Positive Focusing is Directly Useful

Beniamino Accattoli, Jui-Hsuan Wu

Submitted to MFPS 2024

Positive Focusing is Directly Useful

Beniamino Accattoli[1] Jui-Hsuan Wu[2]

Inria & LIX, Ecole Polytechnique, UMR 7161, Palaiseau, France

Abstract

Recently, Miller and Wu introduced the *positive λ-calculus*, a call-by-value λ-calculus with sharing obtained by assigning proof terms to the positively polarized focused proofs for minimal intuitionistic logic. The positive λ-calculus stands out among λ-calculi with sharing for a *compactness* property related to the sharing of variables. We show that—thanks to compactness—the positive calculus neatly captures the core of *useful sharing*, a technique for the study of reasonable time cost models.

Keywords: λ-calculus, sharing, focusing, reasonable cost models

1 Introduction

Andreoli's focusing is a technique for restricting the space of proofs to a subset with good structural properties, called *focused proofs* [14]. Notably, it helps in the *proof search* approach to computation.

Focused proof systems have quickly become a widespread tool in proof theory and—via the Curry-Howard isomorphism—in the study of λ-calculi. They are used well beyond proof search, for instance, in connection to pattern matching by Zeilberger [41] and Krishnaswami [24], proof nets and expansion proofs by Chaudhuri et al. [18,17], synthetic connectives by Chaudhuri [16], abstract machines by Brock-Nannestad et al. [15], decidability of contextual equivalence for sum types by Scherer and Rémy [36] and Scherer [35], contextual equivalence by Rioux and Zdancewic [31], synthetic inference rules by Marin et al. [27], refinement types by Economou et al. [21], and certainly in even more studies. The present paper adds one more instance, studying a connection between focusing and sharing for λ-terms.

Positive Focusing and λ-Terms. In [28], Miller and Wu show how the focused intuitionistic proof system LJF by Liang and Miller [26] can be used to design term structures. In LJF, formulas are polarized. A key theorem of LJF is that different polarizations do not affect provability: if a formula is provable (resp. not provable) in *LJ*, then the formula is provable (resp. not provable) in LJF with *any* polarization. Different polarizations of a provable formula, however, induce focused proofs of very different shapes.

Via annotations of proofs with proof terms, Miller and Wu study the shape of polarized proofs of *minimal* intuitionistic logic, that is, for the logic of implications, which is the basic setting of the Curry-Howard isomorphism. They consider the two natural uniform polarizations having either all atomic formulas polarized negatively, or positively (non-atomic formulas are all negative, because implication is negative). Two very different term structures arise. The negative polarity assignment yields the usual λ-calculus. The

[1] Email: beniamino.accattoli@inria.fr
[2] Email: jwu@lix.polytechnique.fr

positive polarity assignment, instead, yields a quite different syntax, with a restricted form of application and accounting for sharing via a notion of explicit substitution.

The Positive λ-Calculus. Based on this positively-polarized syntax, Wu introduced the *positive λ-calculus* λ_{pos} [40], a calculus with explicit substitutions endowed with call-by-value (shortened to CbV) evaluation, because substituting applications (as in call-by-name) would break the shape of positive terms.

The positive λ-calculus is an unusual calculus and the aim of this paper is to show its relevance. At first sight, it is yet another sharing-based presentations of CbV evaluation. It stands out, however, for a peculiar treatment of variables in relation to sharing, here dubbed *compactness*, that to our knowledge is new. The main contribution of this paper is to show that compactness considerably simplifies the definition and the study of *useful sharing*, a form of shared evaluation first introduced by Accattoli and Dal Lago [5] to study reasonable time cost models for the λ-calculus.

In order to properly explain the novelty of the positive λ-calculus, we are now going to outline three concepts: sharing-based presentations of CbV evaluation, the sharing of variables, and useful sharing.

700 Sharing-Based Presentations of CbV Evaluation. In the literature, sharing is an overloaded word. The most basic form of sharing is *sub-term sharing*, which can be obtained by simply adding to the standard syntax of the λ-calculus a $\mathsf{let}\ x = u\ \mathsf{in}\ t$ construct, which shares u between all the occurrences of x in t. The use of let-expressions is pervasive in presentations of CbV λ-calculi, typically in Moggi [29,30]. In $\mathsf{let}\ x = u\ \mathsf{in}\ t$, it is usually assumed that u shall be evaluated before t, but such an assumption is often dropped when studying sharing. Additionally, $\mathsf{let}\ x = u\ \mathsf{in}\ t$ is often rather more concisely noted as an explicit substitution $t[x{\leftarrow}u]$ (in this paper, meta-level substitution is noted $t\{x{\leftarrow}u\}$).

In a CbV setting, having both explicit substitutions $t[x{\leftarrow}u]$ and applications tu is somewhat redundant, as explicit substitutions can be used to constrain the shape of applications. It is possible, indeed, to have only values v for one or both sub-terms of applications tu, that is, they can be constrained to be of shape vu (or tv, or even vv'). The idea is to apply a transformation $[\![\cdot]\!]$ turning tu into $(x[\![u]\!])[x{\leftarrow}[\![t]\!]]$ with x fresh (or into $([\![t]\!]x)[x{\leftarrow}[\![u]\!]]$, or $(xy)[x{\leftarrow}[\![t]\!]][y{\leftarrow}[\![u]\!]]$ with y fresh). It is typical of CbV (rather than call-by-name), because, for the restriction to be stable by reduction, it should be forbidden to substitute applications. Instances of CbV calculi with restrained applications abound, two notable examples being the calculus of *A-normal forms* by Sabry and Felleisen [32,22] and the *fine-grained* CbV calculus by Levy et al. [25]. Applications are also restricted in Sestoft's study of call-by-need [37], or that of Walker's on substructural type systems [38].

