ROSAEC center
Research On Software Analysis for Error-free Computing
소프트웨어 무결점 연구센터 NRF ERC

ROSAEC MEMO
ROSAEC-2011-012
September 6, 2011

CPS Transformation of Lisp-Like Multi-Staged Languages

Ludovic Patey
École Normale Supérieure
ludovic.patey@ens.fr

Kwangkeun Yi
Seoul National University
kwang@ropas.snu.ac.kr

September 6, 2011

Abstract

We present a Continuation Passing Style transformation of a multi-staged calculus and prove that Plotkin's main theorems, ie. Indifference, Simulation and Translation theorems still hold. This transformation can serve as a basis of CPS optimisation-based multi-staged compilers.

In this paper, we formally present a core Multi-Staged Calculus with its call-by-value and call-by-name operational semantics, a corresponding Multi-Staged Theory and a CPS transformation with proofs of its correctness.

1 Multi-Staged Calculus

In this section, we give a formal definition of the core multi-staged calculus λ_S with its syntax and small-step operational semantics.

1.1 Syntax

$Expr$	$e ::=$	$x \mid \lambda x.e \mid e\ e \mid \textbf{box}\ e \mid \textbf{unbox}\ e \mid \textbf{run}\ e$
$Value^0$	$v^0 ::=$	$x \mid \lambda x.e \mid \textbf{box}\ v^1$
$Value^n (n > 0)$	$v^n ::=$	$x \mid \lambda x.v^n \mid v^n\ v^n \mid \textbf{box}\ v^{n+1} \mid \textbf{unbox}\ v^{n-1}\ (n > 1) \mid \textbf{run}\ v^n$

This language handles pure lambda calculus, ie. variables (x), abstraction ($\lambda x.e$) and application ($e\ e$), and staging constructs: **box** e encapsulates an *open code template* e which is considered in another *stage*. **unbox** e serves as template composition and can be seen as providing a "hole" in the code template. It is then well-defined only enclosed by a **box** operator. **run** e executes a *closed* code template in the current environment. Staging constructs can be arbitrarily nested.

We define inductively the set of free variables (Fig.1) and bounded variables (Fig.2) of an expression. A *closed expression* e is an expression such that $FV^0(e) = \varnothing$.

1.2 Operational Semantics

We provide small-step call-by-value (Fig.3) and call-by-name (Fig.4) operational semantics with substitution (Fig.5). A statement of the form $e \xrightarrow{n}_{\mathtt{v}} e'$ (resp. $e \xrightarrow{n}_{\mathtt{n}} e'$) means "$e$ *expression evaluates to* e' *at stage* n *with a call-by-value (resp. call-by-name) strategy*".

Free Variables

$$
\begin{array}{llll}
FV^0(x) & = \{x\} & FV^n(e_1\ e_2) & = FV^n(e_1) \cup FV^n(e_2) \\
FV^{n+1}(x) & = \varnothing & FV^n(\mathbf{box}\ e) & = FV^{n+1}(e) \\
FV^0(\lambda x.e) & = FV^0(e) \setminus \{x\} & FV^{n+1}(\mathbf{unbox}\ e) & = FV^n(e) \\
FV^{n+1}(\lambda x.e) & = FV^{n+1}(e) & FV^n(\mathbf{run}\ e) & = FV^n(e)
\end{array}
$$

Figure 1: Free Variables of an Expression

Bounded Variables

$$
\begin{array}{llll}
BV^n(x) & = \varnothing & BV^n(e_1\ e_2) & = BV^n(e_1) \cup BV^n(e_2) \\
BV^0(\lambda x.e) & = BV^0(e) \cup \{x\} & BV^n(\mathbf{box}\ e) & = BV^{n+1}(e) \\
BV^{n+1}(\lambda x.e) & = BV^{n+1}(e) & BV^{n+1}(\mathbf{unbox}\ e) & = BV^n(e) \\
 & & BV^n(\mathbf{run}\ e) & = BV^n(e)
\end{array}
$$

Figure 2: Bounded Variables of an Expression

Operational Semantics ($n \geq 0$)

$$
(\mathtt{ABS_v})\quad \frac{e \xrightarrow{n+1}_{\mathrm{v}} e'}{\lambda x.e \xrightarrow{n+1}_{\mathrm{v}} \lambda x.e'}
$$

$$
(\mathtt{APP_v})\quad \frac{e_1 \xrightarrow{n}_{\mathrm{v}} e_1'}{e_1\ e_2 \xrightarrow{n}_{\mathrm{v}} e_1'\ e_2}
$$

$$
\frac{e \xrightarrow{n}_{\mathrm{v}} e' \qquad v \in Value^n}{v\ e \xrightarrow{n}_{\mathrm{v}} v\ e'}
$$

$$
(\lambda x.e)\, v \xrightarrow{0}_{\mathrm{v}} [x \overset{0}{\mapsto} v]e
$$

$$
(\mathtt{BOX_v})\quad \frac{e \xrightarrow{n+1}_{\mathrm{v}} e'}{\mathbf{box}\ e \xrightarrow{n}_{\mathrm{v}} \mathbf{box}\ e'}
$$

$$
(\mathtt{UNB_v})\quad \frac{e \xrightarrow{n}_{\mathrm{v}} e'}{\mathbf{unbox}\ e \xrightarrow{n+1}_{\mathrm{v}} \mathbf{unbox}\ e'}
$$

$$
\frac{v \in Value^1}{\mathbf{unbox}\ \ (\mathbf{box}\ v) \xrightarrow{1}_{\mathrm{v}} v}
$$

$$
(\mathtt{RUN_v})\quad \frac{e \xrightarrow{n}_{\mathrm{v}} e'}{\mathbf{run}\ e \xrightarrow{n}_{\mathrm{v}} \mathbf{run}\ e'}
$$

$$
\frac{v \in Value^1 \qquad FV_0(v) = \varnothing}{\mathbf{run}\ \ (\mathbf{box}\ v) \xrightarrow{0}_{\mathrm{v}} v}
$$

Figure 3: Call-by-value Operational Semantics of λ_S

Operational Semantics ($n \geq 0$)

$$(\mathtt{ABS_n}) \quad \frac{e \xrightarrow{n+1}_{\mathtt{n}} e'}{\lambda x.e \xrightarrow{n+1}_{\mathtt{n}} \lambda x.e'} \qquad (\mathtt{BOX_n}) \quad \frac{e \xrightarrow{n+1}_{\mathtt{n}} e'}{\mathbf{box}\ e \xrightarrow{n}_{\mathtt{n}} \mathbf{box}\ e'}$$

$$(\mathtt{APP_n}) \quad \frac{e_1 \xrightarrow{n}_{\mathtt{n}} e_1'}{e_1\ e_2 \xrightarrow{n}_{\mathtt{n}} e_1'\ e_2} \qquad (\mathtt{UNB_n}) \quad \frac{e \xrightarrow{n+1}_{\mathtt{n}} e'}{\mathbf{unbox}\ e \xrightarrow{n+2}_{\mathtt{n}} \mathbf{unbox}\ e'}$$

$$\frac{e \xrightarrow{n+1}_{\mathtt{n}} e' \quad v \in Value^n}{v\ e \xrightarrow{n+1}_{\mathtt{v}} v\ e'} \qquad \frac{v \in Value^1}{\mathbf{unbox}\ (\mathbf{box}\ v) \xrightarrow{1}_{\mathtt{n}} v}$$

$$\frac{e \xrightarrow{1}_{\mathtt{n}} e' \quad v \in Value^0}{v\ (\mathbf{box}\ e) \xrightarrow{0}_{\mathtt{v}} v\ (\mathbf{box}\ e')} \qquad (\mathtt{RUN_n}) \quad \frac{e \xrightarrow{n+1}_{\mathtt{n}} e'}{\mathbf{run}\ e \xrightarrow{n+1}_{\mathtt{n}} \mathbf{run}\ e'}$$

$$(\lambda x.e_1)\, e_2 \xrightarrow{0}_{\mathtt{n}} [x \overset{0}{\mapsto} e_2]e_1 \qquad \frac{v \in Value^1 \quad FV_0(v) = \varnothing}{\mathbf{run}\ (\mathbf{box}\ v) \xrightarrow{0}_{\mathtt{n}} v}$$

Figure 4: Call-by-name Operational Semantics of λ_S

Notation 1. *Trough this paper we will introduce a few relations subscripted by* $_{\mathtt{v}}$ *or* $_{\mathtt{n}}$*, which stands respectively for Call By Value and Call By Name. We will also use in some results previously defined relations without subscript, meaning that it is valid with bothCBV and CBN strategies (eg. Lem.5).*

Finally when designing a same extension for relations having CBV and CBN definitions, we will consider that subscript is induced (eg. Fig.18).

Notice that only closed template code can be executed with **run** operator, and that unbox parameter is evaluated before code template. An open template code can be closed by unhygienic composition:

$$\begin{aligned}(\lambda x.\mathbf{box}\ (\lambda y.\mathbf{unbox}\ x))\,\mathbf{box}\ y &\xrightarrow{0}_{\mathtt{v}} \mathbf{box}\ (\lambda y.\mathbf{unbox}\ (\mathbf{box}\ y)) \\ &\xrightarrow{0}_{\mathtt{v}} \mathbf{box}\ (\lambda y.y)\end{aligned}$$

Substitution

$$[x \overset{0}{\mapsto} v]y = \begin{cases} v & \text{if } x = y \\ y & \text{otherwise} \end{cases}$$

$$[x \overset{n+1}{\mapsto} v]y = y$$

$$[x \overset{0}{\mapsto} v]\lambda y.e = \begin{cases} \lambda y.e \text{ if } x = y \\ \lambda y.[x \overset{0}{\mapsto} v]e \text{ o.w.} \end{cases}$$

$$[x \overset{n+1}{\mapsto} v]\lambda y.e = \lambda y.[x \overset{n+1}{\mapsto} v]e$$

$$[x \overset{n}{\mapsto} v]e_1\ e_2 = [x \overset{n}{\mapsto} v]e_1\ [x \overset{n}{\mapsto} v]e_2$$

$$[x \overset{n}{\mapsto} v]\mathbf{box}\ e = \mathbf{box}\ [x \overset{n+1}{\mapsto} v]e$$

$$[x \overset{n+1}{\mapsto} v]\mathbf{unbox}\ e = \mathbf{unbox}\ [x \overset{n}{\mapsto} v]e$$

$$[x \overset{n}{\mapsto} v]\mathbf{run}\ e = \mathbf{run}\ [x \overset{n}{\mapsto} v]e$$

Figure 5: Substitution over λ_S

Notation 2. *Being given a relation, we define its transitive closure, notated with the superscript* $^+$ *and its reflexive transitive closure with superscript* **. For example, the transitive reflexive closure of* $\xrightarrow{n}_{\mathtt{v}}$ *is* $\xrightarrow{n^*}_{\mathtt{v}}$.

We also define evaluation functions (Fig.7).

Depth Function

$$
\begin{array}{lll}
depth(x) & = 0 & \\
depth(e_1\ e_2) & = max(depth(e_1), depth(e_2)) & \\
depth(\textbf{box}\ e) & = \begin{cases} depth(e) - 1 & \text{if } depth(e) > 0 \\ 0 & \text{otherwise} \end{cases} &
\end{array}
\qquad
\begin{array}{ll}
depth(\lambda x.e) & = depth(e) \\
depth(\textbf{unbox}\ e) & = depth(e) + 1 \\
depth(\textbf{run}\ e) & = depth(e)
\end{array}
$$

Figure 6: Depth Function of λ_S

Evaluation Functions

$$
\frac{e \in Expr \quad v \in Value^0 \quad e \xrightarrow{0^\star}_{\mathtt{v}} v}{eval_{\mathtt{v}}(e) = v}
\qquad
\frac{e \in Expr \quad v \in Value^0 \quad e \xrightarrow{0^\star}_{\mathtt{n}} v}{eval_{\mathtt{n}}(e) = v}
$$

Figure 7: Evaluation Functions for λ_S

1.3 Discussion about Call-by-name Operational Semantics

Intuitively, in a CBN strategy, a function call is directly applied without reducing its arguments. Here we consider **unbox** e and **run** e as function calls. Hence we have removed 0-staged reductions steps in our CBN operational semantics. As our functions **unbox** and **run** are undefined for terms others than **box**, if there we give them an unreduced term as argument, the execution sticks. Let's see a few examples about differences between call-by-name and call-by-value operational semantics.

The first one is substitution of arbitrary expressions instead of evaluating arguments to value before making the application:

$$
\begin{aligned}
(\lambda x.x)\ ((\lambda y.y)\ z) &\xrightarrow{0}_{\mathtt{v}} (\lambda x.x)\ z \\
&\xrightarrow{0}_{\mathtt{n}} (\lambda y.y)\ z
\end{aligned}
$$

Another difference concerns operators applications at stage 0, like **run** and **unbox**. In Call-by-value operational semantics, an expression is reduced to a value before applying the operator while in Call-by-name evaluation order, if the expression isn't a value, then the evaluator sticks.

$$
\begin{aligned}
\textbf{run}\ ((\lambda x.x)\ (\textbf{box}\ \lambda y.y)) &\xrightarrow{0}_{\mathtt{v}} \textbf{run}\ (\textbf{box}\ \lambda y.y) \xrightarrow{0}_{\mathtt{v}} \lambda y.y \\
&\xrightarrow{0}_{\mathtt{n}} ERROR
\end{aligned}
$$

Because of the absence of 0-staged reductions for arguments of an **unbox**, any n-staged, $n > 0$, corresponds to a code composition, ie. $(\mathtt{UNB}_{\mathtt{n}})[2]$ rule.

Until now, our interpretation of CBN for multi-staged calculus is quite natural. However, in order to obtain a valid CPS transformation, it has been necessary to add an extra and unnatural reduction rule in the case of application:

$$
\frac{e \xrightarrow{1}_{\mathtt{n}} e' \quad v \in Value^0}{v\ (\textbf{box}\ e) \xrightarrow{0}_{\mathtt{v}} v\ (\textbf{box}\ e')}
$$

As explained, 1-staged reductions correspond to a $\textbf{unbox}\ (\textbf{box}\ v) \xrightarrow{n}_{\mathtt{n}} v$ reduction, so the call-by-name behaviour remains at 0-stage in the sense that there is no 0-stage reduction as argument of a function call. It is as well interesting to notice that there is no 1-stage reduction as argument of **unbox** and **run**, ie the following code will stick in a CBN evaluation strategy:

$$\textbf{run}\ (\textbf{box}\ (\textbf{unbox}\ (\textbf{box}\ e)))$$

whereas the following code reduces to $(\lambda x.e_1)\ \textbf{box}\ e_2$ still with CBN strategy:

$$(\lambda x.e_1)\ (\textbf{box}\ (\textbf{unbox}\ (\textbf{box}\ e_2)))$$

Remaining of this section will be dedicated to prove that there is no "natural" CPS-transformation without this extra rule.

Let's consider the following expression:

$$(\lambda x.x)\ \textbf{box}\ (\textbf{unbox}\ (\textbf{box}\ x))$$

In a call-by-value reduction strategy, it evaluates to $\textbf{box}\ x$. Now let's consider a CPS transformation $[\![]\!] : Expr \to Expr$ having the simulation property. Then we have

$$[\![(\lambda x.x)\ \textbf{box}\ (\textbf{unbox}\ (\textbf{box}\ x))]\!] : k \xrightarrow{0^+} [\![(\lambda x.x)\ \textbf{box}\ x]\!] : k \xrightarrow{0^+} [\![\textbf{box}\ x]\!] : k$$

Hence $[\![(\lambda x.x)\ \textbf{box}\ x]\!] : k$ is not a value, but if the first sequence of reductions involves a $\textbf{unbox}\ (\textbf{box}\ v) \xrightarrow{n}_{\mathtt{n}} v$ reduction, let $e_1, e_2 \in Expr$ such that

$$[\![(\lambda x.x)\ \textbf{box}\ (\textbf{unbox}\ (\textbf{box}\ x))]\!] : k \xrightarrow{0^\star} e_1 \xrightarrow{0} e_2 \xrightarrow{0^\star} [\![(\lambda x.x)\ \textbf{box}\ x]\!] : k$$

and the rules applied between e_1 and e_2 involves $(\mathtt{UNB_n})[2]$. As the reduction between e_1 and e_2 is a 0-stage reduction involving $(\mathtt{UNB_n})[2]$, we have one of the following possible root rules applied:

- $(\mathtt{BOX_n})$. In this case $e_i = (\textbf{box}\ e_i')$ for $i \in \{1, 2\}$.

  In this case, because of indifference theorem, reduction rules involve only rules common to CBV and CBN operational semantics, then $e_2 = \textbf{box}\ e_2'$ with $e_2' \in Value^1$ which is a contradiction with the fact that $e_2 \xrightarrow{0^\star} [\![(\lambda x.x)\ \textbf{box}\ x]\!] : k$ which is not a value.

- $(\mathtt{APP_n})[1]$. In this case $e_1 = e_a\ e_b$, $e_2 = e_a'\ e_b$ with $e_a \overset{0}{\rightsquigarrow} e_a'$. By induction, we arrive to a case where $e_1 = \left(\left((\textbf{box}\ e_1')\ e_b^1\right) \dots e_b^n\right)$. By the same reasoning as previous case, $e_2 = \left(\left((\textbf{box}\ e_2')\ e_c^1\right) \dots e_c^n\right)$ with $e_2' \in Value^1$. Then evaluation sticks, which is as well a contradiction with $e_2 \xrightarrow{0^\star} [\![(\lambda x.x)\ \textbf{box}\ x]\!] : k$

We conclude from previous observation that there doesn't exist a CPS transformation which verifies together Indifference property, Simulation property and uses code composition in its destination language if we don't have our unatural extra rule. Intuitively, it comes from the fact that when code composition occurs, there is no way to apply continuation afterwards: we obtain either a value or evaluation sticks.

Another approach used in Plotkin's paper consist of changing **box** semantics in destination language as well as Plotkin changed his Constapply function.

2 Multi-staged Theory λ^S

In this section we introduce $\lambda^S_{\mathtt{v}}$ and $\lambda^S_{\mathtt{n}}$ theories and state the Church-Rosser theorem for equality. Proofs will be given in a further section.

Definition 1. *$\lambda^S_{\mathtt{v}}$ is a theory formulated in the following language:*

Alphabet:	$a,\ b,\ c\ \ldots$	*variables*
	$\lambda,\ (,\ )$	*improper symbols*
	$\boldsymbol{box}\ ,\ \boldsymbol{unbox}\ ,\ \boldsymbol{run}$	*operator symbols*
	$=^n_{\mathtt{v}}$	*equality (Fig.8)*
	$\geq^n_{\mathtt{v}}$	*reduction*
	$\geq^n_1$	*parallel reduction (Fig.24)*

Terms: *Terms are defined as in λ_S calculus*

Formulas: *If e_1, e_2 are terms, then $e_1 =^n e_2$, $e_1 \geq^n e_2$ and $e_1 \geq^n_1 e_2$ are formulas*

Definition 2. *$\lambda^S_{\mathtt{n}}$ is a theory formulated in the following language:*

Alphabet:	$a,\ b,\ c\ \ldots$	*variables*
	$\lambda,\ (,\ )$	*improper symbols*
	$\boldsymbol{box}\ ,\ \boldsymbol{unbox}\ ,\ \boldsymbol{run}$	*operator symbols*
	$=^n_{\mathtt{n}}$	*equality (Fig.9)*
	$\geq^n_{\mathtt{n}}$	*reduction*

Terms: *Terms are defined as in λ_S calculus*

Formulas: *If e_1, e_2 are terms, then $e_1 =^n e_2$, $e_1 \geq^n e_2$ and $e_1 \geq^n_1 e_2$ are formulas*

Equality $(n \geq 0)$

$$(\mathtt{EREF_v}) \quad e =^n_{\mathtt{v}} e$$

$$(\mathtt{ESYM_v}) \quad \frac{e =^n_{\mathtt{v}} e'}{e' =^n_{\mathtt{v}} e}$$

$$(\mathtt{ETRA_v}) \quad \frac{e =^n_{\mathtt{v}} e' \qquad e' =^n_{\mathtt{v}} e''}{e' =^n_{\mathtt{v}} e''}$$

$$(\mathtt{EAPP_v}) \quad \frac{e_1 =^n_{\mathtt{v}} e_1'}{e_1\ e_2 =^n_{\mathtt{v}} e_1'\ e_2} \qquad \frac{e_2 =^n_{\mathtt{v}} e_2'}{e_1\ e_2 =^n_{\mathtt{v}} e_1\ e_2'}$$

$$(\mathtt{EALP_v}) \quad (\lambda x.e)\, v =^0_{\mathtt{v}} [x \overset{0}{\mapsto} v]e \qquad \lambda x.e =^0_{\mathtt{v}} \lambda y.[x \overset{0}{\mapsto} y]e$$

$$(\mathtt{EABS_v}) \quad \frac{e =^n_{\mathtt{v}} e'}{\lambda x.e =^n_{\mathtt{v}} \lambda x.e'}$$

$$(\mathtt{EBOX_v}) \quad \frac{e =^{n+1}_{\mathtt{v}} e'}{\mathbf{box}\ e =^n_{\mathtt{v}} \mathbf{box}\ e'}$$

$$(\mathtt{EUNB_v}) \quad \frac{e =^n_{\mathtt{v}} e'}{\mathbf{unbox}\ e =^{n+1}_{\mathtt{v}} \mathbf{unbox}\ e'} \qquad \frac{v \in Value^1}{\mathbf{unbox}\ (\mathbf{box}\ v) =^1_{\mathtt{v}} v}$$

$$(\mathtt{ERUN_v}) \quad \frac{e =^n_{\mathtt{v}} e'}{\mathbf{run}\ e =^n_{\mathtt{v}} \mathbf{run}\ e'} \qquad \frac{v \in Value^1 \qquad FV_0(v) = \varnothing}{\mathbf{run}\ (\mathbf{box}\ v) =^0_{\mathtt{v}} v}$$

Figure 8: Call-by-value Equality for λ_S

Notation 3. *Reduction relation $\geq^n_{\mathtt{v}}$ (resp. $\geq^n_{\mathtt{n}}$) is defined from equality $=^n_{\mathtt{v}}$ (resp. $=^n_{\mathtt{n}}$) without the* (ESYM) *rule.*

Theorem 1 (Church-Rosser)**.** *Let $e_1, e_2, e_3 \in Expr$ such that $e_1 =^n_{\mathtt{v}} e_2$.*
$\exists e_3 \in Expr$ such that $e_1 \geq^n_{\mathtt{v}} e_3$ and $e_2 \geq^n_{\mathtt{v}} e_3$

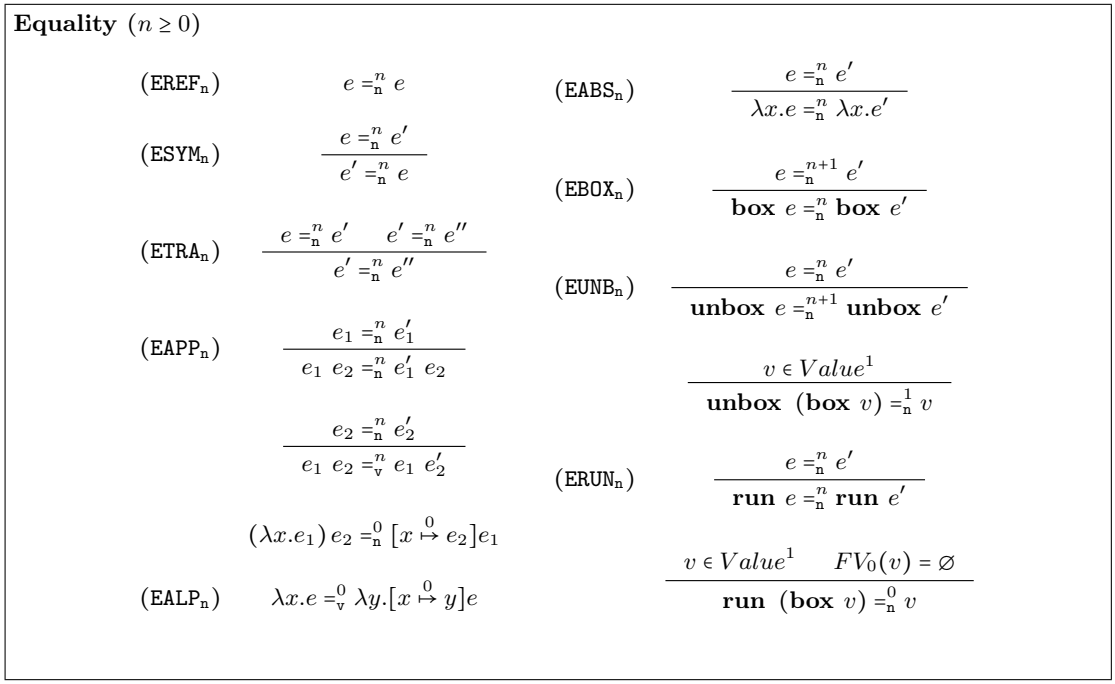

Figure 9: Call-by-name Equality for λ_S

The proof of this theorem is given in section 4.1 and follows a proof scheme à la Martin-Löf, ie. defining parallel reductions, proving local confluence and deducing from it global confluence. We will present it under the version of Barendregt [1]

3 CPS Transformation

In this section we give a CPS transformation of the λ_S calculus (Fig.10) and prove its main properties, ie. Indifference, Simulation and Translation theorems. This transformation is an extension of Fischer [2] & Plotkin [4] version to multi-staged calculus.

Assumption 1. *All new variables are unique in the program.* (A1)

We can respect this constraint by renaming variables when there might be some conflicts.

The CPS Transformation defined in Fig.10 is a function of type $Expr \to Expr \times Context\ Stack$. We define the regular transformation $[\![]\!] : Expr \to Expr$ from the previous one as follow:

Definition 3 (CPS Transformation)**.** *Let $e \in Expr$ such that $e \mapsto (\underline{e}, \bot)$.*

$$[\![e]\!] =_{def} \underline{e}$$

3.1 Input Set of CPS Transformation

Our CPS transformation is defined only when resulting context stack is empty. Lem.1 gives a characterization of this subset as the set of expressions of depth 0. For example $[\![\textbf{box}\ (\textbf{unbox}\ x)]\!]$ is defined, but not $[\![\textbf{box}\ (\textbf{unbox}\ (\textbf{unbox}\ x))]\!]$.

Our validity results will be proved on closed expressions of depth 0. This restriction is quite natural as we can think of a program as a closed expression, and because having a code composition operator outside a box would have no meaning. We can then reasonably assume that any futur type system would reject expressions of higher depth.

Let define the following sets:

Definitions

$$\begin{array}{ll} \textit{Context} & \kappa ::= (e\ \lambda h.[\cdot]) \mid (e\ \lambda h.\kappa) \\ \textit{Context stack} & K ::= \bot \mid K, \kappa \end{array}$$

Transformation

$$\texttt{(TVAR)} \quad x \mapsto (\lambda k.\,(k\ x), \bot)$$

$$\texttt{(TABS)} \quad \frac{e \mapsto (\underline{e}, K)}{\lambda x.e \mapsto (\lambda k.\,(k\ \lambda x.\underline{e}), K)}$$

$$\texttt{(TAPP)} \quad \frac{e_1 \mapsto (\underline{e_1}, K_1) \qquad e_2 \mapsto (\underline{e_2}, K_2)}{e_1\ e_2 \mapsto \left(\lambda k.\underline{e_1}\ \left(\lambda m.\underline{e_2}\ \left(\lambda n.\,((m\ n)\ k)\right)\right), K_1 \bowtie K_2\right)}$$

$$\texttt{(TBOX)} \quad \frac{e \mapsto (\underline{e}, (K, \kappa))}{\mathbf{box}\ e \mapsto (\lambda k.\kappa[k\ (\mathbf{box}\ \underline{e})], K)}$$

$$\frac{e \mapsto (\underline{e}, \bot)}{\mathbf{box}\ e \mapsto (\lambda k.k\ (\mathbf{box}\ \underline{e}), \bot)}$$

$$\texttt{(TUNB)} \quad \frac{e \mapsto (\underline{e}, K)}{\mathbf{unbox}\ e \mapsto (\mathbf{unbox}\ h, (K, (\underline{e}\ \lambda h.[\cdot])))}$$

$$\texttt{(TRUN)} \quad \frac{e \mapsto (\underline{e}, K)}{\mathbf{run}\ e \mapsto (\lambda k.\,(\underline{e}\ (\lambda m.\,((\mathbf{run}\ m)\ k))), K)}$$

Context Stack Merge Operator

$$\begin{array}{rcl} \bot \bowtie K & = & K \\ K \bowtie \bot & = & K \\ (K_1, \kappa_1) \bowtie (K_2, \kappa_2) & = & (K_1 \bowtie K_2), (\kappa_1[\kappa_2]) \end{array}$$

Figure 10: CPS Transformation of λ_S

- *Expr* is the set of all expressions.
- *WT Expr* is the set of well-typed expressions with respect to Choi & al. type system [3]
- *CPS Expr* the domain of $[\![]\!]$, ie. the set of all expressions of depth 0.
- *Eval Expr* the domain of $\mathbf{eval}_{\mathtt{v}}$, ie. the set of expressions reducing to a value using a call-by-value strategy.

$$WT\ Expr \subsetneq CPS\ Expr \subsetneq Expr$$

It seems difficult to extend our validity result to the set of all expressions but because all type systems are likely to accept a strict subset of *CPS Expr*, this challenge is not motivated. Fig.11 shows relations between sets.

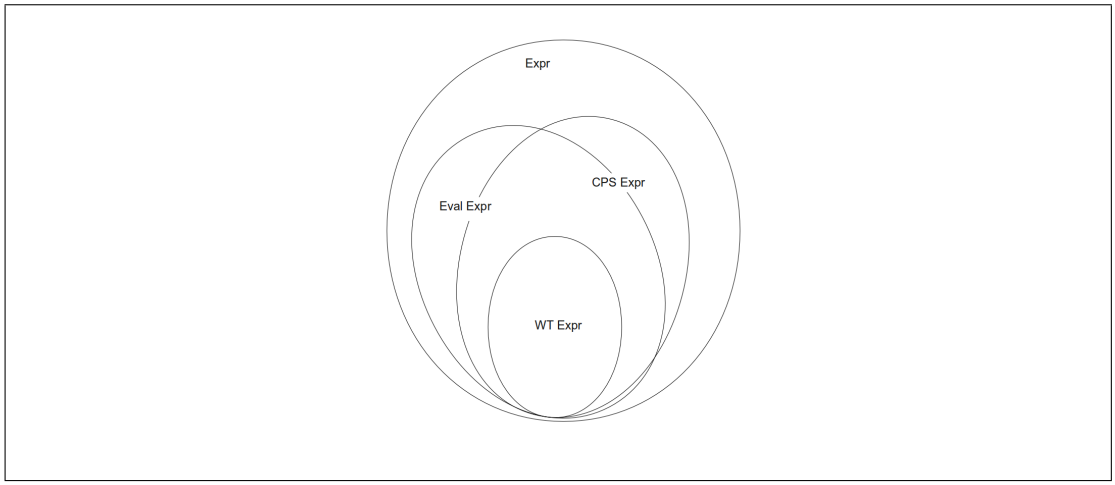

Figure 11: Diagram of Input Sets

We can define Plotkin's Ψ function sending values to values (Fig.12).

