WIP

Jamie Brine

February 24, 2025

Abstract

TODO:

- indent all proof environments and remove space
- why are there big gaps in contents?
- work through maps for rewrites
- finish functors for connectives
- explain how functors properly model contexts, and thus prove contextual closure
- ultimately, prove naturality of rewrites

Contents

1 Introduction

- Brief background in modelling computation

- Deep inference

- BV, NEL

- Reddy's ideas

- Why we need my system

- Result we will prove

2 System sWIP

This system is a modification of NEL, first proposed by Guglielmi and Straßburger in 2002 [2]. The main difference between NEL and sWIP is that the ! and ? structures, which I refer to collectively as the *exponentials*, have been replaced by $\dagger$ and ϑ. These new exponentials can be considered ordered, which is reflected in the modified $\mathsf{b}{\downarrow}$ and $\mathsf{b}{\uparrow}$ rules governing their regeneration.

2.1 Structures and Equivalence

Definition 2.1. *Structures* in sWIP, denoted R,S,T,V.., are defined by the following grammar:

$$R ::= \circ \mid a \mid \overline{R} \mid \dagger R \mid \vartheta R \mid (R, .., R) \mid \langle R; ..; R\rangle \mid [R, .., R]$$

where

- $\circ$ is the *unit*, which is common to all structures and also self dual; this can be thought of as an empty structure
- a is an *atom*, of which there are countably many
- $\overline{R}$ is the *dual* of R
- $\dagger R$ and ϑR are the *dagger* and *hash* of R respectively
- $(R, .., R)$, $\langle R; ..; R\rangle$, and $[R, .., R]$ are *copar*, *seq*, and *par* structures; they are considered *proper* if they contain at least two elements

Definition 2.2. A structure with an empty hole, $S\{\ \}$, is called a *context*. We say R is a *substructure* of $S\{R\}$. If structural parentheses fill the hole exactly, the curly braces will be ommitted, so for example $S\{[R, T]\}$ becomes $S[R, T]$.

Definition 2.3. Structures R and S are considered *equivalent* modulo the relation $=$, which is the smallest congruence defined by the equations in Figure 1.

Associativity

$$(R, (T)) = (R, T)$$
$$\langle R; \langle T \rangle; U \rangle = \langle R; T; U \rangle$$
$$[R, [T]] = [R, T]$$

Commutativity

$$(R, T) = (T, R)$$
$$[R, T] = [T, R]$$

Singleton

$$(R) = \langle R \rangle = [R] = R$$

Unit

$$(R, T, \circ) = (R, T)$$
$$\langle R; T; \circ \rangle = \langle R; T \rangle$$
$$\langle \circ; R; T \rangle = \langle R; T \rangle$$
$$[R, T, \circ] = [R, T]$$

Dual

$$\overline{\circ} = \circ$$
$$\overline{(R, T)} = [\overline{R}, \overline{T}]$$
$$\overline{\langle R; T \rangle} = \langle \overline{R}; \overline{T} \rangle$$
$$\overline{[R, T]} = (\overline{R}, \overline{T})$$
$$\overline{\dagger R} = \vartheta \overline{R}$$
$$\overline{\vartheta R} = \dagger \overline{R}$$
$$\overline{\overline{R}} = R$$

Exponentials

$$\dagger \circ = \circ$$
$$\vartheta \circ = \circ$$

Contextual Closure

if $S\{R\} = S\{T\}$,
then $R = T$

Figure 1: Syntactic Equivalence

2.2 Rewrite Rules, Derivations, and Proofs

We describe how structures can be modified using a set of rules, and how these rules can be chained together to form longer derivations. In particular, we are interested in those derivations which begin with an empty premise, and describe a method with which to transform any derivation into one of this kind, which we call a proof.

Definition 2.4. A *rewrite rule* is a rule ρ with *premise* T and *conclusion* R, denoted $\rho\,\dfrac{T}{R}$. All of the rewrite rules in sWIP are given in Figure 2.

$$\mathsf{ai}{\downarrow}\,\frac{S\{\circ\}}{S[\bar{a},a]} \qquad \mathsf{w}{\downarrow}\,\frac{S\{\circ\}}{S\{\vartheta R\}} \qquad \mathsf{q}{\uparrow}\,\frac{S(\langle R;T\rangle,\langle U;V\rangle)}{S\langle (R,U);(T,V)\rangle}$$

$$\mathsf{s}\,\frac{S([R,T],U)}{S[(R,U),T]} \qquad \mathsf{b}{\downarrow}\,\frac{S\langle R;\vartheta R\rangle}{S\{\vartheta R\}} \qquad \mathsf{p}{\uparrow}\,\frac{S(\vartheta R,\dagger S)}{S\{\vartheta (R,S)\}}$$

$$\mathsf{q}{\downarrow}\,\frac{S\langle [R,T];[U,V]\rangle}{S[\langle R;U\rangle,\langle T;V\rangle]} \qquad \mathsf{ai}{\uparrow}\,\frac{S(a,\bar{a})}{S\{\circ\}} \qquad \mathsf{w}{\uparrow}\,\frac{S\{\dagger R\}}{S\{\circ\}}$$

$$\mathsf{p}{\downarrow}\,\frac{S\{\dagger[R,T]\}}{S[\dagger R,\vartheta T]} \qquad \mathsf{b}{\uparrow}\,\frac{S\{\dagger R\}}{S\langle R;\dagger R\rangle}$$