We shall follow Accattoli et al. [3] and refer to these decompositions of applications as to *crumbled* calculi. By tweaking the rewriting rules, one can also further restrict the immediate sub-terms of applications to be variables (rather than values). This gives rise to at least nine different CbV calculi, one with ordinary application tu and eight crumbled variants vu, xu, tv', vv', xv', ty, vy, and xy. One can also decide at least (each choice doubling the number of possible presentations):

- *Nested vs flattened ES*: whether explicit substitutions can be nested (as in $t[x{\leftarrow}u[y{\leftarrow}r]]$) or have to be flattened (as in $t[x{\leftarrow}u][y{\leftarrow}r]$);
- *Granularity of substitutions*: whether evaluation is small-step (substitution acts on all occurrences of a variable at once) or micro-step (variable occurrences are replaced one at a time);
- *Variables vs values*: whether variables are values, and thus can trigger substitution redexes, or not, that is, only abstractions can be substituted.

None of these choices affects the expressivity of the calculus, but the various calculi have different properties and degrees of manageability. In particular, the proof that a crumbled calculus is as expressive as the full one depends on the choices, and the proof might be non-trivial, as we shall see.

According to this classification, Wu's positive λ-calculus is a crumbled calculus with applications of shape xy (the minimalistic one), flattened substitutions, micro-step, and where variables are not values. It distinguishes itself from all these calculi, however, because it also has a new *compactness* feature that solves a pervasive issue of sub-term sharing.

Sharing of Variables and Compactness. In λ-calculi with sharing, usually variables can be shared, that

is $t[x{\leftarrow}y]$ is a valid term. This leads to the possibility of having *renaming chains*, such as:

$$t[x_1{\leftarrow}x_2][x_2{\leftarrow}x_3]\dots[x_{n-1}{\leftarrow}x_n] \tag{1}$$

These chains are an issue because they lead to both space and time inefficiencies. Optimizations to prevent their creation are adopted for instance by Sands et al. [34], Wand [39], Friedman et al. [23], and Sestoft [37]. The first systematic study of renaming chains is by Accattoli and Sacerdoti Coen [13], with respect to time, where they show that it is enough to remove variables from values to avoid time inefficiencies related to renaming chains. Recently, Accattoli et al. have shown that the dynamic removal of renaming chains is essential for the only known reasonable notion of logarithmic space in the λ-calculus [6].

The new feature of the positive λ-calculus λ_{pos} is that *it does not share variables*, that is, $t[x{\leftarrow}y]$ does *not* belong to the syntax of λ_{pos}. This fact removes the issue of renaming chains altogether, with no need to design optimizations to prevent their creations, or removing variables from values, because renaming chains simply *cannot be expressed*.

In fact, λ_{pos} pushes things one step further, forging a sort of syntactical duality with respect to sharing:

- Variables cannot appear in explicit substitutions and, additionally, are the only constructors beside explicit substitutions;
- Dually, abstractions and applications appear in explicit substitutions but *not* out of them.

That is, x is a positive term but, for instance, xy and $\lambda x.x$ are not positive terms, only $z[z{\leftarrow}xy]$ and $z[z{\leftarrow}\lambda x.x]$ (or more generally $u[z{\leftarrow}xy]$ and $u[z{\leftarrow}\lambda x.x]$, where u is a positive term) are positive terms. This removes the further issue of whether xy and $z[z{\leftarrow}xy]$ are distinct terms, which is relevant in the study of CbV program equivalences, see Accattoli et al. [7]. One might argue that a similar approach to applications is already present in calculi related to the sequent calculus (in the style of Curien and Herbelin [19]), but to our knowledge the extension of this approach to abstractions is new, and relevant for useful sharing, see Sect. 5.

We dub *compactness* the fact that variables are not shared and are the only terms out of ESs. With compact sub-term sharing, terms have shape $x_1[x_1{\leftarrow}u_1]\dots[x_n{\leftarrow}u_1]$, where each of the $u_1,\dots,\ u_n$ is an abstraction or an application (which do not belong to the grammar of terms).

Useful Sharing. Our aim here is promoting the compactness of λ_{pos} by showing its impact on *useful sharing*, a concept apparently unrelated to focusing. Useful sharing is a technique introduced by Accattoli and Dal Lago [5], and then refined in call-by-value by Accattoli et al. [4], to study reasonable time cost models for λ-calculi. It works at the level of micro-step evaluation, that is, of replacements of single variable occurrences. The basic idea is that (in CbV) one should replace a variable occurrence with a copy of a shared abstraction only when it is useful to create β-redexes, that is, in the following case:

$$(xt)[x{\leftarrow}\lambda y.u] \quad\to\quad ((\lambda y.u)t)[x{\leftarrow}\lambda y.u] \tag{2}$$

While one should avoid replacements that do not create β-redexes, i.e. are not useful, as the following one:

$$(tx)[x{\leftarrow}\lambda y.u] \quad\to\quad (t(\lambda y.u))[x{\leftarrow}\lambda y.u] \tag{3}$$

Avoiding non-useful replacements leads to considerable speed-ups, that can even be *exponential* for some terms, in the case of strong evaluation (that is, when evaluation goes under abstraction) [5,4].