Ψ Function

$$\begin{array}{ll} \Psi(x) & = x \\ \Psi(\lambda x.e) & = \lambda x.\underline{e} \\ \Psi(\mathbf{box}\ e) & = \mathbf{box}\ \underline{e} \end{array}$$

Figure 12: Plotkin's Ψ Function

We also define the *length* function over context stacks (Fig.13).

Length Function

$$\begin{array}{c} length(\bot) = 0 \\ length(K, \kappa) = length(K) + 1 \end{array}$$

Figure 13: Length Function over Context Stacks

Lemma 1 (Expression Depth is Context Stack Length)**.** *If* $e \mapsto (\underline{e}, K)$ *then* $depth(e) = length(K)$.

Lemma 2 (Value Preservation wrt. CPS Transformation)**.**

$$\frac{v \in Value^0 \qquad depth(v) = 0}{v \mapsto (\lambda k.k\ \Psi(v), \bot)}$$

We need to extend substitution to contexts and context stacks (Fig.14) to prove our substitution preservation wrt. CPS Transformation.

Substitution

$$\begin{array}{lll}
[x \overset{n}{\mapsto} v]\bot & = \bot & \\
[x \overset{n}{\mapsto} v]\,(K, \kappa) & = [x \overset{m}{\mapsto} v]K, [x \overset{n}{\mapsto} v]\kappa & m = max(n-1, 0) \\
[x \overset{0}{\mapsto} v]\,(e\ \lambda h.\kappa) & = \begin{cases} [x \overset{0}{\mapsto} v]e\ \lambda h.\kappa & \text{if } x = h \\ [x \overset{0}{\mapsto} v]e\ \lambda h.[x \overset{0}{\mapsto} v]\kappa & \text{otherwise} \end{cases} & \\
[x \overset{n+1}{\mapsto} v]\,(e\ \lambda h.\kappa) & = [x \overset{n+1}{\mapsto} v]e\ \lambda h.[x \overset{n+1}{\mapsto} v]\kappa & \\
[x \overset{n}{\mapsto} v]\,(e\ \lambda h.[\cdot]) & = [x \overset{n}{\mapsto} v]e\ \lambda h.[\cdot] &
\end{array}$$

Figure 14: Substitution over Contexts and Context Stacks

Lemma 3 (Substitution Preservation wrt. CPS Transformation)**.**

$$\frac{v \in Value^0 \qquad v \mapsto (\underline{v}, \bot) \qquad e \mapsto (\underline{e}, K) \qquad length(K) = n \qquad m = max(n-1, 0)}{[x \overset{n}{\mapsto} v]e \mapsto \left([x \overset{n}{\mapsto} \Psi(v)]\underline{e}, [x \overset{m}{\mapsto} \Psi(v)]K\right)}$$

For our Free Variable Preservation lemma, we need to extend Free Variables to Context Stacks (Fig.15) and define Context Variables (Fig.16). Intuitively, Context Variables correspond to variables binded by contexts, eg. h in $(\underline{e}\ \lambda h.[])$.

Free Variables

$$\begin{array}{lll}
FV^n(\bot) & = \varnothing & \\
FV^n\,(K, \kappa) & = FV^m(K) \cup FV^n(\kappa) & m = max(n-1, 0) \\
FV^0\,(e\ \lambda h.\kappa) & = FV^0(e) \cup (FV^0(\kappa) \setminus \{h\}) & \\
FV^{n+1}\,(e\ \lambda h.\kappa) & = FV^{n+1}(e) \cup FV^{n+1}(\kappa) & \\
FV^n\,(e\ \lambda h.[\cdot]) & = FV^n(e) &
\end{array}$$

Figure 15: Free Variables over Contexts and Context Stacks

Context Variables

$$\begin{array}{lll}
CV^n(\bot) & = \varnothing & \\
CV^n\,(K, \kappa) & = CV^m(K) \cup CV^n(\kappa) & m = max(n-1, 0) \\
CV^0\,(e\ \lambda h.\kappa) & = \{h\} \cup CV^0(\kappa) & \\
CV^{n+1}\,(e\ \lambda h.\kappa) & = CV^{n+1}(\kappa) & \\
CV^0\,(e\ \lambda h.[\cdot]) & = \{h\} & \\
CV^{n+1}\,(e\ \lambda h.[\cdot]) & = \varnothing &
\end{array}$$

Figure 16: Context Variables over Contexts and Context Stacks

Lemma 4 (Free Variables Preservation wrt. CPS Transformation)**.**

$$\frac{e \in Expr \qquad depth(e) = n \qquad e \mapsto (\underline{e}, K) \qquad m = max(n-1, 0)}{CV^m(K) \cup FV^n(e) = FV^n(\underline{e}) \cup FV^m(K)}$$

Corollary 1 (Free Variables Preservation at Stage 0 wrt. CPS Transformation)**.**

$$\frac{e \in Expr \qquad depth(e) = 0 \qquad e \mapsto (\underline{e}, K)}{FV^0(e) = FV^0(\underline{e})}$$

Proof. As $depth(e) = 0$, by Lem.1 $K = \bot$, and then $CV^0(\bot) = FV^0(\bot) = \varnothing$ and apply Lem.4. □

Theorem 2 (Indifference)**.** *Let $e \in ClosedExpr$ such that $depth(e) = 0$.*
$eval_{\mathtt{n}}(\underline{e}\ \lambda x.x) = eval_{\mathtt{v}}(\underline{e}\ \lambda x.x)$

Proof. This theorem uses the same proof as Simulation's one, noticing that each reduction of translated expression is in the common subset of CBV and CBN operational semantics. See section 4.2 for more details. □

Theorem 3 (Simulation)**.** *Let $e \in ClosedExpr$ such that $depth(e) = 0$.*
$\Psi(eval_{\mathtt{v}}(e)) = eval_{\mathtt{n}}(\underline{e}\ \lambda x.x)$

Proof. Like Fischer & Plotkin's version, our CPS Transformation generates administrative redexes and we don't have a direct simulation. We reuse then the *Colon Translation* which removes administrative redexes and ensures diagram commutation:

Counter-example			Simulation Property		
			e	$\xrightarrow{0}_{\mathtt{v}}$	e'
$(\lambda x.x)\ y$	$\xrightarrow{0}_{\mathtt{v}}$	y	$\downarrow$		$\downarrow$
$\downarrow$		$\downarrow$	$\underline{e}\ k$	$\not\xrightarrow{0}_{\mathtt{v}}$	$\underline{e'}\ k$
$\underline{(\lambda x.x)\ y}\ k$	$\not\xrightarrow{0}_{\mathtt{v}}$	$\underline{y}\ k$	$\downarrow$		$\downarrow$
			$e : k$	$\xrightarrow{0}$	$e' : k$

We need to extend Colon Translation to staged calculus and contexts. It is then of type $(Closed\ Expr \cup Context) \times Expr \to Expr$ (Fig.17).

We first prove that $\underline{e}\ k \xrightarrow{0} e : k$ (Lem.5), then that if $e \xrightarrow{0}_{\mathtt{v}} e'$ then $e : k \xrightarrow{0^+} e' : k$ (Lem.6). And to conclude the proof, we prove that if e sticks, then $\underline{e}\ k$ will eventually stick (Lem.7).

A complete proof is given is section 4.2. □

Theorem 4 (Translation)**.** *Let $e_1, e_2 \in Expr$ such that $depth(e_1) = 0$.*
If $e_1 =^0_{\mathtt{v}} e_2$ then $\underline{e_1} =^0_{\mathtt{v}} \underline{e_2}$ and $\underline{e_1} =^0_{\mathtt{n}} \underline{e_2}$. The converse is not true.

Proof. We prove that $\geq^n_{\mathtt{v}}$ is stable under CPS transformation Lem.8, and conclude using Church-Rosser theorem. Details are given in section 4.3. □

Colon Translation

$v : k$	$= k\ \Psi(v)$	$v \in Value^0$
$(e_1\ e_2) : k$	$= e_1 : \big(\lambda m.\underline{e_2}\ (\lambda n.(m\ n)\ k)\big)$	$e_1 \notin Value^0$
$(v\ e) : k$	$= e : (\lambda n.(\Psi(v)\ n)\ k)$	$v \in Value^0, e \notin Value^0$
$(v_1\ v_2) : k$	$= (\Psi(v_1)\ \Psi(v_2))\ k$	$v_1, v_2 \in Value^0$
$(\mathbf{box}\ e) : k$	$= \kappa : (k\ (\mathbf{box}\ \underline{e}))$	$e \mapsto (\underline{e}, (\bot, \kappa))$
$(\mathbf{run}\ e) : k$	$= e : (\lambda m.(\mathbf{run}\ m)\ k)$	$e \notin Value^0$
$(\mathbf{run}\ v) : k$	$= (\mathbf{run}\ \Psi(v))\ k$	$v \in Value^0$
$(\underline{e}\ \lambda h.[\cdot]) : k$	$= e : \lambda h.k$	
$(\underline{e}\ \lambda h.\kappa) : k$	$= e : \lambda h.\kappa[k]$	

Figure 17: Colon Translation over λ_S

Reduction Rules

$$\frac{K \xrightarrow{n-1} K'}{K,\kappa \xrightarrow{n} K',\kappa} \qquad \frac{\kappa \xrightarrow{n} \kappa'}{K,\kappa \xrightarrow{n} K,\kappa'}$$

$$\frac{e \xrightarrow{n} e'}{(e\ \lambda h.[\cdot]) \xrightarrow{n} (e'\ \lambda h.[\cdot])} \qquad \frac{e \xrightarrow{n} e'}{(e\ \lambda h.\kappa) \xrightarrow{n} (e'\ \lambda h.\kappa} \qquad \frac{\kappa \xrightarrow{n} \kappa'}{(e\ \lambda h.\kappa) \xrightarrow{n} (e\ \lambda h.\kappa'}$$

Figure 18: Reduction Rules over Contexts and Context Stacks

Outermost Context

$$\begin{array}{ll} \kappa,\bot & = \bot,\kappa \\ \kappa,(K,\kappa') & = (\kappa,K),\kappa' \end{array}$$

Figure 19: Outermost Context over Context Stacks

Lemma 5 (CPS Transformation Evaluates to Colon Translation)**.**

$$\frac{k \in Closed\ Value^0 \qquad e \mapsto (\underline{e},\bot)}{\underline{e}\ k \xrightarrow{0^+} e : k}$$

Lemma 6 (Reduction Preservation wrt. Colon Translation)**.** *Let* $e_1, e_2 \in Expr$ *and* $k \in Closed\ Value^0$ *such that* $depth(e_1) = n$, $e_1 \xrightarrow{n} e_2$, $e_1 \mapsto (\underline{e_1}, K_1)$ *and* $e_2 \mapsto (\underline{e_2}, K_2)$.

- *If* $K_1 = \bot$ *then* $e_1 : k \xrightarrow{0^+} e_2 : k$ *and* $K_2 = \bot$.
- *If* $K_1 = \kappa_1, K_1'$ *(ie.* κ_1 *is the outermost context of* K_1*) then*
  - *If* $\kappa_1 = (\underline{e_h}\ \lambda h.[\cdot])$ *with* $e_h \in Value^0$ *then* $[h \overset{n}{\mapsto} \Psi(e_h)]\underline{e_1} \xrightarrow{n^\star} \underline{e_2}$ *and* $[h \overset{n-1}{\mapsto} \Psi(e_h)]K_1' \xrightarrow{n-1^\star} K_2$.
  - *If* $\kappa_1 = (\underline{e_h}\ \lambda h.\kappa_1')$ *with* $e_h \in Value^0$ *then* $[h \overset{n}{\mapsto} \Psi(e_h)]\underline{e_1} \xrightarrow{n^\star} \underline{e_2}$ *and* $[h \overset{n-1}{\mapsto} \Psi(e_h)]\,(\kappa_1', K_1') \xrightarrow{n-1^\star} K_2$.
  - *If* $\kappa_1 = (\underline{e_h}\ \lambda h.[\cdot])$ *with* $e_h \notin Value^0$ *then* $\underline{e_1} = \underline{e_2}$ *and* $\exists e_h'$ *such that* $e_h : k \xrightarrow{0^+} e_h' : k$ *and* $K_2 = \left(\underline{e_h'}\ \lambda h.[\cdot]\right), K_1'$.
  - *If* $\kappa_1 = (\underline{e_h}\ \lambda h.\kappa_1')$ *with* $e_h \notin Value^0$ *then* $\underline{e_1} = \underline{e_2}$ *and* $\exists e_h'$ *such that* $e_h : k \xrightarrow{0^+} e_h' : k$ *and* $K_2 = \left(\underline{e_h'}\ \lambda h.\kappa_1'\right), K_1'$.

In order to complete our Simulation theorem, we need to prove that if evaluation isn't defined, so is evaluation of translated expression whatever the reduction stragegy. An evaluator is undefined if no further reduction rule applies, but current expression isn't a value. We define then inductively $Sticks_{\mathrm{v}}^n$ Fig.20 and $Sticks_{\mathrm{n}}^n$ Fig.21 sets corresponding respectively to expressions which sticks with a CBV evaluation and a CBN one. Note that we assume expression is closed, ie. we omit cases like $x\ e$, **run** x ... $\in Sticks^0$.

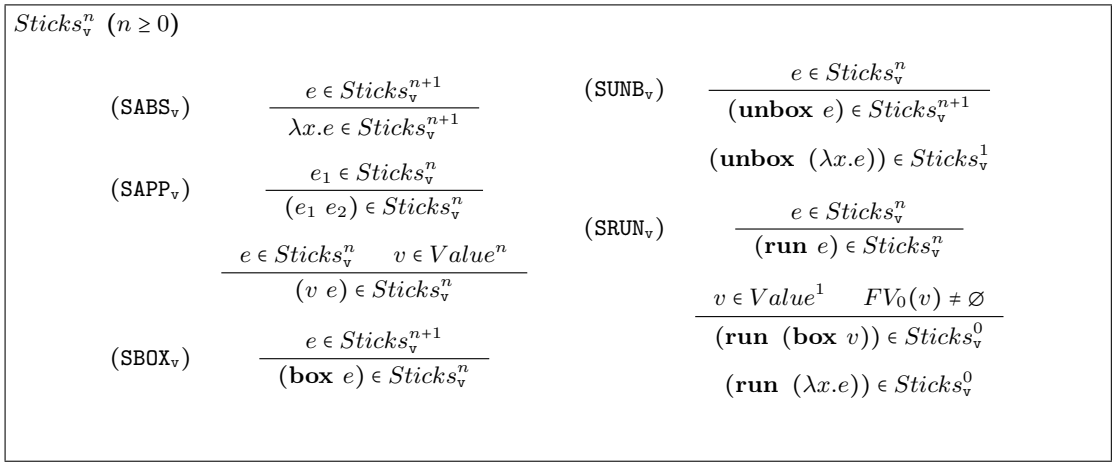

Figure 20: $Sticks^n_{\mathtt{v}}$ for Call-by-value Operational Semantics of λ_S

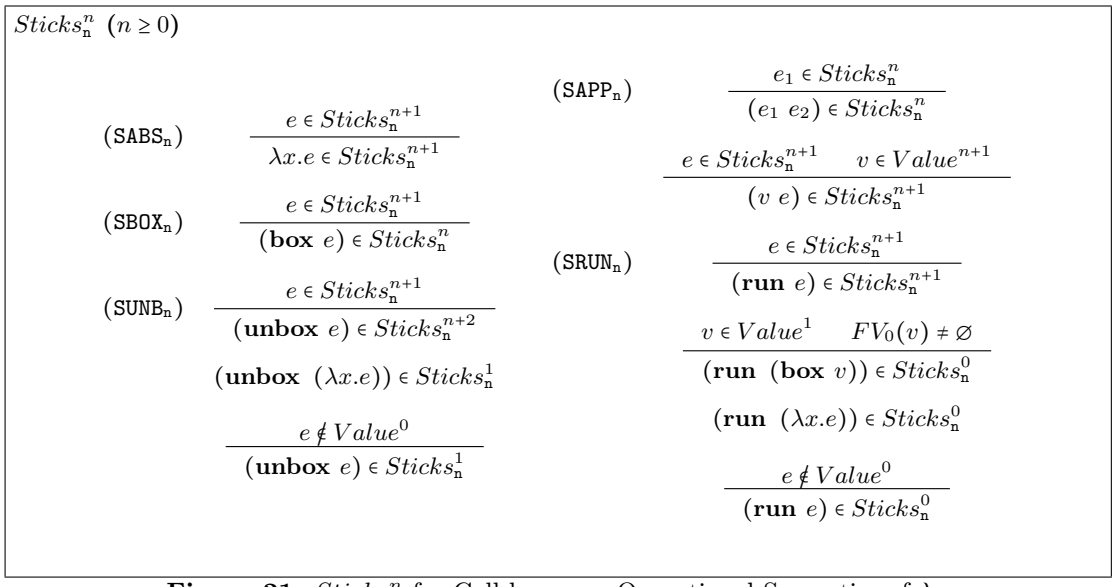

Figure 21: $Sticks^n_{\mathtt{n}}$ for Call-by-name Operational Semantics of λ_S

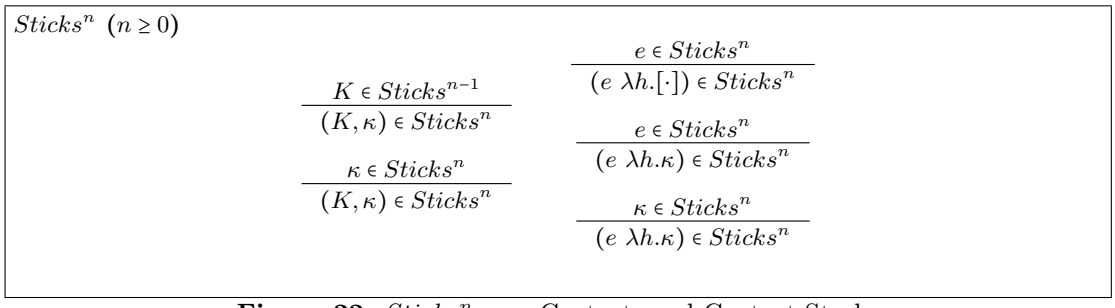

Figure 22: $Sticks^n$ over Contexts and Context Stacks

Lemma 7 ($Sticks^n$ Preservation wrt. Colon Translation)**.** *Let* $e \in Expr$, $k \in Closed\ Value^0$ *such that* $depth(e) = n$, $e \in Sticks_{\mathtt{v}}^n$ *and* $e \mapsto (\underline{e}, K)$

- *If* $K = \bot$ *then* $\exists e_s$ *such that* $e : k \xrightarrow{0^\star} e_s$ *and* $e_s \in Sticks_{\mathtt{v}}^0 \cap Sticks_{\mathtt{n}}^0$
- *If* $K = \kappa, K'$ *with* $\kappa = \left(\underline{e_h}\ \lambda h.[\cdot]\right)$ *or* $\kappa = \left(\underline{e_h}\ \lambda h.\kappa'\right)$ *we have three cases :*
  - *If* $e_h \in Value^0$ *and* $K' = \bot$ *then* $[h \overset{1}{\mapsto} \Psi(e_h)]\underline{e} \in Sticks_{\mathtt{v}}^1 \cap Sticks_{\mathtt{n}}^1$.
  - *If* $e_h \in Value^0$ *and* $K' = \kappa_2, K_2'$ *then* $[h \overset{1}{\mapsto} \Psi(e_h)]\kappa_2 \in Sticks_{\mathtt{v}}^1 \cap Sticks_{\mathtt{n}}^1$.
  - *If* $e_h \notin Value^0$ *then* $\exists e_s$ *such that* $e_h : k \xrightarrow{0^\star} e_s$ *and* $e_s \in Sticks_{\mathtt{v}}^0 \cap Sticks_{\mathtt{n}}^0$.

Equality ($n \geq 0$)

$$\frac{K_1 =^{n-1} K_2}{(K_1, \kappa) =^n (K_2, \kappa)} \qquad \frac{\kappa_1 =^n \kappa_2}{(K, \kappa_1) =^n (K, \kappa_2)}$$

$$\frac{e_1 =^n e_2}{(e_1\ \lambda h.[\cdot]) =^n (e_2\ \lambda h.[\cdot])} \qquad \frac{e_1 =^n e_2}{(e_1\ \lambda h.\kappa) =^n (e_2\ \lambda h.\kappa)} \qquad \frac{\kappa_1 =^n \kappa_2}{(e\ \lambda h.\kappa_1) =^n (e\ \lambda h.\kappa_2)}$$

Figure 23: Equality over Contexts and Context Stacks

Lemma 8 ($\geq_{\mathtt{v}}^n$ Preservation wrt. CPS Transformation)**.** *Let* $e_1, e_2 \in Expr$ *with* $depth(e_1) = n$ *and* $e_i \mapsto \left(\underline{e_i}, K_i\right)$ *for* $i \in \{1, 2\}$ *such that* $e_1 \geq_{\mathtt{v}}^n e_2$.

- *If* $K_1 = \bot$ *then* $\underline{e_1} \geq^n \underline{e_2}$ *and* $K_2 = \bot$
- *If* $K_1 = \kappa_1, K_1'$ *then we have four cases.*
  - *If* $\kappa_1 = \left(\underline{e_h}\ \lambda h.[\cdot]\right)$ *with* $e_h \in Value^0$ *then* $[h \overset{n}{\mapsto} \Psi(e_h)]\underline{e_1} \geq^n \underline{e_2}$ *and* $[h \overset{n-1}{\mapsto} \Psi(e_h)]K_1' \geq^{n-1} K_2$
  - *If* $\kappa_1 = \left(\underline{e_h}\ \lambda h.\kappa_1'\right)$ *with* $e_h \in Value^0$ *then* $[h \overset{n}{\mapsto} \Psi(e_h)]\underline{e_1} \geq^n \underline{e_2}$ *and* $[h \overset{n-1}{\mapsto} \Psi(e_h)]\,(\kappa_1', K_1') \geq^{n-1} K_2$
  - *If* $\kappa_1 = \left(\underline{e_h}\ \lambda h.[\cdot]\right)$ *with* $e_h \notin Value^0$ *then* $\underline{e_1} = \underline{e_2}$ *and* $\exists e_h'$ *such that* $\underline{e_h} \geq^0 \underline{e_h'}$ *and* $K_2 = \left(\underline{e_h'}\ \lambda h.[\cdot]\right), K_1'$
  - *If* $\kappa_1 = \left(\underline{e_h}\ \lambda h.\kappa_1'\right)$ *with* $e_h \notin Value^0$ *then* $\underline{e_1} = \underline{e_2}$ *and* $\exists e_h'$ *such that* $\underline{e_h} \geq^0 \underline{e_h'}$ *and* $K_2 = \left(\underline{e_h'}\ \lambda h.\kappa_1'\right), K_1'$

and in each case, at most one inequality isn't an equality.

4 Proofs

4.1 Proof of Church Rosser Theorem

Lemma 9 (Depth Preservation wrt. $\geq^n$)**.** *Let* $e_1, e_2 \in Expr$*. If* $e_1 \geq^n e_2$ *then* $depth(e_1) \geq depth(e_2)$*.*

Proof. By induction on the size of derivation tree and cases on root (IH).

- Case (`EREF`).
  Then $e_1 = e_2$ hence $depth(e_1) \geq depth(e_2)$.
- Case (`ETRA`). $\exists e$ such that $e_1 \geq^n e$ and $e \geq^n e_2$.
  By (IH) $depth(e_1) \geq depth(e) \geq depth(e_2)$.
- Case (`EAPP`)[1] with $e_1 = e_1'\ e_2'$, $e_2 = e_1''\ e_2'$ and $e_1' \geq^n e_1''$.
  By (IH) $depth(e_1') \geq depth(e_1'')$ and as $depth(e_1) = max(depth(e_1'), depth(e_2'))$ and $depth(e_2) = max(depth(e_1''), depth(e_2'))$ then $depth(e_1) \geq depth(e_2)$.
- Case (`EAPP`)[2]. Similar to case (`EAPP`)[1].
- Case (`EAPP`)[3].
  As *depth* is stable by substitution, $depth(e_1) = depth(e_2)$.
- Case (`EALP`).
  As *depth* is stable by α-renaming, $depth(e_1) = depth(e_2)$.
- Case (`EABS`) with $e_1 = \lambda x.e_1'$, $e_2 = \lambda x.e_2'$ and $e_1' \geq^n e_2'$.
  By (IH) $depth(e_1') \geq depth(e_2')$ and then $depth(e_1) \geq depth(e_2)$.
- Case (`EBOX`) with $e_1 = \mathbf{box}\ e_1'$, $e_2 = \mathbf{box}\ e_2'$ and $e_1' \geq^{n+1} e_2'$.
  By (IH) $depth(e_1') \geq depth(e_2')$ and then $depth(e_1) = depth(e_1') - 1 \geq depth(e_2') - 1 = depth(e_2)$.
- Case (`EUNB`)[1] with $e_1 = \mathbf{unbox}\ e_1'$, $e_2 = \mathbf{unbox}\ e_2'$ and $e_1' \geq^{n-1} e_2'$.
  By (IH) $depth(e_1') \geq depth(e_2')$ and then $depth(e_1) = depth(e_1') + 1 \geq depth(e_2') + 1 = depth(e_2)$.
- Case (`EUNB`)[2] with $e_1 \mathbf{unbox}\ (\mathbf{box}\ v)$ and $e_2 = v$.
  Then $depth(e_1) = max(depth(e_2) - 1, 0) + 1 \geq depth(e_2)$.
- Case (`ERUN`)[1] with $e_1 = \mathbf{run}\ e_1'$, $e_2 = \mathbf{run}\ e_2'$ and $e_1' \geq^n e_2'$.
  By (IH) $depth(e_1') \geq depth(e_2')$ and then $depth(e_1) \geq depth(e_2)$.
- Case (`ERUN`)[2] with $e_1 = \mathbf{run}\ (\mathbf{box}\ v)$ and $e_2 = v$.
  As $v \in Value^0$, $depth(v) = 0$ and $depth(e_1) \geq 0 = depth(e_2)$.

□

Lemma 10 (Minimal Depth for Non Trivial $\geq^n$)**.** *Let* $e_1, e_2 \in Expr$*. If* $e_1 \geq^n e_2$ *then either* $e_1 = e_2$ *or* $depth(e_1) \geq n$*.*

Proof. By induction on the size of derivation tree and cases on root (IH).

- Case (`EREF`).
  Obvious: $e_1 \geq^n e_2$ and $e_1 = e_2$.
- Case (`ETRA`). $\exists e$ such that $e_1 \geq^n e$ and $e \geq^n e_2$.
  By (IH) we have 4 cases:
  - Case $e_1 = e = e_2$ then $e_1 = e_2$.
  - Case $depth(e_1) \geq n$ and $depth(e) \geq n$ then $depth(e_1) = n$.
  - Case $e_1 = e$ and $depth(e) \geq n$ then $depth(e_1) \geq n$.
  - Case $depth(e_1) \geq n$ and $e = e_2$ then $depth(e_1) \geq n$.
- Case (`EAPP`)[1] with $e_1 = e_1'\ e_2'$, $e_2 = e_1''\ e_2'$ and $e_1' \geq^n e_1''$.
  By (IH) either $e_1' = e_1''$ and then $e_1 = e_2$, or $depth(e_1') \geq n$ and $depth(e_1) \geq n$.
- Case (`EAPP`)[2]. Similar to case (`EAPP`)[1].
- Case (`EAPP`)[3]. Then $n = 0$ so $depth(e_1) \geq n$.
- Case (`EALP`). Then $n = 0$ so $depth(e_1) \geq n$.
- Case (`EABS`) with $e_1 = \lambda x.e_1'$, $e_2 = \lambda x.e_2'$ and $e_1' \geq^n e_2'$.
  By (IH) either $e_1' = e_2'$ and then $e_1 = e_2$, or $depth(e_1') \geq n$ and then $depth(e_1) \geq n$.
- Case (`EBOX`) with $e_1 = \mathbf{box}\ e_1'$, $e_2 = \mathbf{box}\ e_2'$ and $e_1' \geq^{n+1} e_2'$.
  By (IH) either $e_1' = e_2'$ and then $e_1 = e_2$, or $depth(e_1') \geq n+1$ and then $depth(e_1) \geq n$.
- Case (`EUNB`)[1] with $e_1 = \mathbf{unbox}\ e_1'$, $e_2 = \mathbf{unbox}\ e_2'$ and $e_1' \geq^{n-1} e_2'$.
  By (IH) either $e_1' = e_2'$ and then $e_1 = e_2$, or $depth(e_1') \geq n-1$ and then $depth(e_1) \geq n$.
- Case (`EUNB`)[2] with $e_1\mathbf{unbox}\ e'$.
  Then $n = 1$ and as $depth(e_1) = 1 + depth(e')$, $depth(e_1) \geq n$.
- Case (`ERUN`)[1] with $e_1 = \mathbf{run}\ e_1'$, $e_2 = \mathbf{run}\ e_2'$ and $e_1' \geq^n e_2'$.
  By (IH) either $e_1' = e_2'$ and then $e_1 = e_2$, or $depth(e_1') \geq n$ and then $depth(e_1) \geq n$.
- Case (`ERUN`)[2]. Then $n = 0$ and so $depth(e_1) \geq n$.