Figure 2: Rewrite Rules for System sWIP

Definition 2.5. A *derivation* Δ from R to T is a finite chain of rewrites with premise R and conclusion T, denoted $\begin{array}{c}R\\ \Delta \| \\ T\end{array}$. A derivation whose premise is $\circ$ is called a *proof*, to which we can add a topmost instance of the *axiom* rule $\circ{\downarrow}\,\dfrac{}{\circ}$; these are denoted $\begin{array}{c}\Delta \| \\ T\end{array}$.

We can turn any derivation into a proof with the following proposition, whose proof uses standard proof theoretical notions common to many systems.

Proposition 2.6. *If* $\begin{array}{c} R \\ \Delta \Vert \\ T \end{array}$ *is a derivation, then there exists a proof* $\begin{array}{c} \Delta \Vert \\ (\overline{R}, T) \end{array}$.

Remark. As sWIP is a deep inference system, a rewrite rule can be applied at arbitrary depth within any structure.

Example 2.7. We give a derivation where, when read top down, the substructure on which the rule is being applied is bolded. Note that this is not the simplest derivation from this premise to this conclusion, but has been chosen to demonstrate applying the rules at different levels within the structure.

$$\mathsf{b}\!\downarrow \frac{\mathsf{q}\!\downarrow \frac{\mathsf{p}\!\downarrow \frac{\mathsf{b}\!\uparrow \frac{\boldsymbol{\dagger[R,T]}}{\langle [R,T]; \boldsymbol{\dagger[R,T]}\rangle}}{\boldsymbol{\langle [R,T]; [\dagger R, \vartheta T]\rangle}}}{[\langle R, \dagger R\rangle, \boldsymbol{\langle T; \vartheta T\rangle}]}}{[\langle R; \dagger R\rangle, \vartheta T]}$$

Example 2.8. Negating the premise of the derivation from example 2.7, and then use the equivalence relation defined in Figure 1 to push negation to the level of atoms, we get the following:

$$\overline{\dagger[R,T]} = \vartheta\overline{[R,T]} = \vartheta(\overline{R}, \overline{T})$$

Using Proposition 2.6, we can hence guarantee that it is possible to contruct the following proof:

$$\begin{array}{c} \Gamma \Vert \\ (\vartheta(\overline{R}, \overline{T}), [\langle R; \dagger R\rangle, \vartheta T]) \end{array}$$

3 Coherence Space Semantics

We give denotational semantics to system sWIP using a coherence space model. Broadly speaking, the aim of this model of the system is to give a representation of proofs of some structure A as *cliques* of the corresponding coherence space $[\![A]\!]$.

One of the advantages of doing so is that it allows us to derive properties about the logical system by working with purely categorical constructs. Some desirable results may be difficult to prove using the logic of the system alone, but much simpler when we work with only the model.

Remark. We will give the semantics for sWIP in terms of only binary versions of the connectives copar, seq, and par. For example, instead of working with an arbitrary copar structure $(R_1, .., R_n)$, we will only consider the simpler case (R, T), and instead rely on the associative property to inductively construct more complex structures.

3.1 Introduction to Coherence Spaces

The idea of coherence spaces was first proposed by Jean-Yves Girard as a model for linear logic [1], and has since been adapted by a range of researchers to model their own systems. Most notably, Uday Reddy extended linear logic with an operator representing one-way communication [4], and gave a coherence space semantics on which much of this work has been based.

We introduce the following definitions which will be used in defining our model. For a more structured, category theoretic introduction to coherence spaces, the reader is referred to Paul-André Melliès' lecture notes [3].

Definition 3.1. A *coherence space* is a pair $(|A|, \supset_A)$, where $|A|$ is some underlying set and $\supset_A$ is the *coherence relation* defined on that set. The relation is symmetric, reflexive, and transitive. We also define a strictly irreflexive version $\frown_A$ which we call *strict coherence*, such that $a \frown_A a' \iff a \supset_A a'$ and $a \neq a'$

Definition 3.2. We say a and a' are *incoherent*, written $a \asymp_A a'$, if either $a \not\frown_A a'$ or $a = a'$

Remark. We will often refer to the coherence space $(|A|, \frown\!\!\!\smile_A)$ as simply A when appropriate. Similarly, we will often write $\frown\!\!\!\smile$ or $\frown$ without the subscript when the coherence space is obvious from the context.

Definition 3.3. A *clique* of a coherence space A is some $C \subseteq |A|$, whose elements are all pairwise coherent.

Remark. Note that as the clique is a set, the pairwise coherence is necessarily strict.

