"] & \\
\llbracket \Gamma \rrbracket & & (\!| A |\!)
\end{tikzcd}
$$

by induction on the derivation.

The bagged interpretation

Variable case: for $\Gamma = x_1 : A_1, \ldots, x_n : A_n$ and $i \in \{ 1, \ \ldots, \ n \}$,

$$
[\![x_i]\!] \quad = \quad
\begin{tikzcd}
 & (\!|A_i|\!) \arrow[dl, "{\langle [],\ldots,\eta_{(\!|A_i|\!)},\ldots,[] \rangle}"'] \arrow[dr, "\mathrm{id}_{(\!|A_i|\!)}"] & \\
!(\!|A_1|\!) \times \cdots \times !(\!|A_n|\!) & & (\!|A_i|\!)
\end{tikzcd}
$$

$\rightsquigarrow$ "for every $a \in (\!|A_i|\!)$, there is one computation consuming $([], \ldots, [a], \ldots, [])$ and producing a"

The bagged interpretation

λ-**abstraction case**: given a derivation of $(\Gamma, x : A) \vdash t : B$,

The bagged interpretation

λ-**abstraction case**: given a derivation of $(\Gamma, x : A) \vdash t : B$,

$$\begin{tikzcd}
 & \llbracket t \rrbracket \arrow[dl, "\partial_l^{\llbracket t \rrbracket}"'] \arrow[dr, "\partial_r^{\llbracket t \rrbracket}"] & \\
\llbracket \Gamma \rrbracket \times !\llparenthesis A \rrparenthesis & & \llparenthesis B \rrparenthesis
\end{tikzcd}$$

The bagged interpretation

λ-**abstraction case**: given a derivation of $(\Gamma, x : A) \vdash t : B$,

$$
\begin{tikzcd}
 & {[\![t]\!]} \arrow[dl, "l \circ \partial_l^{[\![t]\!]}"'] \arrow[d, "r \circ \partial_r^{[\![t]\!]}"] \arrow[dr, "\partial_r^{[\![t]\!]}"] & \\
{[\![\Gamma]\!]} & !(\!|A|\!) & (\!|B|\!)
\end{tikzcd}
$$

The bagged interpretation

λ-**abstraction case**: given a derivation of $(\Gamma, x : A) \vdash t : B$,

$$\begin{tikzcd}
 & \llbracket t \rrbracket \arrow[dl, "l \circ \partial_l^{\llbracket t \rrbracket}"'] \arrow[dr, "\langle r \circ \partial_l^{\llbracket t \rrbracket}, \partial_r^{\llbracket t \rrbracket} \rangle"] & \\
\llbracket \Gamma \rrbracket & & !(\!|A|\!) \times (\!|B|\!)
\end{tikzcd}$$

The bagged interpretation

λ-**abstraction case**: given a derivation of $(\Gamma, x : A) \vdash t : B$,

$$[\![\lambda x.t]\!] \quad = \quad \begin{tikzcd} & {[\![t]\!]} \arrow[dl, "l \circ \partial_l^{[\![t]\!]}"'] \arrow[dr, "\langle r \circ \partial_l^{[\![t]\!]}, \partial_r^{[\![t]\!]} \rangle"] & \\ {[\![\Gamma]\!]} & & {!(\!|A|\!) \times (\!|B|\!)} \end{tikzcd}$$

$\rightsquigarrow$ "if t consumes $(\gamma, [a_i]_i)$ to produce b, then $\lambda x.t$ consumes γ to produce $[a_i]_i \multimap b$"