□

Lemma 11 (η-reduction for $\geq^0$)**.**

$$\lambda\,(x.\,(\lambda y.e)\ \ x) \geq^0 \lambda y.e$$

Proof.

$$\begin{aligned}
\lambda\,(x.\,(\lambda y.e)\ \ x) &\geq^0 \lambda x.[y \overset{0}{\mapsto} x]e && \text{(EAPP[3], EABS)}\\
&\geq^0 \lambda y.[x \overset{0}{\mapsto} y][y \overset{0}{\mapsto} x]e && \text{(EALP, EABS)}\\
&\geq^0 \lambda y.e
\end{aligned}$$

□

Parallel Reduction ($n \geq 0$)

$$(\texttt{PREF}_\texttt{v}) \quad e \geq_1^n e \qquad (\texttt{PBOX}_\texttt{v}) \quad \frac{e \geq_1^{n+1} e'}{\textbf{box}\ e \geq_1^n \textbf{box}\ e'}$$

$$(\texttt{PAPP}_\texttt{v}) \quad \frac{e_1 \geq_1^n e_1' \qquad e_2 \geq_1^n e_2'}{e_1\ e_2 \geq_1^n e_1'\ e_2'} \qquad (\texttt{PUNB}_\texttt{v}) \quad \frac{e \geq_1^n e'}{\textbf{unbox}\ e \geq_1^{n+1} \textbf{unbox}\ e'}$$

$$\frac{e \geq_1^0 e' \qquad v \geq_1^0 v'}{(\lambda x.e)\ v \geq_1^0 [x \overset{0}{\mapsto} v']e'} \qquad \frac{e \geq_1^1 v \qquad v \in Value^1}{\textbf{unbox}\ \ (\textbf{box}\ e) \geq_1^1 v}$$

$$(\texttt{PABS}_\texttt{v}) \quad \frac{e \geq_1^n e'}{\lambda x.e \geq_1^n \lambda x.e'} \qquad (\texttt{PRUN}_\texttt{v}) \quad \frac{e \geq_1^n e'}{\textbf{run}\ e \geq_1^n \textbf{run}\ e'}$$

$$(\texttt{PALP}_\texttt{v}) \quad \frac{e \geq_1^0 e'}{\lambda x.e =_\texttt{v}^0 \lambda y.[x \overset{0}{\mapsto} y]e'} \qquad \frac{e \geq_1^1 v \qquad v \in Value^1 \qquad FV_0(v) = \varnothing}{\textbf{run}\ \ (\textbf{box}\ e) \geq_1^0 v}$$

Figure 24: Call-by-value Parallel Reduction for λ_S

Lemma 12 ($\geq_\texttt{v}^n$ is $\geq_1^n$ Transitive Closure)**.** *Let* $e, e' \in Expr$*.*

$$e \geq_\texttt{v}^n e' \Leftrightarrow \exists e_1, \ldots, e_k \in Expr \text{ such that } e = e_1 \geq_1^n \ldots \geq_1^n e_k = e'$$

Proof. $\Leftarrow$ is straightforward. We will prove $\Rightarrow$ by induction on the size of proof of $e \geq_\texttt{v}^n e'$ (IH).

- ($\texttt{EREF}_\texttt{v}$) Then $e = e'$. Obvious.
- ($\texttt{ETRA}_\texttt{v}$) Then $e \geq_\texttt{v}^n e''$ and $e'' \geq_\texttt{v}^n e'$.
  By (IH) $\exists e_1, \ldots, e_k, \ldots e_m$ such that $e = e_1 \geq_1^n \ldots \geq_1^n e_k = e'' \geq_1^n \ldots \geq_1^n e_m = e'$.
- ($\texttt{EAPP}_\texttt{v}$)[1] Then $e = e_a\ e_b$, $e' = e_a'\ e_b$ and $e_a \geq_\texttt{v}^n e_a'$.
  By (IH) $\exists e_1, \ldots, e_k$ such that $e_a = e_1 \geq_1^n \ldots \geq_1^n e_k = e_a'$. By ($\texttt{PREF}_\texttt{v}$) and ($\texttt{PAPP}_\texttt{v}$)[1] $e_a\ e_b = e_1\ e_b \geq_1^n \ldots \geq_1^n e_k\ e_b = e_a'\ e_b$.
- ($\texttt{EAPP}_\texttt{v}$)[2] Similar to previous case.
- ($\texttt{EAPP}_\texttt{v}$)[3] Then $e = (\lambda x.e'')\ \ v$ and $e' = [x \overset{0}{\mapsto} v]e''$.
  By ($\texttt{PREF}_\texttt{v}$) and ($\texttt{PAPP}_\texttt{v}$)[2].
- ($\texttt{EALP}_\texttt{v}$) Then $e = \lambda x.e''$ and $e' = \lambda y.[x \overset{0}{\mapsto} y]e''$.
  By ($\texttt{PREF}_\texttt{v}$) and ($\texttt{PALP}_\texttt{v}$).
- ($\texttt{EABS}_\texttt{v}$) Then $e = \lambda x.e_a$, $e' = \lambda x.e_a'$ and $e_a \geq_\texttt{v}^n e_a'$.
  By (IH) $\exists e_1, \ldots, e_k$ such that $e_a = e_1 \geq_1^n \ldots \geq_1^n e_k = e_a'$. By ($\texttt{PREF}_\texttt{v}$) and ($\texttt{PABS}_\texttt{v}$) $\lambda x.e_a = \lambda x.e_1 \geq_1^n \ldots \geq_1^n \lambda x.e_k = \lambda x.e_a'$.
- ($\texttt{EBOX}_\texttt{v}$) Then $e = \textbf{box}\ e_a$, $e' = \textbf{box}\ e_a'$ and $e_a \geq_\texttt{v}^{n+1} e_a'$.
  By (IH) $\exists e_1, \ldots, e_k$ such that $e_a = e_1 \geq_1^n \ldots \geq_1^n e_k = e_a'$. By ($\texttt{PREF}_\texttt{v}$) and ($\texttt{PBOX}_\texttt{v}$) $\textbf{box}\ e_a = \textbf{box}\ e_1 \geq_1^{n+1} \ldots \geq_1^{n+1} \textbf{box}\ e_k = \textbf{box}\ e_a'$.

- ($\texttt{EUNB}_\texttt{v}$)[1] Then $e = \textbf{unbox}\ e_a$, $e' = \textbf{unbox}\ e'_a$ and $e_a \geq_\texttt{v}^{n-1} e'_a$.

  By (IH) $\exists e_1, \ldots, e_k$ such that $e_a = e_1 \geq_1^n \ldots \geq_1^n e_k = e'_a$. By ($\texttt{PREF}_\texttt{v}$) and ($\texttt{PUNB}_\texttt{v}$)[1] $\textbf{unbox}\ e_a = \textbf{unbox}\ e_1 \geq_1^{n-1} \ldots \geq_1^{n-1} \textbf{unbox}\ e_k = \textbf{unbox}\ e'_a$.

- ($\texttt{EUNB}_\texttt{v}$)[2] Then $e = \textbf{unbox}\ (\textbf{box}\ v)$ and $e' = v$, $v \in Value^1$.

  By ($\texttt{PREF}_\texttt{v}$) and ($\texttt{PUNB}_\texttt{v}$)[2].

- ($\texttt{ERUN}_\texttt{v}$)[1] Then $e = \textbf{run}\ e_a$, $e' = \textbf{run}\ e'_a$ and $e_a \geq_\texttt{v}^n e'_a$.

  By (IH) $\exists e_1, \ldots, e_k$ such that $e_a = e_1 \geq_1^n \ldots \geq_1^n e_k = e'_a$. By ($\texttt{PREF}_\texttt{v}$) and ($\texttt{PRUN}_\texttt{v}$)[1] $\textbf{run}\ e_a = \textbf{run}\ e_1 \geq_1^n \ldots \geq_1^n \textbf{run}\ e_k = \textbf{run}\ e'_a$.

- ($\texttt{ERUN}_\texttt{v}$)[2] Then $e = \textbf{run}\ (\textbf{box}\ e')$ and $e' \in Value^1$.

  By ($\texttt{PREF}_\texttt{v}$) and ($\texttt{PRUN}_\texttt{v}$)[2].

□

Lemma 13 (Substitution Distributivity)**.** *Let* $e_1, e_2, e_3 \in Expr$.

$$\frac{x \neq y \qquad x \in BV^n(e_1) \Rightarrow x \notin FV^n(e_3) \qquad y \notin FV^n(e_2)}{[x \overset{n}{\mapsto} e_2]\left([y \overset{n}{\mapsto} e_3]e_1\right) = \left[y \overset{n}{\mapsto} [x \overset{n}{\mapsto} e_2]e_3\right]\left([x \overset{n}{\mapsto} e_2]e_1\right)}$$

Proof. By induction over the structure of e_1.

- Case $e_1 = x$, $n = 0$.

$$\begin{aligned}\left[y \overset{0}{\mapsto} [x \overset{0}{\mapsto} e_2]e_3\right]\left([x \overset{0}{\mapsto} e_2]x\right) &= \left[y \overset{0}{\mapsto} [x \overset{0}{\mapsto} e_2]e_3\right]e_2 \\ &= e_2 && (y \notin FV^0(e_2)) \\ &= [x \overset{0}{\mapsto} e_2]x \\ &= [x \overset{0}{\mapsto} e_2]\left([y \overset{0}{\mapsto} e_3]x\right)\end{aligned}$$

- Case $e_1 = y$, $n = 0$.

$$\begin{aligned}\left[y \overset{0}{\mapsto} [x \overset{0}{\mapsto} e_2]e_3\right]\left([x \overset{0}{\mapsto} e_2]y\right) &= \left[y \overset{0}{\mapsto} [x \overset{0}{\mapsto} e_2]e_3\right]y \\ &= [x \overset{0}{\mapsto} e_2]e_3 \\ &= [x \overset{0}{\mapsto} e_2]\left([y \overset{0}{\mapsto} e_3]y\right)\end{aligned}$$

- Case $e_1 = z$, $z \notin \{x, y\}$ or $n > 0$

$$\begin{aligned}\left[y \overset{n}{\mapsto} [x \overset{n}{\mapsto} e_2]e_3\right]\left([x \overset{n}{\mapsto} e_2]z\right) &= \left[y \overset{n}{\mapsto} [x \overset{n}{\mapsto} e_2]e_3\right]z \\ &= z \\ &= [x \overset{n}{\mapsto} e_2]z \\ &= [x \overset{n}{\mapsto} e_2]\left([y \overset{n}{\mapsto} e_3]z\right)\end{aligned}$$

- Case $e_1 = \lambda x.e$, $n = 0$. Then $x \notin FV^0(e_3)$.

$$\begin{aligned}\left[y \overset{0}{\mapsto} [x \overset{0}{\mapsto} e_2]e_3\right]\left([x \overset{0}{\mapsto} e_2]\lambda x.e\right) &= \left[y \overset{0}{\mapsto} [x \overset{0}{\mapsto} e_2]e_3\right]\lambda x.e \\ &= [y \overset{0}{\mapsto} e_3]\lambda x.e && (x \notin FV^0(e_3)) \\ &= [x \overset{0}{\mapsto} e_2]\left([y \overset{0}{\mapsto} e_3]\lambda x.e\right) && (x \notin FV^0(e_3))\end{aligned}$$

- Case $e_1 = \lambda y.e$, $n = 0$.

$$\begin{aligned}\left[y \overset{0}{\mapsto} [x \overset{0}{\mapsto} e_2]e_3\right]\left([x \overset{0}{\mapsto} e_2]\lambda y.e\right) &= \left[y \overset{0}{\mapsto} [x \overset{0}{\mapsto} e_2]e_3\right]\lambda y.\left([x \overset{0}{\mapsto} e_2]e\right) \\ &= \lambda y.[x \overset{0}{\mapsto} e_2]e \\ &= [x \overset{0}{\mapsto} e_2]\lambda y.e \\ &= [x \overset{0}{\mapsto} e_2]\left([y \overset{0}{\mapsto} e_3]\lambda y.e\right)\end{aligned}$$

- Case $e_1 = \lambda z.e$, $z \notin \{x, y\}$ or $n > 0$

$$\begin{aligned}\left[y \overset{n}{\mapsto} [x \overset{n}{\mapsto} e_2]e_3\right]\left([x \overset{n}{\mapsto} e_2]\lambda z.e\right) &= \lambda z.\left[y \overset{n}{\mapsto} [x \overset{n}{\mapsto} e_2]e_3\right]\left([x \overset{n}{\mapsto} e_2]e\right) \\ &= \lambda z.[x \overset{n}{\mapsto} e_2]\left([y \overset{n}{\mapsto} e_3]e\right) && \text{(IH)} \\ &= [x \overset{n}{\mapsto} e_2]\left([y \overset{n}{\mapsto} e_3]\lambda z.e\right)\end{aligned}$$

- Case $e_1 = e_a\ e_b$.

$$\begin{aligned}&\left[y \overset{n}{\mapsto} [x \overset{n}{\mapsto} e_2]e_3\right]\left([x \overset{n}{\mapsto} e_2]\,(e_a\ e_b)\right) \\ &= \left[y \overset{n}{\mapsto} [x \overset{n}{\mapsto} e_2]e_3\right]\left([x \overset{n}{\mapsto} e_2]e_a\right)\ \left[y \overset{n}{\mapsto} [x \overset{n}{\mapsto} e_2]e_3\right]\left([x \overset{n}{\mapsto} e_2]e_b\right) \\ &= [x \overset{n}{\mapsto} e_2]\left([y \overset{n}{\mapsto} e_3]e_a\right)\ [x \overset{n}{\mapsto} e_2]\left([y \overset{n}{\mapsto} e_3]e_b\right) && \text{(IH)} \\ &= [x \overset{n}{\mapsto} e_2]\left([y \overset{n}{\mapsto} e_3]\,(e_a\ e_b)\right)\end{aligned}$$

- Case $e_1 = \mathbf{box}\ e$.

$$\begin{aligned}\left[y \overset{n}{\mapsto} [x \overset{n}{\mapsto} e_2]e_3\right]\left([x \overset{n}{\mapsto} e_2]\mathbf{box}\ e\right) &= \mathbf{box}\ \left[y \overset{n+1}{\mapsto} [x \overset{n+1}{\mapsto} e_2]e_3\right]\left([x \overset{n+1}{\mapsto} e_2]e\right) \\ &= \mathbf{box}\ [x \overset{n+1}{\mapsto} e_2]\left([y \overset{n+1}{\mapsto} e_3]e\right) && \text{(IH)} \\ &= [x \overset{n}{\mapsto} e_2]\left([y \overset{n}{\mapsto} e_3]\mathbf{box}\ e\right)\end{aligned}$$

- Case $e_1 = \mathbf{unbox}\ e$.

$$\begin{aligned}\left[y \overset{n+1}{\mapsto} [x \overset{n+1}{\mapsto} e_2]e_3\right]\left([x \overset{n+1}{\mapsto} e_2]\mathbf{unbox}\ e\right) &= \mathbf{unbox}\ \left[y \overset{n}{\mapsto} [x \overset{n}{\mapsto} e_2]e_3\right]\left([x \overset{n}{\mapsto} e_2]e\right) \\ &= \mathbf{unbox}\ [x \overset{n}{\mapsto} e_2]\left([y \overset{n}{\mapsto} e_3]e\right) && \text{(IH)} \\ &= [x \overset{n+1}{\mapsto} e_2]\left([y \overset{n+1}{\mapsto} e_3]\mathbf{unbox}\ e\right)\end{aligned}$$

- Case $e_1 = \mathbf{run}\ e$.

$$\begin{aligned}\left[y \overset{n}{\mapsto} [x \overset{n}{\mapsto} e_2]e_3\right]\left([x \overset{n}{\mapsto} e_2]\mathbf{run}\ e\right) &= \mathbf{run}\ \left[y \overset{n}{\mapsto} [x \overset{n}{\mapsto} e_2]e_3\right]\left([x \overset{n}{\mapsto} e_2]e\right) \\ &= \mathbf{run}\ [x \overset{n}{\mapsto} e_2]\left([y \overset{n}{\mapsto} e_3]e\right) && \text{(IH)} \\ &= [x \overset{n}{\mapsto} e_2]\left([y \overset{n}{\mapsto} e_3]\mathbf{run}\ e\right)\end{aligned}$$

□

Lemma 14 ($\geq_1^n$ Preservation wrt. Substitution)**.** *Let* $e_1, e_2, e_2' \in Expr$.

$$\frac{e_2 \geq_1^n e_2' \qquad FV^n(e_2') \cap BV^n(e_1) = \varnothing}{[x \overset{n}{\mapsto} e_2]e_1 \geq_1^n [x \overset{n}{\mapsto} e_2']e_1}$$

Proof. By induction over the structure of e_1. (IH).

- Case $e_1 = x$, $n = 0$.

$$\begin{aligned}[x \overset{0}{\mapsto} e_2]x &= e_2 \\ &\geq_1^0 e_2' \\ &\geq_1^0 [x \overset{0}{\mapsto} e_2']x\end{aligned}$$

- Case $e_1 = y$, $y \neq x$ or $n > 0$.

$$\begin{aligned}[x \overset{n}{\mapsto} e_2]y &= y \\ &\geq_1^n y && ((\texttt{PREF}_\texttt{v})) \\ &\geq_1^n [x \overset{n}{\mapsto} e_2']y\end{aligned}$$

- Case $e_1 = \lambda x.e$, $n = 0$.

$$\begin{aligned}[x \overset{0}{\mapsto} e_2]\lambda x.e &= \lambda x.e \\ &\geq_1^0 \lambda x.e && ((\texttt{PREF}_\texttt{v})) \\ &\geq_1^0 [x \overset{n}{\mapsto} e_2']\lambda x.e\end{aligned}$$

- Case $e_1 = \lambda y.e$, $y \neq x$ or $n > 0$.

$$\begin{aligned}
[x \overset{n}{\mapsto} e_2]\lambda y.e &= \lambda y.[x \overset{n}{\mapsto} e_2]e \\
&\geq_1^n \lambda y.[x \overset{n}{\mapsto} e_2']e && \text{(IH, } (\texttt{PABS}_\texttt{v})\text{)} \\
&\geq_1^n [x \overset{n}{\mapsto} e_2']\lambda y.e && (FV^n(e_2') \cap FV(e_1) = \varnothing)
\end{aligned}$$

- Case $e_1 = e_a\ e_b$.

$$\begin{aligned}
[x \overset{n}{\mapsto} e_2]\,(e_a\ e_b) &= [x \overset{n}{\mapsto} e_2]e_a\ [x \overset{n}{\mapsto} e_2]e_b \\
&\geq_1^n [x \overset{n}{\mapsto} e_2']e_a\ [x \overset{n}{\mapsto} e_2']e_b && \text{(IH, } (\texttt{PAPP}_\texttt{v})[1]\text{)} \\
&\geq_1^n [x \overset{n}{\mapsto} e_2']\,(e_a\ e_b) && (FV^n(e_2') \cap FV(e_1) = \varnothing)
\end{aligned}$$

- Case $e_1 = \mathbf{box}\ e$.

$$\begin{aligned}
[x \overset{n}{\mapsto} e_2]\,(\mathbf{box}\ e) &= \mathbf{box}\ [x \overset{n+1}{\mapsto} e_2]e \\
&\geq_1^n \mathbf{box}\ [x \overset{n+1}{\mapsto} e_2']e && \text{(IH, } (\texttt{PBOX}_\texttt{v})\text{)} \\
&\geq_1^n [x \overset{n}{\mapsto} e_2']\,(\mathbf{box}\ e)
\end{aligned}$$

- Case $e_1 = \mathbf{unbox}\ e$.

$$\begin{aligned}
[x \overset{n+1}{\mapsto} e_2]\,(\mathbf{unbox}\ e) &= \mathbf{unbox}\ [x \overset{n}{\mapsto} e_2]e \\
&\geq_1^{n+1} \mathbf{unbox}\ [x \overset{n}{\mapsto} e_2']e && \text{(IH, } (\texttt{PUNB}_\texttt{v})\text{)} \\
&\geq_1^{n+1} [x \overset{n+1}{\mapsto} e_2']\,(\mathbf{unbox}\ e)
\end{aligned}$$

- Case $e_1 = \mathbf{run}\ e$.

$$\begin{aligned}
[x \overset{n}{\mapsto} e_2]\,(\mathbf{run}\ e) &= \mathbf{run}\ [x \overset{n}{\mapsto} e_2]e \\
&\geq_1^n \mathbf{run}\ [x \overset{n}{\mapsto} e_2']e && \text{(IH, } (\texttt{PRUN}_\texttt{v})\text{)} \\
&\geq_1^n [x \overset{n}{\mapsto} e_2']\,(\mathbf{run}\ e)
\end{aligned}$$

□

Lemma 15 ($\geq_1^n$ Preservation wrt. Substitution)**.** *Let* $e_1, e_1', e_2, e_2' \in Expr$.

$$\frac{e_i \geq_1^n e_i' \qquad FV^n(e_2') \cap BV^n(e_1\ e_1'\ e_2') = \varnothing \qquad x \notin BV^n(e_1')}{[x \overset{n}{\mapsto} e_2]e_1 \geq_1^n [x \overset{n}{\mapsto} e_2']e_1'}$$

Proof. By induction over the size of proof of $e_1 \geq_1^n e_1'$ (IH).

- ($\texttt{PREF}_\texttt{v}$) Then $e_1 = e_1'$ and we apply Lem.14.

- ($\texttt{PAPP}_\texttt{v}$)[1] Then $e_1 = e_a\ e_b$, $e_1' = e_a'\ e_b'$, $e_a \geq_1^n e_a'$ and $e_b \geq_1^n e_b'$.

$$\begin{aligned}
[x \overset{n}{\mapsto} e_2]\,(e_a\ e_b) &= [x \overset{n}{\mapsto} e_2]e_a\ [x \overset{n}{\mapsto} e_2]e_b \\
&\geq_1^n [x \overset{n}{\mapsto} e_2']e_a'\ [x \overset{n}{\mapsto} e_2']e_b' && \text{(IH, } \texttt{PAPP}_\texttt{v}\text{)} \\
&\geq_1^n [x \overset{n}{\mapsto} e_2']\,(e_a'\ e_b') && (FV^n(e_2') \cap BV^n(e_1') = \varnothing)
\end{aligned}$$

- ($\texttt{PAPP}_\texttt{v}$)[2] Then $e_1 = (\lambda y.e)\ v$, $e_1' = [y \overset{0}{\mapsto} v']e'$, $e \geq_1^0 e'$ and $v \geq_1^0 v'$.
  - Case $x = y$

$$\begin{aligned}
[x \overset{0}{\mapsto} e_2]\,(\lambda x.e)\ v &= (\lambda x.e)\ [x \overset{0}{\mapsto} e_2]v \\
&\geq_1^n \left[x \overset{0}{\mapsto} [x \overset{0}{\mapsto} e_2']v'\right]e' && \text{(IH, } \texttt{PAPP}_\texttt{v}[2]\text{)} \\
&\geq_1^n [x \overset{0}{\mapsto} e_2'][x \overset{0}{\mapsto} v']e'
\end{aligned}$$

  - Case $x \neq y$.

$$\begin{aligned}
[x \overset{0}{\mapsto} e_2]\,(\lambda y.e)\ v &= \left(\lambda y.[x \overset{0}{\mapsto} e_2]e\right)\ [x \overset{0}{\mapsto} e_2]v \\
&\geq_1^n \left[y \overset{0}{\mapsto} [x \overset{0}{\mapsto} e_2']v'\right]\left([x \overset{0}{\mapsto} e_2']e'\right) && \text{(IH, } \texttt{PAPP}_\texttt{v}[2]\text{)} \\
&\geq_1^n [x \overset{0}{\mapsto} e_2'][y \overset{0}{\mapsto} v']\left([x \overset{0}{\mapsto} e_2']e'\right) \\
&\geq_1^n [x \overset{0}{\mapsto} e_2'][y \overset{0}{\mapsto} v']e' && (y \notin FV^0(e_2'))
\end{aligned}$$

- ($\texttt{PABS}_\texttt{v}$) Then $e_1 = \lambda y.e$, $e_1' = \lambda y.e'$ and $e \geq_1^n e'$.

$$\begin{aligned}
[x \overset{n}{\mapsto} e_2]\lambda y.e = \lambda y.[x \overset{n}{\mapsto} e_2]e && (x \notin BV^n(e_1')) \\
&\geq_1^n \lambda y.[x \overset{n}{\mapsto} e_2']e' && \text{(IH, } \texttt{PABS}_\texttt{v}\text{)} \\
&\geq_1^n [x \overset{n}{\mapsto} e_2']\lambda y.e' && (FV^n(e_2') \cap BV^n(e_1') = \varnothing)
\end{aligned}$$

- ($\texttt{PALP}_\texttt{v}$) Then $e_1 = \lambda y.e$, $e_1' = \lambda z.[y \overset{0}{\mapsto} z]e'$ and $e \geq_1^0 e'$.
  - Case $y = x$.

$$\begin{aligned}
[x \overset{0}{\mapsto} e_2]\lambda x.e = \lambda x.e \\
&\geq_1^0 \lambda z.[x \overset{0}{\mapsto} z]e' && \text{(PALP}_\texttt{v}\text{)} \\
&\geq_1^0 [x \overset{0}{\mapsto} e_2']\lambda z.[x \overset{0}{\mapsto} z]e' && (FV^0(e_2') \cap BV^0(e_1') = \varnothing)
\end{aligned}$$

– Case $y \neq x$.

$$
\begin{aligned}
&[x \overset{0}{\mapsto} e_2]\lambda y.e \\
&= \lambda y.[x \overset{0}{\mapsto} e_2]e && (x \notin BV^0(e_1')) \\
&\geq_1^0 \lambda z.[y \overset{0}{\mapsto} z][x \overset{0}{\mapsto} e_2']e' && (\text{IH, } \mathtt{PALP_v}) \\
&\geq_1^0 \lambda z.[x \overset{0}{\mapsto} [y \overset{0}{\mapsto} z]e_2']\left([y \overset{0}{\mapsto} z]e'\right) && (\text{Lem.13, } x \notin BV^0(e_1'),\ y \notin FV^0(e_2'),\ \mathtt{PABS_v}) \\
&\geq_1^0 \lambda z.[x \overset{0}{\mapsto} e_2']\left([y \overset{0}{\mapsto} z]e'\right) && (FV^0(e_2') \cap BV^0(e_1) = \varnothing) \\
&\geq_1^0 [x \overset{0}{\mapsto} e_2']\lambda z.[y \overset{0}{\mapsto} z]e' && (FV^0(e_2') \cap BV^0(e_1') = \varnothing)
\end{aligned}
$$

- ($\mathtt{PBOX_v}$) Then $e_1 = \mathbf{box}\ e$, $e_1' = \mathbf{box}\ e'$ and $e \geq_1^{n+1} e'$.

$$
\begin{aligned}
[x \overset{n}{\mapsto} e_2]\mathbf{box}\ e &= \mathbf{box}\ [x \overset{n+1}{\mapsto} e_2]e \\
&\geq_1^n \mathbf{box}\ [x \overset{n+1}{\mapsto} e_2']e' && (\text{IH, } \mathtt{PBOX_v}) \\
&\geq_1^n [x \overset{n}{\mapsto} e_2']\mathbf{box}\ e'
\end{aligned}
$$

- ($\mathtt{PUNB_v}$)[1] Then $e_1 = \mathbf{unbox}\ e$, $e_1' = \mathbf{unbox}\ e'$ and $e \geq_1^n e'$.

$$
\begin{aligned}
[x \overset{n+1}{\mapsto} e_2]\mathbf{unbox}\ e &= \mathbf{unbox}\ [x \overset{n}{\mapsto} e_2]e \\
&\geq_1^{n+1} \mathbf{unbox}\ [x \overset{n}{\mapsto} e_2']e' && (\text{IH, } \mathtt{PUNB_v}[1]) \\
&\geq_1^{n+1} [x \overset{n+1}{\mapsto} e_2']\mathbf{unbox}\ e'
\end{aligned}
$$

- ($\mathtt{PUNB_v}$)[2] Then $e_1 = \mathbf{unbox}\ \ (\mathbf{box}\ e)$, $e_1' \in Value^1$ and $e \geq_1^1 e_1'$.

$$
\begin{aligned}
[x \overset{1}{\mapsto} e_2]\left(\mathbf{unbox}\ \ (\mathbf{box}\ e)\right) &= \left(\mathbf{unbox}\ \left(\mathbf{box}\ [x \overset{1}{\mapsto} e_2]e\right)\right) \\
&\geq_1^1 [x \overset{1}{\mapsto} e_2]e_1' && (\text{IH, } \mathtt{PUNB_v}[2])
\end{aligned}
$$

- ($\mathtt{PRUN_v}$)[1] Then $e_1 = \mathbf{run}\ e$, $e_1' = \mathbf{run}\ e'$ and $e \geq_1^n e'$.

$$
\begin{aligned}
[x \overset{n}{\mapsto} e_2]\mathbf{run}\ e &= \mathbf{run}\ [x \overset{n}{\mapsto} e_2]e \\
&\geq_1^n \mathbf{run}\ [x \overset{n}{\mapsto} e_2']e' && (\text{IH, } \mathtt{PRUN_v}[1]) \\
&\geq_1^n [x \overset{n}{\mapsto} e_2']\mathbf{run}\ e'
\end{aligned}
$$

- ($\mathtt{PRUN_v}$)[2] Then $e_1 = \mathbf{run}\ \ (\mathbf{box}\ e)$, $e_1' \in Value^1$ and $e \geq_1^1 e_1'$.

$$
\begin{aligned}
[x \overset{0}{\mapsto} e_2]\mathbf{run}\ \ (\mathbf{box}\ e) &= \mathbf{run}\ \left(\mathbf{box}\ [x \overset{n}{\mapsto} e_2]e\right) \\
&\geq_1^n [x \overset{n}{\mapsto} e_2']e_1' && (\text{IH, } \mathtt{PRUN_v}[2])
\end{aligned}
$$

□

Lemma 16 (Expression Constructor Preservation wrt. $\geq_1^n$)**.**

- *If* $\lambda x.e_1 \geq_1^n e_2$ *then* $\exists e_1' \in Expr$ *such that* $e_1 \geq_1^n e_1'$ *and* $(e_2 = \lambda x.e_1'$ *or* $e_2 = \lambda y.[x \overset{0}{\mapsto} y]e_1')$.
- *If* $e_a\ e_b \geq_1^n e_2$ *then* $\exists e_a', e_b', e_a'' \in Expr$ *such that* $(e_i \geq_1^n e_i'$ *for* $i \in \{a, b\}$ *and* $e_2 = e_a'\ e_b')$ *or* $(e_a = \lambda x.e_a'$ *and* $e_2 = [x \overset{0}{\mapsto} e_b']e_a''$ *where* $e_a' \geq_1^0 e_a''$ *and* $e_b \geq_1^0 e_b')$.
- *If* $\textbf{\textit{box}}\ e_1 \geq_1^n e_2$ *then* $\exists e_1' \in Expr$ *such that* $e_1 \geq_1^{n+1} e_1'$ *and* $e_2 = \textbf{\textit{box}}\ e_1'$.
- *If* $\textbf{\textit{unbox}}\ e_1 \geq_1^{n+1} e_2$ *then* $\exists e_1' \in Expr$ *such that* $(e_1 \geq_1^n e_1'$ *and* $e_2 = \textbf{\textit{unbox}}\ e_1')$ *or* $(e_1 = \textbf{\textit{box}}\ e_1'$, $e_2 \in Value^1$ *and* $e_1' \geq_1^1 e_2)$.
- *If* $\textbf{\textit{run}}\ e_1 \geq_1^n e_2$ *then* $\exists e_1' \in Expr$ *such that* $(e_1 \geq_1^n e_1'$ *and* $e_2 = \textbf{\textit{run}}\ e_1')$ *or* $(e_1 = \textbf{\textit{box}}\ e_1'$, $e_2 \in Value^1$ *and* $e_1' \geq_1^1 e_2)$.