Definition 3.4. A *linear map* between coherence spaces A and B is some relation f on $|A|$ and $|B|$, such that if f relates a to b, and also relates a' to b', we have that

$$\begin{aligned} a \frown\!\!\!\smile_A a' &\implies b \frown\!\!\!\smile_B b' \\ a \frown_A a' &\implies b \frown_B b' \end{aligned}$$

We write $f : A \multimap B$, and use the notation $(\!|a, b|\!) \in f$ to mean "f relates a to b".

Definition 3.5. For linear maps $f : A \multimap B$ and $g : B \multimap C$, we define the *composition* $g \circ f : A \multimap C$ by:

$$g \circ f = \{(\!|a, c|\!) : \exists b \in |B| \text{ such that } (\!|a, b|\!) \in f, (\!|b, c|\!) \in g\}$$

Armed with these definitions, we can begin to construct a semantic model of the system sWIP.

3.2 Semantics of Structures

For each structure x in the grammar of sWIP, we will define the semantics $[\![x]\!]$. Thus, we can inductively define the semantics of an arbitrarily complex structure, starting by translating the atoms and then working outwards towards the outermost connective. Here we will state the semantics of each structure, then in the next section unpack and justify each definition.

Remark. Throughout this section, for any coherence space A we will only define $\frown\!\!\!\smile_A$. Deriving $\frown_A$ and $\asymp_A$ is a simple exercise in applying the definitions of a coherence space, and thus is left to the reader.

- **Unit**: $[\![\circ]\!] = (\{*\}, \{(*, *)\})$, a trivial coherence space whose underlying set is a singleton, with that single element related to itself. Notice that $* \not\frown *$ as the relation is strictly irreflexive, so $\frown_{[\![\circ]\!]} = \emptyset$.

- **Atom**: For each atom $a, b, .., z$ in a structure, we are able to choose any coherence space $A, B, .., Z$ to be the model $[\![a]\!], [\![b]\!], .., [\![z]\!]$. For example, an atom b representing the base type of booleans may have the semantics $[\![b]\!] = (\{\top, \bot\}, \{(\top, \top), (\bot, \bot)\})$.

- **Dual**: $[\![\overline{R}]\!] = (|[\![R]\!]|, \asymp_{[\![R]\!]})$.

- **Copar**: $[\![(R, T)]\!] = ([\![R]\!] \times [\![T]\!], \smile_{[\![(R,T)]\!]})$, such that:

$$(r_1, t_1) \smile (r_2, t_2) \iff r_1 \smile_{[\![R]\!]} r_2, t_1 \smile_{[\![T]\!]} t_2$$

- **Seq**: $[\![\langle R; T\rangle]\!] = ([\![R]\!] \times [\![T]\!], \smile_{[\![\langle R;T\rangle]\!]})$, such that:

$$(r_1, t_1) \smile (r_2, t_2) \iff r_1 \frown_{[\![R]\!]} r_2 \text{ or } r_1 = r_2, t_1 \smile_{[\![T]\!]} t_2$$

- **Par**: $[\![[R, T]]\!] = ([\![R]\!] \times [\![T]\!], \smile_{[\![[R,T]]\!]})$, such that:

$$(r_1, t_1) \smile (r_2, t_2) \iff r_1 \frown_{[\![R]\!]} r_2 \text{ or } t_1 \frown_{[\![T]\!]} t_2 \text{ or } (r_1, t_1) = (r_2, t_2)$$

- **Dagger**: $[\![\dagger R]\!] = ([\![R]\!]^*, \smile_{[\![\dagger R]\!]})$, such that:

$$r_1 ... r_m \smile r'_1 ... r'_n \iff \text{one of:}$$

  - $\exists 1 \leq l \leq \min(m, n)$ s.t. $r_i = r'_i \ \forall i < l$, and $r_l \frown_{[\![R]\!]} r'_l$
  - $r_i = r'_i \ \forall i \leq \min(m, n)$

- **Hash**: $[\![\vartheta R]\!] = ([\![R]\!]^*, \smile_{[\![\vartheta R]\!]})$, such that:

$$r_1 ... r_m \smile r'_1 ... r'_n \iff \text{one of:}$$

  - $\exists 1 \leq l \leq \min(m, n)$ s.t. $r_i = r'_i \ \forall i < l$, and $r_l \frown_{[\![R]\!]} r'_l$
  - $r_1 ... r_m = r'_1 ... r'_n$

3.3 Functoriality of Connectives

We will now take a step back from modelling the system sWIP, and view the coherence space model from a purely categorical perspective. For the rest of this section, we abstract out the underlying structure of any given coherence space, and instead consider how rewrite rules and connectives behave when applied to arbitrary ones.

We require that connectives in our model behave nicely when working in the category of coherence spaces. We will briefly define this category, and then show that we can model each of the unary and binary connectives as functors.