The bagged interpretation

Application case: given a derivation of $\Gamma \vdash t : A \to B$ and of $\Gamma \vdash u : A$,

$$
\llbracket t\ u \rrbracket \quad = \quad
\begin{tikzcd}
 & & & \llbracket t\ u \rrbracket \arrow[dl, dotted, "l"'] \arrow[dr, dotted, "r"] \arrow[d, phantom, "\vee"] & & & \\
 & & \llbracket t \rrbracket \times !\llbracket u \rrbracket \arrow[dl, "\partial_l^{\llbracket t \rrbracket} \times !\partial_l^{\llbracket u \rrbracket}"'] \arrow[dr, "\partial_r^{\llbracket t \rrbracket} \times !\partial_r^{\llbracket u \rrbracket}"] & {} & \mathrm{linev}_{!\llparenthesis A \rrparenthesis, \llbracket B \rrbracket} \arrow[dl, "\partial_l^{\mathrm{linev}_{!\llparenthesis A \rrparenthesis, \llbracket B \rrbracket}}"'] \arrow[dr, "\partial_r^{\mathrm{linev}_{!\llparenthesis A \rrparenthesis, \llbracket B \rrbracket}}"] & & \\
 & \llbracket \Gamma \rrbracket \times !\llbracket \Gamma \rrbracket \arrow[dl, "\cdots"'] & & (!\llparenthesis A \rrparenthesis \times \llparenthesis B \rrparenthesis) \times !\llparenthesis A \rrparenthesis & & \llparenthesis B \rrparenthesis \arrow[dr, equal] & \\
\llbracket \Gamma \rrbracket & & & & & & \llparenthesis B \rrparenthesis
\end{tikzcd}
$$

$\rightsquigarrow$ "if t consumes γ to produce $[a_i]_i \multimap b$ and u consumes δ to produce $[a_i]_i$, then $t\ u$ consumes $\gamma \vec{\oplus} \delta$ to produce b"

The bagged interpretation

Given $\Gamma = (x_1 : A_1, \ldots, x_n : A_n)$, write see_Γ for the Seely morphism

$$\mathsf{see}_\Gamma : \quad !(\llparenthesis A_1 \rrparenthesis \sqcup \cdots \sqcup \llparenthesis A_n \rrparenthesis) \quad \to \quad ![\![A_1]\!] \times \cdots \times ![\![A_n]\!] \quad \in \mathbf{Gpd}$$

Theorem (Compatibility)

Given a derivation $\Gamma \vdash t : A$, we have a canonical isomorphism of spans

$$\begin{tikzcd}
 & & \llparenthesis t \rrparenthesis \arrow[dl, "\partial_l^{\llparenthesis t \rrparenthesis}"'] \arrow[ddrr, "\partial_r^{\llparenthesis t \rrparenthesis}"] & & & & & [\![t]\!] \arrow[ddll, "\partial_l^{[\![t]\!]}"'] \arrow[ddrr, "\partial_r^{[\![t]\!]}"] & & \\
 & !\llparenthesis \Gamma \rrparenthesis \arrow[dl, "\mathsf{see}_\Gamma"'] & & & \overset{\sim}{\Rightarrow} & & & & & \\
[\![\Gamma]\!] & & & & \llparenthesis A \rrparenthesis & [\![\Gamma]\!] & & & & \llparenthesis A \rrparenthesis
\end{tikzcd}$$

in $\mathbf{Thin}$.

Outline

Intersection type systems

[de2018execution]: interpretations of programs in **Rel** can be presented syntactically through an **intersection type system**.

$$\Gamma \vdash t : A \qquad \rightsquigarrow \qquad \Theta \vdash t : \alpha \triangleleft A \quad \text{and} \quad \Theta \vdash_{\mathrm{m}} t : [\alpha_1, \ldots, \alpha_n] \triangleleft A$$

[olimpieri2021intersection]: interpretations of programs in **Esp** can also be presented syntactically through an **intersection type system**

Can we have a similar presentation for **Thin**?

Intersection type systems

Idea of intersection type system: give several types to a pure λ-term:

$$\Gamma \vdash t : \tau_1 \cap \cdots \cap \tau_n$$

In this setting, we can type $\lambda x.xx$:

$$\vdash \lambda x.xx : ((A \to A) \cap A) \to A$$

The system is said

- **non-commutative** when $\sigma \cap \tau \neq \tau \cap \sigma$
- **non-idempotent** when $\sigma \cap \sigma \neq \sigma$

Intersection type systems

In the context of **Rel** and **Thin**, it is better to change perspective.