Proof. By induction over proof length of $e \geq_1^n e_2$ (IH).

- ($\texttt{PREF}_\texttt{v}$).
  - Case $e = \lambda x.e_1$. Then $e_2 = \lambda x.e_1$, so for $e_1' = e_1$ and by ($\texttt{PREF}_\texttt{v}$) $e_1 \geq_1^n e_1'$ and $e_2 = \lambda x.e_1'$.
  - Case $e = e_a\ e_b$. Then $e_2 = e_a\ e_b$ so for $e_a' = e_a$, $e_b' = e_b$ and by ($\texttt{PREF}_\texttt{v}$) $e_a \geq_1^n e_a'$, $e_b \geq_1^n e_b'$ and $e_2 = e_a'\ e_b'$.
  - Case $e = \textbf{box}\ e_1$. Then $e_2 = \textbf{box}\ e_1$ so for $e_1' = e_1$ and by ($\texttt{PREF}_\texttt{v}$) $e_1 \geq_1^{n+1} e_1'$ and $e_2 = \textbf{box}\ e_1'$.
  - Case $e = \textbf{unbox}\ e_1$. Then $e_2 = \textbf{unbox}\ e_1$ so for $e_1' = e_1$ and by ($\texttt{PREF}_\texttt{v}$) $e_1 \geq_1^{n-1} e_1'$ and $e_2 = \textbf{unbox}\ e_1'$.
  - Case $e = \textbf{run}\ e_1$. Then $e_2 = \textbf{run}\ e_1$ so for $e_1' = e_1$ and by ($\texttt{PREF}_\texttt{v}$) $e_1 \geq_1^n e_1'$ and $e_2 = \textbf{run}\ e_1'$.
- ($\texttt{PAPP}_\texttt{v}$)[1] Then $e = e_a\ e_b$, $e_2 = e_a'\ e_b'$, $e_a \geq_1^n e_a'$ and $e_b \geq_1^n e_b'$.
- ($\texttt{PAPP}_\texttt{v}$)[2] Then $e = (\lambda x.e_a')\ e_b$, $e_2 = [x \overset{0}{\mapsto} e_b']e_a''$, $e_a' \geq_1^0 e_a''$ and $e_b \geq_1^0 e_b'$.
- ($\texttt{PABS}_\texttt{v}$) Then $e = \lambda x.e_1$, $e_2 = \lambda y.e_1'$ and $e_1 \geq_1^n e_1'$.
- ($\texttt{PALP}_\texttt{v}$) Then $e = \lambda x.e_1$, $e_2 = \lambda y.[x \overset{0}{\mapsto} y]e_1'$ and $e_1 \geq_1^0 e_1'$.
- ($\texttt{PBOX}_\texttt{v}$) Then $e = \textbf{box}\ e_1$, $e_2 = \textbf{box}\ e_1'$ and $e_1 \geq_1^{n+1} e_1'$.
- ($\texttt{PUNB}_\texttt{v}$)[1] Then $e = \textbf{unbox}\ e_1$, $e_2 = \textbf{unbox}\ e_1'$ and $e_1 \geq_1^n e_1'$.
- ($\texttt{PUNB}_\texttt{v}$)[2] Then $e = \textbf{unbox}\ (\textbf{box}\ e_1')$, $e_2 \in Value^1$ and $e_1' \geq_1^1 e_2$.
- ($\texttt{PRUN}_\texttt{v}$)[1] Then $e = \textbf{run}\ e_1$, $e_2 = \textbf{run}\ e_1'$ and $e_1 \geq_1^n e_1'$.
- ($\texttt{PRUN}_\texttt{v}$)[2] Then $v = \textbf{run}\ (\textbf{box}\ e_1')$, $e_2 \in Value^1$ and $e_1' \geq_1^1 e_2$.

□

Lemma 17 (Value Preservation wrt. $\geq_1^n$)**.** *Let* $v, v' \in Expr$ *such that* $v \in Value^n$ *and* $v \geq_1^n v'$. *Then* $v' \in Value^n$.

Proof. By induction over the length of proof of $v \geq_1^n v'$ (IH).

- ($\texttt{PREF}_\texttt{v}$) Then $v = v'$. Trivial.
- ($\texttt{PAPP}_\texttt{v}$)[1] Then $v = e_a\ e_b$, $v' = e'_a\ e'_b$, $e_a \geq_1^n e'_a$ and $e_b \geq_1^n e'_b$.
  $e_a, e_b \in Value^n$ and by (IH) $e'_a, e'_b \in Value^n$. So $e'_a\ e'_b \in Value^n$.
- ($\texttt{PAPP}_\texttt{v}$)[2] Then $v = (\lambda y.e)\ v_1$, $v' = [y \overset{0}{\mapsto} v_2]e'$, $e \geq_1^0 e'$ and $v_1 \geq_1^0 v_2$.
  $e, v_1 \in Value^0$ and by (IH) $e', v_2 \in Value^0$ so $[y \overset{0}{\mapsto} v_2]e' \in Value^0$.
- ($\texttt{PABS}_\texttt{v}$) Then $v = \lambda y.e$, $v' = \lambda y.e'$ and $e \geq_1^n e'$.
  $e \in Value^n$ and by (IH) $e' \in Value^n$, so $\lambda y.e' \in Value^n$.
- ($\texttt{PALP}_\texttt{v}$) Then $v = \lambda y.e$, $v' = \lambda z.[y \overset{0}{\mapsto} z]e'$ and $e \geq_1^0 e'$.
  $e \in Value^0$ and by (IH) $e' \in Value^0$, so $\lambda z.[y \overset{0}{\mapsto} z]e' \in Value^0$.
- ($\texttt{PBOX}_\texttt{v}$) Then $v = \textbf{box}\ e$, $v' = \textbf{box}\ e'$ and $e \geq_1^{n+1} e'$.
  $e \in Value^{n+1}$ and by (IH) $e' \in Value^{n+1}$ so $\textbf{box}\ e' \in Value^n$.
- ($\texttt{PUNB}_\texttt{v}$)[1] Then $v = \textbf{unbox}\ e$, $v' = \textbf{unbox}\ e'$ and $e \geq_1^n e'$.
  $e \in Value^{n-1}$ and by (IH) $e' \in Value^{n-1}$, so $\textbf{unbox}\ e' \in Value^n$.
- ($\texttt{PUNB}_\texttt{v}$)[2] Then $v = \textbf{unbox}\ (\textbf{box}\ e)$, $v' \in Value^1$ and $e \geq_1^1 e'_1$. Trivial.
- ($\texttt{PRUN}_\texttt{v}$)[1] Then $v = \textbf{run}\ e$, $v' = \textbf{run}\ e'$ and $e \geq_1^n e'$.
  $e \in Value^n$ and by (IH) $e' \in Value^n$, so $\textbf{run}\ e' \in Value^n$.
- ($\texttt{PRUN}_\texttt{v}$)[2] Then $v = \textbf{run}\ (\textbf{box}\ e)$, $v' \in Value^1$ and $e \geq_1^1 e'_1$. Trivial.

□

Lemma 18 ($\geq_1^n$ Confluence)**.** *Let $e_1, e_2, e_3 \in Expr$ such that $e_1 \geq_1^n e_2$ and $e_1 \geq_1^n e_3$.*
$\exists e_4 \in Expr$ such that $e_2 \geq_1^n e_4$ and $e_3 \geq_1^n e_4$

Proof. By induction over the sum of lengths of proof of $e_1 \geq_1^n e_2$ and $e_1 \geq_1^n e_3$ (IH) and by case analysis of last rule of $e_1 \geq_1^n e_2$.

- ($\texttt{PREF}_\texttt{v}$) Then $e_1 = e_2$ and we can take $e_4 = e_3$.
- ($\texttt{PAPP}_\texttt{v}$)[1] Then $e_1 = e_a\ e_b$, $e_2 = e'_a\ e'_b$, $e_a \geq_1^n e'_a$ and $e_b \geq_1^n e'_b$.
  By Lem.16 we have two cases:
  - Case $e_3 = e''_a\ e''_b$ and $e_i \geq_1^n e''_i$ for $i \in \{a, b\}$. By (IH) $\exists e'''_a, e'''_b \in Expr$ such that $e'_i \geq_1^n e'''_i$ and $e''_i \geq_1^n e'''_i$ for $i \in \{a, b\}$. Then for $e_4 = e'''_a\ e'''_b$ by ($\texttt{PAPP}_\texttt{v}$)[1] we are done.
  - Case $e_a = \lambda x.e$, $e_3 = [x \overset{0}{\mapsto} e''_b]e''_a$ with $e \geq_1^0 e''_a$ and $e_b \geq_1^0 e''_b$. By (IH) $\exists e'''_a, e'''_b \in Expr$ such that $e'_i \geq_1^n e'''_i$ and $e''_i \geq_1^n e'''_i$ for $i \in \{a, b\}$. Then for $e_4 = [x \overset{0}{\mapsto} e'''_b]e'''_a$ by ($\texttt{PAPP}_\texttt{v}$)[2] and Lem.15 we are done.
- ($\texttt{PAPP}_\texttt{v}$)[2] Then $e_1 = (\lambda y.e)\ e_b$, $e_2 = [y \overset{0}{\mapsto} e'_b]e'_a$, $e \geq_1^0 e'_a$ and $e_b \geq_1^0 e'_b$.
  By Lem.16 we also have two cases:
  - Case $e_3 = \lambda y.[x \overset{0}{\mapsto} y]e''_a\ e''_b$ and $e_i \geq_1^0 e''_i$ for $i \in \{a, b\}$. By (IH) $\exists e'''_a, e'''_b \in Expr$ such that $e'_i \geq_1^0 e'''_i$ and $e''_i \geq_1^0 e'''_i$ for $i \in \{a, b\}$. Then for $e_4 = [x \overset{0}{\mapsto} e'''_b]e'''_a$ by ($\texttt{PAPP}_\texttt{v}$)[2] and Lem.15 we are done.

– Case $e_3 = [x \overset{0}{\mapsto} e_b'']e_a''$ with $e \geq_1^0 e_a''$ and $e_b \geq_1^0 e_b''$. By (IH) $\exists e_a''', e_b''' \in Expr$ such that $e_i' \geq_1^n e_i'''$ and $e_i'' \geq_1^n e_i'''$ for $i \in \{a, b\}$. Then for $e_4 = [x \overset{0}{\mapsto} e_b''']e_a'''$ by Lem.15 we are done.

- ($\texttt{PABS}_\texttt{v}$) Then $e_1 = \lambda x.e_1'$, $e_2 = \lambda x.e_2'$ and $e_1' \geq_1^n e_2'$.

  By Lem.16 $e_3 = \lambda x.e_3'$ or $e_3 = \lambda y.[x \overset{0}{\mapsto} y]e_3'$, with $e_1' \geq_1^n e_3'$. By (IH) $\exists e_4' \in Expr$ such that $e_2' \geq_1^n e_4'$ and $e_3' \geq_1^n e_4'$. For $e_4 = \lambda x.e_4'$ and by ($\texttt{PALP}_\texttt{v}$) we are done.

- ($\texttt{PALP}_\texttt{v}$) Then $e_1 = \lambda x.e_1'$, $e_2 = \lambda y.[x \overset{0}{\mapsto} y]e_2'$ and $e_1' \geq_1^0 e_2'$.

  By Lem.16 $e_3 = \lambda x.e_3'$ or $e_3 = \lambda y.[x \overset{0}{\mapsto} y]e_3'$, with $e_1' \geq_1^n e_3'$. By (IH) $\exists e_4' \in Expr$ such that $e_2' \geq_1^n e_4'$ and $e_3' \geq_1^n e_4'$. For $e_4 = \lambda y.[x \overset{0}{\mapsto} y]e_4'$ and by ($\texttt{PALP}_\texttt{v}$) we are done.

- ($\texttt{PBOX}_\texttt{v}$) Then $e_1 = \textbf{box}\ e_1'$, $e_2 = \textbf{box}\ e_2'$ and $e_1' \geq_1^{n+1} e_2'$.

  By Lem.16 $e_3 = \textbf{box}\ e_3'$ with $e_1' \geq_1^{n+1} e_3'$. By (IH) $\exists e_4' \in Expr$ such that $e_2' \geq_1^{n+1} e_4'$ and $e_3' \geq_1^{n+1} e_4'$. For $e_4 = \textbf{box}\ e_4'$ and by ($\texttt{PBOX}_\texttt{v}$) we are done.

- ($\texttt{PUNB}_\texttt{v}$)[1] Then $e_1 = \textbf{unbox}\ e_1'$, $e_2 = \textbf{unbox}\ e_2'$ and $e_1' \geq_1^n e_2'$.

  By Lem.16 we have two cases:

  – Case $\exists e_3' \in Expr$ such that $e_3 = \textbf{unbox}\ e_3'$ and $e_1' \geq_1^{n-1} e_3'$. By (IH) $\exists e_4' \in Expr$ such that $e_2' \geq_1^{n-1} e_4'$ and $e_3' \geq_1^{n-1} e_4'$. For $e_4 = \textbf{unbox}\ e_4'$ and by ($\texttt{PUNB}_\texttt{v}$)[1] we are done.

  – Case $\exists e_1'' \in Expr$ such that $e_1' = \textbf{box}\ e_1''$, $e_1'' \geq_1^1 e_3$ and $e_3 \in Value^1$. By Lem.16 $e_2' = \textbf{box}\ e_2''$ with $e_1'' \geq_1^1 e_2''$. By (IH) $\exists e_4 \in Expr$ such that $e_2'' \geq_1^1 e_4$ and $e_3 \geq_1^1 e_4$. By Lem.17 and ($\texttt{PUNB}_\texttt{v}$)[2] we are done.

- ($\texttt{PUNB}_\texttt{v}$)[2] Then $e_1 = \textbf{unbox}\ (\textbf{box}\ e_1')$, $e_1' \geq_1^1 e_2$ and $e_2 \in Value^1$.

  By Lem.16 we have two cases:

  – Case $\exists e_3' \in Expr$ such that $e_3 = \textbf{unbox}\ (\textbf{box}\ e_3')$ and $e_1' \geq_1^1 e_3'$. By (IH) $\exists e_4 \in Expr$ such that $e_2 \geq_1^1 e_4$ and $e_3' \geq_1^1 e_4$. By ($\texttt{PUNB}_\texttt{v}$)[1] we are done.

  – Case $e_1' \geq_1^1 e_3$ with $e_3 \in Value^1$. By (IH) $\exists e_4 \in Expr$ such that $e_2 \geq_1^1 e_4$ and $e_3 \geq_1^1 e_4$. By Lem.17 $e_4 \in Value^1$ and by ($\texttt{PUNB}_\texttt{v}$)[2] we are done.

- ($\texttt{PRUN}_\texttt{v}$)[1] Then $e_1 = \textbf{run}\ e_1'$, $e_2 = \textbf{run}\ e_2'$ and $e_1' \geq_1^n e_2'$.

  By Lem.16 we have two cases:

  – Case $\exists e_3' \in Expr$ such that $e_3 = \textbf{run}\ e_3'$ and $e_1' \geq_1^n e_3'$. By (IH) $\exists e_4' \in Expr$ such that $e_2' \geq_1^n e_4'$ and $e_4' \geq_1^n e_4'$. For $e_4 = \textbf{run}\ e_4'$ and by ($\texttt{PRUN}$)[1] we are done.

  – Case $\exists e_1'' \in Expr$ such that $e_1' = \textbf{box}\ e_1''$, $e_1'' \geq_1^1 e_3$ and $e_3 \in Value^1$. By Lem.16 $\exists e_2'' \in Expr$ such that $e_2' = \textbf{box}\ e_2''$ and $e_1'' \geq_1^1 e_2''$. By (IH) $\exists e_4 \in Expr$ such that $e_2'' \geq_1^1 e_4$ and $e_3 \geq_1^1 e_4$. By Lem.17 $e_4 \in Value^1$ and by ($\texttt{PRUN}$)[2] we are done.

- ($\texttt{PRUN}_\texttt{v}$)[2] Then $e_1 = \textbf{run}\ (\textbf{box}\ e_1')$, $e_1' \geq_1^1 e_2$ and $e_2 \in Value^1$.

  By Lem.16 we have two cases:

  – Case $\exists e_3' \in Expr$ such that $e_3 = \textbf{run}\ (\textbf{box}\ e_3')$ and $e_1' \geq_1^1 e_3'$. By (IH) $\exists e_4 \in Expr$ such that $e_2 \geq_1^1 e_4$ and $e_3' \geq_1^1 e_4$. By Lem.17 $e_4 \in Value^1$ and by ($\texttt{PRUN}$)[2] we are done.

  – Case $e_1' \geq_1^1 e_3$ and $e_3 \in Value^1$. By (IH) $\exists e_4 \in Expr$ such that $e_2 \geq_1^1 e_4$ and $e_3 \geq_1^1 e_4$. By Lem.17 $e_4 \in Value^1$ and by ($\texttt{PRUN}$)[2] we are done.

□

Lemma 19 ($\geq_{\mathtt{v}}^n$ Confluence)**.** *Let* $e_1, e_2, e_3 \in Expr$ *such that* $e_1 \geq_{\mathtt{v}}^n e_2$ *and* $e_1 \geq_{\mathtt{v}}^n e_3$. $\exists e_4 \in Expr$ *such that* $e_2 \geq_{\mathtt{v}}^n e_4$ *and* $e_3 \geq_{\mathtt{v}}^n e_4$

Proof. By a straightforward induction using Lem.12 and Lem.18. □

Proof of Church-Rosser Theorem (Th.1).
Let's recall the theorem:

Let $e_1, e_2, e_3 \in Expr$ *such that* $e_1 =_{\mathtt{v}}^n e_2$.
$\exists e_3 \in Expr$ *such that* $e_1 \geq_{\mathtt{v}}^n e_3$ *and* $e_2 \geq_{\mathtt{v}}^n e_3$

By induction on the proof length of $e_1 =_{\mathtt{v}}^n e_2$ (IH).

- ($\mathtt{EREF_v}$) Then $e_1 = e_2$ so for $e_3 = e_1$ we are done.
- ($\mathtt{ESYM_v}$) Then $e_2 =_{\mathtt{v}}^n e_1$ with a shorter proof. We can apply (IH).
- ($\mathtt{ETRA_v}$) Then $e_1 =_{\mathtt{v}}^n e$ and $e =_{\mathtt{v}}^n e_2$.
  By (IH) $\exists e_1', e_2' \in Expr$ such that $e_1 \geq_{\mathtt{v}}^n e_1'$, $e \geq_{\mathtt{v}}^n e_1'$, $e \geq_{\mathtt{v}}^n e_2'$ and $e_2 \geq_{\mathtt{v}}^n e_2'$. Then by Lem.19 $\exists e_3 \in Expr$ such that $e_1' \geq_{\mathtt{v}}^n e_3$ and $e_2' \geq_{\mathtt{v}}^n e_3$. By ($\mathtt{ETRA_v}$).
- ($\mathtt{EAPP_v}$)[1] Then $e_1 = e_a\ e_b$, $e_2 = e_a'\ e_b$ and $e_a =_{\mathtt{v}}^n e_a'$.
  By (IH) $\exists e_3' \in Expr$ such that $e_a \geq_{\mathtt{v}}^n e_3'$ and $e_a' \geq_{\mathtt{v}}^n e_3'$. Then for $e_3 = e_3'\ e_b$ and by ($\mathtt{EAPP_v}$)[1].
- ($\mathtt{EAPP_v}$)[2] Similar to previous case.
- ($\mathtt{EAPP_v}$)[3] Then $e_1 = (\lambda x.e'')\ v$ and $e_2 = [x \overset{0}{\mapsto} v]e''$.
  For $e_3 = e_2$ and using ($\mathtt{EAPP_v}$)[3] and ($\mathtt{EREF_v}$).
- ($\mathtt{EALP_v}$) Then $e_1 = \lambda x.e$ and $e_2 = \lambda y.[x \overset{0}{\mapsto} y]e$.
  For $e_3 = e_2$ and using ($\mathtt{EALP_v}$) and ($\mathtt{EREF_v}$).
- ($\mathtt{EABS_v}$) Then $e_1 = \lambda x.e$, $e_2 = \lambda x.e'$ and $e =_{\mathtt{v}}^n e'$.
  By (IH) $\exists e_3' \in Expr$ such that $e \geq_{\mathtt{v}}^n e_3'$ and $e' \geq_{\mathtt{v}}^n e_3'$. Then for $e_3 = \lambda x.e_3'$ and by ($\mathtt{EABS_v}$).
- ($\mathtt{EBOX_v}$) Then $e_1 = \mathbf{box}\ e$, $e_2 = \mathbf{box}\ e'$ and $e =_{\mathtt{v}}^{n+1} e'$.
  By (IH) $\exists e_3' \in Expr$ such that $e \geq_{\mathtt{v}}^{n+1} e_3'$ and $e' \geq_{\mathtt{v}}^{n+1} e_3'$. Then for $e_3 = \mathbf{box}\ e_3'$ and by ($\mathtt{EBOX_v}$).
- ($\mathtt{EUNB_v}$)[1] Then $e_1 = \mathbf{unbox}\ e$, $e_2 = \mathbf{unbox}\ e'$ and $e =_{\mathtt{v}}^{n-1} e'$.
  By (IH) $\exists e_3' \in Expr$ such that $e \geq_{\mathtt{v}}^{n-1} e_3'$ and $e' \geq_{\mathtt{v}}^{n-1} e_3'$. Then for $e_3 = \mathbf{unbox}\ e_3'$ and by ($\mathtt{EUNB_v}$)[1].
- ($\mathtt{EUNB_v}$)[2] Then $e_1 = \mathbf{unbox}\ (\mathbf{box}\ e_2)$, $e_2 \in Value^1$.
  For $e_3 = e_2$ and by ($\mathtt{EUNB_v}$)[2] and ($\mathtt{EREF_v}$).
- ($\mathtt{ERUN_v}$)[1] Then $e_1 = \mathbf{run}\ e$, $e_2 = \mathbf{run}\ e'$ and $e =_{\mathtt{v}}^n e'$.
  By (IH) $\exists e_3' \in Expr$ such that $e \geq_{\mathtt{v}}^n e_3'$ and $e' \geq_{\mathtt{v}}^n e_3'$. Then for $e_3 = \mathbf{run}\ e_3'$ and by ($\mathtt{ERUN_v}$)[1].
- ($\mathtt{ERUN_v}$)[2] Then $e_1 = \mathbf{run}\ (\mathbf{box}\ e_2)$, $e_2 \in Value^1$.
  For $e_3 = e_2$ and by ($\mathtt{ERUN_v}$)[2] and ($\mathtt{EREF_v}$).

□

4.2 Proof of Indifference and Simulation Theorems

Lemma 20 (Context Stack Length Preservation wrt. $\bowtie$)**.** *Let K_1 and K_2 be two context stacks. $length(K_1 \bowtie K_2) = max(length(K_1), length(K_2))$.*

Proof. By induction on structure of K_1, K_2 and definition of $\bowtie$.

- $K_1 = \bot$. Then $length(K_2) \geq length(K_1)$ and $length(K_1 \bowtie K_2) = length(K_2)$ si $length(K_1 \bowtie K_2) = max(length(K_1), length(K_2))$.
- $K_2 = \bot$. Then $length(K_1) \geq length(K_2)$ and $length(K_1 \bowtie K_2) = length(K_1)$ si $length(K_1 \bowtie K_2) = max(length(K_1), length(K_2))$.
- $K_1 = (K_1', \kappa_1)$ and $K_2 = (K_2', \kappa_2)$.

$$\begin{aligned} length(K_1 \bowtie K_2) &= length(K_1' \bowtie K_2', \kappa_1[\kappa_2]) \\ &= length(K_1' \bowtie K_2') + 1 \\ &= max(length(K_1'), length(K_2')) + 1 && \text{(IH)} \\ &= max(length(K_1') + 1, length(K_2') + 1) \\ &= max(length(K_1', \kappa_1), length(K_2', \kappa_2)) \\ &= max(length(K_1), length(K_2)) \end{aligned}$$

□

Proof of Lem.1. By structural induction on *Expr* (IH).

- $e = x$. $depth(x) = 0$. By (TVAR) $K = \bot$. Hence $depth(x) = length(K)$.
- $e = \lambda x.e'$, $e' \mapsto (\underline{e'}, K')$.

$$\begin{aligned} depth(e) &= depth(e') \\ &= length(K') && \text{(IH)} \\ &= length(K) && \text{(TABS)} \end{aligned}$$

- $e = e_1\ e_2$, $e_1 \mapsto (\underline{e_1}, K_1)$, $e_2 \mapsto (\underline{e_2}, K_2)$.

$$\begin{aligned} depth(e) &= max(depth(e_1), depth(e_2)) \\ &= max(length(K_1), length(K_2)) && \text{(IH)} \\ &= length(K_1 \bowtie K_2) && \text{(Lem.20)} \\ &= length(K) && \text{(TAPP)} \end{aligned}$$

- $e = \mathbf{box}\ e'$,
  - Case $e' \mapsto (\underline{e'}, (K', \kappa))$.

$$\begin{aligned} depth(e) &= max(depth(e') - 1, 0) \\ &= max(length(K', \kappa) - 1, 0) && \text{(IH)} \\ &= max(length(K'), 0) \\ &= length(K') && (length(K') \geq 0) \\ &= length(K) - 1 && \text{(TBOX)} \end{aligned}$$

– Case $e' \mapsto (\underline{e'}, \bot)$.

$$\begin{aligned} depth(e) &= max(depth(e') - 1, 0) \\ &= max(length(\bot) - 1, 0) && \text{(IH)} \\ &= 0 \\ &= length(K) && \text{(TBOX)} \end{aligned}$$

- $e = \textbf{unbox}\ e'$, $e' \mapsto (\underline{e'}, K')$.

$$\begin{aligned} depth(e) &= depth(e') + 1 \\ &= depth(K') + 1 && \text{(IH)} \\ &= depth(K) && \text{(TUNB)} \end{aligned}$$

- $e = \textbf{run}\ e'$, $e' \mapsto (\underline{e'}, K')$.

$$\begin{aligned} depth(e) &= depth(e') \\ &= depth(K') && \text{(IH)} \\ &= depth(K) && \text{(TRUN)} \end{aligned}$$

□

Lemma 21 (Context Stack Length Upper Bound for Values)**.**

$$\frac{v \in Value^n \qquad v \mapsto (\underline{v}, K)}{length(K) < n}$$

Proof. By structural induction over v and case analysis over $Value^n (n > 0)$ (H1).

- $v = x$. By (TVAR), $K = \bot$ so $length(K) = 0 < n$.
- $v = \lambda x.v'$, $v' \in Value^n$.
  By (H1), $v' \mapsto (\underline{v'}, K)$ and $length(K) < n$.
  By (TABS) $v \mapsto (\underline{v}, K)$.
- $v = v_1\ v_2$, $v_1, v_2 \in Value^n$.
  By (H1), $v_i \mapsto (\underline{v_i}, K_i)$ and $length(K_i) < n$. , $i \in \{1, 2\}$.
  By (TAPP), $v \mapsto (\underline{v}, K_1 \bowtie K_2)$.
  By Lem.20, $length(K_1 \bowtie K_2) = max(length(K_1), length(K_2)) < n$.
- $v = \textbf{box}\ v'$, $v' \in Value^{n+1}$.
  By (H1), $v' \mapsto (\underline{v'}, K)$ and $length(K) < n + 1$.
  By (TBOX)
  - $K = \bot$. $v \mapsto (\underline{v}, \bot)$ and $length(\bot) = 0 < n$
  - $K = K', \kappa$. $v \mapsto (\underline{v}, K')$ and $length(K') < length(K) - 1 < n$
- $v = \textbf{unbox}\ v'$, $v' \in Value^{n-1}$.
  Because $n > 1$, $v' \in Value^n (n > 0)$ and we can apply (H1).
  By (H1), $v' \mapsto (\underline{v'}, K)$ and $length(K) < n - 1$.
  By (TUNB), $v \mapsto (\underline{v}, (K, \kappa))$.
  Then $length(K, \kappa) = length(K) + 1 < n$.
- $v = \textbf{run}\ v'$, $v' \in Value^n$.
  By (H1), $v' \mapsto (\underline{v'}, K)$ and $length(K) < n$.
  By (TRUN), $v \mapsto (\underline{v}, K)$.

□

Proof of Lem.2. As $depth(v) = 0$, $v \mapsto (\underline{v}, \bot)$ Lem.1.
We now prove by cases on definition of $Value^0$ that $\underline{v} = \lambda k.k\ \Psi(v)$.

- $v = x$.

$$\begin{aligned} \underline{x} &= \lambda k.\,(k\ x) && \text{(TVAR)} \\ &= \lambda k.k\ \Psi(x) \end{aligned}$$

- $v = \lambda x.e$.

$$\begin{aligned} \underline{\lambda x.e} &= \lambda k.k\ \lambda x.\underline{e} && \text{(TABS)} \\ &= \lambda k.k\ \Psi(x) \end{aligned}$$

- $v = \mathbf{box}\ v'$, $v' \in Value^1$.

$$\begin{aligned} \underline{\mathbf{box}\ v'} &= \lambda k.k\ \ (\mathbf{box}\ \underline{v'}) && \text{(TBOX, Lem.21)} \\ &= \lambda k.k\ \Psi(\mathbf{box}\ v') \end{aligned}$$

□

Let's define the `NewVars` predicate over contexts:

$$\texttt{NewVars}(e\ \lambda h.[\cdot]) = \{h\} \quad ; \quad \texttt{NewVars}(e\ \lambda h.\kappa) = \{h\} \cup \texttt{NewVars}(\kappa)$$

Lemma 22 (Substitution Preservation wrt. Context Closure)**.** *Let $v \in Value^0$, $\kappa \in Context$, $e \in Expr \cup Context$ and $x \notin \texttt{NewVars}(\kappa)$.*

$$[x \overset{n}{\mapsto} v]\,(\kappa[e]) = \Big([x \overset{n}{\mapsto} v]\kappa\Big)\Big[[x \overset{n}{\mapsto} v]e\Big]$$

Proof. By induction over the definition of substitution extension (IH).

