Spinal Atomic Lambda-Calculus

Tom Gundersen

Red Hat, Inc.

teg@jklm.no

Willem Heijltjes

University of Bath, England, UK

http://www.cs.bath.ac.uk/~wbh22/

w.b.heijltjes@bath.ac.uk

Michel Parigot

Laboratoire PPS, UMR 7126, CNRS & Université Paris 7 (France)

michel.parigot@gmail.com

David R. Sherratt

Friedrich-Schiller University Jena, Germany

david.rhys.sherratt@uni-jena.de

───── **Abstract** ─────

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Praesent convallis orci arcu, eu mollis dolor. Aliquam eleifend suscipit lacinia. Maecenas quam mi, porta ut lacinia sed, convallis ac dui. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Suspendisse potenti.

1 Introduction

We investigate the computational meaning of the following *switch* rule of deep-inference proof theory [**?**, **?**]:

$$\frac{(A \to B) \land C}{A \to (B \land C)} s$$

On its own, it corresponds to an *end-of-scope* marker in λ-calculus. This is a special annotation of a subterm, to indicate that a given variable does not occur free, so that a substitution on that variable can be aborted early. In the above rule, A corresponds to the binding variable of an abstraction and C to the subterm of said abstraction where it doesn't occur, while B represents those subterms where it does occur.

The main thrust of our work is to incorporate this rule, and its computational interpretation as a term construct, into the *atomic λ-calculus* [**?**]. This calculus results from an investigation of the following *medial* rule:

$$\frac{(A \lor B) \to (C \land D)}{(A \to C) \land (B \to D)} m$$

The medial rule enables duplication to proceed *atomically*: on individual constructors (abstraction and application) rather than entire subterms. The atomic λ-calculus implements *full laziness*, a standard notion of sharing where only the *skeleton* of a term needs to be duplicated. Given a term t which needs to be duplicated, full laziness allows to share all maximal subterms $u_1, \ldots, u_k$ of t that do not contain occurrences of a variable bound in t outside u_i. The constructors in t not in any u_i are then part of the skeleton.

Our investigation is then focused on the interaction of switch and medial. Based on this we develop the *spinal atomic λ-calculus*, a natural evolution of the atomic λ-calculus. The new calculus improves on full laziness with *spinal full laziness*, which duplicates the *spine* rather than the skeleton. The spine of an abstraction are the direct paths from the binder to bound variables [**?**]. The graph below provides an example of this for the term $\lambda x.(\lambda z.z)\lambda y.(yy)x$, where the spine of λx is the very thick red line and the largest subterms that could be identied by an end-of-scope operator in the term calculus (or the switch rule in the proof theory) are enclosed by yellow boxes.
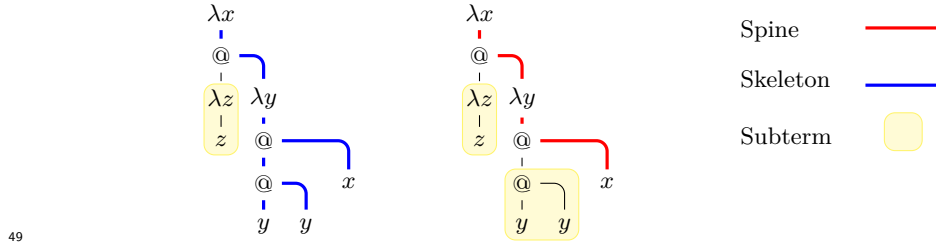
1.1   Related Work

Spine duplication has been implemented by Blanc et al. in [**?**], by making use of labels in a dag-implementation. The main purpose of their work is to study sharing in Wadsworth's *weak λ-calculus* [**?**] (further studied in [**?**]). Balabonski [**?**] showed that spine duplication allows for an optimal reduction in the sense of Lévy [**?**] for *weak reduction* i.e. where a β-reduction $(\lambda x.t)s$ occurring in a subterm u can only reduce if all free variables in the redex are also free in the term u. Blelloch and Greiner [**?**] showed that the weak call-by-value reduction strategy can be implemented in polynomial time with respect to the size of the initial term and the number of β steps in said term. Given the restriction that u is a closed term, this is then the same as *closed reduction* [**?**, **?**]. Our work generalizes spine duplication to the λ-calculus. It uses environments to implement sharing, and does not make use of labels, while maintaining a close intuition to dag-implementations.

End-of-scope markers in the λ-calculus have been seen throughout literature. *Berkling's lambda bar* [**?**] has shown to remove the need for variable names while maintaining correctness; improving efficiency by removing the need for alpha conversion [**?**]. This result was generalized by *Adbmal* (invert of "Lambda") [**?**]. It was shown by using multiple variables, scopes can be sequenced rather than nested which correspond closely to the boxes in MELL proof nets of linear logic [**?**]. This approach was studied further in [**?**] as graph reduction that satisfies optimality [**?**]. Although these approaches could identify the skeleton of a term, none however identify the spine of a term, which meant the scopes explicitly displayed may be larger than necessary from the perspective of performing substitution. This problem was solved by *director strings*, introduced by Kennaway and Sleep in [**?**] for combinator reduction and then generalized for any strategy by Fernández et al. in [**?**]. Director strings are an annotation on terms detailing the location of variable occurrences. An apt annotation on the body of an abstraction will consequently identify the spine of it. Despite being implementations that use director strings [**?**, **?**, **?**], an implementation with sharing techniques allowing for duplicating solely the spine could not be found.

Introduce the rest of the paper.

2   Deep Inference

Deep inference is a methodology for designing proof systems. Inference rules in deep
inference, such as switch and medial, can be applied 'deeply' i.e. there is no concept of
the main connective of a formula. The *open deduction* formalism [**?**] is designed around
this principle, where logical connectives can be applied at the level of derivations as well as
formulae. A derivation from premise A to conclusion C (over the connectives conjunction
and implication) is constructed as follows,

▶ **Definition 1.** *A* derivation *in open deduction is defined as follows.*

$$
\begin{array}{c} A \\ \| \\ C \end{array}
\quad ::= \quad
A \quad | \quad
\begin{array}{cc} A_1 & A_2 \\ \| \wedge \| \\ C_1 & C_2 \end{array}
\quad | \quad
\begin{array}{cc} C_1 & A_2 \\ \| \rightarrow \| \\ A_1 & C_2 \end{array}
\quad | \quad
\begin{array}{c} A \\ \| \\ \dfrac{B_1}{B_2} r \\ \| \\ C \end{array}
$$

where from left to right, (1) the premise and the conclusion can be the same formula i.e.
$A = C$. (2) We can compose derivations horizontally with a conjunction $\wedge$, where $A = A_1 \wedge A_2$
and $C = C_1 \wedge C_2$. (3) We can compose derivations horizontally with an implication $\rightarrow$ where
$A = A_1 \rightarrow A_2$ and $C = C_1 \rightarrow C_2$. Note that the derivation on the antecedent of the implication
is inverted; it can be interpreted as a derivation where we treat the premise as the conclusion
and the conclusion as the premise. Lastly (4) derivations can be composed vertically with
an inference rule r from B_1 to B_2. We work modulo symmetry, associativity, and unit laws
of conjunction. Additionally the generic vertical composition of two derivations (without a
mediating rule) exists as a derived operation in [**?**].

Open deduction was used to type the *basic calculus*, which was introduced in [**?**] as a
basis for the atomic λ-calculus. We follow the same approach here; we reintroduce the basic
calculus and show its typing system can be extended with the switch rule, and later expand
on this to introduce the spinal atomic λ-calculus. We obtain a formulation of minimal logic
together with the switch rule, from embedding its usual natural deduction system into open
deduction. The rules are (respectively) called abstraction, switch, application and (n-ary)
contraction from left to right.

$$
\frac{\top}{A \rightarrow A} \lambda \qquad
\frac{(A \rightarrow B) \wedge C}{A \rightarrow (B \wedge C)} s \qquad
\frac{A \wedge (A \rightarrow B)}{B} @ \qquad
\frac{A}{A \wedge \cdots \wedge A} \vartriangle
$$

These rules are used to type terms of the *basic calculus*, given by the grammar

▶ **Definition 2.** $s, t \quad ::= \quad x \quad | \quad \lambda x.t \quad | \quad s\,t \quad | \quad s[x_1, \ldots, x_n \leftarrow t]$

where the four constructors are called, from left to right, variable, abstraction, application
and *sharing*. This is a *linear* calculus, so each variable occurs at most once, and a sharing
construct is used to represent multiple occurrences of a variable (or term). The variable
bound by an abstraction must occur within the body of the abstraction i.e. in the term $\lambda x.t$,
$x \in (t)_{fv}$. Lastly and similarly, each variable bound by the sharing construct must occur and
become bound i.e. in the term $s[x_1, \ldots, x_n \leftarrow t]$, each $x_i \in (u)_{fv}$ for all $1 \le i \le n$.

For a given set $\{a, b, c, \ldots\}$ of *atomic formulae*, the following two grammars define *minimal
formulae* and *conjunctive formulae* respectively.

$$
A, B, C \quad ::= \quad a \mid A \rightarrow B \qquad\qquad \Gamma, \Delta \quad ::= \quad A \mid \top \mid \Gamma \wedge \Delta
$$

The typing derivations in open deduction for this calculus is displayed in Figure 1, where the corresponding types and derivations for terms are in red. We add the boxes to aid the reader see the horizontal composition of derivations. The colour of the box has no meaning other than to help identify derivations. A variable x may be typed by any minimal formula A, while the other constructors each correspond to inference rules, used within the context of further derivations. A term t with free variables $x_1, \ldots, x_n$ can be typed by a derivation from assumption $A_1^{x_1} \wedge \cdots \wedge A_n^{x_n}$ to conclusion C. A *typing judgement* $t : C$ then expresses that t is typeable by a derivation with conclusion C. Note that a derivation occurring as the antecedent of an implication is always a minimal formula.
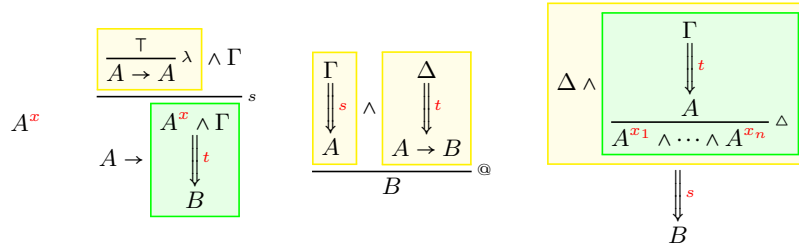
Figure 1 Typing derivations for terms in the basic calculus

The semantics of the basic calculus, including the compilation and readback into the λ-calculus, can be found in [**?**] and will not be repeated here.

Although in the derivation for an abstraction in Figure 1 we place the switch rule directly after the abstraction rule, this is not always the case. Consider the term $t = \lambda x.\lambda y.xy$, where $x : A \rightarrow B = C$ and $y : A$. The following derivations are both typing judgements of $t : C \rightarrow A \rightarrow B$.
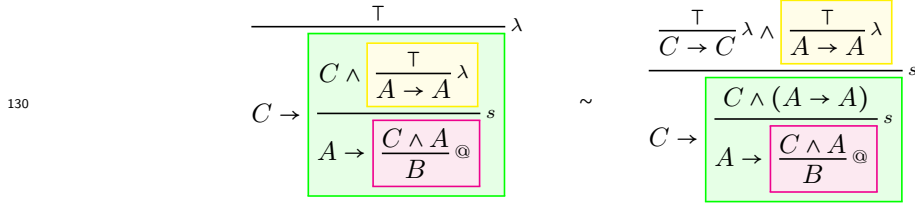
The main difference between the two is that the first is *scope-balanced* while the second is *unbalanced* (terminology taken from [**?**]). In derivations, the *scope* of an abstraction is considered to be the subderivation found underneath the corresponding switch rule. In the scope-balanced derivation, the scope of the binder λx is the whole body of the term; any scopes of any binders underneath λx (such as λy) are considered nested within the scope of λx. The scope of an abstraction in the scope-balanced derivation corresponds to the skeleton of the term. In the unbalanced derivation, scopes are not strictly nested but can overlap. The variable y (with type A) is now identified by the switch rule corresponding with λx, and is not considered within the scope of λx in contrast to the balanced derivation where is was nested. The unbalanced scope of an abstraction corresponds with the spine of the term. Both derivations identify the application in all scope.

The atomic λ-calculus can be seen as using only scope-balanced derivations, and here we show that by using unbalanced derivations we can identify the spine of an abstraction, and moreover when introducing the medial rule to the typing system duplicate said spine by proof normalisation.

3 The Spinal Atomic λ-Calculus

We now formally introduce the syntax of the spinal atomic λ-calculus (Λ_a^S).

▶ **Definition 3** (Pre-Terms). *The pre-terms $t \in \Lambda_a^S$ are defined by the following syntax*

$$t \quad ::= \quad x \quad | \quad t\,t \quad | \quad x\langle\, \vec{y}\,\rangle.t \quad | \quad t[\Gamma]$$

$$[\Gamma] \quad ::= \quad [\vec{x} \leftarrow t] \quad | \quad [e_1\langle\, \vec{x_1}\,\rangle \ldots e_n\langle\, \vec{x_n}\,\rangle \,|\, d\langle\, \vec{y}\,\rangle\, \overline{[\Gamma]}]$$

$$\overline{[\Gamma]} \quad ::= \quad [\Gamma] \quad | \quad \overline{[\Gamma]}[\Gamma]$$

We write $\vec{x}$ for a sequence of variables $x_1, \ldots, x_n$ for $n \geq 0$. An abstraction $x\langle\, x\,\rangle.t$ and a phantom-abstraction $x\langle\, \vec{y}\,\rangle.t$ are two instances of the same construct. We call the variables inside the brackets the *cover* of the abstraction. If the cover is the same as the preceeding variable, then we consider it to be an abstraction, otherwise it is a phantom-abstraction and we call the preceeding variable a *phantom-variable*. The distributor $u[e_1\langle\, \vec{x_1}\,\rangle \ldots e_n\langle\, \vec{x_n}\,\rangle \,|\, d\langle\, \vec{y}\,\rangle\, \overline{[\Gamma]}]$ captures the phantom-variables $e_1, \ldots, e_n$ in u and the covers associated with those phantom-variables are captured by the environment $\overline{[\Gamma]}$, which is a collection of closures $[\Gamma]$. We sometimes write the distributor as $u[\overrightarrow{e\langle\, x\,\rangle} \,|\, d\langle\, \vec{y}\,\rangle\, \overline{[\Gamma]}]$ when we are not concerned about the binding of phantom-variables. Terms are then pre-terms with sensible and correct bindings. To define terms, we first define *free* and *bound* variables and phantom variables.