Broke:

- simply-typed λ-calculus is pure λ-calculus with types
- intersection types $=$ "typing a term with several types"

Woke:

- pure λ-calculus is simply-typed λ-calculus with a reflexive type/object
- intersection types $=$ "assigning different **values** to a term"

ITS for **Thin**

[**de2018execution**]: the relational model be described syntactically by an intersection type system, with multisets as bags

For **Thin**: we use the same system with **lists** as bags

ITS for **Thin**

Simple types considered:

$$A, B, \ldots \quad ::= \quad \mathbf{Bool} \quad | \quad A \to B$$

Refinement ~~types~~ **values** and intersection ~~types~~ **values**:

$$\begin{array}{lcl} \alpha, \beta, \ldots & ::= & \mathbf{tt} \mid \mathbf{ff} \mid \kappa \multimap \alpha \\ \kappa, \lambda, \ldots & ::= & [\alpha_1, \ldots, \alpha_n] \quad (n \in \mathbb{N}) \end{array}$$

where $[\alpha_1, \ldots, \alpha_n]$ is a **list** of elements.

Refinement judgements $\alpha \lhd A$ and $\kappa \lhd_{\mathrm{m}} A$ and their rules:

$$\frac{}{\mathbf{ff} \lhd \mathbf{Bool}} \qquad \frac{}{\mathbf{tt} \lhd \mathbf{Bool}} \qquad \frac{\kappa \lhd_{\mathrm{m}} A \qquad \beta \lhd B}{\kappa \multimap \alpha \lhd A \to B} \qquad \frac{\forall i \in \{\, 1, \ldots, n \,\} \qquad \alpha_i \lhd A}{[\alpha_1, \ldots, \alpha_n] \lhd_{\mathrm{m}} A}$$

ITS for **Thin**

Resource contexts: sequences Θ of bindings of the form

$$\Theta, \Sigma, \ldots \qquad ::= \qquad (x_i : [a_{i,1}, \ldots, a_{i,n_i}] \triangleleft A_i)_{1 \leq i \leq n} \quad (n \in \mathbb{N})$$

ITS for **Thin**

Addition of resource contexts: given

$$\Theta = (x_i : \kappa_i \lhd A_i)_{1 \leq i \leq n} \qquad \Sigma = (x_i : \lambda_i \lhd A_i)_{1 \leq i \leq n}$$

we put

$$\Theta \mathbin{\vec{\oplus}} \Sigma = (x_i : (\kappa_i +\!\!+ \lambda_i) \lhd A_i)_{1 \leq i \leq n}$$

where $\kappa_i +\!\!+ \lambda_i$ stands for the **concatenation of lists**.

ITS for **Thin**

Intersection type judgements $\Theta \vdash t : \alpha \triangleleft A$ and their rules:

$$(\text{IT-Var}) \quad \frac{\alpha \triangleleft A_i}{(x_1 : [] \triangleleft A_1, \cdots, x_i : [\alpha] \triangleleft A_i, \cdots, x_n : [] \triangleleft A_n) \vdash x_i : \alpha \triangleleft A_i}$$

$$(\text{IT-App}) \quad \frac{\Theta \vdash t : \kappa \multimap \beta \triangleleft A \to B \qquad \Theta' \vdash_{\mathrm{m}} u : \kappa}{\Theta \mathbin{\vec{\oplus}} \Theta' \vdash t\ u : \beta \triangleleft B}$$

$$(\text{IT-Lam}) \quad \frac{(\Theta, x : \kappa \triangleleft A) \vdash t : \beta \triangleleft B}{\Theta \vdash \lambda x.t : \kappa \multimap \beta \triangleleft A \to B}$$

$$(\text{IT-Int}) \quad \frac{\Theta_i \vdash t : \alpha_i \triangleleft A \qquad \text{for } i \in \{\, 1, \ldots, n \,\}}{\Theta_1 \mathbin{\vec{\oplus}} \cdots \mathbin{\vec{\oplus}} \Theta_n \vdash_{\mathrm{m}} t : [\alpha_1, \ldots, \alpha_n] \triangleleft A}$$