While the intuition behind useful sharing is easy to convey, the technical details are complex, because cases (2) and (3) are not the only possible ones, and various complications arise. A first simplified setting for useful sharing is given by crumbled λ-calculi where arguments can only be variables (that is, with applications of shape ty, vy, or xy), since then non-useful replacements such as those in (3) cannot be expressed and are then ruled out. This is already known and used by Accattoli et al. [4].

One of the complex aspects of useful sharing is related to renaming chains (as in (1)). They force the distinction between steps as in (2), which are *directly useful*, and steps over a renaming chain such as:

$$(x_1t)[x_1{\leftarrow}x_2]\dots[x_{n-1}{\leftarrow}x_n][x_n{\leftarrow}\lambda y.u] \quad\to\quad (x_1t)[x_1{\leftarrow}x_2]\dots[x_{n-1}{\leftarrow}\lambda y.u][x_n{\leftarrow}\lambda y.u] \tag{4}$$

These replacements are *indirectly useful*: they do not directly create a β-redex and yet they contribute to the *future* creation of β-redexes. Continuing with evaluation, indeed, $\lambda y.u$ shall replace the content of all the explicit substitutions in the chain and finally be substituted for x_1, creating a β-redex. Indirectly useful steps cannot be avoided, otherwise some β-redexes are never created, and evaluation gets stuck.

Thanks to compactness, the positive λ-calculus λ_{pos} has no renaming chains and thus indirectly useful replacements are simply *ruled out*. Evaluation is not stuck in λ_{pos}, though, because indirectly useful steps somehow transform into directly useful steps in λ_{pos}, as explained in Sect. 6. Since λ_{pos} also has minimalistic applications of shape xy, non-useful replacements are ruled out as well. Therefore, λ_{pos} has only useful replacements, and only the *directly* useful ones (hence the title). Essentially, λ_{pos} captures the *core* of useful sharing, ruling out the technicalities.

Open CbV. Concretely, we develop our study with respect to Accattoli and Guerrieri's framework of *Open CbV* [8], that is, we consider *weak evaluation* (i.e. not under abstraction) and *possibly open terms*. It is an intermediate setting between:

(i) Weak evaluation and closed terms, where many subtleties of useful sharing are not observable, and

(ii) Strong evaluation (which implies dealing with open terms), where further technicalities arise.

The same approach is followed by other works on useful sharing by Accattoli and co-authors [12,9,10]—see in particular the introduction of [10] for an extensive discussion of this choice.

Architecture of Our Study. After all these preliminary explanations, we can now explain our results. As a reference for a sharing-based λ-calculus for CbV, we take (a micro-step presentation of) Accattoli and Paolini's *value substitution calculus* (VSC) [11], which is also the reference for the study of CbV reasonable time and useful sharing by Accattoli et al. [4]. The VSC has (possibly nested) explicit substitutions and ordinary applications of shape tu. We then define (our slight variant of) Wu's positive λ-calculus λ_{pos}.

Intuitively, the paper is devoted to proving the equivalence of the Open λ_{pos} and the Open VSC. Often, the equivalence of two calculi is stated as a bisimulation property. For crumbled calculi ruling out some steps—as it is the case for the positive λ-calculus—this is not possible, because the ruled out steps cannot be simulated. A finer approach is needed.

We thus define a *core sub-calculus* of the Open VSC, which has only directly useful replacements, and prove a *factorization theorem* for the Open VSC, stating that every reduction sequence can be factored into a core one followed by a non-core one. Moreover, we provide a theorem ensuring that the Open VSC and the Core Open VSC are termination equivalent.

Next, we define a translation $[\![\cdot]\!]$ from the VSC to λ_{pos} and show that:

(i) *Simulation*: it induces a simulation of the Core Open VSC by the Open λ_{pos}, and

(ii) *Normal forms*: it sends core normal forms into appropriate normal forms of λ_{pos}.

From these properties, it follows that the Core Open VSC and the Open λ_{pos} are termination equivalent.

This proof technique is an original contribution of the paper. In particular, it is the first time that usefulness is justified via a factorization theorem.

Lastly, we show that the compactness of λ_{pos} induces a further relevant property: its open evaluation has the *diamond property*, a strong form of confluence, which is not true for the (micro-step) Open VSC.

Proofs. Proofs are in the Appendix.

2 Preliminaries: Rewriting Notions and Notations

Given a rewriting relation $\to_{\mathsf{r}}$, we write $d\colon t \to_{\mathsf{r}}^* u$ for a $\to_{\mathsf{r}}$-reduction sequence from t to u, the length of which is noted $|d|$. Moreover, we use $|d|_a$ for the number of a-*steps* in d, for a sub-relation $\to_a$ of $\to_{\mathsf{r}}$.

A term t is *weakly* r-*normalizing* if $d\colon t \to_{\mathsf{r}}^* u$ with u r-normal; and t is *strongly* r-*normalizing* if there are no diverging r-sequences from t, or, equivalently, if all its reducts are strongly r-normalizing.