▶ **Definition 4** (Free and Bound Variables). *The free variables $(-)_{fv}$ and bound variables $(-)_{bv}$ of a pre-term t is defined as follows*

$$(x)_{fv} = \{x\} \qquad\qquad\qquad\qquad (x)_{bv} = \{\}$$

$$(s\,t)_{fv} = (s)_{fv} \cup (t)_{fv} \qquad\qquad\qquad (s\,t)_{bv} = (s)_{bv} \cup (t)_{bv}$$

$$(x\langle\, x\,\rangle.t)_{fv} = (t)_{fv} - \{x\} \qquad\qquad (x\langle\, x\,\rangle.t)_{bv} = (t)_{bv} \cup \{x\}$$

$$(c\langle\, \vec{x}\,\rangle.t)_{fv} = (t)_{fv} \qquad\qquad\qquad (c\langle\, \vec{x}\,\rangle.t)_{bv} = (t)_{bv}$$

$$(u[\vec{x} \leftarrow t])_{fv} = (u)_{fv} \cup (t)_{fv} - \vec{x} \qquad (u[\vec{x} \leftarrow t])_{bv} = (u)_{bv} \cup (t)_{bv} \cup \{\vec{x}\}$$

$$(u[\overrightarrow{e\langle\, x\,\rangle} \,|\, c\langle\, c\,\rangle\, \overline{[\Gamma]}])_{fv} = (u\overline{[\Gamma]})_{fv} - \{c\} \qquad (u[\overrightarrow{e\langle\, x\,\rangle} \,|\, c\langle\, c\,\rangle\, \overline{[\Gamma]}])_{bv} = (u\overline{[\Gamma]})_{bv}$$

$$(u[\overrightarrow{e\langle\, x\,\rangle} \,|\, c\langle\, \vec{y}\,\rangle\, \overline{[\Gamma]}])_{fv} = (u\overline{[\Gamma]})_{fv} \cup \{c\} \qquad (u[\overrightarrow{e\langle\, x\,\rangle} \,|\, c\langle\, \vec{y}\,\rangle\, \overline{[\Gamma]}])_{bv} = (u\overline{[\Gamma]})_{bv}$$

▶ **Definition 5** (Free and Bound Phantom-Variables). *The free phantom-variables $(-)_{fp}$ and bound phantom-variables $(-)_{bp}$ of the pre-term t is defined as follows*

$$(x)_{fp} = \{\} \qquad\qquad\qquad\qquad (x)_{bp} = \{\}$$

$$(s\,t)_{fp} = (s)_{fp} \cup (t)_{fp} \qquad\qquad\qquad (s\,t)_{bp} = (s)_{bp} \cup (t)_{bp}$$

$$(x\langle\, x\,\rangle.t)_{fp} = (t)_{fp} \qquad\qquad\qquad (x\langle\, x\,\rangle.t)_{bp} = (t)_{bp}$$

$$(c\langle\, \vec{x}\,\rangle.t)_{fp} = (t)_{fp} \cup \{c\} \qquad\qquad (c\langle\, \vec{x}\,\rangle.t)_{bp} = (t)_{bp}$$

$$(u[\vec{x} \leftarrow t])_{fp} = (u)_{fp} \cup (t)_{fp} \qquad\qquad (u[\vec{x} \leftarrow t])_{bp} = (u)_{bp} \cup (t)_{bp}$$

$$(u[e_1\langle\, \vec{x_1}\,\rangle \ldots e_n\langle\, \vec{x_n}\,\rangle \,|\, c\langle\, c\,\rangle\, \overline{[\Gamma]}])_{fp} = (u\overline{[\Gamma]})_{fp} - \{e_1, \ldots, e_n\}$$

$$(u[e_1\langle\, \vec{x_1}\,\rangle \ldots e_n\langle\, \vec{x_n}\,\rangle \,|\, c\langle\, c\,\rangle\, \overline{[\Gamma]}])_{bp} = (u\overline{[\Gamma]})_{bp} \cup \{e_1, \ldots, e_n\}$$

$$(u[e_1\langle\, \vec{x_1}\,\rangle \ldots e_n\langle\, \vec{x_n}\,\rangle \,|\, c\langle\, \vec{y}\,\rangle\, \overline{[\Gamma]}])_{fp} = (u\overline{[\Gamma]})_{fp} \cup \{c\} - \{e_1, \ldots, e_n\}$$

$$(u[e_1\langle\, \vec{x_1}\,\rangle \ldots e_n\langle\, \vec{x_n}\,\rangle \,|\, c\langle\, \vec{y}\,\rangle\, \overline{[\Gamma]}])_{bp} = (u\overline{[\Gamma]})_{bp} \cup \{e_1, \ldots, e_n\}$$

187 Variables are bound by abstractions (not phantoms) and sharings. Phantom-variables are
188 bound by distributors. With these definitions, we can formally define the terms of Λ_a^S.

189 ▶ **Definition 6** (Terms). *A term $t \in \Lambda_a^S$ is a pre-term with the following constraints*
190 **1.** *Each variable may occur at most once.*
191 **2.** *In an abstraction $x\langle x \rangle.t$, $x \in (t)_{fv}$.*
192 **3.** *In a phantom-abstraction $c\langle x_1, \ldots, x_n \rangle.t$, $\{x_1, \ldots, x_n\} \subset (t)_{fv}$.*
193 **4.** *In a sharing $u[x_1, \ldots, x_n \leftarrow t]$, $\{x_1, \ldots, x_n\} \subset (u)_{fv}$.*
194 **5.** *In a distributor $u[e_1\langle w_1^1, \ldots, w_{k_1}^1 \rangle \ldots e_n\langle w_1^n, \ldots, w_{k_n}^n \rangle | c\langle c \rangle \overline{[\Gamma]}]$*
195   **a.** *For all $1 \leq i \leq n$ and $1 \leq m \leq k_n$, $w_m^i (u)_{fv}$ and becomes bound by $\overline{[\Gamma]}$.*
196   **b.** *$\{e_1\langle w_1^1, \ldots, w_{k_1}^1 \rangle, \ldots, e_n\langle w_1^n, \ldots, w_{k_n}^n \rangle\} \subset (u)_{fc}$, and $\{e_1, \ldots, e_n\} \subset (u)_{fp}$, and each e_i*
197   *becomes bound.*
198   **c.** *The variable c occurs somewhere in the environments $\overline{[\Gamma]}$.*
199 **6.** *In a distributor $u[e_1\langle w_1^1, \ldots, w_{k_1}^1 \rangle \ldots e_n\langle w_1^n, \ldots, w_{k_n}^n \rangle | c\langle y_1, \ldots, y_m \rangle \overline{[\Gamma]}]$*
200   **a.** *Both 5(a) and 5(b) hold.*
201   **b.** *For all $1 \leq i \leq m$, y_i occurs in the environments $\overline{[\Gamma]}$.*

202 We consider terms equal up to the congruence induced by the exchange of closures.
203 Consider the term $t[\Gamma_1][\Gamma_2]$ where $[\Gamma_1]$ and $[\Gamma_2]$ are both closures. Then $t[\Gamma_1][\Gamma_2] \sim$
204 $t[\Gamma_2][\Gamma_1]$ iff $[\Gamma_2]$ only binds variables and phantom-variables located in t. This equivalence is
205 essential to the rewriting theory. We also consider terms equal up to symmetry of contraction.
206 We consider the sequence of variables xs modulo permutations. Let $\vec{x}$ be a list of variables
207 and let $\vec{x_P}$ be a permutation of that list, then the following terms are considered equal.

208 $$u[\vec{x} \leftarrow t] \sim u[\vec{x_P} \leftarrow t] \qquad c\langle \vec{x} \rangle.t \sim c\langle \vec{x_P} \rangle.t$$

209 ## 3.1 Typing System
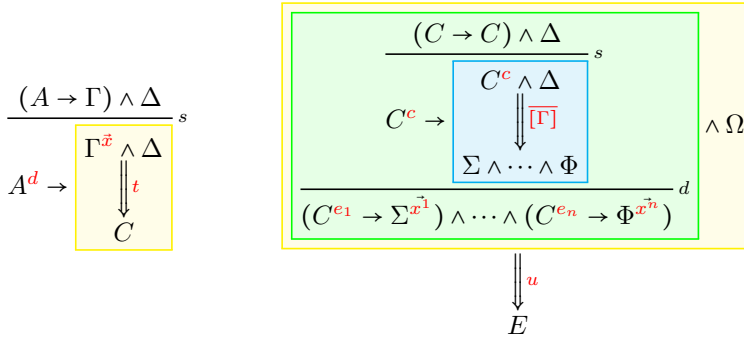
■ **Figure 2** Typing derivations for phantom-abstractions and distributors

210 The terms typed by the derivations in Figure 1 and Figure 2. Figure 2 shows the derivations
211 for the terms $d\langle \vec{x} \rangle.t$ and $u[e_1\langle \vec{x_1} \rangle \ldots e_n\langle \vec{x_n} \rangle | c\langle c \rangle \overline{[\Gamma]}]$. The distributor construct is typed
212 using the medial rule as in [**?**]. Notice that the medial rule in Figure 2 does not use disjunction
213 compared to the medial rule in the introduction. In the derivation we combine the medial
214 rule with a co-contraction rule to form the *distribution rule* (d). Since the formula in the
215 ante-cedent of an implication is always a minimal formula, doing this allows us to avoid
216 introducing disjunction into the typing system.
217 The main difference between our calculus is the bindings. We create a new class of
218 bindings, where phantom-variables are captured by the distributor but variables are captured

by the environment of the distributor. This shows in the derivations since the types of the variables (Σ and Φ) are not captured by the distribution rule.

3.2   Compilation and Readback

We now define the translations between Λ_a^S and the original λ-calculus. First we define the interpretation $\Lambda \to \Lambda_a^S$ (*compilation*). Intuitively, it replaces each abstraction $\lambda x.-$ with the term $x\langle x \rangle. - [x_1, \ldots, x_n \leftarrow x]$ where $x_1, \ldots, x_n$ replace the occurrences of x. Actual substitutions are denoted as $\{t/x\}$. Let $|M|_x$ denote the number of occurrences of x in M, and if $|M|_x = n$ let $M\frac{n}{x}$ denote M with the occurrences of x by fresh, distinct variables $x^1, \ldots, x^n$. First, the translation of a *closed* term M is $(\!| M |\!)'$, defined below

▶ **Definition 7** (Compilation). *The interpretation for closed lambda terms, $(\!| \Lambda |\!)' : \Lambda \to \Lambda_a^S$ is defined below*

$$(\!| x |\!)' = x$$

$$(\!| M\,N |\!)' = (\!| M |\!)'\,(\!| N |\!)'$$

$$(\!| \lambda x.M |\!)' = \begin{cases} x\langle x \rangle.(\!| M |\!)' & \textit{if } |M|_x = 1 \\ x\langle x \rangle.(\!| M\frac{n}{x} |\!)'[x^1, \ldots, x^n \leftarrow x] & \textit{if } |M|_x = n \neq 1 \end{cases}$$

For an arbitrary term M, if $x_1, \ldots, x_k$ are the free variables of M such that $|M|_{x_i} = n_i > 1$, the translation $(\!| M |\!)$ is

$$(\!| M\frac{n_1}{x_1} \ldots \frac{n_k}{x_k} |\!)'[x_1^1, \ldots, x_1^{n_1} \leftarrow x_1] \ldots [x_k^1, \ldots, x_k^{n_k} \leftarrow x_k]$$

The readback into the λ-calculus is slightly more complicated, specifically due to the bindings induced by the distributor. Interpreting a distributor $[\![u[e_1\langle \vec{x_1} \rangle \ldots e_n\langle \vec{x_n} \rangle | c\langle c \rangle \overline{[\Gamma]}]]\!]$ construct as a λ-term requires (1) converting the phantom-abstractions it binds in u into abstractions (2) collapsing the environment (3) maintaining the bindings between the converted abstractions and the intended variables located in the environment.

▶ **Definition 8.** *Given a total function σ with domain D and codomain C, we* overwrite *the function with case $x \mapsto V$ where $x \in D$ and $V \in C$ such that*

$$\sigma[x \mapsto V](z) = \begin{cases} V & z = x \\ \sigma(z) & \textit{otherwise} \end{cases}$$

When using the map σ as part of the translation, the intuition is that for all bound variables x in the term we are translatings, it should be that $\sigma(x) = x$. The map $\gamma : V \to V$ is defined similarly, and the purpose is to keep track of the binding of phantom-variables.

▶ **Definition 9.** *The interpretation $[\![- | - | -]\!] : \Lambda_a^S \times (V \to \Lambda) \times (V \to V) \to \Lambda$ is defined as*

$$[\![x | \sigma | \gamma]\!] = \sigma(x)$$

$$[\![s\,t | \sigma | \gamma]\!] = [\![s | \sigma | \gamma]\!]\,[\![t | \sigma | \gamma]\!]$$

$$[\![c\langle c \rangle.t | \sigma | \gamma]\!] = \lambda c.[\![t | \sigma[c \mapsto c] | \gamma]\!]$$

$$[\![c\langle x_1, \ldots, x_n \rangle.t | \sigma | \gamma]\!] = \lambda c.[\![t | \sigma[x_i \mapsto \sigma(x_i)\{c/\gamma(c)\}]_{i\in[n]} | \gamma]\!]$$

$$[\![u[x_1, \ldots, x_n \leftarrow t] | \sigma | \gamma]\!] = [\![u | \sigma[x_i \mapsto [\![t | \sigma | \gamma]\!]]_{i\in[n]} | \gamma]\!]$$

$$253 \qquad [\![u[e_1\langle \vec{w_1} \rangle, \ldots, e_n\langle \vec{w_n} \rangle \,|\, c\langle c \rangle \overline{[\Gamma]}] \,|\, \sigma \,|\, \gamma]\!] = [\![u\overline{[\Gamma]} \,|\, \sigma \,|\, \gamma[e_i \mapsto c]_{i\in[n]}]\!]$$

$$254 \qquad [\![u[e_1\langle \vec{w_1} \rangle, \ldots, e_n\langle \vec{w_n} \rangle \,|\, c\langle x_1, \ldots, x_m \rangle \overline{[\Gamma]}] \,|\, \sigma \,|\, \gamma]\!] = [\![u\overline{[\Gamma]} \,|\, \sigma' \,|\, \gamma[e_i \mapsto c]_{i\in[n]}]\!]$$

$$255 \qquad \qquad \qquad \qquad \qquad \qquad \qquad \qquad \qquad \textit{where } \sigma' = \sigma[x_i \mapsto \sigma(x_i)\{c/\gamma(c)\}]_{i\in[n]}$$

256
257

258 ▶ **Lemma 10.** *For $s, t \in \Lambda_a^S$, if $s \sim t$ then $[\![s]\!] = [\![t]\!]$*

▶ **Lemma 11.** *For a closed $t \in \Lambda_a^S$, where t has no distributor constructs and only variables are shared, and a closed $N \in \Lambda$. the following*

$$[\![(\! | N | \!)']\!] = N \qquad \qquad (\! | [\![t]\!] | \!)' = t \qquad \qquad \exists_{M\in\Lambda}.t = (\! | M | \!)'$$

259 3.3 Rewrite Rules

260 Both the spinal atomic λ-calculus and the atomic λ-calculus of [**?**] follow atomic reduction
261 steps, i.e. they apply on individual constructors. The biggest difference is that our calculus
262 is capable of duplicating not only the skeleton but also the spine. The rewrite rules in our
263 calculus make use of 3 operations, *substitution*, *book-keeping*, and *exorcism*.
264   The operation *substitution* $t\{s/x\}$ propagates through the term t, and replaces the free
265 occurences of the variable x with the term s. Moreover, if x occurs in the cover of a phantom-
266 variable $e\langle \vec{y} \cdot x \rangle$, then substitution replaces the x in the cover with $(s)_{fv}$, $e\langle \vec{y} \cdot (s)_{fv} \rangle$.
267   Although substitution performs some book-keeping on phantom-abstractions, we define
268 an explicit notion of book-keeping $\{\vec{y}/e\}_b$ that updates the variables stored in a free cover i.e.
269 for a term t, $e\langle \vec{x} \rangle \in (t)_{fc}$ then $e\langle \vec{y} \rangle \in (t\{\vec{y}/e\}_b)_{fc}$.
270   The last operation we introduce is called *exorcism* $\{c\langle \vec{x} \rangle\}_e$. We perform exorcisms on
271 phantom-abstractions to convert them to abstractions. Intuitively, this will be performed on
272 phantom-abstractions with phantom-variables bound to a distributor when said distributor is
273 eliminated. It converts phantom-abstractions to abstractions by introducing a sharing of the
274 phantom-variable that captures the variables in the cover, i.e. $c\langle \vec{x} \rangle.t\{c\langle \vec{x} \rangle\}_e = c\langle c \rangle.t[\vec{x} \leftarrow c]$.