ITS for **Thin**

Since we are considering **Bool** and **if**'s:

$$(\text{IT-If-True}) \quad \frac{\Theta \vdash t : \mathbf{tt} \lhd \mathbf{Bool} \qquad \Theta_{\mathbf{then}} \vdash u : \alpha \lhd A}{\Theta \mathbin{\vec{\oplus}} \Theta_{\mathbf{then}} \vdash_{\mathrm{m}} \mathbf{if}\ t\ \mathbf{then}\ u\ \mathbf{else}\ v : \alpha \lhd A}$$

$$(\text{IT-If-Else}) \quad \frac{\Theta \vdash t : \mathbf{tt} \lhd \mathbf{Bool} \qquad \Theta_{\mathbf{else}} \vdash v : \alpha \lhd A}{\Theta \mathbin{\vec{\oplus}} \Theta_{\mathbf{else}} \vdash_{\mathrm{m}} \mathbf{if}\ t\ \mathbf{then}\ u\ \mathbf{else}\ v : \alpha \lhd A}$$

ITS for **Thin**

Theorem

Given a derivation $\Gamma \vdash t : A$ and its bagged interpretation

$$\begin{tikzcd} & \llbracket t \rrbracket \arrow[dl, "\partial_l^{\llbracket t \rrbracket}"'] \arrow[dr, "\partial_r^{\llbracket t \rrbracket}"] & \\ \llbracket \Gamma \rrbracket & & (\!|A|\!) \end{tikzcd}$$

we have a bijection

$$\mathrm{Ob}(\llbracket t \rrbracket) \qquad \cong \qquad \{\ p \mid p \textit{ derivation of } \Theta \vdash t : \alpha : A \textit{ for } \Theta \triangleleft \Gamma\ \}.$$

- Contrarily to **Rel** and **Esp** [olimpieri2021intersection], **no quotient is required** here for the bijection.

A known broken system

That system is the direct system we obtain when dropping commutativity in the ITS of **Rel**.

It is known for **not** satisfying subject reduction!

A known broken system

In the context $f : \mathbf{Bool} \to \mathbf{Bool} \to \mathbf{Bool}, x : \mathbf{Bool}$ consider

$$t_1 = (\lambda y.\lambda z.f\ z\ y)\ x\ x \qquad t_2 = f\ x\ x$$

A known broken system

In the context $f : \mathbf{Bool} \to \mathbf{Bool} \to \mathbf{Bool}, x : \mathbf{Bool}$ consider

$$t_1 = (\lambda y.\lambda z.f\ z\ y)\ x\ x \qquad t_2 = f\ x\ x$$

Given the resource contexts

$$\Theta_{\mathbf{ff},\mathbf{tt}} = (f : [[\mathbf{ff}] \multimap [\mathbf{tt}] \multimap \mathbf{tt}] \triangleleft \mathbf{Bool} \to \mathbf{Bool}, x : [\mathbf{ff}; \mathbf{tt}] \triangleleft \mathbf{Bool})$$
$$\Theta_{\mathbf{tt},\mathbf{ff}} = (f : [[\mathbf{ff}] \multimap [\mathbf{tt}] \multimap \mathbf{tt}] \triangleleft \mathbf{Bool} \to \mathbf{Bool}, x : [\mathbf{tt}; \mathbf{ff}] \triangleleft \mathbf{Bool})$$

we have

$$\Theta_{\mathbf{ff},\mathbf{tt}} \nvdash t_1 : \mathbf{tt} \triangleleft \mathbf{Bool} \qquad \Theta_{\mathbf{ff},\mathbf{tt}} \vdash t_2 : \mathbf{tt} \triangleleft \mathbf{Bool}$$
$$\Theta_{\mathbf{tt},\mathbf{ff}} \vdash t_1 : \mathbf{tt} \triangleleft \mathbf{Bool} \qquad \Theta_{\mathbf{tt},\mathbf{ff}} \nvdash t_2 : \mathbf{tt} \triangleleft \mathbf{Bool}$$

while $t_1 \to^* t_2 \rightsquigarrow$ **subject reduction not satisfied**

A known broken system

But in **Thin**, the reduction $t_1 \to^* t_2$ is interpreted as a reindexing

$$[\![t_1 \to^* t_2]\!] : [\![t_1]\!] \to [\![t_2]\!]$$

so that subject reduction is weakly recovered:

$$\Theta_{\mathbf{tt},\mathbf{ff}} \vdash t_1 : \mathbf{tt} \lhd \mathbf{Bool} \qquad \cong \qquad \Theta_{\mathbf{ff},\mathbf{tt}} \vdash t_2 : \mathbf{tt} \lhd \mathbf{Bool}$$

(The above is informal!)