Modelling connectives as functors allows rewrite rules to be applied within arbitrary contexts, by composing the rule's model with the functor representing the context. This abstraction simplifies deep inference by making rule application uniform and modular, ensuring necessary properties are preserved at all levels of nesting. Contexts can be constructed compositionally, providing a scalable and elegant framework for handling rules in arbitrary settings.

Definition 3.6. The *category of coherence spaces* is written COHS, with coherence spaces as objects and linear maps between them as morphisms.

In order to prove that the some functor $F : \mathsf{COHS} \to \mathsf{COHS}$ preserves coherence of some map $f : A \multimap B$, we will show 2 things:

1. For $a \frown_A a'$, and pairs $(\!|a, b|\!), (\!|a', b'|\!) \in Ff$, we have that $b \frown_B b'$
2. For pairs $(\!|a, b|\!), (\!|a, b'|\!) \in Ff$, we have that $b \coh_B b'$

(1) directly shows preservation of strict coherence. As strict coherence is a stronger condition than coherence, it also shows preservation of coherence in all cases aside from that where $a = a'$; (2) verifies this case directly.

Definition 3.7. Let $(\_, \_) : \mathsf{COHS} \times \mathsf{COHS} \to \mathsf{COHS}$ be defined as follows:

- For $R, T \in Ob(\mathsf{COHS})$, $(R, T) \coloneqq (R \times T, \frown_{(R,T)})$ such that:

$$(r, t) \frown_{(R,T)} (r', t') \iff r \frown_R r' \text{ and } t \frown_T t'$$

- For $f : R \multimap T$ and $g : U \multimap V$, $(f, g) : (R, U) \multimap (T, V)$ is defined as follows:

$$(f, g) = \{ \llparenthesis (r, u), (t, v) \rrparenthesis : \llparenthesis r, t \rrparenthesis \in f, \llparenthesis u, v \rrparenthesis \in g \}$$

Lemma 3.8. $(\_, \_)$ *is a functor.*

Proof. $(\_, \_)$ preserves identity:

$$\begin{aligned}
(id_R, id_T) &= \{ \llparenthesis (r, t), (r, t) \rrparenthesis : \llparenthesis r, r \rrparenthesis \in id_R, \llparenthesis t, t \rrparenthesis \in id_T \} \\
&= \{ \llparenthesis (r, t), (r, t) \rrparenthesis : r \in |R|, t \in |T| \} \\
&= \{ \llparenthesis (r, t), (r, t) \rrparenthesis : (r, t) \in |(R, T)| \} \\
&= id_{(R,T)}
\end{aligned}$$

$(\_, \_)$ preserves composition:

Define 4 linear maps $f : R \multimap T$, $h : T \multimap U$, $g : V \multimap W$, and $k : W \multimap X$. Then we have $(f, g) : (R, V) \multimap (T, W), (h, k) : (T, W) \multimap (U, X)$, and:

$$\begin{aligned}
(h, k) \circ (f, g) = \{ &\llparenthesis (r, v)(u, x) \rrparenthesis : \exists (t, w) \in |(T, W)| \text{ such that} \\
&\llparenthesis (r, v), (t, w) \rrparenthesis \in (f, g), \llparenthesis (t, w), (u, x) \rrparenthesis \in (h, k) \} \\
= \{ &\llparenthesis (r, v)(u, x) \rrparenthesis : \exists t \in |T|, w \in |W| \text{ such that} \\
&\llparenthesis r, t \rrparenthesis \in f, \llparenthesis v, w \rrparenthesis \in g, \llparenthesis t, u \rrparenthesis \in h, \llparenthesis w, x \rrparenthesis \in k \} \\
= \{ &\llparenthesis (r, v), (u, x) \rrparenthesis : \llparenthesis r, u \rrparenthesis \in h \circ f, \llparenthesis v, x \rrparenthesis \in k \circ g \} \\
= (&h \circ f, k \circ g)
\end{aligned}$$

$(\_, \_)$ preserves coherence:

Define $f : R \multimap T$, $g : U \multimap V$.

First assume that $(r,u) \frown_{(R,U)} (r',u')$, and consider pairs $(\!|(r,u),(t,v)|\!)$, $(\!|(r',u'),(t',v')|\!) \in (f,g)$. Without loss of generality, assume that $r \frown_R r'$ and $u \smile_U u'$. Linearity of f and g implies that $t \frown_T t'$ and $v \smile_V v'$. This gives us $(t,v) \smile_{(T,V)} (t',v')$, and as $t \neq t'$ we find that the coherence is strict as required.

Now consider pairs $(\!|(r,u),(t,v)|\!), (\!|(r,u),(t',v')|\!) \in (f,g)$. Linearity of f and g implies that $t \smile_T t'$ and $v \smile_V v'$. This gives us $(t,v) \smile_{(T,V)} (t',v')$ as required.