- $\kappa = (e'\ \lambda h.[\cdot])$.

$$\begin{aligned} [x \overset{n}{\mapsto} v]\,((e'\ \lambda h.[\cdot])\,[e]) &= [x \overset{n}{\mapsto} v]\,(e'\ \lambda h.e) \\ &= \Big([x \overset{0}{\mapsto} v]e'\ \lambda h.[x \overset{n}{\mapsto} v]e\Big) && (x \notin \texttt{NewVars}(\kappa)) \\ &= \Big([x \overset{0}{\mapsto} v]e'\ \lambda h.[\cdot]\Big)\,[x \overset{n}{\mapsto} v]e \\ &= \Big([x \overset{0}{\mapsto} v]\,(e'\ \lambda h.[\cdot])\Big)\,[x \overset{n}{\mapsto} v]e \end{aligned}$$

- $\kappa = (e'\ \lambda h.\kappa')$.

$$\begin{aligned} [x \overset{n}{\mapsto} v]\,((e'\ \lambda h.\kappa')\,[e]) &= [x \overset{n}{\mapsto} v]\,(e'\ \lambda h.\kappa'[e]) \\ &= \Big([x \overset{0}{\mapsto} v]e'\ \lambda h.[x \overset{n}{\mapsto} v]\,(\kappa'[e])\Big) && (x \notin \texttt{NewVars}(\kappa)) \\ &= \Big([x \overset{0}{\mapsto} v]e'\ \lambda h.\Big([x \overset{n}{\mapsto} v]\kappa'\Big)\Big[[x \overset{n}{\mapsto} v]e\Big]\Big) && \text{(IH)} \\ &= \Big([x \overset{0}{\mapsto} v]e'\ \lambda h.[x \overset{n}{\mapsto} v]\kappa'\Big)\,[x \overset{n}{\mapsto} v]e \\ &= \Big([x \overset{0}{\mapsto} v]\,(e'\ \lambda h.\kappa')\Big)\,[x \overset{n}{\mapsto} v]e \end{aligned}$$

□

Lemma 23 (Substitution Preservation wrt. $\bowtie$ Extension)**.** *Let $v \in Value^0$ and K_1, K_2 two context stacks and $n \geq length(K_1 \bowtie K_2)$.*

$$[x \overset{n}{\mapsto} v](K_1 \bowtie K_2) = [x \overset{n}{\mapsto} v]K_1 \bowtie [x \overset{n}{\mapsto} v]K_2$$

Proof. By induction over the definition of context stack merge operator (IH).

- $K_1 = \bot$. $[x \overset{n}{\mapsto} v](\bot \bowtie K_2) = [x \overset{n}{\mapsto} v]K_2 = [x \overset{n}{\mapsto} v]\bot \bowtie [x \overset{n}{\mapsto} v]K_2$.
- $K_2 = \bot$. $[x \overset{n}{\mapsto} v](K_1 \bowtie \bot) = [x \overset{n}{\mapsto} v]K_1 = [x \overset{n}{\mapsto} v]K_1 \bowtie [x \overset{n}{\mapsto} v]\bot$.
- $K_1 = K_1', \kappa_1$ and $K_2 = K_2', \kappa_2$.

$$\begin{aligned}
[x \overset{n}{\mapsto} v](K_1', \kappa_1 \bowtie K_2', \kappa_2) &= [x \overset{n}{\mapsto} v](K_1' \bowtie K_2', \kappa_1[\kappa_2]) \\
&= [x \overset{m}{\mapsto} v](K_1' \bowtie K_2'), [x \overset{n}{\mapsto} v]\kappa_1[\kappa_2] && (\text{m} = \max(\text{n-1},0)) \\
&= [x \overset{m}{\mapsto} v](K_1' \bowtie K_2'), \Big([x \overset{n}{\mapsto} v]\kappa_1\Big)\Big[[x \overset{n}{\mapsto} v]\kappa_2\Big] && (\text{Lem.22}) \\
&= [x \overset{m}{\mapsto} v]K_1' \bowtie [x \overset{m}{\mapsto} v]K_2', \Big([x \overset{n}{\mapsto} v]\kappa_1\Big)\Big[[x \overset{n}{\mapsto} v]\kappa_2\Big] && (\text{IH}) \\
&= \Big([x \overset{m}{\mapsto} v]K_1', [x \overset{n}{\mapsto} v]\kappa_1\Big) \bowtie \Big([x \overset{m}{\mapsto} v]K_2', [x \overset{n}{\mapsto} v]\kappa_2\Big) \\
&= [x \overset{n}{\mapsto} v](K_1', \kappa_1) \bowtie [x \overset{n}{\mapsto} v](K_2', \kappa_2)
\end{aligned}$$

□

Proof of Lem.3. By structural induction over *Expr*.

- $e = y$
  - Case $y = x$ and $n = 0$.

$$\begin{aligned}
[x \overset{0}{\mapsto} v]x &= v \\
&\mapsto (\lambda k.k\ \Psi(v), \bot) && (\text{Lem.2}) \\
&\mapsto ([x \overset{0}{\mapsto} \Psi(v)](\lambda k.k\ x), [x \overset{0}{\mapsto} \Psi(v)]\bot) \\
&\mapsto ([x \overset{0}{\mapsto} \Psi(v)]\underline{x}, [x \overset{0}{\mapsto} \Psi(v)]K) && (\text{TVAR})
\end{aligned}$$

  - Case $y \neq x$ or $n > 0$.

$$\begin{aligned}
[x \overset{n}{\mapsto} v]y &= y \\
&\mapsto (\lambda k.k\ y, \bot) && (\text{TVAR}) \\
&\mapsto ([x \overset{n}{\mapsto} \Psi(v)](\lambda k.k\ y), [x \overset{n-1}{\mapsto} \Psi(v)]\bot) \\
&\mapsto ([x \overset{n}{\mapsto} \Psi(v)]\underline{y}, [x \overset{n-1}{\mapsto} \Psi(v)]K) && (\text{TVAR})
\end{aligned}$$

- $e = \lambda y.e'$

– Case $x = y$ and $n = 0$. $e' \mapsto (\underline{e'}, K')$.
As $depth(e') = depth(e) = length(K) = length(K') = 0$, $K' = \bot$.

$$\begin{aligned}
[x \overset{0}{\mapsto} v]\lambda x.e' &= \lambda x.e' \\
&\mapsto (\lambda k.k\ \lambda x.\underline{e'}, \bot) && \text{(TABS)} \\
&\mapsto ([x \overset{0}{\mapsto} \Psi(v)]\,(\lambda k.k\ \lambda x.\underline{e'})\,, [x \overset{0}{\mapsto} \Psi(v)]\bot) \\
&\mapsto ([x \overset{0}{\mapsto} \Psi(v)]\underline{\lambda x.e'}, [x \overset{0}{\mapsto} \Psi(v)]\bot) && \text{(TABS)}
\end{aligned}$$

– Case $y \neq x$ or $n > 0$. $e' \mapsto (\underline{e'}, K')$.
By (IH) $[x \overset{n}{\mapsto} v]e' \mapsto ([x \overset{n}{\mapsto} \Psi(v)]\underline{e'}, [x \overset{n-1}{\mapsto} \Psi(v)]K')$. (1)

$$\begin{aligned}
[x \overset{n}{\mapsto} v]\lambda y.e' &= \lambda y.[x \overset{n}{\mapsto} v]e' \\
&\mapsto (\lambda k.k\ \lambda y.[x \overset{n}{\mapsto} \Psi(v)]\underline{e'}, [x \overset{n-1}{\mapsto} \Psi(v)]K') && \text{(TABS, 1)} \\
&\mapsto ([x \overset{n}{\mapsto} \Psi(v)]\,(\lambda k.k\ \lambda y.\underline{e'})\,, [x \overset{n-1}{\mapsto} \Psi(v)]K') \\
&\mapsto ([x \overset{n}{\mapsto} \Psi(v)]\underline{\lambda y.e'}, [x \overset{n-1}{\mapsto} \Psi(v)]K') && \text{(TABS)}
\end{aligned}$$

- $e = e_1\ e_2$. $e_i \mapsto (\underline{e_i}, K_i)$ for $i = 1, 2$
By (IH) $[x \overset{n}{\mapsto} v)]e_i \mapsto ([x \overset{n}{\mapsto} \Psi(v)\underline{e_i}, [x \overset{m}{\mapsto} \Psi(v)]K_i)$. (1)

$$\begin{aligned}
[x \overset{n}{\mapsto} v]\,(e_1\ e_2) &= [x \overset{n}{\mapsto} v)]e_1\ [x \overset{n}{\mapsto} v)]e_2 \\
&\mapsto (\lambda k.[x \overset{n}{\mapsto} \Psi(v)]\underline{e_1}\ \left(\lambda m.[x \overset{n}{\mapsto} \Psi(v)]\underline{e_2}\,(\lambda n.\,((m\ n)\ k))\right), \\
&\qquad [x \overset{m}{\mapsto} \Psi(v)]K_1 \bowtie [x \overset{m}{\mapsto} \Psi(v)]K_2) && \text{(TAPP, 1)} \\
&\mapsto ([x \overset{n}{\mapsto} \Psi(v)]\left(\lambda k.\underline{e_1}\ \left(\lambda m.\underline{e_2}\,(\lambda n.\,((m\ n)\ k))\right)\right), \\
&\qquad [x \overset{m}{\mapsto} \Psi(v)]\,(K_1 \bowtie K_2)) && \text{(Lem.23)} \\
&\mapsto \left([x \overset{n}{\mapsto} \Psi(v)]\left(\underline{e_1\ e_2}\right), [x \overset{m}{\mapsto} \Psi(v)]\,(K_1 \bowtie K_2)\right) && \text{(TAPP)}
\end{aligned}$$

- $e = \mathbf{box}\ e'$. $e' \mapsto (\underline{e'}, K')$.
By (IH) $[x \overset{n}{\mapsto} v]e' \mapsto ([x \overset{n}{\mapsto} \Psi(v)]\underline{e'}, [x \overset{n}{\mapsto} \Psi(v)]K')$. (1)
Case $K' = \bot$.

$$\begin{aligned}
[x \overset{n}{\mapsto} v]\mathbf{box}\ e' &= \mathbf{box}\ [x \overset{n+1}{\mapsto} v]e' \\
&\mapsto \left(\lambda k.k\ \left(\mathbf{box}\ [x \overset{n+1}{\mapsto} \Psi(v)]\underline{e'}\right), \bot\right) && \text{(TBOX, 1)} \\
&\mapsto \left([x \overset{n}{\mapsto} \Psi(v)]\,(\lambda k.k\ \ (\mathbf{box}\ \underline{e'}))\,, \bot\right) \\
&\mapsto \left([x \overset{n}{\mapsto} \Psi(v)]\,(\underline{\mathbf{box}\ e'})\,, [x \overset{m}{\mapsto} \Psi(v)]\bot\right) && \text{(TBOX)}
\end{aligned}$$

Case $K' = K'', \kappa$.

$$[x \overset{n}{\mapsto} v]\mathbf{box}\ e' = \mathbf{box}\ [x \overset{n+1}{\mapsto} v]e'$$
$$\mapsto \left(\lambda k.k\ ([x \overset{n}{\mapsto} \Psi(v)]\kappa)\left[\mathbf{box}\ [x \overset{n+1}{\mapsto} \Psi(v)]\underline{e'}\right], [x \overset{m}{\mapsto} \Psi(v)]K''\right) \quad \text{(TBOX, 1)}$$
$$\mapsto \left([x \overset{n}{\mapsto} \Psi(v)]\left(\lambda k.k\ \kappa[\mathbf{box}\ \underline{e'}]\right), [x \overset{m}{\mapsto} \Psi(v)]K''\right) \quad \text{(Lem.22)}$$
$$\mapsto \left([x \overset{n}{\mapsto} \Psi(v)]\left(\underline{\mathbf{box}\ e'}\right), [x \overset{m}{\mapsto} \Psi(v)]K''\right) \quad \text{(TBOX)}$$

- $e = \mathbf{unbox}\ e'$. $e' \mapsto (\underline{e'}, K')$.
  As $length(K) = n$, $n > 0$ by (TUNB).
  By (IH) $[x \overset{n}{\mapsto} v]e' \mapsto ([x \overset{n}{\mapsto} \Psi(v)]\underline{e'}, [x \overset{n}{\mapsto} \Psi(v)]K')$. (1)

  $$[x \overset{n}{\mapsto} v]\mathbf{unbox}\ e' = \mathbf{unbox}\ [x \overset{n-1}{\mapsto} v]e'$$
  $$\mapsto \left(\mathbf{unbox}\ h, [x \overset{m}{\mapsto} \Psi(v)]K', [x \overset{n-1}{\mapsto} \Psi(v)]\underline{e'}\ (\lambda h.[\cdot])\right) \quad \text{(TUNB,1)}$$
  $$\mapsto \left(\mathbf{unbox}\ h, [x \overset{n-1}{\mapsto} \Psi(v)]\left(K', \underline{e'}\ (\lambda h.[\cdot])\right)\right)$$
  $$\mapsto \left([x \overset{n}{\mapsto} \Psi(v)]\left(\mathbf{unbox}\ h\right), [x \overset{n-1}{\mapsto} \Psi(v)]\left(K', \underline{e'}\ (\lambda h.[\cdot])\right)\right)$$
  $$\mapsto \left([x \overset{n}{\mapsto} \Psi(v)]\underline{\mathbf{unbox}\ e'}, [x \overset{n-1}{\mapsto} \Psi(v)]K\right) \quad \text{(TUNB)}$$

- $e = \mathbf{run}\ e'$. $e' \mapsto (\underline{e'}, K')$.
  By (IH) $[x \overset{n}{\mapsto} v]e' \mapsto ([x \overset{n}{\mapsto} \Psi(v)]\underline{e'}, [x \overset{n}{\mapsto} \Psi(v)]K')$. (1)

  $$[x \overset{n}{\mapsto} v]\mathbf{run}\ e' = \mathbf{run}\ [x \overset{n}{\mapsto} v]e'$$
  $$\mapsto \left(\lambda k.[x \overset{n}{\mapsto} \Psi(v)]\underline{e'}\ (\lambda m.\mathbf{run}\ (m\ k)), [x \overset{m}{\mapsto} \Psi(v)]K'\right) \quad \text{(TRUN, 1)}$$
  $$\mapsto \left([x \overset{n}{\mapsto} \Psi(v)]\left(\lambda k.\underline{e'}\ (\lambda m.\mathbf{run}\ (m\ k))\right), [x \overset{m}{\mapsto} \Psi(v)]K'\right)$$
  $$\mapsto \left([x \overset{n}{\mapsto} \Psi(v)]\underline{\mathbf{run}\ e'}, [x \overset{m}{\mapsto} \Psi(v)]K\right) \quad \text{(TRUN)}$$

□

Lemma 24 (Free (resp. Closed) Variables Preservation wrt. $\bowtie$)**.** *Let $K_1, K_2 \in$ Context Stack such that $\forall m, CV^m(K_1) \cap CV^m(K_2) = \varnothing$*

$$FV^n(K_1 \bowtie K_2) = FV^n(K_1) \cup FV^n(K_2)$$
$$CV^n(K_1 \bowtie K_2) = CV^n(K_1) \cup CV^n(K_2)$$

Proof. By induction over the size of $K_1 \bowtie K_2$ and case analysis of K_1 and K_2 (IH). The proof is done with FV but it is strictly the same for CV.

- $K_1 = \bot$. $FV^n(K_1 \bowtie K_2) = FV^n(K_2) = FV^n(\bot) \cup FV^n(K_2)$.
- $K_2 = \bot$. $FV^n(K_1 \bowtie K_2) = FV^n(K_1) = FV^n(K_1) \cup FV^n(\bot)$.

- $K_1 = K_1', \kappa_1$ and $K_2 = K_2', \kappa_2$.
  As $Indep(K_1, K_2)$, $Indep(\kappa_1, \kappa_2)$ and then $FV^n(\kappa_1[\kappa_2]) = FV^n(\kappa_1) \cup FV^n(\kappa_2)$. (1)

$$\begin{aligned}
FV^n((K_1', \kappa_1) \bowtie (K_2', \kappa_2)) &= FV^n((K_1' \bowtie K_2'), \kappa_1[\kappa_2]) \\
&= FV^{n-1}(K_1' \bowtie K_2') \cup FV^n(\kappa_1[\kappa_2]) \\
&= FV^{n-1}(K_1') \cup FV^{n-1}(K_2') \cup FV^n(\kappa_1[\kappa_2]) && \text{(IH)} \\
&= FV^{n-1}(K_1') \cup FV^{n-1}(K_2') \cup FV^n(\kappa_1) \cup FV^n(\kappa_2) && (1) \\
&= FV^n(K_1', \kappa_1) \cup FV^n(K_2', \kappa_2)
\end{aligned}$$

□

Proof of Lem.4. By induction over the size of e and case analysis of transformation rules (IH).

- $e = x$
  - $CV^0(\bot) \cup FV^0(x) = \varnothing \cup \{x\} = FV^0(\lambda k.(k\ x)) \cup FV^0(\bot)$
  - $CV^n(\bot) \cup FV^{n+1}(x) = \varnothing \cup \varnothing = FV^{n+1}(\lambda k.(k\ x)) \cup FV^n(\bot)$.
- $e = \lambda x.e'$ with $e' \mapsto (\underline{e'}, K)$.

$$\begin{aligned}
CV^0(\bot) \cup FV^0(\lambda x.e') &= FV^0(e') \setminus \{x\} \\
&= \left(FV^0(\underline{e'}) \cup FV^0(\bot)\right) \setminus \{x\} && \text{(IH)} \\
&= \left(FV^0(\lambda k.(k\ \lambda x.\underline{e'})) \cup FV^0(\bot)\right) \setminus \{x\} \\
&= \left(FV^0(\lambda k.(k\ \lambda x.\underline{e'})) \setminus \{x\}\right) \cup FV^0(\bot) \\
&= FV^0(\underline{\lambda x.e'}) \cup FV^0(\bot)
\end{aligned}$$

$$\begin{aligned}
CV^n(K) \cup FV^{n+1}(\lambda x.e') &= CV^n(K) \cup FV^{n+1}(e') \\
&= CV^n(K) \cup FV^{n+1}(\underline{e'}) \cup FV^n(K) && \text{(IH)} \\
&= FV^{n+1}(\underline{e'}) \cup FV^n(K) && \text{(IH)} \\
&= FV^{n+1}(\lambda k.(k\ \lambda x.\underline{e'})) \cup FV^n(K) \\
&= FV^{n+1}(\underline{\lambda x.e'}) \cup FV^n(K)
\end{aligned}$$

- $e = e_1\ e_2$ with $e_i \mapsto (\underline{e_i'}, K_i)$, $i \in \{1, 2\}$. $m = max(n-1, 0)$.

$$\begin{aligned}
&CV^m(K_1 \bowtie K_2) \cup FV^n(e_1\ e_2) \\
&= CV^m(K_1) \cup CV^m(K_2) \cup FV^n(e_1) \cup FV^n(e_2) && \text{(Lem.24, A1)} \\
&= CV^m(K_1) \cup CV^m(K_2) \cup FV^n(\underline{e_1}) \cup FV^n(\underline{e_2}) \cup FV^m(K_1) \cup FV^m(K_2) && \text{(IH)} \\
&= FV^n(\underline{e_1}) \cup FV^n(\underline{e_2}) \cup FV^m(K_1) \cup FV^m(K_2) && \text{(IH)} \\
&= FV^n(\underline{e_1}) \cup FV^n(\underline{e_2}) \cup FV^m(K_1 \bowtie K_2) && \text{(Lem.24, A1)} \\
&= FV^n\left(\lambda k.\underline{e_1}\ \left(\lambda m.\underline{e_2}\ \left(\lambda n.((m\ n)\ k)\right)\right)\right) \cup FV^m(K_1 \bowtie K_2) \\
&= FV^n\left(\underline{e_1\ e_2}\right) \cup FV^m(K)
\end{aligned}$$

- $e = \mathbf{box}\ e'$ with $e' \mapsto (\underline{e'}, K')$.
  - Case $K' = \bot$.

$$\begin{aligned} CV^0(\bot) \cup FV^0(\mathbf{box}\ e') &= FV^1(e') \\ &= FV^1(\underline{e'}) \cup FV^0(\bot) && \text{(IH)} \\ &= FV^0(\lambda k.k\ \ (\mathbf{box}\ \underline{e'})) \cup FV^0(\bot) \\ &= FV^0(\underline{\mathbf{box}\ e'}) \cup FV^0(\bot) \end{aligned}$$

  - Case $K' = K, \kappa$ and $n = 0$. Then $K = \bot$.

$$\begin{aligned} & CV^0(\bot) \cup FV^0(\mathbf{box}\ e') \\ &= FV^1(e') \\ &= \big(FV^1(\underline{e'}) \cup FV^0(K,\kappa)\big) \setminus CV^0(K,\kappa) && \text{(IH)} \\ &= \big(FV^1(\underline{e'}) \cup FV^0(K,\kappa)\big) \setminus CV^0(\kappa) \\ &= \big(FV^0(\mathbf{box}\ \underline{e'}) \cup FV^0(K) \cup FV^0(\kappa)\big) \setminus CV^0(\kappa) \\ &= \big(\big(FV^0(k\ \mathbf{box}\ \underline{e'}) \setminus \{k\}\big) \cup FV^0(K) \cup FV^0(\kappa)\big) \setminus CV^0(\kappa) \\ &= \big(\big(FV^0(k\ \mathbf{box}\ \underline{e'}) \cup FV^0(\kappa)\big) \setminus CV^0(\kappa) \setminus \{k\}\big) \cup FV^0(K) && \text{(A1)} \\ &= \big(FV^0(\kappa[k\ \mathbf{box}\ \underline{e'}]) \setminus \{k\}\big) \cup FV^0(K) && \text{(A1)} \\ &= FV^0(\lambda k.\kappa[k\ \mathbf{box}\ \underline{e'}]) \cup FV^0(K) \\ &= FV^0(\underline{\mathbf{box}\ e'}) \cup FV^0(K) \end{aligned}$$

  - Case $K' = K, \kappa$ and $n > 0$

$$\begin{aligned} & CV^m(K) \cup FV^n(\mathbf{box}\ e') \\ &= CV^m(K) \cup FV^{n+1}(e') \\ &= CV^m(K) \cup \big(\big(FV^{n+1}(\underline{e'}) \cup FV^n(K,\kappa)\big) \setminus CV^n(K,\kappa)\big) && \text{(IH)} \\ &= CV^m(K) \cup \big(\big(FV^{n+1}(\underline{e'}) \cup FV^n(K,\kappa)\big) \setminus CV^m(K) \setminus CV^n(\kappa)\big) \\ &= CV^m(K) \cup \big(\big(FV^{n+1}(\underline{e'}) \cup FV^n(K,\kappa)\big) \setminus CV^n(\kappa)\big) \\ &= (CV^n(K,\kappa) \setminus CV^n(\kappa)) \cup \big(\big(FV^{n+1}(\underline{e'}) \cup FV^n(K,\kappa)\big) \setminus CV^n(\kappa)\big) \\ &= \big(CV^n(K,\kappa) \cup FV^{n+1}(\underline{e'}) \cup FV^n(K,\kappa)\big) \setminus CV^n(\kappa) \\ &= \big(FV^{n+1}(\underline{e'}) \cup FV^n(K,\kappa)\big) \setminus CV^n(\kappa) && \text{(IH)} \\ &= \big(FV^{n+1}(\underline{e'}) \cup FV^m(K) \cup FV^n(\kappa)\big) \setminus CV^n(\kappa) \\ &= (FV^n(k\ \mathbf{box}\ \underline{e'}) \cup FV^m(K) \cup FV^n(\kappa)) \setminus CV^n(\kappa) \\ &= FV^n(\kappa[k\ \mathbf{box}\ \underline{e'}]) \cup FV^m(K) && \text{(A1)} \\ &= FV^n(\lambda k.\kappa[k\ \mathbf{box}\ \underline{e'}]) \cup FV^m(K) \\ &= FV^n(\underline{\mathbf{box}\ e'}) \cup FV^m(K) \end{aligned}$$

- $e = \textbf{unbox}\ e'$ with $e' \mapsto (\underline{e'}, K')$.

$$
\begin{aligned}
& CV^n\,(K', (\underline{e'}\ \lambda h.[\cdot])) \cup FV^{n+1}\,(\textbf{unbox}\ e') \\
&= CV^m(K') \cup CV^n\,(\underline{e'}\ \lambda h.[\cdot]) \cup FV^n(e') \\
&= CV^m(K') \cup FV^n\,(h) \cup FV^n(e') \\
&= CV^m(K') \cup FV^n\,(h) \cup FV^n\,(\underline{e'}) \cup FV^m(K') && \text{(IH)} \\
&= FV^n\,(h) \cup FV^n\,(\underline{e'}) \cup FV^m(K') && \text{(IH)} \\
&= FV^n\,(h) \cup FV^n\,(\underline{e'}\ \ (\lambda h.[\cdot])) \cup FV^m(K') \\
&= FV^n\,(h) \cup FV^n\,(K', (\underline{e'}\ \ (\lambda h.[\cdot]))) \\
&= FV^{n+1}\,(\textbf{unbox}\ h) \cup FV^n\,(K', (\underline{e'}\ \ (\lambda h.[\cdot]))) \\
&= FV^{n+1}\,(\underline{\textbf{unbox}\ e'}) \cup FV^n\,(K)
\end{aligned}
$$

- $e = \textbf{run}\ e'$.

$$
\begin{aligned}
CV^m(K) \cup FV^n(\textbf{run}\ e') &= CV^m(K) \cup FV^n(e') \\
&= CV^m(K) \cup FV^n\,(\underline{e'}) \cup FV^m(K) && \text{(IH)} \\
&= FV^n\,(\underline{e'}) \cup FV^m(K) && \text{(IH)} \\
&= FV^n\,(\lambda k.\,(\underline{e'}\ \ (\lambda m.\,((\textbf{run}\ m)\ \ k)))) \cup FV^m(K) && \text{(A1)} \\
&= FV^n\,(\underline{\textbf{run}\ e'}) \cup FV^m(K)
\end{aligned}
$$

□

Proof of Lem.5. As $e \mapsto (\underline{e}, \bot)$, then because of Lem.1 $depth(e) = 0$, then e isn't an **unbox**. Hence e concides with one of the cases of colon translation. By induction on the size of e and cases of colon translation (IH):

- $e \in Value^0$.

$$
\begin{aligned}
\underline{e}\ k &= (\lambda k.k\ \Psi(e))\ \ k && \text{(Lem.2)} \\
&\xrightarrow{0} k\ \Psi(e) \\
&\xrightarrow{0} e : k
\end{aligned}
$$

- $e = v\ e'$. $\qquad v \in Value^0$ and $e' \notin Value^0$

$$
\begin{aligned}
(\underline{v\ e'})\ \ k &= (\lambda k.\underline{v}\ \ (\lambda m.\underline{e'}\ \ (\lambda n.\,((m\ n)\ k))))\ \ k && \text{(TAPP)} \\
&= (\lambda k.\,(\lambda k.k\ \Psi(v))\ \ (\lambda m.\underline{e'}\ \ (\lambda n.\,((m\ n)\ k))))\ \ k && \text{(Lem.2)} \\
&\xrightarrow{0} (\lambda k.k\ \Psi(v))\ \ (\lambda m.\underline{e'}\ \ (\lambda n.\,((m\ n)\ k))) \\
&\xrightarrow{0} (\lambda m.\underline{e'}\ \ (\lambda n.\,((m\ n)\ k)))\ \ \Psi(v) \\
&\xrightarrow{0} \underline{e'}\ \ (\lambda n.(\Psi(v)\ n)\ k) \\
&\xrightarrow{0^+} e' : (\lambda n.(\Psi(v)\ n)\ k) && \text{(IH)} \\
&\xrightarrow{0^+} (v\ e') : k
\end{aligned}
$$

- $e = e_1\ e_2$. $e_1, e_2 \notin Value^0$

$$\begin{aligned}
\left(\underline{e_1\ e_2}\right)\ k &= \left(\lambda k.\underline{e_1}\ \left(\lambda m.\underline{e_2}\ \left(\lambda n.((m\ n)\ k)\right)\right)\right)\ k && \text{(TAPP)}\\
&\xrightarrow{0} \underline{e_1}\ \left(\lambda m.\underline{e_2}\ \left(\lambda n.((m\ n)\ k)\right)\right)\\
&\xrightarrow{0^+} e_1 : \left(\lambda m.\underline{e_2}\ \left(\lambda n.((m\ n)\ k)\right)\right) && \text{(IH)}\\
&\xrightarrow{0^+} (e_1\ e_2) : k
\end{aligned}$$

  - Case $\kappa = (\underline{e_h}\ \lambda h.[\cdot])$.

$$\begin{aligned}
\underline{\mathbf{box}\ e'}\ k &= \left(\lambda k.\underline{e_h}\ \lambda h.(k\ \ (\mathbf{box}\ \underline{e'}))\right)\ k && \text{(TBOX)}\\
&\xrightarrow{0} \underline{e_h}\ \lambda h.(k\ \ (\mathbf{box}\ \underline{e'}))\\
&\xrightarrow{0^+} e_h : \lambda h.(k\ \ (\mathbf{box}\ \underline{e'})) && \text{(IH)}\\
&\xrightarrow{0^+} \left(\underline{e_h}\ \lambda h.[\cdot]\right) : (k\ \ (\mathbf{box}\ \underline{e'}))\\
&\xrightarrow{0^+} (\mathbf{box}\ e') : k
\end{aligned}$$

  - Case $\kappa = (\underline{e_h}\ \lambda h.\kappa')$.

$$\begin{aligned}
\underline{\mathbf{box}\ e'}\ k &= \left(\lambda k.\underline{e_h}\ \lambda h.\kappa'\left[k\ \ (\mathbf{box}\ \underline{e'})\right]\right)\ k && \text{(TBOX)}\\
&\xrightarrow{0} \underline{e''}\ \lambda h.\kappa'\left[k\ \ (\mathbf{box}\ \underline{e'})\right]\\
&\xrightarrow{0^+} e_h : \lambda h.\kappa'\left[k\ \ (\mathbf{box}\ \underline{e'})\right] && \text{(IH)}\\
&\xrightarrow{0^+} \left(\underline{e_h}\ \lambda h.\kappa\right) : (k\ \ (\mathbf{box}\ \underline{e'}))\\
&\xrightarrow{0^+} (\mathbf{box}\ e') : k
\end{aligned}$$

- $e = \mathbf{run}\ e'$. $e' \notin Value^0$

$$\begin{aligned}
(\underline{\mathbf{run}\ e'})\ k &= (\lambda k.(\underline{e'}\ \ (\lambda m.(\mathbf{run}\ m)\ \ k)))\ k && \text{(TRUN)}\\
&\xrightarrow{0} \underline{e'}\ \ (\lambda m.(\mathbf{run}\ m)\ \ k)\\
&\xrightarrow{0^+} e' : (\lambda m.(\mathbf{run}\ m)\ \ k) && \text{(IH)}\\
&\xrightarrow{0^+} (\mathbf{run}\ e') : k
\end{aligned}$$

- $e = \mathbf{run}\ v.$ $\quad v \in Value^0$

$$\begin{aligned}
(\underline{\mathbf{run}\ v})\ k &= (\lambda k.(\underline{v}\ (\lambda m.(\mathbf{run}\ m)\ k)))\ k && \text{(TRUN)}\\
&= (\lambda k.((\lambda k.k\ \Psi(v))\ (\lambda m.(\mathbf{run}\ m)\ k)))\ k && \text{(Lem.2)}\\
&\xrightarrow{0} (\lambda k.k\ \Psi(v))\ (\lambda m.(\mathbf{run}\ m)\ k)\\
&\xrightarrow{0^+} (\lambda m.(\mathbf{run}\ m)\ k)\ \Psi(v)\\
&\xrightarrow{0^+} (\mathbf{run}\ \Psi(v))\ k\\
&\xrightarrow{0^+} (\mathbf{run}\ v) : k
\end{aligned}$$

□

Lemma 25 (Context Closure Evaluates to Colon Translation)**.**

$$\frac{k \in Closed\ Expr \qquad \kappa \in Context}{\kappa[k] \xrightarrow{0^+} \kappa : k}$$

Proof. By case analysis over the definition of κ.