275 ▶ **Proposition 12.** *Given $M \in \Lambda$ such that for all $v \in V$, $\gamma(v) \notin (M)_{fv}$ and $\sigma(x) = x$, the*
276 *translation $[\![u \,|\, \sigma \,|\, \gamma]\!]$ commutes with substitution $\{M/x\}$ in the following way*

$$[\![u\{t/x\} \,|\, \sigma \,|\, \gamma]\!] = [\![u \,|\, \sigma[x \mapsto [\![t \,|\, \sigma \,|\, \gamma]\!]] \,|\, \gamma]\!]$$

277 ▶ **Proposition 13.** *Book-keeping commutes with the translation in the following way*
278 *if $c\langle y_1, \ldots, y_m \rangle. \in (u)_{fc}$ such that $\{x_1, \ldots, x_n\} \subset \{y_1, \ldots, y_m\}$*
279 *and for those $z \in \{y_1, \ldots, y_m\}/\{x_1, \ldots, x_n\}$, $\gamma(c) \notin (\sigma(z))_{fv}$*
280 *or if simply $\{x_1, \ldots, x_n\} \cap (u)_{fv} = \{\}$*

$$[\![u\{x_1, \ldots, x_n/c\}_b \,|\, \sigma \,|\, \gamma]\!] = [\![u \,|\, \sigma \,|\, \gamma]\!]$$

281 ▶ **Proposition 14.** *Exorcisms commute with the translation in the following way*
282 *if $c\langle x_1, \ldots, x_n \rangle. \in (u)_{fc}$ or $\{x_1, \ldots, x_n\} \cap (u)_{fv} = \{\}$*

$$[\![u\{c\langle x_1, \ldots, x_n \rangle\}_e \,|\, \sigma \,|\, \gamma]\!] = [\![u \,|\, \sigma[x_i \mapsto c]_{i\in[n]} \,|\, \gamma]\!]$$

283   Using these operations, we define the rewrite rules that allow for spinal duplication. Firstly
284 we have beta reduction $(\leadsto_\beta)$, which requires an abstraction and not a phantom-abstraction.

$$285 \qquad (x\langle x \rangle.t)\, s \leadsto_\beta t\{s/x\} \qquad \qquad \qquad \qquad \qquad \qquad \qquad \qquad \qquad (\beta)$$

However, its effect is very different: here β-reduction is a linear operation, since the bound variable x occurs exactly once in the body t. Any duplication of the term t in the atomic lambda-calculus proceeds via the sharing reductions, which we define next. The first set of sharing reduction rules move closures towards the outside of a term. Most of these rewrite rules only change the typing derivations in the way that subderivations are composed, with the exception of moving a closure out of scope of a distributor.

$$s[\Gamma]\,t \rightsquigarrow_L (s\,t)[\Gamma] \qquad\qquad (l_1)$$

$$s\,t[\Gamma] \rightsquigarrow_L (s\,t)[\Gamma] \qquad\qquad (l_2)$$

$$d\langle\,\vec{x}\,\rangle.t[\Gamma] \rightsquigarrow_L (d\langle\,\vec{x}\,\rangle.t)[\Gamma] \text{ if } \{\vec{x}\} \cap (t)_{fv} = \{\vec{x}\} \qquad\qquad (l_3)$$

$$u[x_1,\ldots,x_n \leftarrow t[\Gamma]] \rightsquigarrow_L u[x_1,\ldots,x_n \leftarrow t][\Gamma] \qquad\qquad (l_4)$$

For the case of lifting a closure outside a distributor, we use a notation $\|\,[\Gamma]\,\|$ to identify the variables captured by a closure, i.e.$\|\,[\vec{x} \leftarrow t]\,\| = \{\vec{x}\}$ and $\|\,[e_1\langle\,\vec{x_1}\,\rangle,\ldots,e_n\langle\,\vec{x_x}\,\rangle\,|\,c\langle\,c\,\rangle\,\overline{[\Gamma]}]\,\| = \{\vec{x_1},\ldots,\vec{x_n}\}$. Then let $\{\vec{z}\} = \|\,[\Gamma]\,\|$ in the following rewrite rule, that can only occur if $\{\vec{x}\} \cap ([\Gamma])_{fv} = \{\}$.

$$u[e_1\langle\,\vec{w_1}\,\rangle\ldots e_n\langle\,\vec{w_n}\,\rangle\,|\,c\langle\,\vec{x}\,\rangle\,\overline{[\Gamma]}[\Gamma]]$$
$$\rightsquigarrow_L u\{(\vec{w_i}/\vec{z})/e_i\}_{b_i\in[n]}[e_1\langle\,\vec{w_1}/\vec{z}\,\rangle\ldots e_n\langle\,\vec{w_n}/\vec{z}\,\rangle\,|\,c\langle\,\vec{x}\,\rangle\,\overline{[\Gamma]}][\Gamma] \qquad\qquad (l_5)$$

The proof rewrite rule corresponding with the rewrite rule l_5 can be broken down into two parts. The first part is readjusting how the derivations compose as shown below.

The second part of the rewrite rule justifies the need for the book-keeping operation. In the rewrite below, let A be the type of a variable z where $z \in \vec{z}$. After lifting, we want to remove the variable from the cover as to ensure correctness since the variables in the cover denote the variables captured by the environment. Book-keeping allows us to remove these variables simultaneously.

The lifting rules (l_i) are justified by the need to lift closures out of the distributor, as opposed to duplicating them. The second set of rewrite rules, consecutive sharings are compounded and unary sharings are applied as substitutions.

$$u[w_1, \ldots, w_m \leftarrow y_i][y_1, \ldots, y_n \leftarrow t] \rightsquigarrow_C u[y_1, \ldots, y_{i-1}, w_1, \ldots, w_m, y_{i+1}, \ldots, y_n \leftarrow t] \qquad (c_1)$$

$$u[x \leftarrow t] \rightsquigarrow_C u\{t/x\} \qquad (c_2)$$

The atomic steps for duplicating are given in the third and final set of rewrite rules. The first being the atomic duplication step of an application, which is the same rule used in [**?**]. The proof rewrite steps for each rule are also provided. For simplicity, we only show the binary case for each rule.

$$u[x_1 \ldots x_n \leftarrow s\,t] \rightsquigarrow_D u\{z_1\,y_1/x_1\} \ldots \{z_n\,y_n/x_n\}[z_1 \ldots z_n \leftarrow s][y_1 \ldots y_n \leftarrow t] \qquad (d_1)$$

$$\cfrac{(A \to B) \land A}{\cfrac{B}{B \land B}\,\triangle}\,@ \qquad \cfrac{\cfrac{(A \to B)}{(A \to B) \land (A \to B)}\,\triangle \land \cfrac{B}{B \land B}\,\triangle}{\cfrac{(A \to B) \land A}{B}\,@ \land \cfrac{(A \to B) \land A}{B}\,@}$$

$$\begin{aligned} u[x_1, \ldots, x_n \leftarrow c\langle \vec{y}\rangle.t] \rightsquigarrow_D & \\ u\{e_i\langle w_1^i\rangle.w_1^i/x_i\}_{1 \le i \le n}[e_1\langle w_1^1\rangle \ldots e_n\langle w_1^n\rangle \,|\, c\langle \vec{y}\rangle\,[w_1^1, \ldots, w_1^n \leftarrow t]] & \end{aligned} \qquad (d_2)$$

$$u[e_1\langle \vec{w_1}\rangle \ldots e_n\langle \vec{w_n}\rangle \,|\, c\langle c\rangle\,[\vec{w_1}, \ldots, \vec{w_n} \leftarrow c]] \rightsquigarrow_D u\{e_1\langle \vec{w_1}\rangle\}_e \ldots \{e_n\langle \vec{w_n}\rangle\}_e \qquad (d_3)$$

$$\cfrac{A \to \cfrac{A}{A \land A}\,\triangle}{(A \to A) \land (A \to A)}\,d \qquad \cfrac{}{A \to A}\,\lambda \land \cfrac{}{A \to A}\,\lambda$$

▶ **Proposition 15.** *If $s \rightsquigarrow_{L,C,D} t$ and $s : C$, then $t : C$*

▶ **Lemma 16** (Sharing reduction preserves denotation)**.** *If $s \rightsquigarrow_{L,D,C} t$ then $[\![\,s\,|\,\sigma\,|\,\gamma\,]\!] = [\![\,t\,|\,\sigma\,|\,\gamma\,]\!]$*

4   Strong Normalisation of Sharing Reductions

In order to show our calculus is strongly normalising, we first show that the sharing reduction rules are strongly normalising. To do this, we make use of an 'intermediate calculus' called the weakening calculus. Following the approaches of [**?, ?**], we indite a measure on terms based on its connection with the weakening calculus. We show that this measure strictly decreases as sharing reduction progresses.

▶ **Definition 17.** *The w-terms and the weakening calculus (Λ_w) are*

$$T, U, V \quad ::= \quad x \quad | \quad \lambda x.T^* \quad | \quad U\,V \quad | \quad T[\leftarrow U] \quad | \quad \bullet \qquad \textit{(*) where } x \in (T)_{fv}$$

The terms are variable, abstraction, application, weakening, and a bullet. In the weakening $T[\leftarrow U]$, the subterm U is *weakened*. The interpretation of atomic terms to weakening terms $[\![-\,|\,-\,|\,-]\!]_w$ can be seen as an extension of the translation into the λ-calculus (Definition 9)

340 ▶ **Definition 18.** *The interpretation* $[\![-\,|\,-\,|\,-]\!]_{\mathcal{W}} : \Lambda_a^S \times (V \to \Lambda_{\mathcal{W}}) \times (V \to V) \to \Lambda_{\mathcal{W}}$ *with*
341 *maps* $\sigma : V \to \Lambda_{\mathcal{W}}$ *and* $\gamma : V \to V$ *is defined as an extension of the translation in (Definition*
342 *9) with the following additional special cases.*

343
$$[\![\,u[\leftarrow t]\,|\,\sigma\,|\,\gamma\,]\!]_{\mathcal{W}} = [\![\,u\,|\,\sigma\,|\,\gamma\,]\!]_{\mathcal{W}}[\leftarrow [\![\,t\,|\,\sigma\,|\,\gamma\,]\!]_{\mathcal{W}}]$$

344
$$[\![\,u[\,|\,c\langle c\rangle\,\overline{\lceil\Gamma\rceil}\,]\,|\,\sigma\,|\,\gamma\,]\!]_{\mathcal{W}} = [\![\,u\overline{\lceil\Gamma\rceil}\,|\,\sigma[c\mapsto \bullet]\,|\,\gamma\,]\!]_{\mathcal{W}}$$

345
$$[\![\,u[\,|\,c\langle x_1,\ldots,x_n\rangle\,\overline{\lceil\Gamma\rceil}\,]\,|\,\sigma\,|\,\gamma\,]\!]_{\mathcal{W}} = [\![\,u\overline{\lceil\Gamma\rceil}\,|\,\sigma'\,|\,\gamma\,]\!]_{\mathcal{W}}$$

346
$$where\ \sigma'(z) = \begin{cases} \sigma(z)\{\bullet/\gamma(c)\} & z \in \{x_1,\ldots,x_n\} \\ \sigma(z) & otherwise \end{cases}$$

347
348

349 We also have translations of the weakening calculus to and from the lambda calculus. Both
350 of these translations have been provided in [**?**]. The interpretation $\lfloor - \rfloor$ from weakening terms
351 to λ-terms discards all weakenings. The interpretation $(\!|-|\!)^{\mathcal{W}} : \Lambda \to \Lambda_{\mathcal{W}}$ is defined below.

352 ▶ **Definition 19.** *The interpretation* $M \in \Lambda$, $(\!|-|\!)^{\mathcal{W}} : \Lambda \to \Lambda_{\mathcal{W}}$ *is defined by*

353
$$(\!|\,x\,|\!)^{\mathcal{W}} = x$$

354
$$(\!|\,M\,N\,|\!)^{\mathcal{W}} = (\!|\,M\,|\!)^{\mathcal{W}}\,(\!|\,N\,|\!)^{\mathcal{W}}$$

355
356
$$(\!|\,\lambda x.N\,|\!)^{\mathcal{W}} = \begin{cases} \lambda x.(\!|\,N\,|\!)^{\mathcal{W}} & if\ x \in (N)_{fv} \\ \lambda x.(\!|\,N\,|\!)^{\mathcal{W}}[\leftarrow x] & otherwise \end{cases}$$

357 The following equalities can be observed, where $\sigma^\Lambda(z) = \lfloor \sigma^{\mathcal{W}}(z) \rfloor$.

358 ▶ **Proposition 20.** *For* $N \in \Lambda$ *and* $t \in \Lambda_a^S$ *the following properties hold*

359
$$\lfloor [\![\,t\,|\,\sigma^{\mathcal{W}}\,|\,\gamma\,]\!]_{\mathcal{W}} \rfloor = [\![\,t\,|\,\sigma^\Lambda\,|\,\gamma\,]\!] \qquad\qquad [\![\,(\!|\,N\,|\!)\,]\!]^{\mathcal{W}} = (\!|\,N\,|\!)^{\mathcal{W}} \qquad\qquad \lfloor (\!|\,N\,|\!)^{\mathcal{W}} \rfloor = N$$

360 Here we can take advantage that preservation of strong normalisation has been proven for
361 this weakening calculus already in [**?**], providing the proof for Proposition 22. Additionally,
362 similar ideas and results can be found elsewhere, i.e. with memory in [**?**], the λ-I calculus in
363 [**?**], the λ-void calculus [**?**], and the weakening $\lambda\mu$-calculus [**?**].

364 ▶ **Definition 21.** *In the weakening calculus,* β-*reduction is defined as follows, where* $\overline{\lceil\Gamma\rceil}$ *are*
365 *weakening constructs.*

366
$$((\lambda x.T)\overline{\lceil\Gamma\rceil})\,U \to_\beta T\{U/x\}\overline{\lceil\Gamma\rceil} \qquad\qquad\qquad (w_\beta)$$

367 ▶ **Proposition 22.** *If* $N \in \Lambda$ *is strongly normalising, then so is* $(\!|\,N\,|\!)^{\mathcal{W}}$

368 When translating from the spinal atomic λ-calculus to the weakening calculus, weakenings
369 are maintained whilst sharings are interpreted through duplication via substitution. Thus the
370 reduction rules in the weakening calculus cover the spinal reductions for nullary distributors
371 and weakenings.

372 ▶ **Definition 23.** *The weakening reductions* $(\to_{\mathcal{W}})$ *proceeds as follows.*

373
$$\lambda x.T[\leftarrow U] \to_{\mathcal{W}} (\lambda x.T)[\leftarrow U] \quad if\ x \notin (U)_{fv} \qquad\qquad (w_1)$$

374
$$U[\leftarrow T]\,V \to_{\mathcal{W}} (U\,V)[\leftarrow T] \qquad\qquad\qquad\qquad\quad (w_2)$$

375
$$U\,V[\leftarrow T] \to_{\mathcal{W}} (U\,V)[\leftarrow T] \qquad\qquad\qquad\qquad\quad (w_3)$$

$$T[\leftarrow U[\leftarrow V]] \rightarrow_w T[\leftarrow U][\leftarrow V] \tag{w_4}$$

$$T[\leftarrow \lambda x.U] \rightarrow_w T[\leftarrow U\{\bullet/x\}] \tag{w_5}$$

$$T[\leftarrow U\,V] \rightarrow_w T[\leftarrow U][\leftarrow V] \tag{w_6}$$

$$T[\leftarrow \bullet] \rightarrow_w T \tag{w_7}$$

$$T[\leftarrow U] \rightarrow_w T \quad \textit{if } U \textit{ is a subterm of } T \tag{w_8}$$

It is easy to see that these rules correspond to special cases of the sharing reduction rules for Λ_a^S. Lifting a closure relates (w_1) and (l_3), (w_2) and (l_1), (w_3) and (l_2), (w_4) and (l_4), (w_5) and (d_2), and duplicating a term relates (w_6) and (d_1), and (w_7) and (d_3). It is not so obvious to see what the case (w_8) corresponds to. If U is a subterm of T, then in the corresponding Λ_a^S-term this term would be shared and one of the copies would be in a weakening. Thus this reduction relates to the case (c_1), where we remove the weakening. We demonstrate by considering $t[\leftarrow y][\vec{x} \cdot y \cdot \vec{z} \leftarrow u] \rightsquigarrow_C t[\vec{x} \cdot \vec{z} \leftarrow u]$. On the left hand side, the corresponding weakening-term (obtained by $(\!|-|\!)^w$) would have the weakening $[\leftarrow U]$ where $U = (\!| u |\!)^w$. This is because U is substituted into $[\leftarrow y]$, but on the right hand side this would be gone. This situation can only occur if there are other copies of U substituted into the term. This corresponds to if only the corresponding (c_1) reduction rule can occur. This resemblace is confirmed by the following Lemmas.

▶ **Lemma 24.** *If $t \rightsquigarrow_\beta u$ then $[\![t]\!]^w \rightarrow_\beta^+ [\![u]\!]^w$*

▶ **Lemma 25.** *If $t \rightsquigarrow_{(C,D,L)} u$ and for any $x \in (t)_{bv} \cup (t)_{fp}$ and for all z, $x \notin (\sigma(z))_{fv}$.*

$$[\![t \,|\, \sigma \,|\, \gamma]\!]_w \rightarrow_w^* [\![u \,|\, \sigma \,|\, \gamma]\!]_w$$

We now define the components that we use for our measure on spinal atomic λ-terms that we will use to prove strong normalisation of sharing reductions. The *height* of a term is intuitively a multiset of integers that record the distance of each sharing. The distance is measured by the number of constructors from the sharing node to the root of the term in its graphical notation. The height is defined on terms as $\mathcal{H}^i(-)$, where i is an integer. We say $\mathcal{H}(t)$ for $\mathcal{H}^1(t)$. We use $\uplus$ to denote the disjoint union of two multisets. We denote $\mathcal{H}^i([\Gamma_1]) \uplus \cdots \uplus \mathcal{H}^i([\Gamma_n])$ as $\mathcal{H}^i(\overline{[\Gamma]})$ for the environment $\overline{[\Gamma]} = [\Gamma_1], \ldots, [\Gamma_n]$.