□

Definition 3.9. Let $\langle \_;\_ \rangle : \mathsf{COHS} \times \mathsf{COHS} \to \mathsf{COHS}$ be defined as follows:

- For $R, T \in Ob(\mathsf{COHS})$, $\langle R;T \rangle := (|R| \times |T|, \smile_{\langle R;T \rangle})$ such that:

$$(r,t) \smile_{\langle R;T \rangle} (r',t') \iff r \frown_R r' \text{ or } (r = r' \text{ and } t \smile_T t')$$

- For $f : R \multimap T$ and $g : U \multimap V$, $\langle f;g \rangle : \langle R;U \rangle \multimap \langle T;V \rangle$ is defined as follows:

$$\langle f;g \rangle = \{(\!|(r,u),(t,v)|\!) : (\!|r,t|\!) \in f, (\!|u,v|\!) \in g\}$$

Lemma 3.10. *$\langle \_;\_ \rangle$ is a functor.*

Proof. $\langle \_;\_ \rangle$ preserves identity and composition. The argument is the same as that from the proof of Lemma 3.8, with any coherence space (R,T) replaced by $\langle R;T \rangle$

$\langle \_;\_ \rangle$ preserves coherence:

Define $f : R \multimap T$, $g : U \multimap V$.

First assume that $(r,u) \frown_{\langle R;U \rangle} (r',u')$, and consider pairs $(\!|(r,u),(t,v)|\!)$, $(\!|(r',u'),(t',v')|\!) \in \langle f;g \rangle$. We have that either $r \frown_R r'$, or that $r = r'$ and $u \frown_U u'$. As f and g are linear maps, the former implies that $t \frown t'$, while the latter implies that $t \smile t'$ and $v \frown v'$. In either case we get that $(t,v) \smile_{\langle T;V \rangle} (t',v')$, and as either $t \neq t'$ or $v \neq v'$ we find that the coherence is strict as required.

Now consider pairs $\llparenthesis (r,u),(t,v) \rrparenthesis, \llparenthesis (r,u),(t',v') \rrparenthesis \in \langle f; g \rangle$. Linearity of f implies that $t \smile_T t'$. We have that either $t \neq t'$, in which case $t \frown_T t'$, or that $t = t'$. Combining the second case with the linearity of g gives that $t = t'$ and $v \smile_V v'$, so either case gives us $(t,v) \smile_{(T,V)} (t',v')$ as required.

$\square$

Definition 3.11. Let $\overline{\{\quad\}} : \mathsf{COHS}^{op} \to \mathsf{COHS}$ be defined as follows:

- For $R \in Ob(\mathsf{COHS})$, $\overline{R} = (|R|, \asymp_R)$.
- For $f : R \multimap T$, $\overline{f} : \overline{T} \multimap \overline{R}$ is defined as follows:

$$\overline{f} = \{\llparenthesis t, r \rrparenthesis : \llparenthesis r, t \rrparenthesis \in f\}$$

Lemma 3.12. $\overline{\{\quad\}}$ *is a functor*

Proof. $\overline{\{\quad\}}$ preserves identity trivially, as the underlying set of $\overline{R}$ is the same as that of R:

$\overline{\{\quad\}}$ preserves composition:

Define 2 linear maps $f : R \multimap T$, $g : T \multimap U$. Then:

$$\begin{aligned}
\overline{f} \circ \overline{g} &= \{\llparenthesis u, r \rrparenthesis : \exists t \text{ such that } \llparenthesis u, t \rrparenthesis \in \overline{g}, \llparenthesis t, r \rrparenthesis \in \overline{f}\} \\
&= \{\llparenthesis u, r \rrparenthesis : \exists t \text{ such that } \llparenthesis r, t \rrparenthesis \in f, \llparenthesis t, u \rrparenthesis \in g\} \\
&= \{\llparenthesis u, r \rrparenthesis : \llparenthesis r, u \rrparenthesis \in g \circ f\} \\
&= \overline{g \circ f}
\end{aligned}$$

$\overline{\{\quad\}}$ preserves coherence:

Define $f : R \multimap T$

Take $t, t' \in |T|$ such that $t \frown_{\overline{T}} t'$, that is, $t \not\frown_T t'$. Consider pairs $\llparenthesis t, r \rrparenthesis, \llparenthesis t', r' \rrparenthesis \in \overline{f}$, so we have pairs $\llparenthesis r, t \rrparenthesis, \llparenthesis r', t' \rrparenthesis \in f$. As $t \not\frown_T t'$, the contrapositive to the linearity of f gives that $r \not\frown_R r'$, and thus $r \frown_{\overline{R}} r'$ as required.

If we instead consider pairs $(\!|t, r|\!), (\!|t, r'|\!) \in \overline{f}$, we have pairs $(\!|r, t|\!), (\!|r', t|\!) \in f$. Then $t = t' \implies t \asymp_T t'$, and again the contrapositive of linearity of f gives $r \asymp_R r'$, thus $r \supset_{\overline{R}} r'$ as required.