ITS for morphisms

Let's remember that thin spans are spans of groupoids, **with morphisms**.

Can we we describe the ones in the interpretation of λ-terms?

$$\begin{tikzcd} & [\![t]\!] \arrow[dl, "\partial_l^{[\![t]\!]}"'] \arrow[dr, "\partial_r^{[\![t]\!]}"] & \\ [\![\Gamma]\!] & & (\!|A|\!) \end{tikzcd} \quad \in \mathbf{Gpd}$$

ITS for morphisms

Refinement ~~types~~ **value morphisms** and intersection ~~types~~ **value morphisms**:

$$
\begin{array}{lcl}
\alpha, \beta, \ldots & ::= & \mathbf{tt} \mid \mathbf{ff} \mid \kappa \multimap \alpha \\
\kappa, \lambda, \ldots & ::= & [\alpha_1, \ldots, \alpha_n] \quad (n \in \mathbb{N}) \\
\phi, \psi, \ldots & ::= & \mathsf{id}_{\mathbf{tt}} \mid \mathsf{id}_{\mathbf{ff}} \mid \theta \multimap \phi \\
\theta, \zeta, \ldots & ::= & (\pi, [\phi_1, \ldots, \phi_n]) \quad (n \in \mathbb{N}, \pi \in \mathcal{S}_n)
\end{array}
$$

ITS for morphisms

Resource morphism contexts:

$$\Xi, \Xi', \ldots \qquad ::= \qquad (x_i : \theta_i :: \kappa_i \Rightarrow \kappa'_i \triangleleft A_i)_{1 \leq i \leq n}$$

ITS for morphisms

Intersection type morphism judgements

$$\Xi \vdash t : \phi :: \alpha \Rightarrow \alpha' \triangleleft A \qquad \text{and} \qquad \Xi \vdash_{\mathrm{m}} t : \theta :: \kappa \Rightarrow \kappa' \triangleleft A$$

and their rules:

$$\text{(ITM-Var)} \quad \frac{\phi :: \alpha \Rightarrow \alpha' \triangleleft A_i}{(\ldots, x_i : (\mathrm{id}_{\{1\}}, [\phi]) :: [\alpha] \Rightarrow [\alpha'] \triangleleft A_i, \ldots) \vdash x_i : \phi :: \alpha \Rightarrow \alpha' \triangleleft A_i}$$

$$\text{(ITM-App)} \quad \frac{\Xi \vdash t : (\theta \multimap \phi) :: (\kappa \multimap \beta) \Rightarrow (\kappa' \multimap \beta') \triangleleft A \to B \quad \Xi' \vdash_{\mathrm{m}} u : \theta :: \kappa \Rightarrow \kappa'}{\Xi \vec{\oplus} \Xi' \vdash t\, u : \phi :: \beta \Rightarrow \beta' \triangleleft B}$$

$$\text{(ITM-Lam)} \quad \frac{(\Xi, x : \theta :: \kappa \Rightarrow \kappa' \triangleleft A) \vdash t : \phi :: \beta \Rightarrow \beta' \triangleleft B}{\Xi \vdash \lambda x.t : (\theta \multimap \phi) :: (\kappa \multimap \beta) \Rightarrow (\kappa' \multimap \beta') \triangleleft A \to B}$$

$$\text{(ITM-Int)} \quad \frac{n \in \mathbb{N} \quad \sigma \in \mathcal{S}_n \quad \forall i \in \{1, \ldots, n\},\ \Xi_i \vdash t : \phi_i :: \alpha_i \Rightarrow \alpha'_i \triangleleft A}{(\sigma \otimes (\Xi_i)_{i \in [n]}) \vdash_{\mathrm{m}} t : (\sigma, [\phi_1, \ldots, \phi_n]) :: [\alpha_i]_{1 \le i \le n} \Rightarrow [\alpha'_{\sigma^{-1}(i)}]_{1 \le i \le n} \triangleleft A}$$