▶ **Definition 26** (Sharing Height). *The sharing height $\mathcal{H}^i(t)$ of a term t is given by*

$$\mathcal{H}^i(x) = \{\}$$

$$\mathcal{H}^i(s\,t) = \mathcal{H}^{i+1}(s) \uplus \mathcal{H}^{i+1}(t)$$

$$\mathcal{H}^i(c\langle \vec{x} \rangle.t) = \mathcal{H}^{i+1}(t)$$

$$\mathcal{H}^i(t[\Gamma]) = \mathcal{H}^i(t) \uplus \mathcal{H}^i([\Gamma]) \uplus \{i^1\}$$

$$\mathcal{H}^i([x_1, \ldots, x_n \leftarrow t]) = \mathcal{H}^{i+1}(t)$$

$$\mathcal{H}^i([\overrightarrow{e\langle \vec{w} \rangle} \,|\, c\langle \vec{x} \rangle \, \overline{[\Gamma]}]) = \mathcal{H}^{i+1}(\overline{[\Gamma]}) \uplus \{(i+1)^n\} \text{ where } n \text{ is the number of closures in } \overline{[\Gamma]}$$

This measure then strictly decreases for the rewrite rules l_1, l_2, l_3, l_4 and l_5.

▶ **Lemma 27.** *If $t \rightsquigarrow_{(L)} u$ then $\mathcal{H}^i(t) > \mathcal{H}^i(u)$*

The other measure we consider is the *weight* of a term. Intuitively this quantifies the remaining duplications, which are performed with $\rightsquigarrow_D$ reductions. Calculating the weight of a term requires an auxiliary function from variables to integers. This function is defined by assigning integer weights to the variables of a term. This auxiliary function is defined on

terms $\mathcal{V}^i(-)$, where i is an integer. To measure variables independently of binders is vital. It allows to measure distributors, which duplicate λ's but not the bound variable. Also, only bound variables for abstractions are measured since variables bound by sharings are substituted in the interpretation.

▶ **Definition 28** (Variable Weights). *The function $\mathcal{V}^i(t)$ returns a function that assigns integer weights to the free variables of t. It is defined by the following*

$$\mathcal{V}^i(x) = \{x \mapsto i\}$$

$$\mathcal{V}^i(s\,t) = \mathcal{V}^i(s) \uplus \mathcal{V}^i(t)$$

$$\mathcal{V}^i(c\langle c \rangle.t) = \mathcal{V}^i(t)/\{c\}$$

$$\mathcal{V}^i(c\langle \vec{x} \rangle.t) = \mathcal{V}^i(t) \uplus \{c \mapsto i\}$$

$$\mathcal{V}^i(t[\leftarrow s]) = \mathcal{V}^i(t) \uplus \mathcal{V}^1(s)$$

$$\mathcal{V}^i(t[x_1,\ldots,x_n \leftarrow s]) = \mathcal{V}^i(t)/\{x_1,\ldots,x_n\} \uplus \mathcal{V}^{f(x_1)+\cdots+f(x_n)}(s) \ where \ f = \mathcal{V}^i(t)$$

$$\mathcal{V}^i(t[e_1\langle \vec{w_1} \rangle \ldots e_n\langle \vec{w_n} \rangle \,|\, c\langle c \rangle \,\overline{[\Gamma]}]) = \mathcal{V}^i(t\overline{[\Gamma]})/\{c, e_1,\ldots,e_n\}$$

$$\mathcal{V}^i(t[e_1\langle \vec{w_1} \rangle \ldots e_n\langle \vec{w_n} \rangle \,|\, c\langle \vec{x} \rangle \,\overline{[\Gamma]}]) = \mathcal{V}^i(t\overline{[\Gamma]})/\{e_1,\ldots,e_n\} \uplus \{c \mapsto i\}$$

The weight of a term can then be defined via the use of this auxiliary function. The auxiliary function is used when calculating the weight of a sharing, where the sharing weight of the variables bound by the sharing play a significant role in calculating the weight of the shared term. In the case of a weakening, we assign an initial weight of 1 to indicate that the constructor is not duplicated by appears at least once in the weakening calculus. Again we say $\mathcal{W}(t) = \mathcal{W}^1(t)$.

▶ **Definition 29** (Sharing Weight). *The sharing weight $\mathcal{W}^i(t)$ of a term t is a multiset of integers computed by the function defined below*

$$\mathcal{W}^i(x) = \{\}$$

$$\mathcal{W}^i(s\,t) = \mathcal{W}^i(s) \uplus \mathcal{W}^i(t) \uplus \{i\}$$

$$\mathcal{W}^i(c\langle c \rangle.t) = \mathcal{W}^i(t) \uplus \{i\} \uplus \{\mathcal{V}^i(t)(c)\}$$

$$\mathcal{W}^i(c\langle \vec{x} \rangle.t) = \mathcal{W}^i(t) \uplus \{i\}$$

$$\mathcal{W}^i(t[\leftarrow s]) = \mathcal{W}^i(t) \uplus \mathcal{W}^1(s)$$

$$\mathcal{W}^i(t[x_1,\ldots,x_n \leftarrow s]) = \mathcal{W}^i(t) \uplus \mathcal{W}^{f(x_1)+\cdots+f(x_n)}(s) \ where \ f = \mathcal{V}^i(t)$$

$$\mathcal{W}^i(t[e_1\langle \vec{w_1} \rangle \ldots e_n\langle \vec{w_n} \rangle \,|\, c\langle c \rangle \,\overline{[\Gamma]}]) = \mathcal{W}^i(t\overline{[\Gamma]}) \uplus \{\mathcal{V}^i(t\overline{[\Gamma]})(c)\}$$

$$\mathcal{W}^i(t[e_1\langle \vec{w_1} \rangle \ldots e_n\langle \vec{w_n} \rangle \,|\, c\langle \vec{x} \rangle \,\overline{[\Gamma]}]) = \mathcal{W}^i(t\overline{[\Gamma]})$$

We show that this measure then strictly decreases on the rewrite rules d_1, d_2, d_3 and is unaffected by all the other sharing reduction rules.

▶ **Lemma 30.** *If $t \rightsquigarrow_D u$ then $\mathcal{W}^i(t) > \mathcal{W}^i(u)$*

▶ **Lemma 31.** *If $t \rightsquigarrow_{(L,C)} u$ then $\mathcal{W}^i(t) = \mathcal{W}^i(u)$*

The last measure we consider is the number of closures in the term, where is can be easily observed that the rewrite rules c_1 and c_2 strictly decrease this measure, and that the $\rightsquigarrow_L$ rules do not alter the number of closures. We then use this along with height and weight to define a *sharing measure* on terms.

▶ **Definition 32.** *The* sharing measure *of a Λ_a^S-term t is a triple $(\mathcal{W}(t), C, \mathcal{H}(t))$ where C is the number of closures in t. We can compare two different sharing measures by considering the lexicographical preferences according to weight > number of closures > height.*

▶ **Theorem 33.** *Sharing reduction $\leadsto_{(D,L,C)}$ is strongly normalising*

Proof. From Lemma 30, Lemma 31, and Lemma 27, it follows that the sharing measure of a term is strictly decreasing under $\leadsto_{(D,L,C)}$, proving the statement. ◀

Now that we have proven the sharing reductions are strongly normalising, we can prove that they are confluent for closed terms.

▶ **Theorem 34.** *The sharing reduction relation $\leadsto_{(R,D,L,C)}$ is confluent*

Proof. Lemma 10 tells us that the preservation is preserved under reduction i.e. for $s \leadsto_{(R,D,L,C)} t$, $[\![s]\!] = [\![t]\!]$. Therefore given $t \leadsto^*_{(R,D,L,C)} s_1$ and $t \leadsto^*_{(R,D,L,C)} s_2$, $[\![t]\!] = [\![s_1]\!] = [\![s_2]\!]$. Since we know that sharing reductions are strongly normalising, we know there exists terms u_1 and u_2 in sharing normal form such that $s_1 \leadsto^*_{(R,D,L,C)} u_1$ and $s_2 \leadsto^*_{(R,D,L,C)} u_2$. Lemma 11 tells us that terms in closed terms in sharing normal form are in correspondence with their denotations i.e. $(\![[\![t]\!]]\!)' = t$. Since by Lemma 10 we know $[\![u_1]\!] = [\![s_1]\!] = [\![s_2]\!] = [\![u_2]\!]$, and by Lemma 11 $(\![[\![u_1]\!]]\!)' = u_1$ and $(\![[\![u_2]\!]]\!)' = u_2$, we can conclude $u_1 = u_2$. Hence, we prove confluence. ◀

5 Preservation of Strong Normalisation

Here we show how Λ_a^S preserves strong normalisation with respect to the λ-calculus. Recall that by Proposition 20 that for all $N \in \Lambda$, $[\![(\![N]\!)]\!]^\mathcal{W} = (\![N]\!)^\mathcal{W}$, and that Proposition 22 states if a term $N \in \Lambda$ is strongly normalising then so is $(\![N]\!)^\mathcal{W}$. Observe that the statement 'if term M has an infinite reduction sequence then term N has an infinite reduction sequence' is equivalent to 'if term N is strongly normalising then term M is strongly normalising' by contraposition. Therefore, given a strongly normalising term $N \in \Lambda$, we know that its corresponding weakening term is also strongly normalising. Furthermore, since $[\![(\![N]\!)]\!]^\mathcal{W} = (\![N]\!)^\mathcal{W}$, we know that $[\![(\![N]\!)]\!]^\mathcal{W}$ is also strongly normalising.
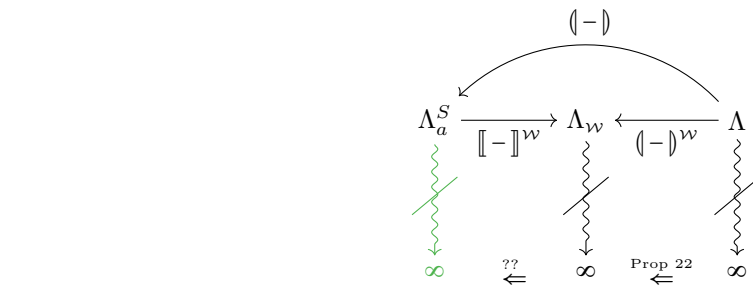
We prove that the spinal atomic λ-calculus preserves strong normalisation with the following.

▶ **Lemma 35.** *For $t \in \Lambda_a^S$ has an infinite reduction path, then $[\![t]\!]^\mathcal{W}$ also has an infinite reduction path.*

Proof. Due to Theorem 34, we know that the infinite reduction path contains an infinite β-reduction. This means in the reduction sequence, between each β-reduction, there are finite many $\leadsto_{(D,L,C)}$ reduction steps. Lemma 25 says each $\leadsto_{(D,L,C)}$ step in Λ_a^S corresponds

to zero or more weakening reductions ($\leadsto^*_\mathcal{W}$). Lemma 24 says that each beta reduction in Λ^S_a corresponds to one or more β-steps in $\Lambda_\mathcal{W}$. Therefore, it is inevitable that $[\![\,t\,]\!]^\mathcal{W}$ also has an infinite reduction path. ◄

▶ **Theorem 36.** *If $N \in \Lambda$ is strongly normalising, then so is $(\!| N |\!)$.*

Proof. For a given $N \in \Lambda$ that is strongly normalising, we know by Lemma 22 that $(\!| N |\!)^\mathcal{W}$ is strongly normalising. Then $[\![\,(\!| N |\!)\,]\!]^\mathcal{W}$ is strongly normalising, since Proposition 20 states that $(\!| N |\!)^\mathcal{W} = [\![\,(\!| N |\!)\,]\!]^\mathcal{W}$. Then by Lemma 35, which states that if $[\![\,t\,]\!]^\mathcal{W}$ is strongly normalising, then t is strongly normalising, proves that $(\!| N |\!)$ is strongly normalising. ◄

6 Conclusion and Further Remarks

A   The Spinal Atomic λ-Calculus

A.1   Compilation and Readback

In this section we provide the proof for Proposition 11: For $s, t \in \Lambda_a^S$, if $s \sim t$ then $[\![\, s \,]\!] = [\![\, t \,]\!]$.

Proof. Let us consider the cases.

$t[\Gamma_1][\Gamma_2] \sim t[\Gamma_2][\Gamma_1]$

Consider $[\![\, t[\Gamma_1][\Gamma_2] \,|\, \sigma \,|\, \gamma \,]\!] = [\![\, t[\Gamma_1] \,|\, \sigma' \,|\, \gamma' \,]\!] = [\![\, t \,|\, \sigma'' \,|\, \gamma'' \,]\!]$. Since due to conditions any variable $x \in \|\, [\Gamma_2] \,\|$ cannot occur in $[\Gamma_1]$, for all subterms s located in $[\Gamma_1]$, $[\![\, s \,|\, \sigma' \,|\, \gamma' \,]\!] = [\![\, s \,|\, \sigma \,|\, \gamma \,]\!]$. Therefore $[\![\, t \,|\, \sigma'' \,|\, \gamma'' \,]\!] = [\![\, t[\Gamma_2] \,|\, \sigma''' \,|\, \gamma''' \,]\!] = [\![\, t[\Gamma_2][\Gamma_1] \,|\, \sigma \,|\, \gamma \,]\!]$.

The remaining cases discuss permutations of variables in sharings and phantom-abstractions. In both these cases, we overwrite σ for the cases of the variables in said sharing or phantom-abstractions. The order in which they appear do not influence the translation since we do this for all variables regardless.   ◄

We also provide the proof for Lemma 11: For a closed $t \in \Lambda_a^S$, where t has no distributor constructs and only variables are shared, and a closed $N \in \Lambda$. the following

$$[\![\, (\!|\, N \,|\!)' \,]\!] = N \qquad\qquad (\!|\, [\![\, t \,]\!] \,|\!)' = t \qquad\qquad \exists_{M \in \Lambda}. t = (\!|\, M \,|\!)'$$

Proof. We prove $[\![\, (\!|\, N \,|\!)' \,]\!] = N$ by induction on N

Base Case: Variable

$[\![\, (\!|\, x \,|\!)' \,]\!] = [\![\, x \,]\!] = x$

Inductive Case: Application

$[\![\, (\!|\, M \, N \,|\!)' \,]\!] = [\![\, (\!|\, M \,|\!)' \,]\!] \, [\![\, (\!|\, N \,|\!)' \,]\!] = M \, N$

Inductive Case: Abstraction

$[\![\, (\!|\, \lambda x. M \,|\!)' \,]\!]$

    Case: $|M|_x = 1$

    $= \lambda x. [\![\, (\!|\, M \,|\!)' \,]\!] = \lambda x. M$

    Case: $|M|_x = n$

    $= \lambda x. [\![\, (\!|\, M \frac{n}{x} \,|\!)'[x_1, \ldots, x_n \leftarrow x] \,]\!] = \lambda x. [\![\, (\!|\, M \frac{n}{x} \,|\!)' \,|\, \sigma \,|\, I \,]\!] = \lambda x. [\![\, (\!|\, M \frac{n}{x} \,|\!)' \,]\!] \{x/x_i\}_{1 \le i \le n}$

    $\overset{\text{I.H.}}{=} \lambda x. M \frac{n}{x} \{x/x_i\}_{1 \le i \le n} = \lambda x. M$

We prove $(\!|\, [\![\, t \,]\!] \,|\!)' = t$ by induction on t

Base Case: Variable

$(\!|\, [\![\, x \,]\!] \,|\!)' = (\!|\, x \,|\!)' = x$

Inductive Case: Application

$(\!|\, [\![\, s \, t \,]\!] \,|\!)' = (\!|\, [\![\, s \,]\!] \,|\!)' \, (\!|\, [\![\, t \,]\!] \,|\!)' \overset{\text{I.H.}}{=} s \, t$

Inductive Case: Abstraction

539   Case: $(\![\,[\![\,x\langle\,x\,\rangle.t\,]\!]\,]\!)' = x\langle\,x\,\rangle.(\![\,[\![\,t\,]\!]\,]\!)' \overset{\text{I.H.}}{=} x\langle\,x\,\rangle.t$

540

541   Case: $(\![\,[\![\,x\langle\,x\,\rangle.t[x_1,\dots,x_n \leftarrow x]\,]\!]\,]\!)' = (\!|\,\lambda x.[\![\,t\,|\,\sigma\,|\,I\,]\!]\,|\!)'$

542     $= (\!|\,\lambda x.[\![\,t\,]\!]\{x/x_i\}_{1\le i\le n}\,|\!)' = x\langle\,x\,\rangle.(\![\,[\![\,t\,]\!]\,]\!)'[x_1,\dots,x_n \leftarrow x]$

543     $\overset{\text{I.H.}}{=} x\langle\,x\,\rangle.t[x_1,\dots,x_n \leftarrow x]$