Given two reductions $\to_1$ and $\to_2$ we use $\to_1\to_2$ for their composition, defined as $t \to_1\to_2 u$ if $t \to_1 r \to_2 u$ for some r. We also write, *e.g.*, $\to_1\to_2 \subseteq \to_3^*$ to state that $t \to_1\to_2 u$ implies $t \to_3^* u$

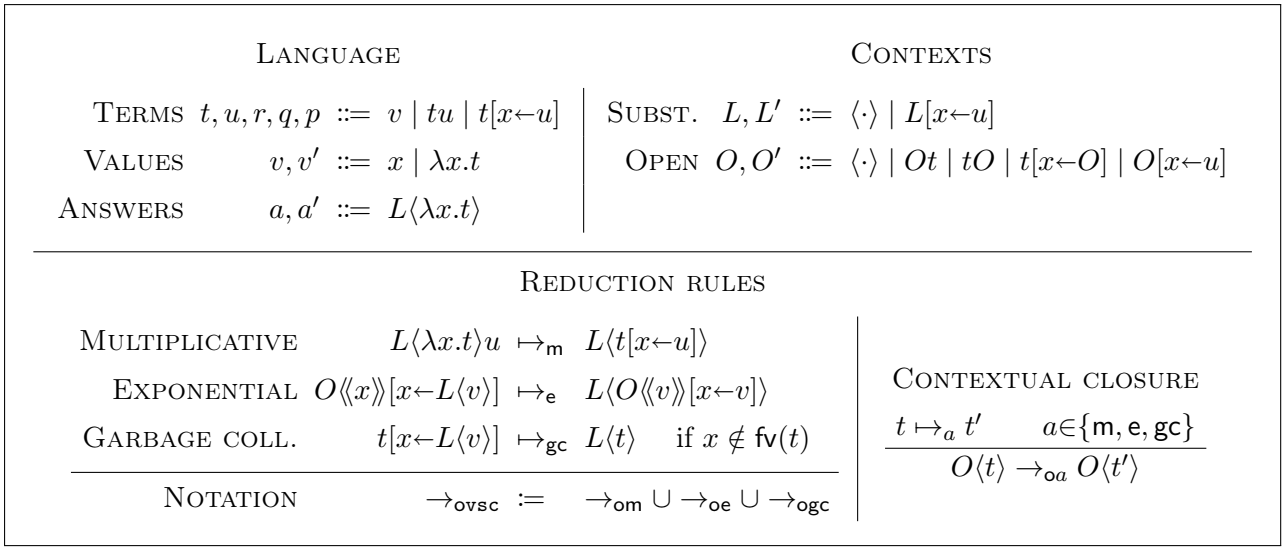

Fig. 1. The Open (Micro-Step) Value Substitution Calculus λ_{ovsc}.

Diamond Property. According to Dal Lago and Martini [20], a relation $\rightarrow_{\mathsf{r}}$ is *diamond* if $u_1\, {}_{\mathsf{r}}\!\leftarrow t \rightarrow_{\mathsf{r}} u_2$ and $u_1 \neq u_2$ imply $u_1 \rightarrow_{\mathsf{r}} r\, {}_{\mathsf{r}}\!\leftarrow u_2$ for some r. If $\rightarrow_{\mathsf{r}}$ is diamond then:

(i) *Confluence*: $\rightarrow_{\mathsf{r}}$ is confluent, that is, $u_1\, {}^*_{\mathsf{r}}\!\leftarrow t \rightarrow^*_{\mathsf{r}} u_2$ implies $u_1 \rightarrow^*_{\mathsf{r}} r\, {}^*_{\mathsf{r}}\!\leftarrow u_2$ for some r;

(ii) *Length invariance*: all r-reduction sequences to normal form with the same start term have the same length (*i.e.* if $d\colon t \rightarrow^*_{\mathsf{r}} u$ and $d'\colon t \rightarrow^*_{\mathsf{r}} u$ with u $\rightarrow_{\mathsf{r}}$-normal then $|d| = |d'|$);

(iii) *Uniform normalization*: t is weakly r-normalizing if and only if it is strongly r-normalizing.

Basically, the diamond property captures a more liberal form of determinism.

3 The (Micro-Step) Open Value Substitution Calculus

In this section, we present our system of reference for Open CbV, Accattoli and Paolini's *value substitution calculus* [11] (shortened to VSC). We shall adopt a micro-step presentation (explained below) of the open fragment, that is, of weak evaluation on possibly open terms. This variant is defined in Fig. 1 and shall be denoted with λ_{ovsc}. We also recall some of its basic properties, which shall either be used in the next section or used to compare λ_{ovsc} with the positive λ-calculus.

Terms. The VSC is a CbV λ-calculus extended with let-expressions, similarly to Moggi's CbV calculus [29,30]. We do however write a let-expression $\mathsf{let}\ x = u\ \mathsf{in}\ t$ as a more compact *explicit substitution* $t[x\leftarrow u]$ (ES for short), which binds x in t. Moreover, our let/ES does not fix an order of evaluation between t and u, in contrast to many papers in the literature (*e.g.* Sabry and Wadler [33] or Levy et al. [25]) where u is evaluated first. Intuitively, $t[x\leftarrow u]$ is a construct for *sub-term sharing*, as for instance is evident when comparing tt and $(xx)[x\leftarrow t]$, where t is shared in the latter.

The set of free variables of a term t is denoted by $\mathsf{fv}(t)$ and terms are identified up to α-renaming. We use $t\{x\leftarrow u\}$ for the capture-avoiding substitution of t for each free occurrence of x in t.