ITS for morphisms

Given a derivation $\Gamma \vdash t : A$, we get a groupoid $[\![t]\!]^{\mathsf{IT}}$:

- objects: derivations $\Theta \vdash t : \alpha \triangleleft A$
- morphisms: derivations $\Xi \vdash t : \phi :: \alpha \Rightarrow \alpha' \triangleleft A$
- composition of derivations as composition

Theorem (C., F.)

The interpretation in **Thin** *is described by the presented ITS, i.e.,*

$$[\![t]\!] \cong [\![t]\!]^{\mathsf{IT}}.$$

Conclusion

Thin spans

- a **quantitative semantic model** based on spans of groupoids
- provide a **proof-relevant extension** of the relational model **Rel**
- **syntactic description** given by an intersection type system

Related works

Other quantitative models:

- **Generalized species of structures fiore2008cartesian**
- **Template games mellies2019template**

Syntactic descriptions through intersection type systems:

- **olimpieri2021intersection**
- **tsukada2017generalised**

The end

Any questions?

Seely equivalence

Recall: a common approach for exhibiting a categorical model of **LL** is to find a Seely isomorphism

$$\mathrm{see}_{A,B} : \, !A \otimes \, !B \to \, !(A \,\&\, B).$$

Seely equivalence

In **Thin**,

$$\mathcal{A} \otimes \mathcal{B} \triangleq (A \times B, \ldots) \qquad \text{and} \qquad \mathcal{A} \,\&\, \mathcal{B} \triangleq (A \sqcup B, \ldots).$$

We have the 2-categorical analogue of a Seely isomorphism, already in **Gpd**:

Proposition

*Given $A, B \in$ **Gpd**, there is an adjoint equivalence of groupoids*

$$\begin{tikzcd} \mathbf{List}^\star(A) \times \mathbf{List}^\star(B) \arrow[rr, bend left, "\mathsf{see}_{A,B}"] \arrow[rr, phantom, "\perp"] & & \mathbf{List}^\star(A \sqcup B) \arrow[ll, bend left, "\overline{\mathsf{see}}_{A,B}"] \end{tikzcd} .$$

Idea: given $a = (a_i)_{i \in I}$ and $b = (b_j)_{j \in J}$, one can merge a and b as $c = (c_k)_{k \in K}$ with $K \cong I \sqcup J$.

The Seely 2-cell

Recall: the Seely isomorphism

$$\mathsf{see}_{A,B} \colon\, !A \otimes !B \to !(A \mathbin{\&} B)$$

is supposed to verify the equality

$$\begin{tikzcd}
!A \otimes !B \arrow[r, "\mathsf{see}_{A,B}"] \arrow[dd, "\delta_A \otimes \delta_B"'] & !(A \mathbin{\&} B) \arrow[d, "\delta_{A \& B}"] \\
{} \arrow[r, phantom, "="] & !!(A \mathbin{\&} B) \arrow[d, "!\langle !l, !r \rangle"] \\
!!A \otimes !!B \arrow[r, "\mathsf{see}_{!A,!B}"'] & !(!A \mathbin{\&} !B)
\end{tikzcd} .$$

The Seely 2-cell

The Seely equality appears here as a non-trivial 2-cell in **Gpd**:

$$
\begin{tikzcd}
!!A \times !!B \arrow[r, equal] \arrow[d, "\mathsf{see}_{!A,!B}"] & !!A \times !!B \arrow[d, "\mu_A \times \mu_B"] \\
!(!A \sqcup !B) \arrow[d, "!{[!(\bar{l}),!(\bar{r})]}"] & !A \times !B \arrow[dd, "\mathsf{see}_{A,B}"] \\
!!(A \sqcup B) \arrow[d, "\mu_{A \sqcup B}"] \arrow[r, phantom, "\overset{\mathsf{See}_{A,B}}{\Rrightarrow}"] & \\
!(A \sqcup B) \arrow[r, equal] & !(A \sqcup B)
\end{tikzcd}
.
$$