□

Definition 3.13. Let $\dagger_- : \mathsf{COHS} \to \mathsf{COHS}$ be defined as follows:

- For $R \in Ob(\mathsf{COHS})$, $\dagger R = (|R|^*, \supset_{\dagger R})$, such that:

  $$r_1 ... r_m \supset r'_1 ... r'_n \iff \text{one of:}$$

  - $\exists 1 \leq l \leq \min(m, n)$ s.t. $r_i = r'_i \ \forall i < l$, and $r_l \frown_R r'_l$
  - $r_i = r'_i \ \forall i \leq \min(m, n)$

  (Informally, 2 words over R are coherent in $\dagger R$ if one is a prefix of the other, or if the first place they differ they do so strictly coherently)

- For $f : R \multimap T$, $\dagger f : \dagger R \multimap \dagger T$ is defined as follows:

  $$\dagger f = \{(\!|r_1 ... r_n, t_1 ... t_n|\!) : (\!|r_i, t_i|\!) \in f \ \ \forall 1 \leq i \leq n\}$$

Lemma 3.14. $\dagger_-$ *is a functor*

Proof. $\dagger_-$ preserves identity:

$$\begin{aligned} \dagger id_R &= \{(\!|r_1 ... r_n, r_1 ... r_n|\!) : (\!|r_i, r_i|\!) \in id_R \ \ \forall 1 \leq i \leq n\} \\ &= \{(\!|r_1 ... r_n, r_1 ... r_n|\!) : r_i \in |R| \ \ \forall 1 \leq i \leq n\} \\ &= id_{\dagger R} \end{aligned}$$

$\dagger_-$ preserves composition:

Define 2 linear maps $f : R \multimap T$, $g : T \multimap U$. Then:

$$
\begin{aligned}
\dagger g \circ \dagger f &= \{ (\!| r_1 ... r_n, u_1 u_2 ... u_n |\!) : \exists t_1 ... t_n \in |T|^* \text{ such that} \\
&\quad (\!| r_1 ... r_n, t_1 ... t_n |\!) \in \dagger f, (\!| t_1 ... t_n, u_1 u_2 ... u_n |\!) \in \dagger g \} \\
&= \{ (\!| r_1 ... r_n, u_1 u_2 ... u_n |\!) : \exists t_i \in |T| \text{ such that} \\
&\quad (\!| r_i, t_i |\!) \in f, (\!| t_i, u_i |\!) \in g \;\; \forall 1 \leq i \leq n \} \\
&= \{ (\!| r_1 ... r_n, u_1 u_2 ... u_n |\!) : (\!| r_i, u_1 |\!) \in g \circ f \;\; \forall 1 \leq i \leq n \} \\
&= \dagger (g \circ f)
\end{aligned}
$$

$\dagger_{\_}$ preserves coherence:

Define $f : R \multimap T$.

First assume that $r_1 ... r_n \frown_{\dagger R} r'_1 ... r'_n$, and consider pairs $(\!| r_1 ... r_n, t_1 ... t_n |\!)$, $(\!| r'_1 ... r'_n, t'_1 ... t'_n |\!) \in \dagger f$. $\exists 1 \leq j \leq n$ where j is the smallest index such that $r_j \frown_R r'_j$. Linearity of f then gives that $t_i \supset\!\!\!\smile_T t'_i \;\; \forall 1 \leq i \leq j$, so in the first position that they differ they must do so coherently (even if all of $t_i = t'_i$, linearity of f ensures that $t_j \frown_T t'_j$). This gives $t_1 ... t_n \frown_{\dagger T} t'_1 ... t'_n$ as required.

Now consider pairs $(\!| r_1 ... r_n, t_1 ... t_n |\!)$, $(\!| r_1 ... r_n, t'_1 ... t'_n |\!) \in \dagger f$. Linearity of f gives that $t_i \supset\!\!\!\smile_T t'_i \;\; \forall 1 \leq i \leq n$. If $t_1 ... t_n = t'_1 ... t'_n$ then the proof is trivial, so assume they are not equal. $\exists 1 \leq j \leq n$ where j is the smallest index such that $t_j \neq t'_j$. However, as $t_j \supset\!\!\!\smile_T t'_j$, we must have that $t_j \frown_T t'_j$, giving $t_1 ... t_n \supset\!\!\!\smile_{\dagger T} t'_1 ... t'_n$ as required.

$\square$

The remaining connectives, Par and Hash, can be constructed compositionally, as they are the duals of Copar and Dagger respectively. That is, we can define:

$$[\_, \_] := \overline{(\overline{\{ \quad \}}, \overline{\{ \quad \}})} : \mathsf{COHS} \times \mathsf{COHS} \to \mathsf{COHS}$$

$$\vartheta_{\_} := \overline{\dagger \overline{\{ \quad \}}} : \mathsf{COHS} \to \mathsf{COHS}$$

Deriving the actions of these functors is left as an exercise to the reader; the results are as follows:

- For $R, T \in Ob(\mathsf{COHS})$, $[R, T] \coloneqq (R \times T, \coh_{[R,T]})$ such that:

$$(r, t) \coh_{[R,T]} (r', t') \iff r \frown_R r' \text{ or } t \frown_T t' \text{ or } (r, t) = (r', t')$$