Contexts. We shall use (many notions of) *contexts*, *i.e.* terms with a hole, noted $\langle\cdot\rangle$. The most general notion of context used here is open contexts O, for which the hole cannot appear under abstraction and where the adjective *open* means *possibly open* (that is, possibly with occurrences of free variables), and not *necessarily open*. We shall also extensively use *substitution contexts* L, which are simply lists of ESs. Plugging a term t in a context O is noted $O\langle t\rangle$ and can capture variables, for instance $(\lambda x.\lambda y.\langle\cdot\rangle)\langle xy\rangle = \lambda x.\lambda y.xy$ (while $(\lambda x.\lambda y.z)\{z\leftarrow xy\} = \lambda x'.\lambda y'.xy$); we use $O\langle\!\langle t\rangle\!\rangle$ when we want to prevent it.

Substitution contexts are in particular used to define *answers*, which are abstractions surrounded by a list of substitutions. Answers shall play a key role in relating λ_{ovsc} with the positive λ-calculus in Sect. 7.

Reduction Rules. The reduction rules of VSC are slightly unusual as they use *contexts* both to allow one to reduce redexes located in sub-terms, which is standard, *and* to define the redexes themselves, which is less standard—this kind of rules is called *at a distance*. The rewriting rules at a distance of λ_{ovsc} are related to cut-elimination on proof nets, via the CbV translation $(A \Rightarrow B)^v = !(A^v \multimap B^v)$ of intuitionistic logic into linear logic, see Accattoli [1]. The linear logic connection is also the reason behind the *multiplicative* and *exponential* terminology for the rewriting rules.

The multiplicative root rule $\mapsto_{\mathsf{m}}$ turns a β-redex (at a distance, because of the substitution context L) into an ES. Note that there is no *by-value* restriction, rule $\mapsto_{\mathsf{m}}$ can fire also if the argument is not a value. The by-value restriction is part of rules $\mapsto_{\mathsf{e}}$ and $\mapsto_{\mathsf{gc}}$ which take care of the substitution process.

The VSC usually appears with a single *small-step* substitution / exponential rule, where small-step refers to the fact that it is based on meta-level substitution $t\{x \leftarrow v\}$, which replaces all the occurrences of the bound variable x at once. Here, we adopt a *micro-step* presentation: the exponential rule $\mapsto_{\mathsf{e}}$ replaces a *single* variable occurrence at a time, and there is an additional garbage collection rule $\mapsto_{\mathsf{gc}}$ for removing ESs bounding variables with no occurrences. These two rules can duplicate / erase only values. Both root rules are *at a distance* in that they involve a substitution context L, which is not duplicated / erased.

The rewriting relation $\to_{\mathsf{ovsc}}$ of λ_{ovsc} is obtained by closing the root rewriting rules $\mapsto_{\mathsf{m}}$, $\mapsto_{\mathsf{e}}$, and $\mapsto_{\mathsf{gc}}$ by open contexts O and by taking the union of the obtained rules.

Confluence and Lack of Diamond. The VSC is confluent and the open small-step VSC even has the *diamond property* (defined in Sect. 2), see Accattoli and Paolini [11]. In the literature, there is no proof that the variant λ_{ovsc} used here is confluent, but this can be easily proved by adjusting the proof in [11]. We omit the details since the proof is absolutely standard. For the study in this paper, instead, it is worth pointing out that λ_{ovsc} is *not* diamond, as the following diagram shows:

$$\begin{array}{ccccc} O\langle\!\langle x \rangle\!\rangle[x \leftarrow y][y \leftarrow v] & & \xrightarrow{\quad\mathsf{oe}\quad} & & O\langle\!\langle x \rangle\!\rangle[x \leftarrow v][y \leftarrow v] \\ \Big\downarrow{\scriptstyle \mathsf{oe}} & & & & \vdots{\scriptstyle \mathsf{oe}} \\ O\langle\!\langle y \rangle\!\rangle[x \leftarrow y][y \leftarrow v] & \xrightarrow{\mathsf{oe}} & O\langle\!\langle y \rangle\!\rangle[x \leftarrow v][y \leftarrow v] & \xrightarrow{\mathsf{oe}} & O\langle\!\langle v \rangle\!\rangle[x \leftarrow v][y \leftarrow v] \end{array}$$

Postponement of Garbage Collection. A typical property of micro-step λ-calculi at a distance is the possibility of postponing garbage collection (GC). In our setting, the postponement preserves both the number of non-GC steps (which is standard) and the number of GC steps (which is not always the case) because GC steps cannot be duplicated in call-by-value weak evaluation (since only values are duplicated, but GC steps cannot take place inside values because of weakness). As it is standard, the (global) postponement of GC is obtained by iterating a local form of postponement. Notation: $\to_{\mathsf{o}\neg\mathsf{gc}} := \to_{\mathsf{om}} \cup \to_{\mathsf{oe}}$.

Proposition 3.1 (Local postponement of garbage collection, proof at p. 20) *For $a \in \{\mathsf{m}, \mathsf{e}\}$, If $t \to_{\mathsf{ogc}} \to_{oa} u$, then $t \to_{oa} \to_{\mathsf{ogc}} u$.*

Proposition 3.2 (Postponement of garbage collection, proof at p. 24) *If $d : t \to_{\mathsf{o}}^* u$, then there exist reduction sequences $e : t \to_{\mathsf{o}\neg\mathsf{gc}}^* u'$ and $f : u' \to_{\mathsf{ogc}}^* u$ with $|e|_{\mathsf{om}} = |d|_{\mathsf{om}}$, $|e|_{\mathsf{oe}} = |d|_{\mathsf{oe}}$, and $|f| = |d|_{\mathsf{ogc}}$.*

Local Termination. As it is often the case when β-reduction is decomposed into smaller rules, every single rule of λ_{ovsc} is strongly normalizing separately (it is only together that they may diverge, namely $\to_{\mathsf{o}\neg\mathsf{gc}}$ diverges on some terms).