544

545 The proof for $\exists_{M\in\Lambda}.t = (\!|\,M\,|\!)'$ is the same as in [**?**].   ◀

546 A.2   Rewrite Rules

547 In this section we provide the proof for Proposition 37: Given $M \in \Lambda$ such that for all $v \in V$,

548 $\gamma(v) \notin (M)_{fv}$ and $\sigma(x) = x$, the translation $[\![\,u\,|\,\sigma\,|\,\gamma\,]\!]$ commutes with substitution $\{M/x\}$ in

549 the following way

$$[\![\,u\{t/x\}\,|\,\sigma\,|\,\gamma\,]\!] = [\![\,u\,|\,\sigma[x \mapsto [\![\,t\,|\,\sigma\,|\,\gamma\,]\!]]\,|\,\gamma\,]\!]$$

550 **Proof.** We prove this by induction on u

551

552 Base Case: Variable

553 $[\![\,x\{t/x\}\,|\,\sigma\,|\,\gamma\,]\!] = [\![\,t\,|\,\sigma\,|\,\gamma\,]\!] = [\![\,x\,|\,\sigma'\,|\,\gamma\,]\!]$

554

555 $[\![\,y\,|\,\sigma\,|\,\gamma\,]\!] = \sigma(y) = \sigma'(y) = [\![\,y\,|\,\sigma'\,|\,\gamma\,]\!]$

556

557 Inductive Case: Application

558 $[\![\,u\,s\{t/x\}\,|\,\sigma\,|\,\gamma\,]\!] = [\![\,u\{t/x\}\,|\,\sigma\,|\,\gamma\,]\!]\,[\![\,s\{t/x\}\,|\,\sigma\,|\,\gamma\,]\!] \overset{\text{I.H.}}{=} [\![\,u\,|\,\sigma'\,|\,\gamma\,]\!]\,[\![\,s\,|\,\sigma'\,|\,\gamma\,]\!] = [\![\,u\,s\,|\,\sigma'\,|\,\gamma\,]\!]$

559

560 Inductive Case: Abstraction

561 $[\![\,(c\langle\,c\,\rangle.s)\{t/x\}\,|\,\sigma\,|\,\gamma\,]\!] = \lambda c.[\![\,s\{t/x\}\,|\,\sigma\,|\,\gamma\,]\!] \overset{\text{I.H.}}{=} \lambda c.[\![\,s\,|\,\sigma'\,|\,\gamma\,]\!] = [\![\,c\langle\,c\,\rangle.s\,|\,\sigma'\,|\,\gamma\,]\!]$

562

563 Inductive Case: Phantom-Abstraction

564 $[\![\,(c\langle\,x_1,\dots,x_n\,\rangle.s)\{t/x\}\,|\,\sigma\,|\,\gamma\,]\!]$

565   Case: $x \in \{x_1,\dots,x_n\}$

566     $= [\![\,(c\langle\,x_1,\dots,x_n,x\,\rangle.s)\{t/x\}\,|\,\sigma\,|\,\gamma\,]\!] = [\![\,c\langle\,x_1,\dots,x_n,y_1,\dots,y_m\,\rangle.s\{t/x\}\,|\,\sigma\,|\,\gamma\,]\!]$

567   where $\{y_1,\dots,y_m\} = (t)_{fv}$

568     $= \lambda c.[\![\,s\{t/x\}\,|\,\sigma''\,|\,\gamma\,]\!] \overset{\text{I.H.}}{=} \lambda c.[\![\,s\,|\,\sigma_1'''\,|\,\gamma\,]\!] = \lambda c.[\![\,s\,|\,\sigma_2'''\,|\,\gamma\,]\!] = [\![\,c\langle\,x_1,\dots,x_n,x\,\rangle.s\,|\,\sigma'\,|\,\gamma\,]\!]$

569   where $\sigma''(z) = \begin{cases} \sigma(z)\{c/\gamma(c)\} & \text{if } z \in \{x_1,\dots,x_n,y_1,\dots,y_m\} \\ \sigma(z) & \text{otherwise} \end{cases}$

570     $\sigma_1''' = \sigma''[x \mapsto [\![\,t\,|\,\sigma''\,|\,\gamma\,]\!]]$

571     $\sigma_2'''(z) = \begin{cases} [\![\,t\,|\,\sigma''\,|\,\gamma\,]\!]\{c/\gamma(c)\} & z = x \\ \sigma(z)\{c/\gamma(c)\} & z \in \{x_1,\dots,x_n\} \\ \sigma(z) & \text{otherwise} \end{cases}$

572

573   Case: $x \notin \{x_1,\dots,x_n\}$

574     $= [\![\,c\langle\,x_1,\dots,x_n\,\rangle.s\{t/x\}\,|\,\sigma\,|\,\gamma\,]\!] = \lambda c.[\![\,s\{t/x\}\,|\,\sigma''\,|\,\gamma\,]\!] \overset{\text{I.H.}}{=} \lambda c.[\![\,t\,|\,\sigma''[x \mapsto [\![\,t\,|\,\sigma''\,|\,\gamma\,]\!]]\,|\,\gamma\,]\!] =$

575 $\lambda c.[\![\,t\,|\,\sigma''[x \mapsto [\![\,t\,|\,\sigma\,|\,\gamma\,]\!]]\,|\,\gamma\,]\!] = [\![\,c\langle\,x_1,\dots,x_n\,\rangle.s\,|\,\sigma[x \mapsto [\![\,t\,|\,\sigma\,|\,\gamma\,]\!]]\,|\,\gamma\,]\!]$

576   where

577     $\sigma'' = \sigma[x_i \mapsto \sigma(x_i)\{c/\gamma(c)\}]_{i\in[n]}$

578

579 Inductive Case: Sharing

580    $[\![\, u[z_1,\ldots,z_n \leftarrow s]\{t/x\}\,|\,\sigma\,|\,\gamma\,]\!] = [\![\, u\{t/x\}[z_1,\ldots,z_n \leftarrow s\{t/x\}]\,|\,\sigma\,|\,\gamma\,]\!] = [\![\, u\{t/x\}\,|\,\sigma''\,|\,\gamma\,]\!]$

581    $\stackrel{\text{I.H.}}{=} [\![\, u\,|\,\sigma'''\,|\,\gamma\,]\!] = [\![\, u[z_1,\ldots,z_n \leftarrow s]\,|\,\sigma'\,|\,\gamma\,]\!]$

582   where

583    $\sigma'' = \sigma[z_1 \mapsto [\![\, s\{t/x\}\,|\,\sigma\,|\,\gamma\,]\!],\ldots,z_n \mapsto [\![\, s\{t/x\}\,|\,\sigma\,|\,\gamma\,]\!]]$

584    $\sigma''' = \sigma'[z_1 \mapsto [\![\, s\,|\,\sigma'\,|\,\gamma\,]\!],\ldots,z_n \mapsto [\![\, s\,|\,\sigma'\,|\,\gamma\,]\!]]$

585

586   Inductive Case: Distributor 1

587    $[\![\, u[e_1\langle \vec{w}_1 \rangle,\ldots,e_n\langle \vec{w}_n \rangle\,|\,c\langle \underline{c} \rangle \overline{[\Gamma]}]\{t/x\}\,|\,\sigma\,|\,\gamma\,]\!]$

588    $= [\![\, u\overline{[\Gamma]}\{t/x\}\,|\,\sigma\,|\,\gamma'\,]\!] \stackrel{\text{I.H.}}{=} [\![\, u\overline{[\Gamma]}\,|\,\sigma'\,|\,\gamma'\,]\!]$

589    $= [\![\, u[e_1\langle \vec{w}_1 \rangle,\ldots,e_n\langle \vec{w}_n \rangle\,|\,c\langle \vec{x} \rangle \overline{[\Gamma]}]\,|\,\sigma'\,|\,\gamma\,]\!]$

590   where

591    $\gamma' = \gamma[e_1 \mapsto c,\ldots,e_n \mapsto c]$

592

593   Inductive Case: Distributor 2

594    $[\![\, u[e_1\langle \vec{w}_1 \rangle,\ldots,e_n\langle \vec{w}_n \rangle\,|\,c\langle \vec{x} \rangle \overline{[\Gamma]}]\{t/x\}\,|\,\sigma\,|\,\gamma\,]\!]$

595    $= [\![\, u\overline{[\Gamma]}\{t/x\}\,|\,\sigma''\,|\,\gamma'\,]\!] \stackrel{\text{I.H.}}{=} [\![\, u\overline{[\Gamma]}\,|\,\sigma'''\,|\,\gamma'\,]\!]$

596    $= [\![\, u[e_1\langle \vec{w}_1 \rangle,\ldots,e_n\langle \vec{w}_n \rangle\,|\,c\langle \vec{x} \rangle \overline{[\Gamma]}]\,|\,\sigma'\,|\,\gamma\,]\!]$

597   where

598    $\gamma' = \gamma[e_1 \mapsto c,\ldots,e_n \mapsto c]$ ◀

599   The proof for Proposition 13 (repeated here) is shown below. Book-keeping commutes

600 with the translation in the following way

601   if $c\langle y_1,\ldots,y_m \rangle. \in (u)_{fc}$ such that $\{x_1,\ldots,x_n\} \subset \{y_1,\ldots,y_m\}$

602   and for those $z \in \{y_1,\ldots,y_m\}/\{x_1,\ldots,x_n\}$, $\gamma(c) \notin (\sigma(z))_{fv}$

603   or if simply $\{x_1,\ldots,x_n\} \cap (u)_{fv} = \{\}$

$$[\![\, u\{x_1,\ldots,x_n/c\}_b\,|\,\sigma\,|\,\gamma\,]\!] = [\![\, u\,|\,\sigma\,|\,\gamma\,]\!]$$

604   **Proof.** We prove this by induction on u

605

606   Base Case: Variable

607    $[\![\, x\{x_1,\ldots,x_n/c\}_b\,|\,\sigma\,|\,\gamma\,]\!] = [\![\, x\,|\,\sigma\,|\,\gamma\,]\!] = \sigma(x) = \sigma'(x) = [\![\, x\,|\,\sigma'\,|\,\gamma'\,]\!]$

608   Since is cannot be that $x \in \{x_1,\ldots,x_n\}$

609

610   Base Case: Phantom-Abstraction

611    $[\![\,(c\langle y_1,\ldots,y_m \rangle.t)\{x_1,\ldots,x_n/c\}_b\,|\,\sigma\,|\,\gamma\,]\!] = [\![\, c\langle x_1,\ldots,x_n \rangle.t\,|\,\sigma\,|\,\gamma\,]\!]$

612    $= \lambda c.[\![\, t\,|\,\sigma''\,|\,\gamma\,]\!] = \lambda c.[\![\, t\,|\,\sigma''\,|\,\gamma'\,]\!] = [\![\, c\langle y_1,\ldots,y_m \rangle.t\,|\,\sigma'\,|\,\gamma'\,]\!]$

613   where

614    $\sigma = \sigma_1[x_1 \mapsto M_1,\ldots,x_n \mapsto M_n]$

615    $\sigma'' = \sigma_1[x_1 \mapsto M_1\{c/d\},\ldots,x_n \mapsto M_n\{c/d\}]$

616    $\gamma(c) = d$

617   Note: due to condition of Proposition any $\{y_i \mapsto M_i\{c/d\}\} = \{y_i \mapsto M_i\}$

618

619   Base Case: Distributor

620    $[\![\, u[e_1\langle \vec{w}_1 \rangle,\ldots,e_n\langle \vec{w}_n \rangle\,|\,c\langle y_1,\ldots,y_m \rangle \overline{[\Gamma]}]\{x_1,\ldots,x_n/c\}_b\,|\,\sigma\,|\,\gamma\,]\!]$

621    $= [\![\, u[e_1\langle \vec{w}_1 \rangle,\ldots,e_n\langle \vec{w}_n \rangle\,|\,c\langle x_1,\ldots,x_n \rangle \overline{[\Gamma]}]\,|\,\sigma\,|\,\gamma\,]\!] = [\![\, u\overline{[\Gamma]}\,|\,\sigma'\,|\,\gamma'\,]\!]$

622    $= [\![\, u[e_1\langle \vec{w}_1 \rangle,\ldots,e_n\langle \vec{w}_n \rangle\,|\,c\langle y_1,\ldots,y_m \rangle \overline{[\Gamma]}]\,|\,\sigma\,|\,\gamma\,]\!]$

623   where $\gamma' = \gamma[e_1 \mapsto c,\ldots,e_n \mapsto c]$

624    $\sigma = \sigma_1[x_1 \mapsto M_1,\ldots,x_n \mapsto M_n]$

625 $\sigma' = \sigma_1[x_1 \mapsto M_1\{c/\gamma(c)\}, \ldots, x_n \mapsto M_n\{c/\gamma(c)\}]$

627 Inductive Case: Application

628 $[\![(s\,t)\{x_1, \ldots, x_n/c\}_b \,|\, \sigma \,|\, \gamma]\!] = [\![s\{x_1, \ldots, x_n/c\}_b \,|\, \sigma \,|\, \gamma]\!] [\![t\{x_1, \ldots, x_n/c\}_b \,|\, \sigma \,|\, \gamma]\!]$

629 $\overset{\text{I.H.}}{=} [\![s \,|\, \sigma \,|\, \gamma]\!] [\![t \,|\, \sigma \,|\, \gamma]\!] = [\![s\,t \,|\, \sigma \,|\, \gamma]\!]$

631 Inductive Case: Abstraction

632 $[\![(z\langle z\rangle.t)\{x_1, \ldots, x_n/c\}_b \,|\, \sigma \,|\, \gamma]\!] = \lambda z.[\![t\{x_1, \ldots, x_n/c\}_b \,|\, \sigma \,|\, \gamma]\!] \overset{\text{I.H.}}{=} \lambda z.[\![t \,|\, \sigma \,|\, \gamma]\!] = [\![z\langle z\rangle.t \,|\, \sigma \,|\, \gamma]\!]$

634 Inductive Case: Phantom-Abstraction

635 $[\![(d\langle z_1, \ldots, z_m\rangle.t)\{x_1, \ldots, x_n/c\}_b \,|\, \sigma \,|\, \gamma]\!] = \lambda d.[\![t\{x_1, \ldots, x_n/c\}_b \,|\, \sigma' \,|\, \gamma]\!]$

636 $\overset{\text{I.H.}}{=} \lambda d.[\![t \,|\, \sigma' \,|\, \gamma]\!] = [\![d\langle z_1, \ldots, z_m\rangle.t \,|\, \sigma \,|\, \gamma]\!]$

638 Inductive Case: Sharing

639 $[\![u[z_1, \ldots, z_m \leftarrow t]\{x_1, \ldots, x_n/c\}_b \,|\, \sigma \,|\, \gamma]\!]$

640 $= [\![u\{x_1, \ldots, x_n/c\}_b[z_1, \ldots, z_m \leftarrow t\{x_1, \ldots, x_n/c\}_b] \,|\, \sigma \,|\, \gamma]\!]$

641 $= [\![u\{x_1, \ldots, x_n/c\}_b \,|\, \sigma' \,|\, \gamma]\!] \overset{\text{I.H.}}{=} [\![u \,|\, \sigma'' \,|\, \gamma]\!] = [\![u[z_1, \ldots, z_m \leftarrow t] \,|\, \sigma \,|\, \gamma]\!]$

643 Inductive Case: Distributor

644 $[\![u[e_1\langle \vec{w_1}\rangle, \ldots, e_m\langle \vec{w_m}\rangle \,|\, d\langle d\rangle \overline{[\Gamma]}]\{x_1, \ldots, x_n/c\}_b \,|\, \sigma \,|\, \gamma]\!]$

645 $= [\![u[\underline{e_1\langle \vec{w_1}\rangle, \ldots, e_m\langle \vec{w_m}\rangle \,|\, d\langle d\rangle \overline{[\Gamma]}}\{x_1, \ldots, x_n/c\}_b] \,|\, \sigma \,|\, \gamma]\!]$

646 $= [\![u\overline{[\Gamma]}\{x_1, \ldots, x_n/c\}_b \,|\, \sigma \,|\, \gamma']\!] \overset{\text{I.H.}}{=} [\![u\overline{[\Gamma]} \,|\, \sigma \,|\, \gamma']\!]$

647 $= [\![u[e_1\langle \vec{w_1}\rangle, \ldots, e_m\langle \vec{w_m}\rangle \,|\, d\langle d\rangle \overline{[\Gamma]}] \,|\, \sigma \,|\, \gamma]\!]$ ◀

648 The proof for 14 (repeated here) is below. Exorcisms commute with the translation in
649 the following way

650 if $c\langle x_1, \ldots, x_n\rangle. \in (u)_{fc}$ or $\{x_1, \ldots, x_n\} \cap (u)_{fv} = \{\}$

$$[\![u\{c\langle x_1, \ldots, x_n\rangle\}_e \,|\, \sigma \,|\, \gamma]\!] = [\![u \,|\, \sigma[x_i \mapsto c]_{i\in[n]} \,|\, \gamma]\!]$$