- For $f : R \multimap T$ and $g : U \multimap V$, $[f, g] : [R, U] \multimap [T, V]$ is defined as follows:

$$[f, g] = \{ \llparenthesis (r, u), (t, v) \rrparenthesis : \llparenthesis r, t \rrparenthesis \in f, \llparenthesis u, v \rrparenthesis \in g \}$$

- For $R \in Ob(\mathsf{COHS})$, $\vartheta R = (|R|^*, \coh_{\vartheta R})$, such that:

$$r_1 ... r_m \coh r'_1 ... r'_n \iff \text{one of:}$$

  - $\exists 1 \leq l \leq \min(m, n)$ s.t. $r_i = r'_i \ \forall i < l$, and $r_l \frown_R r'_l$
  - $r_1 ... r_m = r'_1 ... r'_n$

  (Informally, 2 words over R are coherent in ϑR if they are equal, or if the first place they differ they do so strictly coherently)

- For $f : R \multimap T$, $\vartheta f : \vartheta R \multimap \vartheta T$ is defined as follows:

$$\vartheta f = \{ \llparenthesis r_1 ... r_n, t_1 ... t_n \rrparenthesis : \llparenthesis r_i, t_i \rrparenthesis \in f \ \ \forall 1 \leq i \leq n \}$$

We do not have to prove that these are functors, as composition of functors always results in functors.

With these final 2 functors, we now have a purely categorial model of any structure that could arise from the grammar of sWIP, which preserves all necessary properties no matter the choice of coherence space for atoms.

3.4 Semantics of Rewrite Rules

For each rewrite rule of sWIP, we define a linear map between coherence spaces to be its model. The map representing some rewrite rule $\rho \frac{T}{R}$ will be of the form $F_\rho : T \multimap R$.

We will consider the simplest form of each rule, that is, applying each in an empty context. Extending this to rewrites in arbitrary contexts is automatic, as we can represent the context as a functor.

Crucially, we must show that each of these maps preserve coherence (and thus cliques), which will allow chains of rewrites to model derivations (and thus proofs).

Definition 3.15. To model the rewrite

$$\mathsf{ai}\!\downarrow \frac{\circ}{[a, \overline{a}]}$$

we define $F_{\mathsf{ai}\downarrow} : \circ \multimap [a, \overline{a}]$ as follows:

$$F_{\mathsf{ai}\downarrow} = \{(\!|*, (a_i, a_i)|\!) : a_i \in |a|\}$$

Lemma 3.16. $F_{\mathsf{ai}\downarrow}$ *preserves coherence*

Proof. As $* \not\frown_\circ *$, preservation of strict coherence is vacuously true. Now consider pairs $(\!|*, (a_1, a_1)|\!), (\!|*, (a_2, a_2)|\!) \in F_{\mathsf{ai}\downarrow}$. As $* = *$ and thus $* \smile\!\!\!\!\frown *$, we must show that $(a_1, a_1) \smile\!\!\!\!\frown_{[a,\overline{a}]} (a_2, a_2)$. If $a_1 = a_2$ the coherence is trivial, so suppose that $a_1 \neq a_2$. We either have that $a_1 \frown_a a_2$, in which case we get $(a_1, a_1) \smile\!\!\!\!\frown_{[a,\overline{a}]} (a_2, a_2)$, or that $a_1 \not\frown_a a_2$. In the latter case, we get that $a_1 \frown_{\overline{a}} a_2$, and so $(a_1, a_1) \smile\!\!\!\!\frown_{[a,\overline{a}]} (a_2, a_2)$. □

Definition 3.17. To model the rewrite

$$\mathsf{w}\!\downarrow \frac{\circ}{\vartheta R}$$

we define $F_{\mathsf{w}\downarrow} : \circ \multimap \vartheta R$ as follows:

$$F_{\mathsf{w}\downarrow} = \{(\!|*, \epsilon|\!)\}$$

where ϵ is the empty sequence in$R$

Lemma 3.18. $F_{\mathsf{w}\downarrow}$ *preserves coherence*

Proof. Trivial, as any two empty sequences in R are coherent by equality. $\square$

Definition 3.19. To model the rewrite

$$\mathsf{s}\,\frac{([R,T],U)}{[(R,U),T]}$$

we define $F_{\mathsf{s}} : ([R,T],U) \multimap [(R,U),T]$ as follows:

$$F_{\mathsf{s}} = \{(\!|(r,t,u),(r,u,t)|\!) : r \in |R|, t \in |T|, u \in |U|\}$$

Lemma 3.20. *F_{s} preserves coherence*

Proof. Consider $(r,t,u),(r',t',u') \in |([R,T],U)|$. If the two are equal, it is trivial to show that F_{s} preserves this equality. Assume instead that $(r,t,u) \frown (r',t',u')$. To get the required coherence $(r,u,t) \frown_{[(R,U),T]} (r',u',t')$, it suffices to show that either $(r,u) \frown_{(R,U)} (r',u')$ or $t \frown t'$. We have 2 possible cases:

Case 1: $(r,t) \frown_{[R,T]} (r',t')$ and $u \smile u'$

> $(r,t) \frown (r',t') \implies r \frown r'$ or $t \frown t'$. The latter alone suffices, and the former combined with $u \smile u'$ gives $(r,u) \frown_{(R,U)} (r',u')$ as required.