Proposition 3.3 (Local termination, proof at p. 24) *Let $a \in \{\mathsf{m}, \mathsf{e}, \mathsf{gc}\}$. Relation $\to_{oa}$ is strongly normalizing. Moreover, $\to_{\mathsf{oe}} \cup \to_{\mathsf{ogc}}$ is strongly normalizing.*

Renaming Chains. In λ_{ovsc}, there can be renaming chains such as $t[x_1 \leftarrow x_2][x_2 \leftarrow x_3] \dots [x_{n-1} \leftarrow x_n]$. The issue with these chains is that some terms dynamically create longer and longer chains, slowing down the evaluation process. The simplest term on which the issue can be observed is the looping combinator Ω:

$$\frac{}{\Gamma, x : A \vdash x : A}\ var \qquad set(y : A, z : A) \subseteq \Gamma\ \frac{\Gamma, x : A \vdash t : A}{\Gamma \vdash t[x{\leftarrow}yz] : A}\ app$$

$$\frac{\Gamma, y : A \vdash u : A \qquad \Gamma, x : A \vdash t : A}{\Gamma \vdash t[x{\leftarrow}\lambda y.u] : A}\ abs$$

Fig. 2. The term formation rules for positive λ-terms.

$$\text{Terms} \quad t, u, r ::= x \mid t[x{\leftarrow}yz] \mid t[x{\leftarrow}\lambda y.u]$$
$$\text{Evaluation Ctxs} \quad E ::= \langle\cdot\rangle \mid E[x{\leftarrow}yz] \mid E[x{\leftarrow}\lambda y.t]$$

Root reduction rules

$$E\langle t[x{\leftarrow}yz]\rangle[y{\leftarrow}\lambda w.E'\langle w'\rangle] \mapsto_{\mathsf{eme}_+} E\langle E'\langle t\{x{\leftarrow}w'\}\rangle\{w{\leftarrow}z\}\rangle[y{\leftarrow}\lambda w.E'\langle w'\rangle]$$
$$t[x{\leftarrow}\lambda y.u] \mapsto_{\mathsf{gc}_+} t \quad \text{if } x \notin \mathsf{fv}(t)$$

Notation $\to_{\mathsf{o}_+} := \to_{\mathsf{oeme}_+} \cup \to_{\mathsf{ogc}_+}$

Ctx closure $\dfrac{t \mapsto_a t'}{E\langle t\rangle \to_{\mathsf{o}a} E\langle t'\rangle}\ a \in \{\mathsf{eme}_+, \mathsf{gc}_+\}$

Fig. 3. The open positive λ-calculus λ_{pos}.

$$\begin{array}{lll}
\Omega = & (\lambda x.xx)(\lambda x.xx) & \to_{\mathsf{om}}\ (x_1x_1)[x_1{\leftarrow}\lambda x.xx] \\
\to_{\mathsf{oe}} & ((\lambda x.xx)x_1)[x_1{\leftarrow}\lambda x.xx] & \to_{\mathsf{om}}\ (x_2x_2)[x_2{\leftarrow}x_1][x_1{\leftarrow}\lambda x.xx] \\
\to_{\mathsf{oe}} & (x_1x_2)[x_2{\leftarrow}x_1][x_1{\leftarrow}\lambda x.xx] & \to_{\mathsf{oe}}\ ((\lambda x.xx)x_2)[x_2{\leftarrow}x_1][x_1{\leftarrow}\lambda x.xx] \\
\to_{\mathsf{om}} & (x_3x_3)[x_3{\leftarrow}x_2][x_2{\leftarrow}x_1][x_1{\leftarrow}\lambda x.xx] & \to_{\mathsf{oe}}\ (x_2x_3)[x_3{\leftarrow}x_2][x_2{\leftarrow}x_1][x_1{\leftarrow}\lambda x.xx] \\
\to_{\mathsf{oe}} & (x_1x_3)[x_3{\leftarrow}x_2][x_2{\leftarrow}x_1][x_1{\leftarrow}\lambda x.xx] & \to_{\mathsf{oe}}\ ((\lambda x.xx)x_3)[x_3{\leftarrow}x_2][x_2{\leftarrow}x_1][x_1{\leftarrow}\lambda x.xx]\ \cdots
\end{array}$$

Note that after each $\to_{\mathsf{om}}$ step, evaluation does a sequence of $\to_{\mathsf{oe}}$ steps having length equal to the number of preceeding $\to_{\mathsf{om}}$ steps. Easy calculations show that the number of $\to_{\mathsf{oe}}$ steps is then *quadratic* in the number of $\to_{\mathsf{om}}$ steps, for these sequences. This quadratic overhead was first pointed out and studied by Accattoli and Sacerdoti Coen [13]. They show that it is enough to remove variables from values (as it is done in most implementative studies, but usually without an explanation for this choice), in order to remove this issue, since evaluation then rather proceeds as follows:

$$\begin{array}{lll}
\Omega = & (\lambda x.xx)(\lambda x.xx) & \to_{\mathsf{om}}\ (x_1x_1)[x_1{\leftarrow}\lambda x.xx] \\
\to_{\mathsf{oe}} & ((\lambda x.xx)x_1)[x_1{\leftarrow}\lambda x.xx] & \to_{\mathsf{om}}\ (x_2x_2)[x_2{\leftarrow}x_1][x_1{\leftarrow}\lambda x.xx] \\
\to_{\mathsf{oe}} & (x_2x_2)[x_2{\leftarrow}\lambda x.xx][x_1{\leftarrow}\lambda x.xx] & \to_{\mathsf{oe}}\ ((\lambda x.xx)x_2)[x_2{\leftarrow}\lambda x.xx][x_1{\leftarrow}\lambda x.xx] \\
\to_{\mathsf{om}} & (x_3x_3)[x_3{\leftarrow}x_2][x_2{\leftarrow}\lambda x.xx][x_1{\leftarrow}\lambda x.xx] & \to_{\mathsf{oe}}\ (x_3x_3)[x_3{\leftarrow}\lambda x.xx][x_2{\leftarrow}\lambda x.xx][x_1{\leftarrow}\lambda x.xx] \\
\to_{\mathsf{oe}} & ((\lambda x.xx)x_3)[x_3{\leftarrow}\lambda x.xx][x_2{\leftarrow}\lambda x.xx][x_1{\leftarrow}\lambda x.xx] & \to_{\mathsf{om}}\ \cdots
\end{array}$$

And it is easily seen that the number of $\to_{\mathsf{oe}}$ steps is now *linear* in the number of $\to_{\mathsf{om}}$ steps. The positive λ-calculus of the next section shall subsume this approach, by forbidding altogether ESs containing a variable, thus also removing the ambiguity of whether variables are values or not.

4 The (Explicit) Open Positive λ-Calculus

Here, we present the open fragment λ_{opos} of Wu's positive λ-calculus λ_{pos}, and then slightly refine it into an *explicit* variant λ_{oxpos}. The definition of λ_{opos} is in Fig. 3. Terms of λ_{pos} are a subset of VSC terms.

Terms. In λ_{opos}, as in the λ-calculus, there are only three constructors, variables, applications, and abstractions. There are however, various differences, namely:

- Applications are only between variables, that is, of the shape yz;
- Applications and abstractions are always shared, that is, standalone applications and abstractions are not allowed in the grammar. They can only be introduced by the explicit substitution constructs $[x{\leftarrow}yz]$ and $[x{\leftarrow}\lambda y.u]$;
- Positive sharing is peculiar as positive terms are *not* shared in general, that is, $t[x{\leftarrow}u]$ is not a positive term. There are only two distinct sharing constructors for applications and abstractions, but no construct for sharing variables or applications/abstractions with top-level sharing;

Contexts. The notions of substitution contexts L and open contexts O of the λ_{ovsc} pleasantly collapse onto a single notion of *evaluation contexts* E in λ_{opos}, because the additional clauses for O in λ_{ovsc} (for the sub-terms of applications and inside ESs) do not make sense in λ_{opos}, due to the restricted shape of terms. As it is immediately seen from the grammar of positive terms, every positive term t can be written uniquely as $E\langle x\rangle$ for some x and E, with E possibly capturing x.

Rewriting Rules. There are two rewriting rules at a distance, based on open contexts and defined in Fig. 3. Rule $\to_{\mathsf{ogc}_+}$ handles garbage collection and is standard. Rule $\to_{\mathsf{oeme}_+}$ is quite heavy, essentially it is a macro reduction step concatenating two steps of the VSC plus a meta-level substitution. Let us explain it (it is not necessary to grasp it completely, since right after we shall introduce a slight variant of the positive calculus that has simpler rewriting rules). Rule $\to_{\mathsf{oeme}_+}$ does three operations, and also adopts some notations to respect the constrained shape of terms:

(i) *Useful exponential step*: it does the useful[3] replacement of an applied occurrence of y, i.e. it acts on y on $E\langle t[x{\leftarrow}yz]\rangle$, by an abstraction $\lambda w.u$.

(ii) *Created multiplicative step*: since $t[x{\leftarrow}(\lambda w.u)z]$ is not a construct of λ_{pos}, the rule has to also do on-the-fly what in the VSC is a multiplicative step, which would create the ES $u[w{\leftarrow}z]$.

(iii) *Further meta-level substitution*: but since $u[w{\leftarrow}z]$ is not a construct of λ_{pos} either, the step also does on-the-fly the meta-level substitution associated to $[w{\leftarrow}z]$.

(iv) *Respecting the syntax*: to write the reduct, one needs to respect the constrained shape of positive terms, which requires to write u as $E'\langle w'\rangle$ and then do the re-arrangement of the term structure and the meta-level substitutions of variables specified by the rule in Fig. 3.

Working with such a rule is heavy. A first reason is that, quite simply, it involves a lot of symbols. Another reason is that, by concatenating steps of different nature of the VSC, λ_{opos} is *not* conservative over λ_{ovsc} in the sense that it breaks one of its key property, namely local termination (stated by Prop. 3.3). Rule $\to_{\mathsf{oeme}_+}$ can indeed diverge by itself, as in the following representation of the looping term Ω in λ_{opos}:

$$x[x{\leftarrow}yy][y{\leftarrow}\lambda z.w[w{\leftarrow}zz]] \to_{\mathsf{oeme}_+} x[x{\leftarrow}yy][y{\leftarrow}\lambda z.w[w{\leftarrow}zz]] \to_{\mathsf{oeme}_+} \cdots$$

We are then going to decompose $\to_{\mathsf{oeme}_+}$ into two rules, by adding a new constructor to the calculus.