651 **Proof.** We prove this by induction on u

653 Base Case: Variable

654 $[\![z\{c\langle x_1, \ldots, x_n\rangle\}_e \,|\, \sigma \,|\, \gamma]\!] = [\![z \,|\, \sigma \,|\, \gamma]\!] = \sigma(z) = \sigma'(z) = [\![z \,|\, \sigma' \,|\, \gamma]\!]$

656 Base Case: Phantom-Abstraction

657 $[\![(c\langle x_1, \ldots, x_n\rangle.t)\{c\langle x_1, \ldots, x_n\rangle\}_e \,|\, \sigma \,|\, \gamma]\!] = [\![c\langle c\rangle.t[x_1, \ldots, x_n \leftarrow c] \,|\, \sigma \,|\, \gamma]\!]$

658 $= \lambda c.[\![t[x_1, \ldots, x_n \leftarrow c] \,|\, \sigma \,|\, \gamma]\!] = \lambda c.[\![t \,|\, \sigma' \,|\, \gamma]\!] = [\![c\langle x_1, \ldots, x_n\rangle.t \,|\, \sigma' \,|\, \gamma]\!]$

660 Base Case: Distributor

661 $[\![u[e_1\langle \vec{w_1}\rangle, \ldots, e_m\langle \vec{w_m}\rangle \,|\, c\langle x_1, \ldots, x_n\rangle \overline{[\Gamma]}]\{c\langle x_1, \ldots, x_n\rangle\}_e \,|\, \sigma \,|\, \gamma]\!]$

662 $= [\![u[\underline{e_1\langle \vec{w_1}\rangle, \ldots, e_m\langle \vec{w_m}\rangle \,|\, c\langle c\rangle \overline{[\Gamma]}}[x_1, \ldots, x_n \leftarrow c]] \,|\, \sigma \,|\, \gamma]\!]$

663 $= [\![u\overline{[\Gamma]}[x_1, \ldots, x_n \leftarrow c] \,|\, \sigma \,|\, \gamma']\!] = [\![u\overline{[\Gamma]} \,|\, \sigma' \,|\, \gamma']\!]$

664 $= [\![u[e_1\langle \vec{w_1}\rangle, \ldots, e_m\langle \vec{w_m}\rangle \,|\, c\langle x_1, \ldots, x_n\rangle \overline{[\Gamma]}] \,|\, \sigma' \,|\, \gamma]\!]$

666 Inductive Case: Application

667 $[\![(s\,t)\{c\langle x_1, \ldots, x_n\rangle\}_e \,|\, \sigma \,|\, \gamma]\!] = [\![s\{c\langle x_1, \ldots, x_n\rangle\}_e \,|\, \sigma \,|\, \gamma]\!] [\![t\{c\langle x_1, \ldots, x_n\rangle\}_e \,|\, \sigma \,|\, \gamma]\!]$

668 $\overset{\text{I.H.}}{=} [\![s \,|\, \sigma' \,|\, \gamma]\!] [\![t \,|\, \sigma' \,|\, \gamma]\!] = [\![s\,t \,|\, \sigma' \,|\, \gamma]\!]$

670   Inductive Case: Abstraction

671   $\llbracket (z\langle z\rangle.t)\{c\langle x_1,\ldots,x_n\rangle\}_e \,|\,\sigma\,|\,\gamma\rrbracket = \lambda z.\llbracket t\{c\langle x_1,\ldots,x_n\rangle\}_e \,|\,\sigma\,|\,\gamma\rrbracket$

672   $\overset{\text{I.H.}}{=} \lambda z.\llbracket t\,|\,\sigma'\,|\,\gamma\rrbracket = \llbracket z\langle z\rangle.t\,|\,\sigma'\,|\,\gamma\rrbracket$

673

674   Inductive Case: Phantom-Abstraction

675   $\llbracket (d\langle z_1,\ldots,z_m\rangle.t)\{c\langle x_1,\ldots,x_n\rangle\}_e \,|\,\sigma\,|\,\gamma\rrbracket = \lambda d.\llbracket t\{c\langle x_1,\ldots,x_n\rangle\}_e \,|\,\sigma''\,|\,\gamma\rrbracket$

676   $\overset{\text{I.H.}}{=} \lambda d.\llbracket t\,|\,\sigma'''\,|\,\gamma\rrbracket = \llbracket d\langle z_1,\ldots,z_m\rangle.t\,|\,\sigma'\,|\,\gamma\rrbracket$

677

678   Inductive Case: Sharing

679   $\llbracket u[z_1,\ldots,z_m \leftarrow t]\{c\langle x_1,\ldots,x_n\rangle\}_e \,|\,\sigma\,|\,\gamma\rrbracket$

680   $= \llbracket u\{c\langle x_1,\ldots,x_n\rangle\}_e[z_1,\ldots,z_m \leftarrow t\{c\langle x_1,\ldots,x_n\rangle\}_e]\,|\,\sigma\,|\,\gamma\rrbracket$

681   $= \llbracket u\{c\langle x_1,\ldots,x_n\rangle\}_e \,|\,\sigma''\,|\,\gamma\rrbracket \overset{\text{I.H.}}{=} \llbracket u\,|\,\sigma'''\,|\,\gamma\rrbracket = \llbracket u[z_1,\ldots,z_m \leftarrow t]\,|\,\sigma'\,|\,\gamma\rrbracket$

682

683   Inductive Case: Distributor

684   $\llbracket u[e_1\langle \vec{w_1}\rangle,\ldots,e_m\langle \vec{w_m}\rangle\,|\,d\langle d\rangle\,\overline{[\Gamma]}]\{c\langle x_1,\ldots,x_n\rangle\}_e \,|\,\sigma\,|\,\gamma\rrbracket$

685   $= \llbracket u[\underline{e_1\langle \vec{w_1}\rangle},\ldots,e_m\langle \vec{w_m}\rangle\,|\,d\langle d\rangle\,\overline{[\Gamma]}\{c\langle x_1,\ldots,x_n\rangle\}_e]\,|\,\sigma\,|\,\gamma\rrbracket$

686   $= \llbracket u\overline{[\Gamma]}\{c\langle x_1,\ldots,x_n\rangle\}_e \,|\,\sigma\,|\,\gamma'\rrbracket \overset{\text{I.H.}}{=} \llbracket u\overline{[\Gamma]}\,|\,\sigma'\,|\,\gamma'\rrbracket$

687   $= \llbracket u[e_1\langle \vec{w_1}\rangle,\ldots,e_m\langle \vec{w_m}\rangle\,|\,d\langle d\rangle\,\overline{[\Gamma]}]\,|\,\sigma\,|\,\gamma'\rrbracket$ ◀

688   We prove Lemma 16 on a case by case basis. If $s \rightsquigarrow_{L,D,C} t$ then $\llbracket s\,|\,\sigma\,|\,\gamma\rrbracket = \llbracket t\,|\,\sigma\,|\,\gamma\rrbracket$

Proof. We prove this by induction. First we to a case-by-case basis for the base case.

Case: (c_1)

$$u[\vec{w} \leftarrow y][\vec{x} \cdot y \leftarrow t] \rightsquigarrow_C u[\vec{x} \cdot \vec{w} \leftarrow t]$$

$\llbracket u[\vec{w} \leftarrow y][\vec{x} \cdot y \leftarrow t]\,|\,\sigma\,|\,\gamma\rrbracket = \llbracket u[\vec{w} \leftarrow y]\,|\,\sigma'\,|\,\gamma\rrbracket = \llbracket u\,|\,\sigma''\,|\,\gamma\rrbracket = \llbracket u[\vec{x} \cdot \vec{w} \leftarrow t]\,|\,\sigma\,|\,\gamma\rrbracket$
where
$\sigma' = \sigma[x \mapsto \llbracket t\,|\,\sigma\,|\,\gamma\rrbracket]_{\forall x \in \vec{x}}[y \mapsto \llbracket t\,|\,\sigma\,|\,\gamma\rrbracket]$
$\sigma'' = \sigma'[w \mapsto \llbracket t\,|\,\sigma\,|\,\gamma\rrbracket]_{\forall w \in \vec{w}}$

Case: (c_2)

$$u[x \leftarrow t] \rightsquigarrow_C u\{t/x\}$$

$\llbracket u[x \leftarrow t]\,|\,\sigma\,|\,\gamma\rrbracket = \llbracket u\,|\,\sigma[x \mapsto \llbracket t\,|\,\sigma\,|\,\gamma\rrbracket]\,|\,\gamma\rrbracket = \llbracket u\{t/x\}\,|\,\sigma\,|\,\gamma\rrbracket$

Case: (d_1)

$$u[x_1 \ldots x_n \leftarrow s\,t] \rightsquigarrow_D u\{z_1\,y_1/x_1\}\ldots\{z_n\,y_n/x_n\}[z_1 \ldots z_n \leftarrow s][y_1 \ldots y_n \leftarrow t]$$

$\llbracket u[x_1 \ldots x_n \leftarrow s\,t]\,|\,\sigma\,|\,\gamma\rrbracket = \llbracket u\,|\,\sigma'\,|\,\gamma\rrbracket$
where
$\sigma' = \sigma[x_i \mapsto \llbracket s\,t\,|\,\sigma\,|\,\gamma\rrbracket]_{1 \leq i \leq n} = \sigma[x_i \mapsto \llbracket s\,|\,\sigma\,|\,\gamma\rrbracket\,\llbracket t\,|\,\sigma\,|\,\gamma\rrbracket]_{1 \leq i \leq n}$

$\llbracket u\{z_1\,y_1/x_1\}\ldots\{z_n\,y_n/x_n\}[z_1 \ldots z_n \leftarrow s][y_1 \ldots y_n \leftarrow t]\,|\,\sigma\,|\,\gamma\rrbracket$
$= \llbracket u\{z_1\,y_1/x_1\}\ldots\{z_n\,y_n/x_n\}\,|\,\sigma''\,|\,\gamma\rrbracket$
where
$\sigma'' = \sigma[z_i \mapsto \llbracket s\,|\,\sigma\,|\,\gamma\rrbracket]_{1 \leq i \leq n}[y_i \mapsto \llbracket t\,|\,\sigma\,|\,\gamma\rrbracket]_{1 \leq i \leq n}$ since $y_i \notin (s)_{fv}$
$= \llbracket u\,|\,\sigma'''\,|\,\gamma\rrbracket$
where

$\sigma''' = \sigma''[x_i \mapsto [\![\, z_i\, y_i\, |\, \sigma''\, |\, \gamma\,]\!]\,]_{1 \le i \le n} = \sigma[x_i \mapsto [\![\, s\, |\, \sigma\, |\, \gamma\,]\!]\, [\![\, t\, |\, \sigma\, |\, \gamma\,]\!]\,]_{1 \le i \le n}$
since z_i and $y_i \notin (u)_{fv}$

Case: (d_2)

$$u[x_1, \ldots, x_n \leftarrow c\langle\, \vec{y}\, \rangle.t] \rightsquigarrow_D$$

$$u\{e_i\langle\, w_1^i\, \rangle.w_1^i / x_i\}_{1 \le i \le n}[e_1\langle\, w_1^1\, \rangle \ldots e_n\langle\, w_1^n\, \rangle\, |\, c\langle\, \vec{y}\, \rangle\, [w_1^1, \ldots, w_1^n \leftarrow t]]$$

SubCase: $\vec{y} = c$
$[\![\, u[x_1, \ldots, x_n \leftarrow c\langle\, c\, \rangle.t]\, |\, \sigma\, |\, \gamma\,]\!] = [\![\, u\, |\, \sigma'\, |\, \gamma\,]\!]$
where $\sigma' = \sigma[x_i \mapsto [\![\, c\langle\, c\, \rangle.t\, |\, \sigma\, |\, \gamma\,]\!]\,]_{1 \le i \le n} = \sigma[x_i \mapsto \lambda c.[\![\, t\, |\, \sigma\, |\, \gamma\,]\!]\,]_{1 \le i \le n}$

$[\![\, u\{e_i\langle\, w_1^i\, \rangle.w_1^i / x_i\}_{1 \le i \le n}[e_1\langle\, w_1^1\, \rangle \ldots e_n\langle\, w_1^n\, \rangle\, |\, c\langle\, c\, \rangle\, [w_1^1, \ldots, w_1^n \leftarrow t]]\, |\, \sigma\, |\, \gamma\,]\!]$
$[\![\, u\{e_i\langle\, w_1^i\, \rangle.w_1^i / x_i\}_{1 \le i \le n}[w_1^1, \ldots, w_1^n \leftarrow t]\, |\, \sigma\, |\, \gamma'\,]\!]$
$= [\![\, u\{e_i\langle\, w_1^i\, \rangle.w_1^i / x_i\}_{1 \le i \le n}\, |\, \sigma'\, |\, \gamma'\,]\!]$
where
$\gamma' = \gamma[e_1 \mapsto c, \ldots, e_n \mapsto c]$
$\sigma' = \sigma[w_1^i \mapsto [\![\, t\, |\, \sigma\, |\, \gamma'\,]\!]\,]_{1 \le i \le n} = \sigma[w_1^i \mapsto [\![\, t\, |\, \sigma\, |\, \gamma\,]\!]\,]_{1 \le i \le n}$
$= [\![\, u\, |\, \sigma''\, |\, \gamma'\,]\!] = [\![\, u\, |\, \sigma''\, |\, \gamma\,]\!]$
where
$\sigma'' = \sigma'[x_i \mapsto [\![\, e_i\langle\, w_1^i\, \rangle.w_1^i\, |\, \sigma'\, |\, \gamma'\,]\!]\,]_{1 \le i \le n} = \sigma'[x_i \mapsto \lambda e_i.[\![\, w_1^i\, |\, \sigma_i'\, |\, \gamma'\,]\!]\,]_{1 \le i \le n}$
$\quad = \sigma'[x_i \mapsto \lambda e_i.[\![\, t\, |\, \sigma\, |\, \gamma\,]\!]\{e_i / c\}]_{1 \le i \le n} =_\alpha \sigma'[x_i \mapsto \lambda c.[\![\, t\, |\, \sigma\, |\, \gamma\,]\!]\,]_{1 \le i \le n}$
$\quad = \sigma[x_i \mapsto \lambda c.[\![\, t\, |\, \sigma\, |\, \gamma\,]\!]\,]_{1 \le i \le n}$ since $w_1^i \notin (u)_{fv}$

SubCase: $\vec{y} = \{y_1, \ldots, y_m\}$
$[\![\, u[x_1, \ldots, x_n \leftarrow c\langle\, y_1, \ldots, y_m\, \rangle.t]\, |\, \sigma\, |\, \gamma\,]\!] = [\![\, u\, |\, \sigma'\, |\, \gamma\,]\!]$
where
$\sigma' = \sigma[x_i \mapsto [\![\, c\langle\, y_1, \ldots, y_m\, \rangle.t\, |\, \sigma\, |\, \gamma\,]\!]\,]_{1 \le i \le n} = \sigma[x_i \mapsto \lambda c.[\![\, t\, |\, \sigma''\, |\, \gamma\,]\!]\,]_{1 \le i \le n}$
$\sigma = \sigma_1[y_1 \mapsto M_1, \ldots, y_m \mapsto M_m]$
$\sigma'' = \sigma_1[y_1 \mapsto M_1\{c/\gamma(c)\}, \ldots, y_m \mapsto M_m\{c/\gamma(c)\}]$