- $\kappa = (\underline{e}\ \lambda h.[\cdot])$.

$$\begin{aligned}
\kappa[k] &= \underline{e}\ \lambda h.k\\
&\xrightarrow{0^+} e : \lambda h.k && \text{(Lem.5)}\\
&\xrightarrow{0^+} (\underline{e}\ \lambda h.[\cdot]) : k
\end{aligned}$$

- $\kappa = (\underline{e}\ \lambda h.\kappa')$.

$$\begin{aligned}
\kappa[k] &= \underline{e}\ \lambda h.\kappa'[k]\\
&\xrightarrow{0^+} e : \lambda h.\kappa'[k] && \text{(Lem.5)}\\
&\xrightarrow{0^+} (\underline{e}\ \lambda h.\kappa') : k
\end{aligned}$$

□

Lemma 26 (Outermost Context Preservation wrt. $\bowtie$)**.** *Let* $(\kappa_1, K_1), (\kappa_2, K_2) \in Context\ Stack.$

$$\frac{length(K_1) > length(K_2)}{(\kappa_1, K_1) \bowtie (\kappa_2, K_2) = \kappa_1, K}$$

$$\frac{length(K_1) = lenght(K_2)}{(\kappa_1, K_1) \bowtie (\kappa_2, K_2) = \kappa_1[\kappa_2], K}$$

$$\frac{length(K_1) < length(K_2)}{(\kappa_1, K_1) \bowtie (\kappa_2, K_2) = \kappa_2, K}$$

Proof. By induction over the definition of Context Stack Merge Operator.

- Case $K_1 = K_2 = \bot$.

$$\begin{aligned}(\kappa_1, \bot) \bowtie (\kappa_2, \bot) &= (\bot, \kappa_1) \bowtie (\bot, \kappa_2) \\ &= \bot, \kappa_1[\kappa_2] \\ &= \kappa_1[\kappa_2], \bot \\ &= \kappa_1[\kappa_2], (\bot \bowtie \bot)\end{aligned}$$

- Case $K_1 = \bot$ and $K_2 = K_2', \kappa_2'$.

$$\begin{aligned}(\kappa_1, \bot) \bowtie (\kappa_2, (K_2', \kappa_2')) &= (\bot, \kappa_1) \bowtie ((\kappa_2, K_2'), \kappa_2') \\ &= (\bot \bowtie (\kappa_2, K_2')), \kappa_1[\kappa_2'] \\ &= (\kappa_2, K_2'), \kappa_1[\kappa_2'] \\ &= \kappa_2, (K_2', \kappa_1[\kappa_2'])\end{aligned}$$

- Case $K_1 = K_1', \kappa_1'$ and $K_2 = \bot$.

$$\begin{aligned}(\kappa_1, (K_1', \kappa_1')) \bowtie (\kappa_2, \bot) &= ((\kappa_1, K_1'), \kappa_1') \bowtie (\bot, \kappa_2) \\ &= ((\kappa_1, K_1') \bowtie \bot), \kappa_1'[\kappa_2] \\ &= (\kappa_1, K_1'), \kappa_1'[\kappa_2] \\ &= \kappa_1, (K_1', \kappa_1'[\kappa_2])\end{aligned}$$

- Case $K_1 = K_1', \kappa_1'$ and $K_2 = K_2', \kappa_2'$.

$$\begin{aligned}(\kappa_1, (K_1', \kappa_1')) \bowtie (\kappa_2, (K_2', \kappa_2')) &= ((\kappa_1, K_1'), \kappa_1') \bowtie ((\kappa_2, K_2'), \kappa_2') \\ &= ((\kappa_1, K_1') \bowtie (\kappa_2, K_2')), \kappa_1'[\kappa_2'] \\ &= (\kappa, K'), \kappa_1'[\kappa_2'] \qquad \text{(IH)} \\ &= \kappa, (K', \kappa_1'[\kappa_2'])\end{aligned}$$

□

Lemma 27 (Reduction Preservation wrt. $\bowtie$)**.**

$$\frac{K_1 \xrightarrow{n^\star} K_1' \qquad K_2 \xrightarrow{n^\star} K_2'}{K_1 \bowtie K_2 \xrightarrow{n^\star} K_1' \bowtie K_2'}$$

Proof. By induction over the definition of Context Stack Merge Operator.

- Case $K_1 = \bot$. Then $K_1' = \bot$. $\bot \bowtie K_2 = K_2 \xrightarrow{n^\star} K_2' = \bot \bowtie K_2'$.
- Case $K_2 = \bot$. Then $K_2' = \bot$. $K_1 \bowtie \bot = K_1 \xrightarrow{n^\star} K_1' = K_1' \bowtie \bot$.
- Case $K_1 = K_a, \kappa_1$ and $K_b = K_2'', \kappa_2$. Let $m = max(n-1, 0)$. Then $K_1' = K_a', \kappa_1'$ with $K_a \xrightarrow{m^\star} K_a'$ and $\kappa_1 \xrightarrow{n^\star} \kappa_1'$.

As well $K_2' = K_b', \kappa_2'$ with $K_b \xrightarrow{m^\star} K_b'$ and $\kappa_2 \xrightarrow{n^\star} \kappa_2'$.

$$\begin{aligned}(K_a, \kappa_1) \bowtie (K_b, \kappa_2) &= (K_a \bowtie K_b), \kappa_1[\kappa_2] \\ &\xrightarrow{n^\star} (K_a \bowtie K_b), \kappa_1'[\kappa_2] \\ &\xrightarrow{n^\star} (K_a \bowtie K_b), \kappa_1'[\kappa_2'] \\ &\xrightarrow{n^\star} (K_a' \bowtie K_b'), \kappa_1'[\kappa_2'] \qquad \text{(IH)} \\ &\xrightarrow{n^\star} (K_a', \kappa_1') \bowtie (K_b', \kappa_2')\end{aligned}$$

□

Proof of Lem.6. By induction on the size of e and by cases according to the definition of $\xrightarrow{0}_{\mathtt{v}}$ (IH). We will show only interesting cases. The other cases are straightforward inductions like in $(\mathtt{ABS}_{\mathtt{v}})$.

- Case $(\mathtt{ABS}_{\mathtt{v}})$.

$$(\mathtt{ABS}_{\mathtt{v}}) \quad \frac{e_1' \xrightarrow{n+1}_{\mathtt{v}} e_2'}{\lambda x.e_1' \xrightarrow{n+1}_{\mathtt{v}} \lambda x.e_2'} \qquad (\mathtt{TABS}) \quad \frac{e \mapsto (\underline{e}, K)}{\lambda x.e \mapsto (\lambda k.\,(k\ \lambda x.\underline{e})\,, K)}$$

Then $e_1 = \lambda x.e_1'$, $e_2 = \lambda x.e_2'$, $e_1' \mapsto \left(\underline{e_1'}, (\kappa_1, K_1')\right)$ and $n > 0$.

  - Case $\kappa_1 = \left(\underline{e_h}\ \lambda h.[\cdot]\right)$ and $e_h \in Value^0$.

$$\begin{aligned}[h \overset{n}{\mapsto} \Psi(e_h)]\underline{\lambda x.e_1'} &= [h \overset{n}{\mapsto} \Psi(e_h)]\lambda k.(k\ \lambda x.\underline{e_1'}) \\ &= \lambda k.(k\ \lambda x.[h \overset{n}{\mapsto} \Psi(e_h)]\underline{e_1'}) \qquad \text{(A1)} \\ &\xrightarrow{n^\star} \lambda k.(k\ \lambda x.\underline{e_2'}) \qquad (\text{IH}, n > 0) \\ &\xrightarrow{n^\star} \underline{\lambda x.e_2'}\end{aligned}$$

    By (TABS) and (IH) $[h \overset{n-1}{\mapsto} \Psi(e_h)]K_1' \xrightarrow{n-1^\star} K_2$.
  - Case $\kappa_1 = \left(\underline{e_h}\ \lambda h.[\cdot]\right)$ and $e_h \notin Value^0$.

$$\begin{aligned}\underline{\lambda x.e_1'} &= \lambda k.(k\ \lambda x.\underline{e_1'}) \\ &= \lambda k.(k\ \lambda x.\underline{e_2'}) \qquad \text{(IH)} \\ &= \underline{\lambda x.e_2'}\end{aligned}$$

    By (TABS) and (IH) $K_2 = \left(\underline{e_h'}\ \lambda h.[\cdot]\right), K_1'$ with $e_h : k \xrightarrow{0^+} e_h' : k$.

– Case $\kappa_1 = \left(\underline{e_h}\ \lambda h.\kappa_1'\right)$ and $e_h \in Value^0$.

$$\begin{aligned}
[h \overset{n}{\mapsto} \Psi(e_h)]\underline{\lambda x.e_1'} &= [h \overset{n}{\mapsto} \Psi(e_h)]\lambda k.(k\ \lambda x.\underline{e_1'}) \\
&= \lambda k.(k\ \lambda x.[h \overset{n}{\mapsto} \Psi(e_h)]\underline{e_1'}) && \text{(A1)} \\
&\xrightarrow{n^\star} \lambda k.(k\ \lambda x.\underline{e_2'}) && \text{(IH, } n > 0\text{)} \\
&\xrightarrow{n^\star} \underline{\lambda x.e_2'}
\end{aligned}$$

By (TABS) and (IH) $[h \overset{n-1}{\mapsto} \Psi(e_h)](\kappa_1', K_1') \xrightarrow{n-1^\star} K_2$.

– Case $\kappa_1 = \left(\underline{e_h}\ \lambda h.\kappa_1'\right)$ and $e_h \notin Value^0$.

$$\begin{aligned}
\underline{\lambda x.e_1'} &= \lambda k.(k\ \lambda x.\underline{e_1'}) \\
&= \lambda k.(k\ \lambda x.\underline{e_2'}) && \text{(IH)} \\
&= \underline{\lambda x.e_2'}
\end{aligned}$$

By (TABS) and (IH) $K_2 = \left(\underline{e_h'}\ \lambda h.\kappa_1'\right), K_1'$ with $e_h : k \xrightarrow{0^+} e_h' : k$.

- Case $(\texttt{APP}_\texttt{v})$, $n = 0$.

– Case 1:

$$(\texttt{APP}_\texttt{v}) \qquad \frac{e_1' \xrightarrow{n}_\texttt{v} e_2'}{e_1'\ e_1'' \xrightarrow{n}_\texttt{v} e_2'\ e_1''}$$

$$(\texttt{TAPP}) \qquad \frac{e_1 \mapsto (\underline{e_1}, K_1) \qquad e_2 \mapsto (\underline{e_2}, K_2)}{e_1\ e_2 \mapsto \left(\lambda k.\underline{e_1}\ \left(\lambda m.\underline{e_2}\ (\lambda n.((m\ n)\ k))\right), K_1 \bowtie K_2\right)}$$

Then $e_1 = e_1'\ e_1''$ and $e_2 = e_2'\ e_1''$ with $e_1' \xrightarrow{0} e_2'$.

$$\begin{aligned}
(e_1'\ e_1'') : k = e_1' : \left(\lambda m.\underline{e_1''}\ (\lambda n.(m\ n)\ k)\right) \\
&\xrightarrow{0^+} e_2' : \left(\lambda m.\underline{e_1''}\ (\lambda n.(m\ n)\ k)\right) && \text{(IH)} \\
&\xrightarrow{0^+} e_3 && \text{(for some } e_3\text{)}
\end{aligned}$$

If $e_2' \notin Value^0$ then $e_3 = (e_2'\ e_1'') : k$. Otherwise

$$\begin{aligned}
e_3 = \left(\lambda m.\underline{e_1''}\ (\lambda n.(m\ n)\ k)\right)\Psi(e_2') \\
&\xrightarrow{0} \underline{e_1''}\ (\lambda n.(\Psi(e_2')\ n)\ k) \\
&\xrightarrow{0^+} e_1'' : (\lambda n.(\Psi(e_2')\ n)\ k) \\
&\xrightarrow{0^+} e_4 && \text{(for some } e_4\text{)}
\end{aligned}$$

If $e_1' \notin Value^0$ then $e_4 = (e_2'\ e_1'') : k$. Otherwise

$$\begin{aligned} e_4 &= (\lambda n.\,(\Psi(e_2')\ n)\ k)\,\Psi(e_1'') \\ &\xrightarrow{0} (\Psi(e_2')\ \Psi(e_1''))\ k \\ &\xrightarrow{0} (e_2'\ e_1'') : k \end{aligned}$$

– Case 2:

$$(\texttt{APP}_\texttt{v}) \qquad \frac{e_1' \xrightarrow{n}_\texttt{v} e_2' \qquad v \in Value^n}{v\ e_1' \xrightarrow{n}_\texttt{v} v\ e_2'}$$

$$(\texttt{TAPP}) \qquad \frac{e_1 \mapsto (\underline{e_1}, K_1) \qquad e_2 \mapsto (\underline{e_2}, K_2)}{e_1\ e_2 \mapsto \left(\lambda k.\underline{e_1}\ \left(\lambda m.\underline{e_2}\ (\lambda n.((m\ n)\ k))\right), K_1 \bowtie K_2\right)}$$

Then $e_1 = v\ e_1'$ and $e_2 = v\ e_2'$ with $e_1' \xrightarrow{0} e_2'$.

$$\begin{aligned} (v\ e_1') : k &= e_1' : (\lambda n.\,(\Psi(v)\ n)\ k) \\ &\xrightarrow{0^+} e_2' : (\lambda n.\,(\Psi(v)\ n)\ k) && \text{(IH)} \\ &\xrightarrow{0^+} e_3 && \text{(for some } e_3\text{)} \end{aligned}$$

If $e_2' \notin Value^0$ then $e_3 = (v\ e_2') : k$. Otherwise

$$\begin{aligned} e_3 &= (\lambda n.\,(\Psi(v)\ n)\ k)\,\Psi(e_2') \\ &\xrightarrow{0} (\Psi(v)\ \Psi(e_2'))\ k \\ &\xrightarrow{0} (v\ e_2') : k \end{aligned}$$

In cases 1 and 2, as $\bot \bowtie \bot = \bot$, by (TAPP) and (IH), $K_2 = \bot$.

– Case 3:

$$(\texttt{APP}_\texttt{v}) \qquad (\lambda x.e_1')\,v \xrightarrow{0}_\texttt{v} [x \overset{0}{\mapsto} v]e_1'$$

$$(\texttt{TAPP}) \qquad \frac{e_1 \mapsto (\underline{e_1}, K_1) \qquad e_2 \mapsto (\underline{e_2}, K_2)}{e_1\ e_2 \mapsto \left(\lambda k.\underline{e_1}\ \left(\lambda m.\underline{e_2}\ (\lambda n.((m\ n)\ k))\right), K_1 \bowtie K_2\right)}$$

Then $e_1 = (\lambda x.e_1')\ v$, $v \in Closed\ Value^0$ and $e_2 = [x \overset{0}{\mapsto} v]e_1'$.

$$\begin{aligned} ((\lambda x.e_1')\ v) : k &= (\Psi(\lambda x.e_1')\ \Psi(v))\ k \\ &= \left(\left(\lambda x.\underline{e_1'}\right)\ \Psi(v)\right)\ k \\ &\xrightarrow{0} \left([x \overset{0}{\mapsto} \Psi(v)]\underline{e_1'}\right)\ k \\ &\xrightarrow{0} \underline{[x \overset{0}{\mapsto} v]e_1'}\ k && \text{(Lem.3)} \\ &\xrightarrow{0^+} \left([x \overset{0}{\mapsto} v]e_1'\right) : k && \text{(Lem.5)} \end{aligned}$$

By (TAPP), (IH) and (Lem.3), $K_2 = \bot$.

- Case $(\texttt{APP}_\texttt{v})$, $n > 0$.
  - Case 1:

$$(\texttt{APP}_\texttt{v}) \qquad \frac{e_a \xrightarrow{n}_\texttt{v} e'_a}{e_a\ e_b \xrightarrow{n}_\texttt{v} e'_a\ e_b}$$

$$(\texttt{TAPP}) \qquad \frac{e_1 \mapsto (\underline{e_1}, K_1) \qquad e_2 \mapsto (\underline{e_2}, K_2)}{e_1\ e_2 \mapsto \left(\lambda k.\underline{e_1}\ \left(\lambda m.\underline{e_2}\ (\lambda n.((m\ n)\ k))\right), K_1 \bowtie K_2\right)}$$

Then $e_1 = e_a\ e_b$, $e_2 = e'_a\ e_b$, $e_a \mapsto \left(\underline{e_a}, (\kappa_a, K_a)\right)$, $e_b \mapsto \left(\underline{e_b}, (\kappa_b, K_b)\right)$, $e'_a \mapsto \left(\underline{e'_a}, K''_a\right)$

∗ If $depth(e_b) < n$ then by Lem.1 and Lem.26 $\kappa_1 = \kappa_a$.

· If $\kappa_1 = \left(\underline{e_h}\ \lambda h.[\cdot]\right)$ and $e_h \in Value^0$.

$$\begin{aligned}
[h \overset{n}{\mapsto} \Psi(e_h)]\underline{e_1} &= [h \overset{n}{\mapsto} \Psi(e_h)]\lambda k.\underline{e_a}\ \left(\lambda m.\underline{e_b}\ (\lambda n.((m\ n)\ k))\right) \\
&= \lambda k.[h \overset{n}{\mapsto} \Psi(e_h)]\underline{e_a}\ \left(\lambda m.[h \overset{n}{\mapsto} \Psi(e_h)]\underline{e_b}\ (\lambda n.((m\ n)\ k))\right) \\
&= \lambda k.[h \overset{n}{\mapsto} \Psi(e_h)]\underline{e_a}\ \left(\lambda m.\underline{e_b}\ (\lambda n.((m\ n)\ k))\right) && \text{(A1)} \\
&\xrightarrow{n^\star} \lambda k.\underline{e'_a}\ \left(\lambda m.\underline{e_b}\ (\lambda n.((m\ n)\ k))\right) && \text{(IH)} \\
&\xrightarrow{n^\star} \underline{e'_a\ e_b}
\end{aligned}$$

$$\begin{aligned}
[h \overset{n-1}{\mapsto} \Psi(e_h)]K'_1 &= [h \overset{n-1}{\mapsto} \Psi(e_h)](K'_a \bowtie K_b) \\
&= \left([h \overset{n-1}{\mapsto} \Psi(e_h)]K'_a\right) \bowtie \left([h \overset{n-1}{\mapsto} \Psi(e_h)]K_b\right) && \text{(Lem.23)} \\
&= \left([h \overset{n-1}{\mapsto} \Psi(e_h)]K'_a\right) \bowtie K_b && \text{(A1)} \\
&\xrightarrow{n-1^\star} K''_a \bowtie K_b && \text{(Lem.27, IH)} \\
&\xrightarrow{n-1^\star} K_2
\end{aligned}$$

· If $\kappa_1 = \left(\underline{e_h}\ \lambda h.\kappa'_1\right)$ and $e_h \in Value^0$. The proof of $[h \overset{n}{\mapsto} \Psi(e_h)]\underline{e_1} \xrightarrow{n^\star} \underline{e_2}$ is exactly the same as in previous case.

$$\begin{aligned}
&[h \overset{n-1}{\mapsto} \Psi(e_h)](\kappa'_1, K'_1) \\
&= [h \overset{n-1}{\mapsto} \Psi(e_h)]\,(\kappa'_1, (K'_a \bowtie K_b)) \\
&= [h \overset{n-1}{\mapsto} \Psi(e_h)]\,((\kappa'_1, K'_a) \bowtie K_b) && \text{(Lem.26)} \\
&= [h \overset{n-1}{\mapsto} \Psi(e_h)]\,(\kappa'_1, K'_a) \bowtie [h \overset{n-1}{\mapsto} \Psi(e_h)]K_b && \text{(subst-csmo)} \\
&= [h \overset{n-1}{\mapsto} \Psi(e_h)]\,(\kappa'_1, K'_a) \bowtie K_b && \text{(A1)} \\
&\xrightarrow{n-1^\star} K''_a \bowtie K_b && \text{(Lem.27, IH)} \\
&\xrightarrow{n-1^\star} K_2
\end{aligned}$$

· If $\kappa_1 = \left(\underline{e_h}\ \lambda h.[\cdot]\right)$ and $e_h \notin Value^0$.

$$\begin{aligned}\underline{e_1} &= \lambda k.\underline{e_a}\ \left(\lambda m.\underline{e_b}\ (\lambda n.((m\ n)\ k))\right)\\ &= \lambda k.\underline{e'_a}\ \left(\lambda m.\underline{e_b}\ (\lambda n.((m\ n)\ k))\right) \qquad \text{(IH)}\\ &= \underline{e_2}\end{aligned}$$

By (IH), $\exists e'_h$ such that $e_h : k \xrightarrow{0^\star} e'_h : k$ and $K''_a = \left(\underline{e'_h}\ \lambda h.[\cdot]\right), K'_a$.

$$\begin{aligned}K_2 &= K''_a \bowtie K_b\\ &= \left(\left(\underline{e'_h}\ \lambda h.[\cdot]\right), K'_a\right) \bowtie K_b \qquad \text{(IH)}\\ &= \left(\underline{e'_h}\ \lambda h.[\cdot]\right), (K'_a \bowtie K_b) \qquad \text{(Lem.26)}\end{aligned}$$

· If $\kappa_1 = \left(\underline{e_h}\ \lambda h.\kappa'_1\right)$ and $e_h \notin Value^0$. The proof of $\underline{e_1} = \underline{e_2}$ is exactly the same as in previous case.
By (IH), $\exists e'_h$ such that $e_h : k \xrightarrow{0^\star} e'_h : k$ and $K''_a = \left(\underline{e'_h}\ \lambda h.\kappa'_1\right), K'_a$.

$$\begin{aligned}K_2 &= K''_a \bowtie K_b\\ &= \left(\left(\underline{e'_h}\ \lambda h.\kappa'_1\right), K'_a\right) \bowtie K_b \qquad \text{(IH)}\\ &= \left(\underline{e'_h}\ \lambda h.\kappa'_1\right), (K'_a \bowtie K_b) \qquad \text{(Lem.26)}\end{aligned}$$

∗ If $depth(e_b) = n$ then by Lem.1 and Lem.26 $\kappa_1 = \kappa_a[\kappa_b]$. Similar to previous case.

– Case 2:

$$(\texttt{APP}_\texttt{v}) \qquad \frac{e'_1 \xrightarrow{n}_\texttt{v} e'_2 \qquad v \in Value^n}{v\ e'_1 \xrightarrow{n}_\texttt{v} v\ e'_2}$$

$$(\texttt{TAPP}) \qquad \frac{e_1 \mapsto (\underline{e_1}, K_1) \qquad e_2 \mapsto (\underline{e_2}, K_2)}{e_1\ e_2 \mapsto \left(\lambda k.\underline{e_1}\ \left(\lambda m.\underline{e_2}\ (\lambda n.((m\ n)\ k))\right), K_1 \bowtie K_2\right)}$$

Similar to case 1.

- Case $(\texttt{BOX}_\texttt{v})$, $n = 0$.

$$(\texttt{BOX}_\texttt{v}) \qquad \frac{e'_1 \xrightarrow{n+1}_\texttt{v} e'_2}{\textbf{box}\ e'_1 \xrightarrow{n}_\texttt{v} \textbf{box}\ e'_2} \qquad (\texttt{TBOX}) \qquad \frac{e \mapsto (\underline{e}, (K, \kappa))}{\textbf{box}\ e \mapsto (\lambda k.\kappa\,[k\ \ (\textbf{box}\ \underline{e})], K)}$$

Then $e_1 = \textbf{box}\ e'_1$, $e_2 = \textbf{box}\ e'_2$, $e'_1 \mapsto \left(\underline{e'_1}, (K, \kappa)\right)$ and $e'_1 \xrightarrow{1} e'_2$.
As $depth(e_1) = 0$, $depth(e'_1) \leq 1$ and as $e'_1 \xrightarrow{1} e'_2$, $depth(e'_1) = 1$
We have then $K = \bot$ and $e'_2 \mapsto (\underline{e'_2}, \bot)$. So κ is the outermost context.

– Case $\kappa = \left(\underline{e_h}\ \lambda h.[\cdot]\right)$ and $e_h \in Value^0$. Then $depth(e_2') = 0$. (1)

$$\begin{aligned}
(\mathbf{box}\ e_1') : k &= \left(\underline{e_h}\ \lambda h.[\cdot]\right) : \left(k\ \left(\mathbf{box}\ \underline{e_1'}\right)\right) \\
&= e_h : \lambda h.\left(k\ \left(\mathbf{box}\ \underline{e_1'}\right)\right) \\
&= \left(\lambda h.\left(k\ \left(\mathbf{box}\ \underline{e_1'}\right)\right)\right)\ \Psi(e_h) \\
&\xrightarrow{0^+} k\ \left(\mathbf{box}\ [h \overset{1}{\mapsto} \Psi(e_h)]\underline{e_1'}\right) \\
&\xrightarrow{0^+} k\ \left(\mathbf{box}\ \underline{e_2'}\right) && \text{(IH)} \\
&\xrightarrow{0^+} (\mathbf{box}\ e_2') : k && (1)
\end{aligned}$$

– Case $\kappa = \left(\underline{e_h}\ \lambda h.[\cdot]\right)$ and $e_h \notin Value^0$. By (IH) $\underline{e_1'} = \underline{e_2'}$, then $depth(e_2') \geq 1$ and as $depth(e_2) = 0$ then $depth(e_2') = 1$. (1)

$$\begin{aligned}
(\mathbf{box}\ e_1') : k &= \left(\underline{e_h}\ \lambda h.[\cdot]\right) : \left(k\ \left(\mathbf{box}\ \underline{e_1'}\right)\right) \\
&= e_h : \lambda h.\left(k\ \left(\mathbf{box}\ \underline{e_1'}\right)\right) \\
&= e_h : \lambda h.\left(k\ \left(\mathbf{box}\ \underline{e_2'}\right)\right) && \text{(IH)} \\
&\xrightarrow{0^+} e_h' : \lambda h.\left(k\ \left(\mathbf{box}\ \underline{e_2'}\right)\right) && \text{(IH)} \\
&\xrightarrow{0^+} \left(\underline{e_h'}\ \lambda h.[\cdot]\right) : \left(k\ \left(\mathbf{box}\ \underline{e_2'}\right)\right) \\
&\xrightarrow{0^+} (\mathbf{box}\ e_2') : k && (1)
\end{aligned}$$

– Case $\kappa = \left(\underline{e_h}\ \lambda h.\kappa'\right)$ and $e_h \in Value^0$. Then $depth(e_2') = 0$. (1)

$$\begin{aligned}
(\mathbf{box}\ e_1') : k &= \left(\underline{e_h}\ \lambda h.\kappa'\right) : \left(k\ \left(\mathbf{box}\ \underline{e_1'}\right)\right) \\
&= e_h : \lambda h.\kappa'\left[k\ \left(\mathbf{box}\ \underline{e_1'}\right)\right] \\
&= \left(\lambda h.\kappa'\left[k\ \left(\mathbf{box}\ \underline{e_1'}\right)\right]\right)\ \Psi(e_h) \\
&\xrightarrow{0^+} \left([h \overset{0}{\mapsto} \Psi(e_h)]\kappa'\right)\left[k\ \left(\mathbf{box}\ [h \overset{1}{\mapsto} \Psi(e_h)]\underline{e_1'}\right)\right] && \text{(A1)} \\
&\xrightarrow{0^+} \left([h \overset{0}{\mapsto} \Psi(e_h)]\kappa'\right)\left[k\ \left(\mathbf{box}\ \underline{e_2'}\right)\right] && \text{(IH)} \\
&\xrightarrow{0^+} \left([h \overset{0}{\mapsto} \Psi(e_h)]\kappa'\right) : \left(k\ \left(\mathbf{box}\ \underline{e_2'}\right)\right) && \text{(Lem.25)} \\
&\xrightarrow{0^+} (\mathbf{box}\ e_2') : k && \text{(IH, 1)}
\end{aligned}$$

– Case $\kappa = \left(\underline{e_h}\ \lambda h.\kappa'\right)$ and $e_h \notin Value^0$. By (IH) $\underline{e_1'} = \underline{e_2'}$ and like in second case $depth(e_2') = 1$. (1)

$$
\begin{aligned}
(\mathbf{box}\ e_1') : k &= \left(\underline{e_h}\ \lambda h.\kappa'\right) : \left(k\ \left(\mathbf{box}\ \underline{e_1'}\right)\right) \\
&= e_h : \lambda h.\kappa' \left[k\ \left(\mathbf{box}\ \underline{e_1'}\right)\right] \\
&= e_h : \lambda h.\kappa' \left[k\ \left(\mathbf{box}\ \underline{e_2'}\right)\right] \\
&\xrightarrow{0^+} e_h' : \lambda h.\kappa' \left[\left(k\ \left(\mathbf{box}\ \underline{e_2'}\right)\right)\right] && \text{(IH)} \\
&\xrightarrow{0^+} \left(\underline{e_h'}\ \lambda h.\kappa'\right) : \left(k\ \left(\mathbf{box}\ \underline{e_2'}\right)\right) \\
&\xrightarrow{0^+} (\mathbf{box}\ e_2') : k && \text{(IH, 1)}
\end{aligned}
$$

- Case ($\texttt{UNB}_\texttt{v}$)
  - Case 1:

$$
(\texttt{UNB}_\texttt{v}) \quad \frac{e_1' \xrightarrow{n}_\texttt{v} e_2'}{\mathbf{unbox}\ e_1' \xrightarrow{n+1}_\texttt{v} \mathbf{unbox}\ e_2'} \qquad (\texttt{TUNB}) \quad \frac{e \mapsto (\underline{e}, K)}{\mathbf{unbox}\ e \mapsto (\mathbf{unbox}\ h, (K, (\underline{e}\ \lambda h.[\cdot])))}
$$

Then $e_1 = \mathbf{unbox}\ e_1'$, $e_2 = \mathbf{unbox}\ e_2'$ and $e_1' \xrightarrow{n}_\texttt{v} e_2'$. We have also $e_1 \mapsto \left(\underline{e_1}, \left(K_1'', \left(\underline{e_1'}\ \lambda h.[\cdot]\right)\right)\right)$. and $e_2 \mapsto \left(\underline{e_2}, \left(K_2'', \left(\underline{e_2'}\ \lambda h.[\cdot]\right)\right)\right)$. $\underline{e_1} = \underline{e_2} = \mathbf{unbox}\ h$.