Cartesian structure

Definition

A bicategory $\mathcal{C}$ is **cartesian** when, for every objects Y, Z, there exist

an object $Y \mathbin{\&} Z \in \mathcal{C}$ and morphisms $l\colon Y \mathbin{\&} Z \to Y$ and $r\colon Y \mathbin{\&} Z \to Z$

such that, for every X, there is an adjoint equivalence of categories

$$\begin{tikzcd} \mathcal{C}(X, Y \mathbin{\&} Z) \arrow[rr, bend left, "(l \odot (-), r \odot (-))"] \arrow[rr, phantom, "\perp"] & & \mathcal{C}(X, Y) \times \mathcal{C}(X, Z) \arrow[ll, bend left, "\langle -, - \rangle"] \end{tikzcd}$$

(+ there exists a terminal object expressed as an adjoint equivalence too).

Cartesian structure

Theorem
The bicategory $\mathbf{Thin}_!$ *is cartesian.*

Cartesian structure

Theorem
The bicategory $\mathbf{Thin}_!$ *is cartesian.*

Given two thin groupoids $\mathcal{A}$ and $\mathcal{B}$, we take $\mathcal{A} \,\&\, \mathcal{B} \mathrel{\hat=} (A \sqcup B, \ldots)$ and

$$l = \begin{tikzcd} & & A \arrow[dl, "\bar{l}"'] \arrow[ddr, "\mathrm{id}_A"] & \\ & A \sqcup B \arrow[dl, "\eta_{A \sqcup B}"'] & & \\ \mathbf{List}^\star(A \sqcup B) & & & A \end{tikzcd} \quad \text{and} \quad r = \begin{tikzcd} & & B \arrow[dl, "\bar{r}"'] \arrow[ddr, "\mathrm{id}_B"] & \\ & A \sqcup B \arrow[dl, "\eta_{A \sqcup B}"'] & & \\ \mathbf{List}^\star(A \sqcup B) & & & B \end{tikzcd}$$

for $l\colon \mathcal{A} \,\&\, \mathcal{B} \to \mathcal{A}$ and $r\colon \mathcal{A} \,\&\, \mathcal{B} \to \mathcal{B}$ in $\mathbf{Thin}_!$.

Closure

A cartesian bicategory $\mathcal{C}$ is **closed** when, for every object Y, Z, there exist

an object $Y \Rightarrow Z \in \mathcal{C}$ and a morphism $\mathrm{ev}_{Y,Z} \colon (Y \Rightarrow Z) \,\&\, Y \to Z$

such that, for every $X \in \mathcal{C}$, there is an adjoint equivalence

$$\begin{tikzcd} \mathcal{C}(X, Y \Rightarrow Z) \arrow[rr, bend left, "\mathrm{ev}_{Y,Z} \odot (- \& Y)"] \arrow[rr, phantom, "\perp"] & & \mathcal{C}(X \,\&\, Y, Z) \arrow[ll, bend left, "(-)^{\dagger}"] \end{tikzcd} .$$

Theorem

The cartesian bicategory $\mathbf{Thin}_!$ *is closed.*

Closure

A cartesian bicategory $\mathcal{C}$ is **closed** when, for every object Y, Z, there exist

$$\text{an object } Y \Rightarrow Z \in \mathcal{C} \qquad \text{and} \qquad \text{a morphism } \mathrm{ev}_{Y,Z} \colon (Y \Rightarrow Z) \,\&\, Y \to Z$$

such that, for every $X \in \mathcal{C}$, there is an adjoint equivalence

$$\begin{tikzcd} \mathcal{C}(X, Y \Rightarrow Z) \arrow[rr, bend left, "\mathrm{ev}_{Y,Z} \odot (- \& Y)"] \arrow[rr, phantom, "\perp"] & & \mathcal{C}(X \,\&\, Y, Z) \arrow[ll, bend left, "(-)^{\dagger}"] \end{tikzcd} \; .$$