$[\![\, u\{e_i\langle\, w_1^i\, \rangle.w_1^i / x_i\}_{1 \le i \le n}[e_1\langle\, w_1^1\, \rangle \ldots e_n\langle\, w_1^n\, \rangle\, |\, c\langle\, y_1, \ldots, y_m\, \rangle\, [w_1^1, \ldots, w_1^n \leftarrow t]]\, |\, \sigma\, |\, \gamma\,]\!]$
$[\![\, u\{e_i\langle\, w_1^i\, \rangle.w_1^i / x_i\}_{1 \le i \le n}[w_1^1, \ldots, w_1^n \leftarrow t]\, |\, \sigma''\, |\, \gamma'\,]\!]$
where $\gamma' = \gamma[e_1 \mapsto c, \ldots, e_n \mapsto c]$
$= [\![\, u\{e_i\langle\, w_1^i\, \rangle.w_1^i / x_i\}_{1 \le i \le n}\, |\, \sigma'''\, |\, \gamma'\,]\!]$
where $\sigma''' = \sigma''[w_1^i \mapsto [\![\, t\, |\, \sigma''\, |\, \gamma'\,]\!]\,]_{1 \le i \le n} = \sigma''[w_1^i \mapsto [\![\, t\, |\, \sigma''\, |\, \gamma\,]\!]\,]_{1 \le i \le n}$
$= [\![\, u\, |\, \sigma''''\, |\, \gamma'\,]\!] = [\![\, u\, |\, \sigma''''\, |\, \gamma\,]\!] = [\![\, u\, |\, \sigma''\, |\, \gamma\,]\!]$
where $\sigma'''' = \sigma'''[x_i \mapsto [\![\, e_i\langle\, w_1^i\, \rangle.w_1^i\, |\, \sigma'''\, |\, \gamma'\,]\!]\,]_{1 \le i \le n} = \sigma'''[x_i \mapsto \lambda e_i.[\![\, w_1^i\, |\, \sigma_i'''\, |\, \gamma'\,]\!]\,]_{1 \le i \le n}$
$\quad = \sigma'''[x_i \mapsto \lambda e_i.[\![\, t\, |\, \sigma''\, |\, \gamma\,]\!]\{e_i / \gamma'(e_i)\}]_{1 \le i \le n} =_\alpha \sigma'''[x_i \mapsto \lambda c.[\![\, t\, |\, \sigma''\, |\, \gamma\,]\!]\,]_{1 \le i \le n}$

Case: (d_3)

$$u[e_1\langle\, \vec{w_1}\, \rangle \ldots e_n\langle\, \vec{w_n}\, \rangle\, |\, c\langle\, c\, \rangle\, [\vec{w_1}, \ldots, \vec{w_n} \leftarrow c]] \rightsquigarrow_D u\{e_1\langle\, \vec{w_1}\, \rangle\}_e \ldots \{e_n\langle\, \vec{w_n}\, \rangle\}_e$$

$[\![\, u[e_1\langle\, \vec{w_1}\, \rangle \ldots e_n\langle\, \vec{w_n}\, \rangle\, |\, c\langle\, c\, \rangle\, [\vec{w_1}, \ldots, \vec{w_n} \leftarrow c]]\, |\, \sigma\, |\, \gamma\,]\!]$
$= [\![\, u[\vec{w_1}, \ldots, \vec{w_n} \leftarrow c]\, |\, \sigma\, |\, \gamma'\,]\!] = [\![\, u\, |\, \sigma'\, |\, \gamma'\,]\!]$
$= [\![\, u\{e_1\langle\, \vec{w_1}\, \rangle\}_e \ldots \{e_n\langle\, \vec{w_n}\, \rangle\}_e\, |\, \sigma\, |\, \gamma'\,]\!] = [\![\, u\{e_1\langle\, \vec{w_1}\, \rangle\}_e \ldots \{e_n\langle\, \vec{w_n}\, \rangle\}_e\, |\, \sigma\, |\, \gamma\,]\!]$

For the remaining cases, we say $[\![\, t[\Gamma]\, |\, \sigma\, |\, \gamma\,]\!]$ produces $[\![\, t\, |\, \sigma_\Gamma\, |\, \gamma_\Gamma\,]\!]$ where σ_Γ and γ_Γ are

the resulting maps from interpreting the closure $[\Gamma]$

Case: (l_1)

$$s[\Gamma]\,t \rightsquigarrow_L (s\,t)[\Gamma]$$

$$[\![\,s[\Gamma]\,t\,|\,\sigma\,|\,\gamma\,]\!] = [\![\,s\,|\,\sigma_\Gamma\,|\,\gamma_\Gamma\,]\!]\,[\![\,t\,|\,\sigma\,|\,\gamma\,]\!] = [\![\,s\,|\,\sigma_\Gamma\,|\,\gamma_\Gamma\,]\!]\,[\![\,t\,|\,\sigma_\Gamma\,|\,\gamma_\Gamma\,]\!] = [\![\,(s\,t)[\Gamma]\,|\,\sigma\,|\,\gamma\,]\!]$$

Case: (l_2)

$$s[\Gamma]\,t \rightsquigarrow_L (s\,t)[\Gamma]$$

$$[\![\,s\,(t[\Gamma])\,|\,\sigma\,|\,\gamma\,]\!] = [\![\,s\,|\,\sigma\,|\,\gamma\,]\!]\,[\![\,t\,|\,\sigma_\Gamma\,|\,\gamma_\Gamma\,]\!] = [\![\,s\,|\,\sigma_\Gamma\,|\,\gamma_\Gamma\,]\!]\,[\![\,t\,|\,\sigma_\Gamma\,|\,\gamma_\Gamma\,]\!] = [\![\,(s\,t)[\Gamma]\,|\,\sigma\,|\,\gamma\,]\!]$$

Case: (l_3)

$$d\langle\,\vec{x}\,\rangle.t[\Gamma] \rightsquigarrow_L (d\langle\,\vec{x}\,\rangle.t)[\Gamma]$$

SubCase: $\vec{x} = d$

$$[\![\,d\langle\,d\,\rangle.t[\Gamma]\,|\,\sigma\,|\,\gamma\,]\!] = \lambda d.[\![\,t[\Gamma]\,|\,\sigma\,|\,\gamma\,]\!] = \lambda d.[\![\,t\,|\,\sigma_\Gamma\,|\,\gamma_\Gamma\,]\!] = [\![\,d\langle\,d\,\rangle.t\,|\,\sigma_\Gamma\,|\,\gamma_\Gamma\,]\!] = [\![\,(d\langle\,d\,\rangle.t)[\Gamma]\,|\,\sigma\,|\,\gamma\,]\!]$$

SubCase: $\vec{x} = x_1, \ldots, x_n$

$$[\![\,d\langle\,x_1, \ldots, x_n\,\rangle.t[\Gamma]\,|\,\sigma\,|\,\gamma\,]\!] = \lambda d.[\![\,t[\Gamma]\,|\,\sigma'\,|\,\gamma\,]\!] = \lambda d.[\![\,t\,|\,\sigma'_\Gamma\,|\,\gamma_\Gamma\,]\!] = [\![\,d\langle\,x_1, \ldots, x_n\,\rangle.t\,|\,\sigma_\Gamma\,|\,\gamma_\Gamma\,]\!]$$
$$= [\![\,(d\langle\,x_1, \ldots, x_n\,\rangle.t)[\Gamma]\,|\,\sigma\,|\,\gamma\,]\!]$$

since we know $x_1, \ldots, x_n \notin ([\Gamma])_{fv}$

Case: (l_4)

$$u[\vec{x} \leftarrow t[\Gamma]] \rightsquigarrow_L u[\vec{x} \leftarrow t][\Gamma]$$

$$[\![\,u[\vec{x} \leftarrow t[\Gamma]]\,|\,\sigma\,|\,\gamma\,]\!] = [\![\,u\,|\,\sigma'\,|\,\gamma\,]\!] = [\![\,u\,|\,\sigma''\,|\,\gamma_\Gamma\,]\!] = [\![\,u[\vec{x} \leftarrow t]\,|\,\sigma_\Gamma\,|\,\gamma_\Gamma\,]\!] = [\![\,u[\vec{x} \leftarrow t[\Gamma]]\,|\,\sigma\,|\,\gamma\,]\!]$$
where
$$\sigma' = \sigma[x \mapsto [\![\,t[\Gamma]\,|\,\sigma\,|\,\gamma\,]\!]]_{\forall x \in \vec{x}} = \sigma[x \mapsto [\![\,t\,|\,\sigma_\Gamma\,|\,\gamma_\Gamma\,]\!]]_{\forall x \in \vec{x}}$$
$$\sigma'' = \sigma_\Gamma[x \mapsto [\![\,t\,|\,\sigma_\Gamma\,|\,\gamma_\Gamma\,]\!]]_{\forall x \in \vec{x}}$$

Cases: (l_5)

$$u[e_1\langle\,\vec{w_1}\,\rangle \ldots e_n\langle\,\vec{w_n}\,\rangle\,|\,c\langle\,\vec{x}\,\rangle\,\overline{[\Gamma]}[\Gamma]] \rightsquigarrow_L$$
$$u\{(\vec{w_i}/\vec{z})/e_i\}_{b_{i\in[n]}}[e_1\langle\,\vec{w_1}/\vec{z}\,\rangle \ldots e_n\langle\,\vec{w_n}/\vec{z}\,\rangle\,|\,c\langle\,\vec{x}\,\rangle\,\overline{[\Gamma]}][\Gamma]$$

689   SubCase: $\vec{x} = c$

690   $[\![\,u[e_1\langle\,\vec{w_1}\,\rangle \ldots e_n\langle\,\vec{w_n}\,\rangle\,|\,c\langle\,c\,\rangle\,\overline{[\Gamma]}[\Gamma]]\,|\,\sigma\,|\,\gamma\,]\!] = [\![\,u\overline{[\Gamma]}[\Gamma]\,|\,\sigma\,|\,\gamma'\,]\!] = [\![\,u\overline{[\Gamma]}\,|\,\sigma_\Gamma\,|\,\gamma'_\Gamma\,]\!]$

691   $= [\![\,u\overline{[\Gamma]}\{\vec{z_1}/e_1\}_b \ldots \{\vec{z_n}/e_n\}_b\,|\,\sigma_\Gamma\,|\,\gamma'_\Gamma\,]\!] = [\![\,u\{\vec{z_1}/e_1\}_b \ldots \{\vec{z_n}/e_n\}_b\overline{[\Gamma]}\,|\,\sigma_\Gamma\,|\,\gamma'_\Gamma\,]\!]$

692   $= [\![\,u\{\vec{z_1}/e_1\}_b \ldots \{\vec{z_n}/e_n\}_b[e_1\langle\,\vec{z_1}\,\rangle \ldots e_n\langle\,\vec{z_n}\,\rangle\,|\,c\langle\,c\,\rangle\,\overline{[\Gamma]}]\,|\,\sigma_\Gamma\,|\,\gamma_\Gamma\,]\!]$

693   $= [\![\,u\{\vec{z_1}/e_1\}_b \ldots \{\vec{z_n}/e_n\}_b[e_1\langle\,\vec{z_1}\,\rangle \ldots e_n\langle\,\vec{z_n}\,\rangle\,|\,c\langle\,c\,\rangle\,\overline{[\Gamma]}][\Gamma]\,|\,\sigma\,|\,\gamma\,]\!]$

694

695   SubCase: $\vec{x} = x_1, \ldots, x_m$

696   $[\![\,u[e_1\langle\,\vec{w_1}\,\rangle \ldots e_n\langle\,\vec{w_n}\,\rangle\,|\,c\langle\,x_1, \ldots, x_m\,\rangle\,\overline{[\Gamma]}[\Gamma]]\,|\,\sigma\,|\,\gamma\,]\!]$

697   $= [\![\,u\overline{[\Gamma]}[\Gamma]\,|\,\sigma'\,|\,\gamma'\,]\!] = [\![\,u\overline{[\Gamma]}\,|\,\sigma'_\Gamma\,|\,\gamma'_\Gamma\,]\!] = [\![\,u\overline{[\Gamma]}\{\vec{z_1}/e_1\}_b \ldots \{\vec{z_n}/e_n\}_b\,|\,\sigma_\Gamma\,|\,\gamma'_\Gamma\,]\!]$

698   $= [\![\,u\{\vec{z_1}/e_1\}_b \ldots \{\vec{z_n}/e_n\}_b\overline{[\Gamma]}\,|\,\sigma_\Gamma\,|\,\gamma'_\Gamma\,]\!]$

699   $= [\![\,u\{\vec{z_1}/e_1\}_b \ldots \{\vec{z_n}/e_n\}_b[e_1\langle\,\vec{z_1}\,\rangle \ldots e_n\langle\,\vec{z_n}\,\rangle\,|\,c\langle\,x_1, \ldots, x_n\,\rangle\,\overline{[\Gamma]}]\,|\,\sigma_\Gamma\,|\,\gamma_\Gamma\,]\!]$

700   $= [\![\,u\{\vec{z_1}/e_1\}_b \ldots \{\vec{z_n}/e_n\}_b[e_1\langle\,\vec{z_1}\,\rangle \ldots e_n\langle\,\vec{z_n}\,\rangle\,|\,c\langle\,x_1, \ldots, x_n\,\rangle\,\overline{[\Gamma]}][\Gamma]\,|\,\sigma\,|\,\gamma\,]\!]$

701

702 Inductive Case: Application $t \rightsquigarrow_{(C,D,L)} t'$

703 $[\![\,t\,s\,|\,\sigma\,|\,\gamma\,]\!] = [\![\,t\,|\,\sigma\,|\,\gamma\,]\!]\,[\![\,s\,|\,\sigma\,|\,\gamma\,]\!] \overset{\text{I.H.}}{=} [\![\,t'\,|\,\sigma\,|\,\gamma\,]\!]\,[\![\,s\,|\,\sigma\,|\,\gamma\,]\!] = [\![\,t'\,s\,|\,\sigma\,|\,\gamma\,]\!]$ ◄

B Strong Normalisation of Sharing Reductions

The weakening calculus is used to show preservation of strong normalisation with respect to the λ-calculus. A β-step in our calculus may occur within a weakening, and therefore is simulated by zero β-steps in the λ-calculus. Therefore if there is an infinite reduction path located inside a weakening in Λ_a^S, then the reduction path is not preserved in the corresponding λ-term as there are no weakenings. To deal with this, just as done in [**?, ?, ?**], we make use of the weakening calculus. A β-step is non-deleteing precisely because of the weakening construct. If a β-step would be deleting in the λ-calculus, then the weakening calculus would instead keep the deleted term around as 'garbage', which can continue to reduce unless explicitly 'garbage-collected' by extra (non-β) reduction steps. The weakening calculus has already been shown to preserve strong normalisation through the use of a perpetual strategy in [**?**]. A part of proving PSN is then using the weakening calculus to prove that if $t \in \Lambda_a^S$ has a infinite reduction path, then its translation into the weakening calculus also has an infinite reduction path.

First we demonstrate that our readback translation (Definition 18) is truly an extention of the translation into the λ-calculus (Definition 9). We therefore demonstrate that our operations (substitution, book-keeping, and exorcisms) commute with the two translation functions in the same way.

▶ **Proposition 37.** *Given $M \in \Lambda$ such that for all $v \in V$, $\gamma(v) \notin (M)_{fv}$ and $\sigma(x) = x$, the translation $[\![u\,|\,\sigma\,|\,\gamma]\!]_w$ commutes with substitution $\{M/x\}$ in the following way*

$$[\![u\{t/x\}\,|\,\sigma\,|\,\gamma]\!]_w = [\![u\,|\,\sigma[x \mapsto [\![t\,|\,\sigma\,|\,\gamma]\!]_w]\,|\,\gamma]\!]_w$$

Proof. We prove this by induction on u. The argument is similar to the proof of Proposition 37. We only discuss here to cases involving the three special cases defined in Definition 18.