  - Case $n = 0$. Then $K_1'' = K_2'' = \bot$. By (IH) $e_1' : k \xrightarrow{0^+} e_2' : k$.

    Taking $e_h' = e_2'$ satisfies condition $\exists e_h'$ such that $e_h : k \xrightarrow{0^+} e_h'$ and $K_2 = (\underline{e_h'}\ \lambda h.[\cdot]), \bot$.
  - Case $n > 0$. By (IH) it is straightforward.

  - Case 2:

$$
(\texttt{UNB}_\texttt{v}) \quad \frac{v \in Value^1}{\mathbf{unbox}\ \ (\mathbf{box}\ v) \xrightarrow{1}_\texttt{v} v} \qquad (\texttt{TUNB}) \quad \frac{e \mapsto (\underline{e}, K)}{\mathbf{unbox}\ e \mapsto (\mathbf{unbox}\ h, (K, (\underline{e}\ \lambda h.[\cdot])))}
$$

Then $e_1 = \mathbf{unbox}\ \ (\mathbf{box}\ v)$, $v \in Value^1$ and $e_2 = v$.
Since $n = 1$, $K_1 = \bot, (\underline{\mathbf{box}\ v}\ \lambda h.[\cdot])$ and $K_2 = \bot$.
So trivially $[h \overset{0}{\mapsto} \Psi(e_h)]K_1' = [h \overset{0}{\mapsto} \Psi(e_h)]\bot = \bot = K_2$.

$$
\begin{aligned}
[h \overset{1}{\mapsto} \Psi(e_h)]\underline{e_1} &= [h \overset{1}{\mapsto} \Psi\,(\mathbf{box}\ v)]\mathbf{unbox}\ h \\
&= \mathbf{unbox}\ [h \overset{0}{\mapsto} \Psi\,(\mathbf{box}\ v)]h \\
&= \mathbf{unbox}\ \Psi\,(\mathbf{box}\ v) \\
&= \mathbf{unbox}\ \ (\mathbf{box}\ \underline{v}) \\
&\xrightarrow{0} \underline{v}
\end{aligned}
$$

- Case ($\texttt{RUN}_\texttt{v}$), $n = 0$.
  - Case 1:

$$(\texttt{RUN}_\texttt{v})\quad \frac{e_1' \xrightarrow{n}_\texttt{v} e_2'}{\mathbf{run}\ e_1' \xrightarrow{n}_\texttt{v} \mathbf{run}\ e_2'} \qquad (\texttt{TRUN})\quad \frac{}{\mathbf{run}\ e \mapsto (\underline{e}, K)\ (\lambda k.(\underline{e}\ (\lambda m.((\mathbf{run}\ m)\ k))), K)}$$

Then $e_1 = \mathbf{run}\ e_1'$, $e_2 = \mathbf{run}\ e_2'$ and $e_1' \xrightarrow{0} e_2'$.

$$\begin{aligned}(\mathbf{run}\ e_1') : k &= e_1' : (\lambda m.(\mathbf{run}\ m)\ k)\\ &\xrightarrow{0^+} e_2' : (\lambda m.(\mathbf{run}\ m)\ k) && \text{(IH)}\\ &\xrightarrow{0^+} e_3 && \text{(for some } e_3\text{)}\end{aligned}$$

If $e_2' \notin Value^0$ then $e_3 = (\mathbf{run}\ e_2') : k$. Otherwise

$$\begin{aligned}e_3 &= (\lambda m.(\mathbf{run}\ m)\ k)\,\Psi(e_2')\\ &\xrightarrow{0} (\mathbf{run}\ \Psi(e_2'))\ k\\ &\xrightarrow{0} (\mathbf{run}\ e_2') : k\end{aligned}$$

– Case 2 :

$$(\texttt{RUN}_\texttt{v})\quad \frac{v \in Value^1 \qquad FV_0(v) = \varnothing}{\mathbf{run}\ (\mathbf{box}\ v) \xrightarrow{0}_\texttt{v} v} \qquad (\texttt{TRUN})\quad \frac{}{\mathbf{run}\ e \mapsto (\underline{e}, K)\ (\lambda k.(\underline{e}\ (\lambda m.((\mathbf{run}\ m)\ k))), K)}$$

Then $e_1 = \mathbf{run}\ (\mathbf{box}\ v)$ with $v \in Closed\ Value^1$.

$$\begin{aligned}(\mathbf{run}\ (\mathbf{box}\ v)) : k &= (\mathbf{run}\ \Psi\,(\mathbf{box}\ v))\ k\\ &= (\mathbf{run}\ (\mathbf{box}\ \underline{v}))\ k\\ &\xrightarrow{0} \underline{v}\ k && (FV^0(v) = \varnothing, \text{Cor.1})\\ &\xrightarrow{0^+} v : k && \text{(Lem.5)}\end{aligned}$$

And for both cases $K_2 = \perp$ by (TRUN) and (IH).

□

Lemma 28 ($Sticks^n$ Preservation wrt. Context Closure)**.** *Let $\kappa \in Context$ and $e \in Expr$ such that $\kappa \in Sticks^n$ with $n > 0$. Then $\kappa[e] \in Sticks^n$.*

Proof. By induction on the size of κ and cases of Context definition (IH).

- Case $\kappa = (e_h\ \lambda h.[\cdot])$.
  Then $e_h \in Sticks^n$ and $\kappa[e] = e_h\ \lambda h.e \in Sticks^n$ by (SAPP).
- $\kappa = (e_h\ \lambda h.\kappa')$.
  - If $e_h \in Sticks^n$ then $\kappa[e] = e_h\ \lambda h.\kappa'[e] \in Sticks^n$ by (SAPP).
  - If $\kappa' \in Sticks^n$ the $\kappa[e] = e_h\ \lambda h.\kappa'[e] \in Sticks^n$ by (IH), (SABS) and (SAPP).

□

Lemma 29 (Depth Lower Bound wrt. $Sticks^n$)**.** *Let $e \in Expr$ such that $e \in Sticks^n$. Then $n \leq depth(e)$.*

Proof. By induction on the size of e and cases according to the definition of $Sticks^n$ (IH).
Let $n = depth(e)$.

- Case $(\texttt{SABS})$: $e = \lambda x.e'$ with $e' \in Sticks^n$.
  By (IH), $depth(e') \geq n$ then $depth(e) \geq n$.
- Case $(\texttt{SAPP}[1])$: $e = e_1\ e_2$ with $e_1 \in Sticks^n_{\mathtt{v}}$.
  By (IH) $depth(e_1) \geq n$ and as $depth(e) \geq depth(e_1)$ then $depth(e) \geq n$.
- Case $(\texttt{SAPP})$: $e = v\ e'$ with $v \in Value^n$ and $e' \in Sticks^n_{\mathtt{v}}$.
  By (IH) $depth(e') \geq n$ and as $depth(e) \geq depth(e')$ then $depth(e) \geq n$.
- Case $(\texttt{SBOX})$: $e = \mathbf{box}\ e'$ with $e' \in Sticks^{n+1}_{\mathtt{v}}$.
  By (IH) $depth(e') \geq n+1$. Then $depth(e) = depth(e') - 1 \geq n$.
- Case $(\texttt{SUNB}[1])$, $e = \mathbf{unbox}\ e'$ with $e' \in Sticks^{n-1}_{\mathtt{v}}$.
  By (IH) $depth(e') \geq n-1$. Then $depth(e) = depth(e') + 1 \geq n$.
- Case $(\texttt{SUNB}[2])$: $e = \mathbf{unbox}\ \ (\lambda x.e')$.
  $depth(e) \geq 1 = n$.
- Case $(\texttt{SUNB}_{\mathtt{n}}[3])$: $e = \mathbf{unbox}\ e'$ with $e' \notin Value^0$.
  $depth(e) \geq 1 = n$.
- Case $(\texttt{SRUN}[1])$: $e = \mathbf{run}\ e'$ with $e' \in Sticks^n_{\mathtt{v}}$.
  By (IH) $depth(e') \geq n$. Then $depth(e) = depth(e') \geq n$.
- Case $(\texttt{SRUN}[2])$: $e = \mathbf{run}\ \ (\mathbf{box}\ v)$ with $v \in Value^1$ and $FV_0(v) \neq \emptyset$.
  $depth(e) \geq 0 = n$.
- Case $(\texttt{SRUN}[3])$: $e = \mathbf{run}\ \ (\lambda x.e')$.
  $depth(e) \geq 0 = n$.
- Case $(\texttt{SRUN}_{\mathtt{n}}[4])$: $e = \mathbf{run}\ e'$ with $e' \notin Value^0$.
  $depth(e) \geq 0 = n$.

□

Proof of Lem.7. By induction on the size of e and cases according to the definition of $Sticks^n_{\mathtt{v}}$ (IH).

- Case $(\texttt{SABS}_{\mathtt{v}})$: $e = \lambda x.e'$ with $e' \in Sticks^n_{\mathtt{v}}$ and $e' \mapsto (\underline{e'}, (\kappa, K'))$.
  $depth(e') = depth(e) = n$ and $e' \in Sticks^n_{\mathtt{v}}$ hence we will be able to apply (IH) on e'.
  - If $e_h \in Value^0$ and $K' = \bot$.

$$\begin{aligned}
[h \overset{1}{\mapsto} \Psi(e_h)]\underline{\lambda x.e'} &= [h \overset{1}{\mapsto} \Psi(e_h)]\lambda k.k\ \lambda x.\underline{e'} \\
&= \lambda k.k\ \lambda x.[h \overset{1}{\mapsto} \Psi(e_h)]\underline{e'} \\
&\in Sticks^1_{\mathtt{v}} \cap Sticks^1_{\mathtt{n}} \qquad \text{(IH, SABS, SAPP, SABS)}
\end{aligned}$$

– If $e_h \in Value^0$ and $K' = \kappa_2, K'_2$ then (IH) applies.
– If $e_h \notin Value^0$ then (IH) applies.

- Case $(\texttt{SAPP}_\texttt{v}[1])$, $n = 0 :\ e = e_1\ e_2$ with $e_1 \in Sticks^0_\texttt{v}$.
$depth(e_1) = depth(e_2) = 0$ and $e_1 \in Sticks^0_\texttt{v}$ hence we will be able to apply (IH) on e_1.

$$\begin{aligned}(e_1\ e_2) : k = e_1 : \left(\lambda m.\underline{e_2}\ \lambda n.\,(m\ n)\ \ k\right)& \\ \xrightarrow{0^\star} e_s & \qquad (\text{IH},\ e_s \in Sticks^0_\texttt{v} \cap Sticks^0_\texttt{n})\end{aligned}$$

- Case $(\texttt{SAPP}_\texttt{v}[1])$, $n > 0 :\ e = e_1\ e_2$ with $e_1 \in Sticks^n_\texttt{v}$.
$depth(e) \geq depth(e_1)$ and by Lem.29 $depth(e_1) \geq n = depth(e)$. Hence $depth(e) = depth(e_1) = n$ and we can apply (IH) on e_1.
We will treat only the case $depth(e_1) = depth(e_2)$ then by Lem.26 $\kappa = \kappa_{e_1}[\kappa_{e_2}]$ with $e_1 \mapsto \left(\underline{e_i}, (\kappa_{e_i}, K_{e_i})\right)$.

– Case $e_h \in Value^0$ and $K_{e_1} \bowtie K_{e_2} = \bot$.

$$\begin{aligned}[h \overset{1}{\mapsto} \Psi(e_h)]\underline{e_1\ e_2} &= [h \overset{1}{\mapsto} \Psi(e_h)]\left(\lambda k.\underline{e_1}\ \left(\lambda m.\underline{e_2}\ (\lambda n.\,((m\ n)\ k))\right)\right) \\ &= \left(\lambda k.[h \overset{1}{\mapsto} \Psi(e_h)]\underline{e_1}\ \left(\lambda m.[h \overset{1}{\mapsto} \Psi(e_h)]\underline{e_2}\ (\lambda n.\,((m\ n)\ k))\right)\right) \\ &\in Sticks^0_\texttt{v} \cap Sticks^0_\texttt{n} \qquad (\text{IH, SABS})\end{aligned}$$

– Case $e_h \in Value^0$ and $K_{e_1} \bowtie K_{e_2} = \kappa'_{e_1}[\kappa'_{e_2}], \left(K'_{e_1} \bowtie K'_{e_2}\right)$ with $K_{e_i} = \kappa'_{e_i}, K'_{e_i}$.

$$\begin{aligned}[h \overset{1}{\mapsto} \Psi(e_h)]\kappa'_{e_1}[\kappa'_{e_2}] &= \left([h \overset{1}{\mapsto} \Psi(e_h)]\kappa'_{e_1}\right)[h \overset{1}{\mapsto} \Psi(e_h)]\kappa'_{e_2} \qquad (\text{Lem.22}) \\ &\in Sticks^1_\texttt{v} \cap Sticks^1_\texttt{n} \qquad (\text{IH})\end{aligned}$$

– Case $e_h \notin Value^0$. By (IH), if $\kappa_{e_1} = \left(\underline{e_h}\ \lambda h.[\cdot]\right)$ then $\kappa = \left(\underline{e_h}\ \lambda h.\kappa_{e_2}\right)$ else $\kappa_{e_1} = \left(\underline{e_h}\ \lambda h.\kappa'_{e_1}\right)$ and then $\kappa = \left(\underline{e_h}\ \lambda h.\kappa'_{e_1}[\kappa_{e_2}\right)$ and in both cases, $\exists e_s$ such that $e_h : k \xrightarrow{0^\star} e_s$ and $e_s \in Sticks^0_\texttt{v} \cap Sticks^0_\texttt{n}$.

- Case $(\texttt{SAPP}_\texttt{v}[2])$, $n = 0 :\ e = v\ e'$ with $v \in Value^0$ and $e' \in Sticks^0_\texttt{v}$.
Then $depth(e) = depth(e') = 0$ and as $e' \in Sticks^0_\texttt{v}$ we can apply (IH) on e'.

$$\begin{aligned}(v\ e') : k = e' : (\lambda n.\,(\Psi(v)\ n)\ \ k)& \\ \xrightarrow{0^\star} e_s & \qquad (\text{IH},\ e_s \in Sticks^0_\texttt{v} \cap Sticks^0_\texttt{n})\end{aligned}$$

- Case $(\texttt{SAPP}_\texttt{v}[2])$, $n > 0 :\ e = v\ e'$ with $v \in Value^n$ and $e' \in Sticks^n_\texttt{v}$.
This case is very similar to $(\texttt{SAPP}_\texttt{v}[1]$, $n > 0$.

- Case $(\texttt{SBOX}_\texttt{v})$, $n = 0 :\ e = \textbf{box}\ e'$ with $e' \in Sticks^1_\texttt{v}$.
By Lem.29 $depth(e') \geq 1$ and as $depth(e') \leq depth(e) + 1$, $depth(e') = 1$. By Lem.1 and (TBOX) $e' \mapsto (\underline{e'}, (\kappa', \bot))$ with $\kappa' = \left(\underline{e_h}\ \lambda h.[\cdot]\right)$ or $\kappa' = \left(\underline{e_h}\ \lambda h.\kappa''\right)$. As $depth(e') = 1$ and $e' \in Sticks^1_\texttt{v}$, we will be able to apply (IH) on e'.

– Case $e_h \in Value^0$.

$$\begin{aligned}
(\mathbf{box}\ e') : k &= \kappa' : (k\ \ (\mathbf{box}\ \underline{e'})) \\
&= e_h : \lambda h.k\ \ (\mathbf{box}\ \underline{e'}) \\
&= (\lambda h.k\ \ (\mathbf{box}\ \underline{e'}))\ \ \Psi(e_h) \\
&\xrightarrow{0} [h \overset{0}{\mapsto} \Psi(e_h)]\,(k\ \ (\mathbf{box}\ \underline{e'})) \\
&\xrightarrow{0} k\ \left(\mathbf{box}\ [h \overset{1}{\mapsto} \Psi(e_h)]\underline{e'}\right) \\
&\xrightarrow{0^\star} k\ \ (\mathbf{box}\ e'_s) && (\text{IH},\ e'_s \in Sticks^1_{\mathtt{v}} \cap Sticks^1_{\mathtt{n}}) \\
&\xrightarrow{0^\star} e_s && (e_s \in Sticks^0_{\mathtt{v}} \cap Sticks^0_{\mathtt{n}})
\end{aligned}$$

– Case $e_h \notin Value^0$. We will prove the case $\kappa' = \left(\underline{e_h}\ \lambda h.[\cdot]\right)$. The case $\kappa' = \left(\underline{e_h}\ \lambda h.\kappa''\right)$ is almost the same.

$$\begin{aligned}
(\mathbf{box}\ e') : k &= \kappa' : (k\ \ (\mathbf{box}\ \underline{e'})) \\
&= e_h : \lambda h.k\ \ (\mathbf{box}\ \underline{e'}) \\
&\xrightarrow{0^\star} e_s && (\text{IH},\ e_s \in Sticks^0_{\mathtt{v}} \cap Sticks^0_{\mathtt{n}})
\end{aligned}$$

- Case ($\mathtt{SBOX}_{\mathtt{v}}$), $n > 0$: $e = \mathbf{box}\ e'$ with $e' \in Sticks^{n+1}_{\mathtt{v}}$ with $e' \mapsto (\underline{e'} and\,(\kappa, (\kappa_2, K'_2)))$.
  By Lem.29 $depth(e') \geq n+1$ and as $depth(e') \leq depth(e)+1$, $depth(e') = n+1$. As $e' \in Sticks^{n+1}_{\mathtt{v}}$, by (IH) $[h \overset{1}{\mapsto} \Psi(e_h)]\kappa_2 \in Sticks^1_{\mathtt{v}} \cap Sticks^1_{\mathtt{n}}$.

  – Case $e_h \in Value^0$ and $K'_2 = \bot$.

$$\begin{aligned}
[h \overset{1}{\mapsto} \Psi(e_h)]\underline{e} &= [h \overset{1}{\mapsto} \Psi(e_h)]\lambda k.\kappa_2\,[k\ \ (\mathbf{box}\ \underline{e'}]) \\
&= \lambda k.[h \overset{1}{\mapsto} \Psi(e_h)]\,(\kappa_2\,[k\ \ (\mathbf{box}\ \underline{e'})]) && (\text{SABS}) \\
&= \lambda k.\left([h \overset{1}{\mapsto} \Psi(e_h)]\kappa_2\right)[h \overset{1}{\mapsto} \Psi(e_h)]\,[k\ \ (\mathbf{box}\ \underline{e'})] && (\text{Lem.22}) \\
&\in Sticks^1_{\mathtt{v}} \cap Sticks^1_{\mathtt{n}} && (\text{IH, Lem.28})
\end{aligned}$$

  – Case $e_h \in Value^0$ and $K'_2 \neq \bot$. Then $[h \overset{1}{\mapsto} \Psi(e_h)\kappa_2 \in Sticks^1_{\mathtt{v}} \cap Sticks^1_{\mathtt{n}}$.

  – Case $e_h \notin Value^0$ then by (IH) $\exists e_s$ such that $e_h : k \xrightarrow{0^\star} e_s$ and $e_s \in Sticks^0_{\mathtt{v}} \cap Sticks^0_{\mathtt{n}}$.

- Case ($\mathtt{SUNB}_{\mathtt{v}}[1]$), $n = 1$: $e = \mathbf{unbox}\ e'$ with $e' \in Sticks^0_{\mathtt{v}}$.
  As $depth(e') = depth(e) - 1 = 0$ and $e' \in Sticks^0_{\mathtt{v}}$, we can apply (IH) on e'. $e \mapsto (\mathbf{unbox}\ h, ((\underline{e'}\ \lambda h.[\cdot]), \bot))$. By (IH) $\exists e_s$ such that $e' : k \xrightarrow{0^\star} e_s$ and $e_s \in Sticks^0_{\mathtt{v}} \cap Sticks^0_{\mathtt{n}}$. As $e_s \in Sticks^0_{\mathtt{v}}$ then $e' \notin Value^0$.

- Case ($\mathtt{SUNB}_{\mathtt{v}}[1]$), $n > 1$: $e = \mathbf{unbox}\ e'$ with $e' \in Sticks^{n-1}_{\mathtt{v}}$.
  By assumption $depth(e) = n$ then $depth(e') = n-1$ and we can apply (IH) on e': $e' \mapsto (\underline{e'}, (\kappa, K''))$ and by (TUNB) $e \mapsto (\mathbf{unbox}\ h', (\kappa, (K'', \underline{e'}\ \lambda h'.[\cdot])))$.

– Case $e_h \in Value^0$ and $K'' = \bot$.

$$[h \overset{2}{\mapsto} \Psi(e_h)]\,(\underline{e'}\ \lambda h'.[\cdot]) = \left([h \overset{2}{\mapsto} \Psi(e_h)]\underline{e'}\right)\ \lambda h'.[\cdot] \in Sticks^1_{\mathtt{v}} \cap Sticks^1_{\mathtt{n}} \quad \text{(IH)}$$

– Case $e_h \in Value^0$ and $K'' \neq \bot$. Then by (IH) $[h \overset{1}{\mapsto} \Psi(e_h)]\kappa_2 \in Sticks^1_{\mathtt{v}} \cap Sticks^1_{\mathtt{n}}$.

– Case $e_h \notin Value^0$ then by (IH) $\exists e_s$ such that $e_h : k \xrightarrow{0^\star} e_s$ and $e_s \in Sticks^0_{\mathtt{v}} \cap Sticks^0_{\mathtt{n}}$.

- Case $(\mathtt{SUNB_v}[2])$: $e = \mathbf{unbox}\ (\lambda x.e')$.
  By assumption $depth(e) = 1$ then $depth(e') = 0$.
  By Lem.1 and (TUNB), $e \mapsto (\mathbf{unbox}\ h, (\bot, (\underline{e'}\ \lambda h.[\cdot])))$

$$[h \overset{1}{\mapsto} \Psi(\lambda x.e')]\,(\mathbf{unbox}\ h) = (\mathbf{unbox}\ (\lambda x.\underline{e})) \in Sticks^1_{\mathtt{v}} \cap Sticks^1_{\mathtt{n}}$$

- Case $(\mathtt{SRUN_v}[1])$, $n = 0$: $e = \mathbf{run}\ e'$ with $e' \in Sticks^0_{\mathtt{v}}$.
  $depth(e) = depth(e') = 0$ then we can apply (IH) on e'.

$$(\mathbf{run}\ e') : k = e' : (\lambda m.(\mathbf{run}\ m)\ k) \xrightarrow{0^\star} e_s \quad \text{(IH, } e_s \in Sticks^0_{\mathtt{v}} \cap Sticks^0_{\mathtt{n}})$$

- Case $(\mathtt{SRUN_v}[1])$, $n > 0$: $e = \mathbf{run}\ e'$ with $e' \in Sticks^n_{\mathtt{v}}$.
  $depth(e) = depth(e') = n$. We can apply (IH) on e'. By (TRUN) $e' \mapsto (\underline{e'}, (\kappa, K'))$

– Case $e_h \in Value^0$ and $K' = \bot$.

$$[h \overset{1}{\mapsto} \Psi(e_h)]\underline{\mathbf{run}\ e'} = \lambda k.(\underline{e'}\ (\lambda m.((\mathbf{run}\ m)\ k))) \in Sticks^1_{\mathtt{v}} \cap Sticks^1_{\mathtt{n}} \quad \text{(IH, SABS)}$$

– Case $e_h \in Value^0$ and $K' = \kappa_2, K'_2$. In this case, (IH) applies directly.

– Case $e_h \notin Value^0$. Then (IH) applies.

- Case $(\mathtt{SRUN_v}[2])$: $e = \mathbf{run}\ (\mathbf{box}\ v)$ with $v \in Value^1$ and $FV_0(v) \neq \varnothing$.

$$(\mathbf{run}\ (\mathbf{box}\ v)) : k = (\mathbf{run}\ (\mathbf{box}\ \underline{v}))\ k \in Sticks^0_{\mathtt{v}} \cap Sticks^0_{\mathtt{n}} \quad \text{(Cor.1, SRUN[2], SAPP[1])}$$

- Case $(\mathtt{SRUN_v}[3])$: $e = \mathbf{run}\ (\lambda x.e')$.
  $depth(e) = 0 \geq depth(e') \geq 0$.

$$(\mathbf{run}\ (\lambda x.e')) : k = (\mathbf{run}\ (\lambda x.\underline{e'}))\ k \in Sticks^0_{\mathtt{v}} \cap Sticks^0_{\mathtt{n}} \quad \text{(SRUN[3], SAPP[1])}$$

□

Proof of Th.2 and Th.3. Let $e \in Expr$ such that $depth(e) = 0$.

- Case $eval_{\mathtt{v}}(e)$ is defined. $eval_{\mathtt{v}}(e) = v$ for some $v \in Value^0$. Then $e \xrightarrow{0^\star}_{\mathtt{v}} v$.

$$\begin{aligned} \underline{e}\ (\lambda x.x) &\xrightarrow{0^+} e : (\lambda x.x) && \text{(Lem.5)} \\ &\xrightarrow{0^+} v : (\lambda x.x) && \text{(Lem.6)} \\ &\xrightarrow{0^+} \Psi(v) \\ &\xrightarrow{0^+} \Psi\left(eval_{\mathtt{v}}(e)\right) \end{aligned}$$

  As reductions were available in both call-by-value and call-by-name operational semantics

$$eval_{\mathtt{n}}\left(\underline{e}\ (\lambda x.x)\right) = eval_{\mathtt{v}}\left(\underline{e}\ (\lambda x.x)\right) = \Psi\left(eval_{\mathtt{v}}(e)\right)$$

- Case $eval_{\mathtt{v}}(e)$ is not defined.
  - If $e \xrightarrow{0^\star}_{\mathtt{v}} e'$ for some $e' \in Sticks^0_{\mathtt{v}}$ then

$$\begin{aligned} \underline{e}\ (\lambda x.x) &\xrightarrow{0^+} e : (\lambda x.x) && \text{(Lem.5)} \\ &\xrightarrow{0^+} e' : (\lambda x.x) && \text{(Lem.6)} \\ &\xrightarrow{0^+} e_s && \text{(Lem.7, } e_s \in Sticks^0_{\mathtt{v}} \cap Sticks^0_{\mathtt{n}}\text{)} \end{aligned}$$

  - If there is an infinite series $e_1, e_2, \ldots$, such that $e = e_1 \xrightarrow{0}_{\mathtt{v}} e_2 \xrightarrow{0}_{\mathtt{v}} e3 \xrightarrow{0} \ldots$ then by *Lem.*6 we have

$$\underline{e}\ (\lambda x.x) \xrightarrow{0^+} e : (\lambda x.x) = e_1 : (\lambda x.x) \xrightarrow{0^+} e_2 : (\lambda x.x) \xrightarrow{0^+} \ldots$$

    with the same result.

□

4.3 Proof of Translation Theorem

Lemma 30 ($=^n$ Preservation wrt. Context Closure)**.** *Let $\kappa_1, \kappa_2 \in Context$ and $e \in Expr$.*

$$\frac{\kappa_1 =^n \kappa_2}{\kappa_1[e] =^n \kappa_2[e]}$$

Proof. By induction over size of κ_i ($length(\kappa_1) = length(\kappa_2)$) (IH).

- Case $\kappa_i = ((e_{h_i}\ \lambda h.[\cdot])$ for $i \in \{1, 2\}$. Then $e_{h_1} =^n e_{h_2}$.

$$\frac{\begin{array}{c} e_{h_1} =^n e_{h_2} \\ e_{h_1}\ \lambda h.e =^n e_{h_2}\ \lambda h.e \\ \kappa_1[e] =^n \kappa_2[e] \end{array}}{} \quad \text{(EAPP[1])}$$

- Case $\kappa_i = (e_{h_i}\ \lambda h.\kappa_i')$ for $i \in \{1,2\}$.

  We have then two cases depending on the construction rule:

  – If $e_{h_1} =^n e_{h_2}$ and $\kappa_1' = \kappa_2'$ then

$$\begin{array}{rl} e_{h_1} =^n e_{h_2} & \\ e_{h_1}\ \lambda h.\kappa_1'[e] =^n e_{h_2}\ \lambda h.\kappa_2'[e] & \text{(EAPP[1])} \\ \kappa_1[e] =^n \kappa_2[e] & \end{array}$$

  – If $e_{h_1} = e_{h_2}$ and $\kappa_1' =^n \kappa_2'$ then

$$\begin{array}{rl} \kappa_1'[e] =^n \kappa_2'[e] & \text{(IH)} \\ \lambda h.\kappa_1'[e] =^n \lambda h.\kappa_2'[e] & \text{(EABS)} \\ e_{h_1}\ \lambda h.\kappa_1'[e] =^n e_{h_2}\ \lambda h.\kappa_2'[e] & \text{(EAPP[2])} \\ \kappa_1[e] =^n \kappa_2[e] & \end{array}$$

□

Lemma 31 ($=^n$ Preservation wrt. Context Closure)**.** *Let $\kappa \in Context$ and $e_1, e_2 \in Expr$.*

$$\frac{e_1 =^n e_2}{\kappa[e_1] =^n \kappa[e_2]}$$

Proof. By induction over size of κ (IH).

- Case $\kappa = ((e_h\ \lambda h.[\cdot])$.

$$\begin{array}{rl} e_1 =^n e_2 & \\ \lambda h.e_1 =^n \lambda h.e_2 & \text{(EABS)} \\ e_h\ \lambda h.e_1 =^n e_h\ \lambda h.e_2 & \text{(EAPP[2])} \\ \kappa[e_1] =^n \kappa[e_2] & \end{array}$$

- Case $\kappa = (e_h\ \lambda h.\kappa')$.

$$\begin{array}{rl} e_1 =^n e_2 & \\ \kappa'[e_1] =^n \kappa'[e_2] & \text{(IH)} \\ \lambda h.\kappa'[e_1] =^n \lambda h.\kappa'[e_2] & \text{(EABS)} \\ e_h\ \lambda h.e_1 =^n e_h\ \lambda h.e_2 & \text{(EAPP[2])} \\ \kappa[e_1] =^n \kappa[e_2] & \end{array}$$

□

Lemma 32 ($=^n$ Preservation wrt. $\bowtie$)**.**

$$\frac{K_1 \geq^n K_1'}{K_1 \bowtie K_2 \geq^n K_1' \bowtie K_2}$$

Proof. By induction over size of K_1 and cases of Context Stack Merge Operator (IH).