Theorem

The cartesian bicategory $\mathbf{Thin}_{!}$ *is closed.*

The closed structure for $\mathbf{Thin}_!$

Given thin groupoids $\mathcal{B}, \mathcal{C}$, we take $\mathcal{B} \Rightarrow \mathcal{C} \mathrel{\hat{=}} (!B \times C, \ldots)$ and

$$\mathrm{ev}_{\mathcal{B},\mathcal{C}} : (\mathcal{B} \Rightarrow \mathcal{C}) \mathbin{\&} \mathcal{B} \to \mathcal{C} \qquad = \qquad \begin{tikzcd}
 & & & !B \times C \arrow[dl, "{\langle l,r,l \rangle}"'] \arrow[dddrrr, "r"] & & & \\
 & & !B \times C \times !B \arrow[dl, "\eta_{!B \times C} \times !B"'] & & & & \\
 & !(!B \times C) \times !B \arrow[dl, "\mathrm{see}_{!B \times C, B}"'] & & & & & \\
 !((!B \times C) \sqcup B) & & & & & & C
\end{tikzcd}$$

(writting directly ! for $\mathbf{List}^\star$).

Probabilistic extension

Write **Prob** for the category

- with one object $\bullet$
- with $[0,1] \subset \mathbb{R}$ as the set of arrows
- with multiplication as composition

$$\mathbf{Prob} \quad = \quad \begin{tikzcd} \bullet \arrow[loop above, "{[0,1]}"] \end{tikzcd}$$

Probabilistic extension

A **span enriched in probabilities** is a span $S\colon A \nrightarrow B$ together with a natural transformation

$$
\begin{tikzcd}
& S \arrow[dl, "\partial^S_A"'] \arrow[dr, "\partial^S_B"] & \\
A \arrow[dr] \arrow[rr, phantom, "\overset{\nu^S}{\Longrightarrow}"] & & B \arrow[dl] \\
& \mathbf{Prob} &
\end{tikzcd}
$$

in **Cat**.

Probabilistic extension

A **span enriched in probabilities** is a span $S\colon A \nrightarrow B$ together with a natural transformation

$$
\begin{tikzcd}
 & S \arrow[dl, "\partial^S_A"'] \arrow[dr, "\partial^S_B"] & \\
A \arrow[dr] \arrow[rr, phantom, "\overset{\nu^S}{\Longrightarrow}"] & & B \arrow[dl] \\
 & \mathbf{Prob} &
\end{tikzcd}
$$

in **Cat**.

Concretely, it is the data of $\nu^S(s) \in [0, 1]$ for every $s \in S$ respecting the symmetries in S.

Probabilistic extension

Composing two spans enriched in probabilities S and T:

$$\nu^{T \odot S} \quad = \quad \begin{tikzcd}
 & & T \odot S \arrow[dl, dashed] \arrow[dr, dashed] \arrow[dd, phantom, "\vee", very near start] & & \\
 & S \arrow[dl, "\partial_A^S"'] \arrow[dr, "\partial_B^S"] & = & T \arrow[dl, "\partial_A^S"'] \arrow[dr, "\partial_B^S"] & \\
A \arrow[dr] & \overset{\nu^S}{\Longrightarrow} & B \arrow[dl] \arrow[dr] \arrow[d, phantom, "="] & \overset{\nu^T}{\Longrightarrow} & C \arrow[dl] \\
 & \mathbf{Prob} \arrow[rr, equal] & & \mathbf{Prob} &
\end{tikzcd}$$

Probabilistic extension

By computing the pasting diagram we have

$$\nu^{T \odot S}((s, t)) = \nu^{S}(s) \cdot \nu^{T}(t)$$

so that the composition of spans adequately multiplies the probabilities of the witnesses.

Other effectful extensions

More generally, we can hope for

Theorem (In preparation...)

Given an SMCC $\mathcal{C}$, there is a cartesian closed bicategory $\mathbf{Thin}_{!,\mathcal{C}}$ of spans enriched in $\mathcal{C}$.

We recover in our setting the weighting of gen. species of structures by SMCC:

- **tsukada2018species**

The end

Any questions?

Whiteboard