Inductive Case: Weakening
$[\![u[\leftarrow s]\{t/x\}\,|\,\sigma\,|\,\gamma]\!]_w = [\![u\{t/x\}\,|\,\sigma\,|\,\gamma]\!]_w[\leftarrow [\![s\{t/x\}\,|\,\sigma\,|\,\gamma]\!]_w]$
$\stackrel{\text{I.H.}}{=} [\![u\,|\,\sigma'\,|\,\gamma]\!]_w[\leftarrow [\![s\,|\,\sigma'\,|\,\gamma]\!]_w] = [\![u[\leftarrow s]\,|\,\sigma'\,|\,\gamma]\!]_w$

Inductive Case: Distributor
$[\![u[\,|\,c\langle \vec{x} \rangle\, \overline{[\Gamma]}]\{t/x\}\,|\,\sigma\,|\,\gamma]\!]_w$

    SubCase: $\vec{x} = c$
$[\![u[\,|\,c\langle c \rangle\, \overline{[\Gamma]}]\{t/x\}\,|\,\sigma\,|\,\gamma]\!]_w = [\![u[\,|\,c\langle c \rangle\, \overline{[\Gamma]}\{t/x\}]\,|\,\sigma\,|\,\gamma]\!]_w$
$= [\![u\overline{[\Gamma]}\{t/x\}\,|\,\sigma''\,|\,\gamma']\!]_w \stackrel{\text{I.H.}}{=} [\![u\overline{[\Gamma]}\,|\,\sigma'''\,|\,\gamma']\!]_w = [\![u[\,|\,c\langle c \rangle\, \overline{[\Gamma]}]\,|\,\sigma'\,|\,\gamma]\!]_w$
where
$\sigma'' = \sigma \cup \{c \mapsto \bullet\}$
$\sigma''' = \sigma \cup \{c \mapsto \bullet\} \cup \{x \mapsto [\![t\,|\,\sigma''\,|\,\gamma']\!]_w\} = \sigma \cup \{c \mapsto \bullet\} \cup \{x \mapsto [\![t\,|\,\sigma\,|\,\gamma]\!]_w\}$

    SubCase: $\vec{x} = x_1, \ldots, x_n$
$[\![u[\,|\,c\langle x_1, \ldots, x_n \rangle\, \overline{[\Gamma]}]\{t/x\}\,|\,\sigma\,|\,\gamma]\!]_w$

       SubSubCase: $\vec{x} = x_1, \ldots, x_n, x$
$[\![u[\,|\,c\langle x_1, \ldots, x_n, x \rangle\, \overline{[\Gamma]}]\{t/x\}\,|\,\sigma\,|\,\gamma]\!]_w$
$[\![u[\,|\,c\langle x_1, \ldots, x_n, y_1, \ldots, y_m \rangle\, \overline{[\Gamma]}\{t/x\}]\,|\,\sigma\,|\,\gamma]\!]_w$
where $\{y_1, \ldots, y_m\} = (t)_{fv}$
$= [\![u\overline{[\Gamma]}\{t/x\}\,|\,\sigma''\,|\,\gamma]\!]_w$

749   where

750    $\sigma = \sigma_1 \cup \{x_1 \mapsto M_1, \ldots, x_n \mapsto M_n, y_1 \mapsto N_1, \ldots, y_m \mapsto N_m\}$

751    $\sigma'' = \sigma_1 \cup \{x_1 \mapsto M_1\{\bullet/\gamma(c)\},$

752        $\ldots, x_n \mapsto M_n\{\bullet/\gamma(c)\}, y_1 \mapsto N_1\{\bullet/\gamma(c)\}, \ldots, y_m \mapsto N_m\{\bullet/\gamma(c)\}\}$

753    $\overset{\text{I.H.}}{=} [\![u\overline{[\Gamma]} \,|\, \sigma''' \,|\, \gamma]\!]_{\mathcal{W}} = [\![u[\,|\,c\langle x_1, \ldots, x_n, x \rangle \overline{[\Gamma]}] \,|\, \sigma' \,|\, \gamma]\!]_{\mathcal{W}}$

754   where $\sigma''' = \sigma'' \cup \{x \mapsto [\![t \,|\, \sigma'' \,|\, \gamma]\!]_{\mathcal{W}}\} = \sigma'' \cup \{x \mapsto [\![t \,|\, \sigma' \,|\, \gamma]\!]_{\mathcal{W}}\{\bullet/\gamma(c)\}\}$

755   since $\{y_1, \ldots, y_m\} = (t)_{fv}$

756

757   SubSubCase: $\vec{x} = x_1, \ldots, x_n$

758    $[\![u[\,|\,c\langle x_1, \ldots, x_n \rangle \overline{[\Gamma]}\{t/x\} \,|\, \sigma \,|\, \gamma]\!]_{\mathcal{W}} = [\![u[\,|\,c\langle x_1, \ldots, x_n \rangle \overline{[\Gamma]}\{t/x\}] \,|\, \sigma \,|\, \gamma]\!]_{\mathcal{W}}$

759    $[\![u\overline{[\Gamma]}\{t/x\} \,|\, \sigma'' \,|\, \gamma]\!]_{\mathcal{W}} \overset{\text{I.H.}}{=} [\![u\overline{[\Gamma]} \,|\, \sigma''' \,|\, \gamma]\!]_{\mathcal{W}} = [\![u[\,|\,c\langle x_1, \ldots, x_n \rangle \overline{[\Gamma]}] \,|\, \sigma' \,|\, \gamma]\!]_{\mathcal{W}}$

760    $\sigma = \sigma_1 \cup \{x_1 \mapsto M_1, \ldots, x_n \mapsto M_n\}$

761    $\sigma'' = \sigma_1 \cup \{x_1 \mapsto M_1\{\bullet/\gamma(c)\}, \ldots, x_n \mapsto M_n\{\bullet/\gamma(c)\}\}$

762    $\sigma''' = \sigma'' \cup \{x \mapsto [\![t \,|\, \sigma'' \,|\, \gamma]\!]_{\mathcal{W}}\} = \sigma'' \cup \{x \mapsto [\![t \,|\, \sigma \,|\, \gamma]\!]_{\mathcal{W}}\}$

763   since $\{x_1, \ldots, x_n\} \cap (t)_{fv} = \{\}$ ◀

764   Below we repeat Proposition 20.

765   For $N \in \Lambda$ and $t \in \Lambda_a^S$ the following properties hold

766

$$\lfloor [\![t \,|\, \sigma^{\mathcal{W}} \,|\, \gamma]\!]_{\mathcal{W}} \rfloor = [\![t \,|\, \sigma^{\Lambda} \,|\, \gamma]\!] \qquad [\![(\!(N)\!)]\!]^{\mathcal{W}} = (\!(N)\!)^{\mathcal{W}} \qquad \lfloor (\!(N)\!)^{\mathcal{W}} \rfloor = N$$

767   where $\sigma^{\Lambda}(z) = \lfloor \sigma^{\mathcal{W}}(z) \rfloor$.

768   **Proof.** We prove $\lfloor [\![u \,|\, \sigma^{\mathcal{W}} \,|\, \gamma]\!]_{\mathcal{W}} \rfloor = [\![u \,|\, \sigma^{\Lambda} \,|\, \gamma]\!]$ by induction on u.

769

770   Base Case: Variable

771    $\lfloor [\![x \,|\, \sigma^{\mathcal{W}} \,|\, \gamma]\!]_{\mathcal{W}} \rfloor = \lfloor \sigma^{\mathcal{W}}(x) \rfloor = [\![x \,|\, \sigma^{\Lambda} \,|\, \gamma]\!]$

772

773   Inductive Case: Application

774    $\lfloor [\![s\,t \,|\, \sigma^{\mathcal{W}} \,|\, \gamma]\!]_{\mathcal{W}} \rfloor = \lfloor [\![s \,|\, \sigma^{\mathcal{W}} \,|\, \gamma]\!]_{\mathcal{W}} \rfloor \lfloor [\![t \,|\, \sigma^{\mathcal{W}} \,|\, \gamma]\!]_{\mathcal{W}} \rfloor \overset{\text{I.H.}}{=} [\![s \,|\, \sigma^{\Lambda} \,|\, \gamma]\!] \, [\![t \,|\, \sigma^{\Lambda} \,|\, \gamma]\!] = [\![s\,t \,|\, \sigma^{\Lambda} \,|\, \gamma]\!]$

775

776   Inductive Case: Abstraction

777    $\lfloor [\![x\langle x \rangle.t \,|\, \sigma^{\mathcal{W}} \,|\, \gamma]\!]_{\mathcal{W}} \rfloor = \lambda x.\lfloor [\![t \,|\, \sigma^{\mathcal{W}} \,|\, \gamma]\!]_{\mathcal{W}} \rfloor \overset{\text{I.H.}}{=} \lambda x.[\![t \,|\, \sigma^{\Lambda} \,|\, \gamma]\!] = [\![x\langle x \rangle.t \,|\, \sigma^{\Lambda} \,|\, \gamma]\!]$

778

779   Inductive Case: Phantom-Abstraction

780    $\lfloor [\![c\langle x_1, \ldots, x_n \rangle.t \,|\, \sigma^{\mathcal{W}} \,|\, \gamma]\!]_{\mathcal{W}} \rfloor = \lambda c.\lfloor [\![t \,|\, \sigma_1^{\mathcal{W}} \,|\, \gamma]\!]_{\mathcal{W}} \rfloor \overset{\text{I.H.}}{=} \lambda c.[\![x \,|\, \sigma_1^{\Lambda} \,|\, \gamma]\!] = [\![c\langle x_1, \ldots, x_n \rangle.t \,|\, \sigma^{\Lambda} \,|\, \gamma]\!]$

781   where

782    $\sigma_1^{\mathcal{W}} = \sigma[x_1 \mapsto \sigma(x_1)\{c/\gamma(c)\}, \ldots, x_n \mapsto \sigma(x_n)\{c/\gamma(c)\}]$

783    $\sigma_1^{\Lambda} = \sigma[x_1 \mapsto \lfloor \sigma(x_1) \rfloor\{c/\gamma(c)\}, \ldots, x_n \mapsto \lfloor \sigma(x_n) \rfloor\{c/\gamma(c)\}]$

784

785   Inductive Case: Weakening

786    $\lfloor [\![u[\leftarrow t] \,|\, \sigma^{\mathcal{W}} \,|\, \gamma]\!]_{\mathcal{W}} \rfloor = \lfloor [\![u \,|\, \sigma^{\mathcal{W}} \,|\, \gamma]\!]_{\mathcal{W}} \rfloor \overset{\text{I.H.}}{=} [\![u \,|\, \sigma^{\Lambda} \,|\, \gamma]\!] = [\![u[\leftarrow t] \,|\, \sigma^{\Lambda} \,|\, \gamma]\!]$

787

788   Inductive Case: Sharing

789 $\lfloor [\![u[x_1, \dots, x_n \leftarrow t] \,|\, \sigma^{\mathcal{W}} \,|\, \gamma]\!]_{\mathcal{W}} \rfloor = \lfloor [\![u \,|\, \sigma_1^{\mathcal{W}} \,|\, \gamma]\!]_{\mathcal{W}} \rfloor \overset{\text{I.H.}}{=} [\![u \,|\, \sigma_1^{\Lambda} \,|\, \gamma]\!] = [\![u[x_1, \dots, x_n \leftarrow t] \,|\, \sigma^{\Lambda} \,|\, \gamma]\!]$

790 where

791 $\sigma_1^{\mathcal{W}} = \sigma^{\mathcal{W}}[x_i \mapsto [\![t \,|\, \sigma^{\mathcal{W}} \,|\, \gamma]\!]_{\mathcal{W}}]_{1 \le i \le n}$

792 $\sigma_1^{\Lambda} = \sigma^{\Lambda}[x_i \mapsto \lfloor [\![t \,|\, \sigma^{\mathcal{W}} \,|\, \gamma]\!]_{\mathcal{W}} \rfloor]_{1 \le i \le n} \overset{\text{I.H.}}{=} \sigma^{\Lambda}[x_i \mapsto [\![t \,|\, \sigma^{\Lambda} \,|\, \gamma]\!]]_{1 \le i \le n}$

793

794 Inductive Case: Distributor

795 $\lfloor [\![u[e_1\langle \vec{w_1} \rangle, \dots, e_m \langle \vec{w_m} \rangle \,|\, c\langle \vec{x} \rangle \overline{\Gamma}] \,|\, \sigma^{\mathcal{W}} \,|\, \gamma]\!]_{\mathcal{W}} \rfloor$

796

797   SubCase: $\vec{x} = c$

798 $\lfloor [\![u[e_1\langle \vec{w_1} \rangle, \dots, e_m \langle \vec{w_m} \rangle \,|\, c\langle c \rangle \overline{\Gamma}] \,|\, \sigma^{\mathcal{W}} \,|\, \gamma]\!]_{\mathcal{W}} \rfloor$

799 $= \lfloor [\![u\overline{\Gamma} \,|\, \sigma \,|\, \gamma']\!]_{\mathcal{W}} \rfloor \overset{\text{I.H.}}{=} [\![u\overline{\Gamma} \,|\, \sigma^{\Lambda} \,|\, \gamma']\!]$

800 $= [\![u[e_1\langle \vec{w_1} \rangle, \dots, e_m \langle \vec{w_m} \rangle \,|\, c\langle c \rangle \overline{\Gamma}] \,|\, \sigma^{\Lambda} \,|\, \gamma]\!]$

801

802   SubCase: $\vec{x} = x_1, \dots, x_n$

803 $\lfloor [\![u[e_1\langle \vec{w_1} \rangle, \dots, e_m \langle \vec{w_m} \rangle \,|\, c\langle x_1, \dots, x_n \rangle \overline{\Gamma}] \,|\, \sigma^{\mathcal{W}} \,|\, \gamma]\!]_{\mathcal{W}} \rfloor$

804 $\lfloor [\![u\overline{\Gamma} \,|\, \sigma_1^{\mathcal{W}} \,|\, \gamma']\!]_{\mathcal{W}} \rfloor \overset{\text{I.H.}}{=} [\![u\overline{\Gamma} \,|\, \sigma_1^{\Lambda} \,|\, \gamma']\!]$

805 $= [\![u[e_1\langle \vec{w_1} \rangle, \dots, e_m \langle \vec{w_m} \rangle \,|\, c\langle x_1, \dots, x_n \rangle \overline{\Gamma}] \,|\, \sigma^{\Lambda} \,|\, \gamma]\!]$

806 where

807 $\sigma_1^{\mathcal{W}} = \sigma[x_1 \mapsto \sigma(x_1)\{c/\gamma(c)\}, \dots, x_n \mapsto \sigma(x_n)\{c/\gamma(c)\}]$

808 $\sigma_1^{\Lambda} = \sigma[x_1 \mapsto \lfloor \sigma(x_1) \rfloor\{c/\gamma(c)\}, \dots, x_n \mapsto \lfloor \sigma(x_n) \rfloor\{c/\gamma(c)\}]$

809

810 We prove $[\![(\!| N |\!)]\!]^{\mathcal{W}} = (\!| N |\!)^{\mathcal{W}}$ by induction on N. We prove this statement by first prov-

811 ing it for closed terms.

812

813 Base Case: Variable

814 $[\![(\!| x |\!)']\!]^{\mathcal{W}} = [\![x]\!]^{\mathcal{W}} = x = (\!| x |\!)^{\mathcal{W}}$

815

816 Inductive Case: Application

817 $[\![(\!| M\,N |\!)']\!]^{\mathcal{W}} = [\![(\!| M |\!)']\!]^{\mathcal{W}} [\![(\!| N |\!)']\!]^{\mathcal{W}} \overset{\text{I.H.}}{=} (\!| M |\!)^{\mathcal{W}} (\!| N |\!)^{\mathcal{W}} = (\!| M\,N |\!)^{\mathcal{W}}$

818

819 Inductive Case: Abstraction

820 $[\![(\!| \lambda x.M |\!)']\!]^{\mathcal{W}}$

821   SubCase: $|M|_x = 0$

822 $= \lambda x.[\![(\!| M |\!)'[\leftarrow x]]\!]^{\mathcal{W}} = \lambda x.[\![(\!| M |\!)']\!]^{\mathcal{W}}[\leftarrow x] \overset{\text{I.H.}}{=} \lambda x.(\!| M |\!)^{\mathcal{W}}[\leftarrow x] = (\!| \lambda x.M |\!)^{\mathcal{W}}$

823

824   SubCase: $|M|_x = 1$

825 $= \lambda x.[\![(\!| M |\!)']\!]^{\mathcal{W}} \overset{\text{I.H.}}{=} \lambda x.(\!| M |\!)^{\mathcal{W}} = (\!| \lambda x.M |\!)^{\mathcal{W}}$

826

827   SubCase: $|M|_x = n > 1$

828 $= [\![(\!| M\frac{n}{x} |\!)'[x^1, \dots, x^n \leftarrow x]]\!]^{\mathcal{W}} = [\![(\!| M\frac{n}{x} |\!)' \,|\, \sigma \,|\, I]\!]_{\mathcal{W}} \overset{\text{prop ??}}{=} [\![(\!| M\frac{n}{x} |\!)']\!]^{\mathcal{W}}\{x/x_i\}_{1 \le i \le n}$

829 $\overset{\text{I.H.}}{=} (\!| M\frac{n}{x} |\!)^{\mathcal{W}}\{x/x_i\}_{1 \le i \le n} = (\!| M |\!)^{\mathcal{W}}$

830

831 Now that we have proven is works for closed terms, we can show the statement $[\![(\!| N |\!)]\!]^{\mathcal{W}} =$

832 $(\!| N |\!)^{\mathcal{W}}$ holds

833

834 $[\![(\!| N |\!)]\!]^{\mathcal{W}} = [\![(\!| N\frac{n_1}{x_1} \dots \frac{n_k}{x_k} |\!)'[x_1^1, \dots, x_1^{n_1} \leftarrow x_1] \dots [x_k^1, \dots, x_k^{n_k} \leftarrow x_k]]\!]^{\mathcal{W}}$

835 $\overset{\text{prop ??}}{=} [\![(\!| N\frac{n_1}{x_1} \dots \frac{n_k}{x_k} |\!)']\!]^{\mathcal{W}}\{x_i/x_i^j\}_{1 \le i \le k, 1 \le j \le n_i} = (\!| N\frac{n_1}{x_1} \dots \frac{n_k}{x_k} |\!)^{\mathcal{W}}\{x_i/x_i^j\}_{1 \le i \le k, 1 \le j \le n_i} = (\!| N |\!)^{\mathcal{W}}$ ◀