- Case $K_1 = \bot$. Impossible.
- Case $K_2 = \bot$. Then $K_1 \bowtie K_2 = K_1 \geq^n K_1' = K_1' \bowtie K_2$.
- Case $K_1 = K_1'', \kappa_1$ and $K_2 = K_2', \kappa_2$.
  Then $K_1' = K_1''', \kappa_1'$. We have two cases:
  - Case $\kappa_1 = \kappa_1'$. Then $K_1'' \geq^n K_1'''$.

$$\begin{aligned}(K_1'', \kappa_1) \bowtie (K_2', \kappa_2) &= (K_1'' \bowtie K_2')\,, \kappa_1[\kappa_2] \\ &= (K_1'' \bowtie K_2')\,, \kappa_1'[\kappa_2] \\ &= (K_1''' \bowtie K_2')\,, \kappa_1'[\kappa_2] \\ &= (K_1''', \kappa_1') \bowtie (K_2', \kappa_2)\end{aligned} \tag{IH}$$

  - Case $K_1'' = K_1'''$. Then $\kappa_1 \geq^n \kappa_1'$.

$$\begin{aligned}(K_1'', \kappa_1) \bowtie (K_2', \kappa_2) &= (K_1'' \bowtie K_2')\,, \kappa_1[\kappa_2] \\ &= (K_1''' \bowtie K_2')\,, \kappa_1[\kappa_2] \\ &= (K_1''' \bowtie K_2')\,, \kappa_1'[\kappa_2] \\ &= (K_1''', \kappa_1') \bowtie (K_2', \kappa_2)\end{aligned}$$

□

Proof of Lem.8. By induction on the size of e and cases according to the definition of Equality (IH).

We will show only interesting cases.

- Case ($\texttt{EREF}_\texttt{v}$), $K_1 = \bot$.
  Then $e_1 = e_2$, so $\underline{e_1} = \underline{e_2}$ and by ($\texttt{EREF}$) $\underline{e_1} \geq^0 \underline{e_2}$.
- Case ($\texttt{ETRA}_\texttt{v}$), $K_1 = \bot$.
  Then $\exists e$ such that $e_1 \geq^0_\texttt{v} e$ and $e \geq^0 e_2$. As $e_1 \geq^0 e$ then by Lem.9 $depth(e) = 0$. So by (IH) $\underline{e_1} \geq^0 \underline{e}$ and $\underline{e} \geq^0 \underline{e_2}$ and we can apply ($\texttt{ETRA}$).
- Case ($\texttt{ETRA}_\texttt{v}$), $K_1 = \kappa_1, K_1'$.
  Then $\exists e$ such that $e_1 \geq^n_\texttt{v} e$ and $e \geq^n_\texttt{v} e_2$. As $depth(e_1) = n$ by assumption, we can apply (IH) on e_1 and $\underline{e_1} \geq^n \underline{e}$.
  If $depth(e) < n$ then by Lem.10 $e = e_2$ and we are done. Otherwise $depth(e) = n$ then we can apply (IH) as well on e and $\underline{e} \geq^n \underline{e_2}$. By ($\texttt{ETRA}$) $\underline{e_1} \geq^n \underline{e_2}$.
- Case ($\texttt{EAPP}_\texttt{v}$)[1], $K_1 = \bot$ with $e_1 = e_a\ e_b$, $e_2 = e_a'\ e_b$ and $e_a \geq^0_\texttt{v} e_a'$.

$$\begin{aligned}\underline{e_a\ e_b} &= \lambda k.\underline{e_a}\ \big(\lambda m.\underline{e_b}\ (\lambda n.(m\ n)\ k)\big) && \text{(TAPP)} \\ &\geq^0 \lambda k.\underline{e_a'}\ \big(\lambda m.\underline{e_b}\ (\lambda n.(m\ n)\ k)\big) && \text{(IH, EABS)} \\ &\geq^0 \underline{e_a'\ e_b} && \text{(TAPP)}\end{aligned}$$

- Case $(\texttt{EAPP}_\texttt{v})[1]$, $K_1 = \kappa_1, K_1'$ with $e_1 = e_a\ e_b$, $e_2 = e_a'\ e_b$ and $e_a \geq_\texttt{v}^n e_a'$.

  If $depth(e_a) < n$ then by *Lem.*10 $e_1 = e_2$ and we are done. Otherwise $depth(e_a) = n$. We will handle the case where $depth(e_b) = n$. The case $depth(e_b) < n$ is straightforward. So $e_a \mapsto \left(\underline{e_a}, \kappa_a, K_a'\right)$ and $e_b \mapsto \left(\underline{e_b}, (\kappa_b, K_b')\right)$. By Lem.26 $\kappa_a[\kappa_b] = \kappa_1$. We have four cases:

  - Case $\kappa_a = \left(\underline{e_{h_a}}\ \lambda h.[\cdot]\right)$ with $e_h \in Value^0$.

    By (IH) $[h \overset{n}{\mapsto} \Psi(e_h)]\underline{e_a} \geq^n \underline{e_a'}$, $e_a' \mapsto \left(\underline{e_a'}, (K_a'')\right)$ with $K_a' \geq^{n-1} K_a''$.

    $$\begin{aligned}
    [h \overset{n}{\mapsto} \Psi(e_h)]\underline{e_a\ e_b} &= [h \overset{n}{\mapsto} \Psi(e_h)]\lambda k.\underline{e_a}\ \left(\lambda m.\underline{e_b}\ (\lambda n.(m\ n)\ k)\right) && \text{(TAPP)}\\
    &= \lambda k.\left([h \overset{n}{\mapsto} \Psi(e_h)]\underline{e_a}\right)\ \left(\lambda m.\underline{e_b}\ (\lambda n.(m\ n)\ k)\right) && \text{(A1)}\\
    &\geq^n \lambda k.\underline{e_a'}\ \left(\lambda m.\underline{e_b}\ (\lambda n.(m\ n)\ k)\right) && \text{(IH, EABS)}\\
    &\geq^n \underline{e_a'\ e_b} && \text{(TAPP)}
    \end{aligned}$$

    And by (IH) $K_a' \geq^{n-1} K_a''$ so $(K_a' \bowtie K_b') \geq^{n-1} (K_a'' \bowtie K_b')$ by Lem.32.
  - Case $\kappa_a = \left(\underline{e_h}\ \lambda h.\kappa_a'\right)$ with $e_h \in Value^0$. Similar to previous case.
  - Case $\kappa_a = \left(\underline{e_h}\ \lambda h.[\cdot]\right)$ with $e_h \notin Value^0$.

    By (IH) $\exists e_h'$ such that $\underline{e_h} \geq^0 \underline{e_h'}$ and $e_a' \mapsto \left(\underline{e_a}, \left(\left(\underline{e_h'}\ \lambda h.[\cdot]\right), K_a''\right)\right)$ Then

    $$\begin{aligned}
    \underline{e_a\ e_b} &= \lambda k.\underline{e_a}\ \left(\lambda m.\underline{e_b}\ (\lambda n.(m\ n)\ k)\right) && \text{(TAPP)}\\
    &= \lambda k.\underline{e_a'}\ \left(\lambda m.\underline{e_b}\ (\lambda n.(m\ n)\ k)\right) && \text{(IH)}\\
    &= \underline{e_a'\ e_b} && \text{(TAPP)}
    \end{aligned}$$

    And as $\kappa_1 = \kappa_a[\kappa_b]$ then

    $$\begin{aligned}
    K_2 &= \left(\left(\underline{e_h'}\ \lambda h.[\cdot]\right), K_a''\right) \bowtie (\kappa_b, K_b')\\
    &= \left(\underline{e_h'}\ \lambda h.\kappa_b\right), (K_a'' \bowtie K_b')\\
    &= \left(\underline{e_h'}\ \lambda h.\kappa_b\right), K_1'
    \end{aligned}$$

  - Case $\kappa_a = \left(\underline{e_h}\ \lambda h.\kappa_a'\right)$ with $e_h \notin Value^0$. Similar to previous case.

- Case $(\texttt{EAPP}_\texttt{v})[2]$, $K_1 = \perp$ with $e_1 = e_a\ e_b$, $e_2 = e_a\ e_b'$ and $e_b \geq_\texttt{v}^0 e_b'$.

  $$\begin{aligned}
  \underline{e_a\ e_b} &= \lambda k.\underline{e_a}\ \left(\lambda m.\underline{e_b}\ (\lambda n.(m\ n)\ k)\right) && \text{(TBOX)}\\
  &\geq^0 \lambda k.\underline{e_a}\ \left(\lambda m.\underline{e_b'}\ (\lambda n.(m\ n)\ k)\right) && \text{(IH, EABS, EAPP[2], EABS)}\\
  &\geq^0 \underline{e_a\ e_b'}
  \end{aligned}$$

- Case $(\texttt{EAPP}_\texttt{v})[3]$, $K_1 = \bot$ with $e_1 = (\lambda x.e)\ v$, $e_2 = [x \overset{0}{\mapsto} v]e$ and $v \in Value^0$.

$$\begin{aligned}
\underline{(\lambda x.e)\ v} &= \lambda k.\underline{\lambda x.e}\ (\lambda m.\underline{v}\ (\lambda n.(m\ n)\ k)) && \text{(TAPP)}\\
&= \lambda k.(\lambda k.k\ \lambda x.\underline{e})\ (\lambda m.\underline{v}\ (\lambda n.(m\ n)\ k)) && \text{(TABS)}\\
&\geq^0 \lambda k.((\lambda m.\underline{v}\ (\lambda n.(m\ n)\ k))\ \lambda x.\underline{e}) && \text{(EAPP[3], EABS)}\\
&\geq^0 \lambda k.(\underline{v}\ (\lambda n.((\lambda x.\underline{e})\ n)\ k)) && \text{(EAPP[3], EABS, ETRA)}\\
&\geq^0 \lambda k.((\lambda k.k\ \Psi(v))\ (\lambda n.((\lambda x.\underline{e})\ n)\ k)) && \text{(Lem.2)}\\
&\geq^0 \lambda k.((\lambda n.((\lambda x.\underline{e})\ n)\ k)\ \Psi(v)) && \text{(EAPP[3], EABS, ETRA)}\\
&\geq^0 \lambda k.(((\lambda x.\underline{e})\ \Psi(v))\ k) && \text{(EAPP[3], EABS, ETRA)}\\
&\geq^0 \lambda k.\left(\left([x \overset{0}{\mapsto} \Psi(v)]\underline{e}\right)\ k\right) && \text{(EAPP[3], EABS, ETRA)}\\
&\geq^0 [x \overset{0}{\mapsto} \Psi(v)]\underline{e} && \text{(Lem.11, EABS, ETRA)}\\
&\geq^0 \underline{[x \overset{0}{\mapsto} v]e} && \text{(Lem.3, ETRA)}
\end{aligned}$$

- Case $(\texttt{EALP}_\texttt{v})$ with $e_1 = \lambda x.e$ and $e_2 = \lambda y.[x \overset{0}{\mapsto} y]e$.
  Then $n = depth(e_1) = depth(e_2) = depth(e) = 0$.

$$\begin{aligned}
\underline{\lambda x.e} &= \lambda k.k\ (\lambda x.\underline{e}) && \text{(TABS)}\\
&\geq^0 \lambda k.k\ \left(\lambda y.[x \overset{0}{\mapsto} \Psi(y)]\underline{e}\right) && \text{(EALP, EABS)}\\
&\geq^0 \lambda k.k\ \left(\lambda y.\underline{[x \overset{0}{\mapsto} y]e}\right) && \text{(Lem.3)}\\
&\geq^0 \underline{\lambda y.[x \overset{0}{\mapsto} y]e} && \text{(TABS)}
\end{aligned}$$

- Case $(\texttt{EABS}_\texttt{v})$, $K_1 = \bot$ with $e_1 = \lambda x.e_1'$, $e_2 = \lambda x.e_2'$ and $e_1' \geq^0 e_2'$.

$$\begin{aligned}
\underline{\lambda x.e_1'} &= \lambda k.k\ \left(\lambda x.\underline{e_1'}\right) && \text{(TABS)}\\
&\geq^0 \lambda k.k\ \left(\lambda x.\underline{e_2'}\right) && \text{(EABS, EAPP[2], EABS)}\\
&\geq^0 \underline{\lambda x.e_2'}
\end{aligned}$$

- Case $(\texttt{EBOX}_\texttt{v})$, $K_1 = \bot$ with $e_i = \textbf{box}\ e_i'$ for $i \in \{1, 2\}$.
  Then $n = 0$. By $(\texttt{EBOX}_\texttt{v})$, $e_1' \geq_\texttt{v}^1 e_2'$. If $depth(e_1) = 0$ then by Lem.10 $e_1' = e_2'$ and we are done. As $depth(e_1') \leq depth(e_1) + 1 = 1$, we can apply (IH) on $\underline{e_1'}$. $e_1' \mapsto \left(\underline{e_1'}, (\kappa_1', \bot)\right)$.
  - Case $\kappa_1' = \left(\underline{e_h}\ \lambda h.[\cdot]\right)$ with $e_h \in Value^0$.

By (IH) $[h \overset{1}{\mapsto} \Psi(e_h)]\underline{e'_1} \geq^1 \underline{e'_2}$ and $e'_2 \mapsto \left(\underline{e'_2}, \bot\right)$

$$\begin{aligned}
\underline{e_1} &= \lambda k.\underline{e_h}\ \left(\lambda h.k\ \left(\mathbf{box}\ \underline{e'_1}\right)\right) \\
&= \lambda k.\left(\lambda k.k\ \Psi(e_h)\right)\ \left(\lambda h.k\ \left(\mathbf{box}\ \underline{e'_1}\right)\right) && \text{(Lem.2)} \\
&\geq^0 \lambda k.\left(\left(\lambda h.k\ \left(\mathbf{box}\ \underline{e'_1}\right)\right)\ \Psi(e_h)\right) && \text{(EAPP[3], EABS)} \\
&\geq^0 \lambda k.k\ \left(\mathbf{box}\ [h \overset{1}{\mapsto} \Psi(e_h)]\underline{e'_1}\right) && \text{(EAPP[3], EABS)} \\
&\geq^0 \lambda k.k\ \left(\mathbf{box}\ \underline{e'_2}\right) && \text{(IH, EBOX, EAPP[2], EABS)}
\end{aligned}$$

– Case $\kappa'_1 = \left(\underline{e_h}\ \lambda h.\kappa''_1\right)$ with $e_h \in Value^0$.
By (IH) $\left[h \overset{0}{\mapsto} \Psi(e_h)\right](\kappa''_1, \bot) \geq^0 K'_2$ with $e'_2 \mapsto \left(\underline{e'_2}, (K'_2)\right)$. Then $K'_2 = \kappa''_2, \bot$ and $\kappa''_1 \geq^0 \kappa''_2$.

$$\begin{aligned}
\underline{e_1} &= \lambda k.\underline{e_h}\ \left(\lambda h.\kappa''_1\left[k\ \left(\mathbf{box}\ \underline{e'_1}\right)\right]\right) && \text{(TBOX)} \\
&= \lambda k.\left(\lambda k.k\ \Psi(e_h)\right)\ \left(\lambda h.\kappa''_1\left[k\ \left(\mathbf{box}\ \underline{e'_1}\right)\right]\right) && \text{(Lem.2)} \\
&\geq^0 \lambda k.\left(\left(\lambda h.\kappa''_1\left[k\ \left(\mathbf{box}\ \underline{e'_1}\right)\right]\right)\ \Psi(e_h)\right) && \text{(EAPP[3], EABS)} \\
&\geq^0 \lambda k.\left([h \overset{0}{\mapsto} \Psi(e_h)]\kappa''_1\right)\left[k\ \left(\mathbf{box}\ [h \overset{1}{\mapsto} \Psi(e_h)]\underline{e'_1}\right)\right] && \text{(EAPP[3], EABS)} \\
&\geq^0 \lambda k.\kappa''_2\left[k\ \left(\mathbf{box}\ \underline{e'_2}\right)\right] && \text{(IH, EBOX, EAPP[2], EABS)} \\
&\geq^0 \underline{e_2} && \text{(TBOX)}
\end{aligned}$$

– Case $\kappa'_1 = \left(\underline{e_h}\ \lambda h.[\cdot]\right)$ with $e_h \notin Value^0$.
By (IH) $\exists e'_h$ such that $\underline{e_h} \geq^0 \underline{e'_h}$ and $e'_2 \mapsto \left(\underline{e'_1}, \left(\left(\underline{e'_h}\ \lambda h.[\cdot]\right), \bot\right)\right)$

$$\begin{aligned}
\underline{e_1} &= \lambda k.\underline{e_h}\ \left(\lambda h.k\ \left(\mathbf{box}\ \underline{e'_1}\right)\right) \\
&\geq^0 \lambda k.\underline{e'_h}\ \left(\lambda h.k\ \left(\mathbf{box}\ \underline{e'_1}\right)\right) && \text{(IH, EAPP[1], EABS)} \\
&\geq^0 \underline{e_2}
\end{aligned}$$

– Case $\kappa'_1 = \left(\underline{e_h}\ \lambda h.\kappa''_1\right)$ with $e_h \notin Value^0$.
By (IH) $\exists e'_h$ such that $\underline{e_h} \geq^0 \underline{e'_h}$ and $e'_2 \mapsto \left(\underline{e'_1}, \left(\left(\underline{e'_h}\ \lambda h.\kappa''_1\right), \bot\right)\right)$

$$\begin{aligned}
\underline{e_1} &= \lambda k.\underline{e_h}\ \left(\lambda h.\kappa''_1\left[k\ \left(\mathbf{box}\ \underline{e'_1}\right)\right]\right) \\
&\geq^0 \lambda k.\underline{e'_h}\ \left(\lambda h.\kappa''_1\left[k\ \left(\mathbf{box}\ \underline{e'_1}\right)\right]\right) && \text{(IH, EAPP[1], EABS)} \\
&\geq^0 \underline{e_2}
\end{aligned}$$

- Case ($\texttt{EBOX}_\text{v}$), $K_1 = \kappa_1, K'_1$ with $e_i = \mathbf{box}\ e'_i$ for $i \in \{1, 2\}$ and $e'_1 \geq^{n+1}_\text{v} e'_2$.
If $depth(e'_1) < n + 1$ then by Lem.10 $e'_1 = e'_2$ and we are done. Otherwise we can apply (IH) on e'_1. By ($\texttt{TBOX}$)[1] $e'_1 \mapsto \left(\underline{e'_1}, (\kappa_1, K'_1, \kappa'_1)\right)$. We have four cases.

– Case $\kappa_1 = \left(\underline{e_h}\ \lambda h.[\cdot]\right)$ with $e_h \in Value^0$.

By (IH) $[h \overset{n+1}{\mapsto} \Psi(e_h)]\underline{e'_1} \geq^{n+1} \underline{e'_2}$ and $e'_2 \mapsto \left(\underline{e'_2}, (K_2, \kappa'_2)\right)$ with $(K'_1, \kappa'_1) \geq^n (K_2, \kappa'_2)$.

$$\begin{aligned}
[h \overset{n}{\mapsto} \Psi(e_h)]\underline{e_1} &= [h \overset{n}{\mapsto} \Psi(e_h)]\lambda k.\kappa'_1\left[k\ \left(\mathbf{box}\ \underline{e'_1}\right)\right] \\
&= \lambda k.\left([h \overset{n}{\mapsto} \Psi(e_h)]\kappa'_1\right)\left[k\ \left(\mathbf{box}\ [h \overset{n+1}{\mapsto} \Psi(e_h)]\underline{e'_1}\right)\right] && \text{(A1)} \\
&\geq^n \lambda k.\kappa'_2\left[k\ \left(\mathbf{box}\ \underline{e'_2}\right)\right] && \text{(IH, EBOX, EAPP[2], EABS, Lem.31)} \\
&\geq^n \underline{e_2} && \text{(TBOX)}
\end{aligned}$$

And by (IH) either $K'_1 = K_2$ or $K'_1 \geq^{n-1} K_2$ and so in both cases $K'_1 \geq^{n-1} K_2$. As well, at most one inequality isn't an equality.

– Case $\kappa_1 = \left(\underline{e_h}\ \lambda h.\kappa''_1\right)$ with $e_h \in Value^0$.

As in previous case $[h \overset{n}{\mapsto} \Psi(e_h)]\underline{e_1} \geq^n \underline{e_2}$. By (IH) $(\kappa''_1, K'_1, \kappa'_1) \geq^n (K_2, \kappa'_2)$ so $\kappa''_1, K'_1 \geq^{n-1} K_2$.

– Case $\kappa_1 = \left(\underline{e_h}\ \lambda h.[\cdot]\right)$ with $e_h \notin Value^0$.

By (IH) $\exists e'_h$ such that $\underline{e_h} \geq^0 \underline{e'_h}$ and $e'_2 \mapsto \left(\underline{e'_1}, \left(\left(\underline{e'_h}\ \lambda h.[\cdot]\right), K'_1, \kappa'_1\right)\right)$

By (TBOX)[1] $\mathbf{box}\ e'_2 \mapsto \left(\underline{\mathbf{box}\ e'_1}, \left(\left(\underline{e'_h}\ \lambda h.[\cdot]\right), K'_1\right)\right)$.

– Case $\kappa_1 = \left(\underline{e_h}\ \lambda h.\kappa''_1\right)$ with $e_h \notin Value^0$. Similar to previous case.

• Case (EUNB$_\mathtt{v}$)[1] with $e_i = \mathbf{unbox}\ e'_i$ for $i \in \{1, 2\}$.

$depth(e'_1) = depth(e_1) - 1$ so we can apply (IH) on e'_1.

– Case $e'_i \mapsto \left(\underline{e'_i}, \bot\right)$. By (IH) $\underline{e'_1} \geq^0 \underline{e'_2}$.

By (TUNB) $e_i \mapsto \left(\mathbf{unbox}\ h, \left(\bot, \left(\underline{e'_i}\ \lambda h.[\cdot]\right)\right)\right)$.

So $\underline{e_1} = \underline{e_2}$ and we can take $e'_h = e_i$ to be done.

– Case $e'_1 \mapsto \left(\underline{e'_1}, (\kappa_1, K''_1)\right)$ and $e'_2 \mapsto \left(\underline{e'_2}, K''_2\right)$.

By (TUNB) $e_1 \mapsto \left(\mathbf{unbox}\ h', \left(\kappa_1, K''_1, \left(\underline{e'_1}\ \lambda h'.[\cdot]\right)\right)\right)$

and $e_2 \mapsto \left(\mathbf{unbox}\ h', \left(K''_2, \left(\underline{e'_2}\ \lambda h'.[\cdot]\right)\right)\right)$.

∗ Case $\kappa_1 = \left(\underline{e_h}\ \lambda h.[\cdot]\right)$ with $e_h \in Value^0$.

$$\begin{aligned}
[h \overset{n}{\mapsto} \Psi(e_h)]\,(\mathbf{unbox}\ h') &= \mathbf{unbox}\ h' && \text{(A1)} \\
&\geq^n \mathbf{unbox}\ h' && \text{(EREF)}
\end{aligned}$$

And it is an equality.

$$\begin{aligned}
[h \overset{n-1}{\mapsto} \Psi(e_h)]K'_1 &= [h \overset{n-1}{\mapsto} \Psi(e_h)]\left(K''_1, \left(\underline{e'_1}\lambda h'.[\cdot]\right)\right) \\
&= \left([h \overset{n-2}{\mapsto} \Psi(e_h)]K''_1\right), \left([h \overset{n-1}{\mapsto} \Psi(e_h)]\underline{e'_1}\ \lambda h'.[\cdot]\right) \\
&\geq^{n-1} \left(K''_2, \left(\underline{e'_2}\ \lambda h'.[\cdot]\right)\right) && \text{(IH)}
\end{aligned}$$

∗ Case $\kappa_1 = \left(\underline{e_h}\ \lambda h.\kappa_1''\right)$ with $e_h \in Value^0$.
Like in previous case we have $[h \overset{n}{\mapsto} \Psi(e_h)]\,(\mathbf{unbox}\ h') \geq^n \mathbf{unbox}\ h'$.

$$\begin{aligned}[h \overset{n-1}{\mapsto} \Psi(e_h)]\,(\kappa_1'', K_1') &= [h \overset{n-1}{\mapsto} \Psi(e_h)]\left(\kappa_1'', K_1'', \left(\underline{e_1'}\lambda h'.[\cdot]\right)\right)\\ &= \left([h \overset{n-2}{\mapsto} \Psi(e_h)]\,(\kappa_1'', K_1'')\right), \left([h \overset{n-1}{\mapsto} \Psi(e_h)]\underline{e_1'}\ \lambda h'.[\cdot]\right)\\ &\geq^{n-1} \left(K_2'', \left(\underline{e_2'}\ \lambda h'.[\cdot]\right)\right) \qquad \text{(IH)}\end{aligned}$$

∗ Case $\kappa_1 = \left(\underline{e_h}\ \lambda h.[\cdot]\right)$ with $e_h \notin Value^0$.
We have trivially $\underline{e_1} = \mathbf{unbox}\ h' = \underline{e_2}$. Using (IH), $\exists e_h'$ such that $\underline{e_h} \geq^0 \underline{e_h'}$ and $K_2'' = \left(\underline{e_h'}\ \lambda h.[\cdot]\right)$, so as $K_2 = K_2'', \left(\underline{e_2'}\ \lambda h'.[\cdot]\right)$ we are done.

∗ Case $\kappa_1 = \left(\underline{e_h}\ \lambda h.\kappa_1''\right)$ with $e_h \notin Value^0$. Similar to previous case.

- Case $(\texttt{EUNB}_\texttt{v})[2]$ with $e_1 = \mathbf{unbox}\ \ (\mathbf{box}\ v)$ and $e_2 = v \in Value^0$.
Then $n = 1$. By (TUNB) $e_1 \mapsto (\mathbf{unbox}\ h, (\bot, (\underline{\mathbf{box}\ v}\ \lambda h.[\cdot])))$.

$$\begin{aligned}[h \overset{1}{\mapsto} \Psi\,(\mathbf{box}\ v)]\,(\mathbf{unbox}\ h) &= \mathbf{unbox}\ \ (\mathbf{box}\ \underline{v})\\ &\geq^1 \underline{v} \qquad \text{(EUNB[2])}\end{aligned}$$

$K_1' = \bot$, $K_2 = \bot$ and $[h \overset{0}{\mapsto} \Psi\,(\mathbf{box}\ v)]\bot \geq^0 \bot$ and it is an equality.

- Case $(\texttt{ERUN}_\texttt{v})[1]$, $K_1 = \bot$ with $e_1 = \mathbf{run}\ e_1'$, $e_2 = \mathbf{run}\ e_2'$ and $e_1' \geq^0 e_2'$.

$$\begin{aligned}\underline{\mathbf{run}\ e_1'} &= \lambda k.\left(\underline{e_1'}\ \ (\lambda m.((\mathbf{run}\ m)\ \ k))\right) && \text{(TRUN)}\\ &\geq^0 \lambda k.\left(\underline{e_2'}\ \ (\lambda m.((\mathbf{run}\ m)\ \ k))\right) && \text{(EABS)}\\ &\geq^0 \underline{\mathbf{run}\ e_2'} && \text{(TRUN)}\end{aligned}$$

- Case $(\texttt{ERUN}_\texttt{v})[2]$ with $e_1 = \mathbf{run}\ \ (\mathbf{box}\ v)$, $e_2 = v$ and $v \in Value^0$.

$$\begin{aligned}\underline{\mathbf{run}\ \ (\mathbf{box}\ v)} &= \lambda k.(\underline{\mathbf{box}\ v}\ \ (\lambda m.((\mathbf{run}\ m)\ \ k))) && \text{(TRUN)}\\ &= \lambda k.((\lambda k.k\ \ (\mathbf{box}\ \underline{v}))\ \ (\lambda m.((\mathbf{run}\ m)\ \ k))) && \text{(TBOX)}\\ &\geq^0 \lambda k.((\lambda m.((\mathbf{run}\ m)\ \ k))\ \ (\mathbf{box}\ \underline{v})) && \text{(EAPP[3], EABS)}\\ &\geq^0 \lambda k.((\mathbf{run}\ \ (\mathbf{box}\ \underline{v}))\ \ k) && \text{(EAPP[3], EABS, ETRA)}\\ &\geq^0 \lambda k.(\underline{v}\ k) && \text{(Lem.4, ERUN, EABS, ETRA)}\\ &\geq^0 \underline{v} && \text{(Lem.11, ETRA)}\end{aligned}$$

□

Proof of Th.4.

- If $e_1 =_\texttt{v}^0 e_2$ then by Th.1 $\exists e_3 \in Expr$ such that $e_1 \geq_\texttt{v}^0 e_3$ and $e_2 \geq_\texttt{v}^0 e_3$. As $depth(e_1) = 0$ then by Lem.8 $\underline{e_1} \geq^0 \underline{e_3}$, $\underline{e_2} \geq^0 \underline{e_3}$ and then $\underline{e_1} =^0 \underline{e_3}$.

- Conversely, as our transformation is an extension of Plotkin's one which behaves the same on unstaged expressions, the same example holds: for $e_1 = ((\lambda x.x\ x)\ (\lambda x.x\ x))\ y$ and $e_2 = (\lambda x.x\ y)\ ((\lambda x.x\ x)\ (\lambda x.x\ x))$, then $\underline{e_1} =^0 \underline{e_2}$ but $e_1 \neq^0_{\mathrm{v}} e_2$.

□

References

[1] H.P. Barendregt. *Some extensional term models for combinatory logics and lambda-calculi.* PhD thesis, Utrecht: Rijksuniversiteit Utrecht, 2006.

[2] M.J. Fischer. Lambda calculus schemata. In *ACM SIGACT News*, volume 7, pages 104–109. ACM, 1972.

[3] I.S. Kim, K. Yi, and C. Calcagno. A polymorphic modal type system for lisp-like multi-staged languages. In *CONFERENCE RECORD OF THE ACM SYMPOSIUM ON PRINCIPLES OF PROGRAMMING LANGUAGES*, volume 33, page 257. Citeseer, 2006.

[4] G.D. Plotkin. Call-by-name, call-by-value and the [lambda]-calculus. *Theoretical computer science*, 1(2):125–159, 1975.