Case 2: $(r,t) \smile_{[R,T]} (r',t')$ and $u \frown u'$

> The same 2 cases as above can arise here, or alternatively we may have that $(r,t) = (r',t')$. In this case, we have that $r \smile r'$, which combined with $u \frown u'$ gives $(r,u) \frown_{(R,U)} (r',u')$ as required.

$\square$

Definition 3.21. To model the rewrite

$$\mathsf{q}{\downarrow}\,\frac{\langle [R,T];[U,V]\rangle}{[\langle R;U\rangle,\langle T;V\rangle]}$$

we define $F_{\mathsf{q}\downarrow} : \langle [R,T];[U,V]\rangle \multimap [\langle R;U\rangle,\langle T;V\rangle]$ as follows:

$$F_{\mathsf{q}\downarrow} = \{(\!|(r,t,u,v),(r,u,t,v)|\!) : r \in |R|, t \in |T|, u \in |U|, v \in |V|\}$$

Lemma 3.22. *$F_{\mathsf{q}\downarrow}$ preserves coherence*

Proof. Consider $(r,t,u,v),(r',t',u',v') \in |\langle[R,T];[U,V]\rangle|$. If the two are equal, it is trivial to show that $F_{\mathsf{q}\downarrow}$ preserves this equality. Assume instead that $(r,t,u,v) \frown (r',t',u',v')$. To get the required coherence $(r,u,t,v) \frown_{[\langle R;U\rangle,\langle T;V\rangle]} (r',u',t',v')$, it suffices to show that either $(r,u) \frown_{\langle R;U\rangle} (r',u')$ or $(t,v) \frown_{\langle T;V\rangle} (t',v')$. We have 2 possible cases:

Case 1: $(r,t) \frown_{[R,T]} (r',t')$

> We have that either $r \frown r'$, implying $(r,u) \frown (r',u')$, or $t \frown t'$, in which case $(t,v) \frown (t',v')$. In either case the sufficient condition is met.

Case 2: $(r,t) = (r',t')$ and $(u,v) \frown_{[U,V]} (u',v')$

> We have that either $u \frown u'$ or $v \frown v'$. As $r = r'$, the former implies $(r,u) \frown (r',u')$, and similarly the latter implies $(t,v) \frown (t',v')$. In either case the sufficient condition is met.

$\square$

Definition 3.23. To model the rewrite

$$\mathsf{p}\downarrow \frac{\dagger[R,T]}{[\dagger R, \vartheta T]}$$

we define $F_{\mathsf{p}\downarrow} : \dagger[R,T] \multimap [\dagger R, \vartheta T]$ as follows:

$$F_{\mathsf{p}\downarrow} = \{(\!| r_1 t_1 ... r_m t_m, (r'_1 ... r_n, t'_1 ... t'_p) |\!) : r_i, r'_j \in |R|, t_k, t'_l \in |T|\}$$

Lemma 3.24. *$F_{\mathsf{p}\downarrow}$ preserves coherence*

Proof. DO THIS BIT HERE PLEASE $\square$

Definition 3.25. To model the rewrite

$$\mathsf{b}\downarrow \frac{\langle R; \vartheta R\rangle}{\vartheta R}$$

we define $F_{\mathsf{b}\downarrow} : \langle R; \vartheta R\rangle \multimap \vartheta R$ as follows:

$$F_{\mathsf{b}\downarrow} = \{(\!|\langle r_0; r_1 ... r_m, r'_1 ... r'_n |\!) : r_i, r'_j \in |R|\}$$

Lemma 3.26. $F_{\mathsf{p}\downarrow}$ *preserves coherence*

Proof. DO THIS BIT HERE PLEASE □

SOMETHING SOMETHING DUALITY OF RULES MAKES UP RULES AUTOMATIC.

4 Some Big Result...

Theorem 4.1. *Cor blimey look at the state of this stonking result!*

Proof. Bodge together all of the lemmas from the previous chapters and there you go. Easy. No bother at all. □

5 Conclusion

References

[1] Jean-Yves Girard. Linear logic. *Theoretical computer science*, 50(1):1–101, 1987.

[2] Alessio Guglielmi and Lutz Straßburger. A non-commutative extension of mell. In *Logic for Programming, Artificial Intelligence, and Reasoning: 9th International Conference, LPAR 2002 Tbilisi, Georgia, October 14–18, 2002 Proceedings 9*, pages 231–246. Springer, 2002.

[3] Paul-André Melliès. A survival guide on coherence spaces, 2000.

[4] Uday S. Reddy. A linear logic model of state. 1993.