The Explicit Positive λ-Calculus. The explicit positive calculus λ_{oxpos} is defined in Fig. 4. The idea is to extend λ_{opos} by adding the intermediate construct $t[x{\leftarrow}(\lambda w.u)z]$ which allows us to decompose rule $\mapsto_{\mathsf{eme}_+}$ in two, separating the first operation, the useful exponential step, now noted $\mapsto_{\mathsf{e}_+}$, from the rest of the rule, now noted $\mapsto_{\mathsf{m}_+}$. Rule $\mapsto_{\mathsf{m}_+}$ still corresponds to two actions (the multiplicative step of the VSC and the meta-level substitution of the variable), but it is one of the key points of λ_{opos} that variable substitutions disappear, so we shall not decompose the rule further. Intuitively, the new construct $t[x{\leftarrow}(\lambda w.u)z]$ plays a similar (but dual) role to the introduction of ESs in the λ-calculus to decompose β-redexes in two, as it is an explicit β-redex. This is why we refer to the obtained calculus as to the *explicit positive calculus*. One might also see it as switching the shape of (compact) applications of λ_{pos} from xy to vy.

Clearly, λ_{oxpos} simulates λ_{opos}: if $t \to_{\mathsf{oeme}_+} u$ then $t \to_{\mathsf{oe}_+} \to_{\mathsf{om}_+} u$.

[3] Useful steps shall be defined and studied in the next section. Here we rest on the intuitive description given in the introduction.

$$\text{Terms}\quad t,u,r ::= x \mid t[x{\leftarrow}yz] \mid t[x{\leftarrow}\lambda y.u] \mid [x{\leftarrow}(\lambda y.u)z]$$
$$\text{Evaluation Ctxs}\quad E ::= \langle\cdot\rangle \mid E[x{\leftarrow}yz] \mid E[x{\leftarrow}\lambda y.t] \mid E[x{\leftarrow}(\lambda y.t)z]$$

Root reduction rules

$$t[x{\leftarrow}(\lambda y.E\langle z\rangle)w] \mapsto_{\mathsf{m}_+} E\langle t\{x{\leftarrow}z\}\rangle\{y{\leftarrow}w\}$$
$$E\langle t[x{\leftarrow}yz]\rangle[y{\leftarrow}\lambda w.u] \mapsto_{\mathsf{e}_+} E\langle t[x{\leftarrow}(\lambda w.u)z]\rangle[y{\leftarrow}\lambda w.u]$$
$$t[x{\leftarrow}\lambda y.u] \mapsto_{\mathsf{gc}_+} t \quad \text{if } x \notin \mathsf{fv}(t)$$

Notation $\to_{\mathsf{ox}_+} := \to_{\mathsf{om}_+} \cup \to_{\mathsf{oe}_+} \cup \to_{\mathsf{ogc}_+}$

Ctx closure $\dfrac{t \mapsto_a t'}{E\langle t\rangle \to_{oa} E\langle t'\rangle}\ a{\in}\{\mathsf{m}_+,\mathsf{e}_+,\mathsf{gc}_+\}$

Fig. 4. The open explicit positive λ-calculus λ_{oxpos}.

Proposition 4.1 *λ_{oxpos} simulates λ_{opos}.*

Diamond. A difference between λ_{ovsc} and λ_{oxpos} is that the latter is diamond, as we now show, while the first is not (see Sect. 3). The proof uses the following basic lemma.

Lemma 4.2 (Stability under renamings) *Let t and u be λ_{oxpos} terms. If $t \to_{\mathsf{ox}_+} u$ then $t\{x{\leftarrow}y\} \to_{\mathsf{ox}_+} u\{x{\leftarrow}y\}$ for any x and y.*

Theorem 4.3 (Positive diamond, proof at p. 24) *Relation $\to_{\mathsf{ox}_+}$ is diamond.*

Postponement of GC and Local Termination. In terms of properties, λ_{oxpos} is conservative with respect to λ_{ovsc}, as postponement of GC and local termination still hold. Notation: $\to_{\mathsf{ox}_+\neg\mathsf{gc}} := \to_{\mathsf{om}_+} \cup \to_{\mathsf{oe}_+}$.

Proposition 4.4 (Local postponement of garbage collection, proof at p. 25) *Let t and u be λ_{oxpos} terms and $a \in \{\mathsf{m}_+,\mathsf{e}_+\}$. If $t \to_{\mathsf{ogc}_+} \to_{oa} u$, then $t \to_{oa} \to_{\mathsf{ogc}_+} u$.*

Proposition 4.5 (Postponement of garbage collection, proof at p. 26) *Let t and u be λ_{oxpos} terms, $d : t \to^*_{\mathsf{ox}_+} u$. Then there exist reduction sequences $e : t \to^*_{\mathsf{ox}_+\neg\mathsf{gc}} u'$ and $f : u' \to^*_{\mathsf{ogc}_+} u$ with $|e|_{\mathsf{om}_+} = |d|_{\mathsf{om}_+}$, $|e|_{\mathsf{oe}_+} = |d|_{\mathsf{oe}_+}$, and $|f| = |d|_{\mathsf{ogc}_+}$.*

Proposition 4.6 (Local termination, proof at p. 26) *Let $a \in \{\mathsf{m}_+,\mathsf{e}_+,\mathsf{gc}_+\}$. Relation $\to_{oa}$ is strongly normalizing. Moreover, $\to_{\mathsf{oe}_+} \cup \to_{\mathsf{ogc}_+}$ is strongly normalizing.*

Absence of Renaming Chains. At the end of Sect. 3, we discussed renaming chains and their dynamic creations by the looping term Ω. The following term is the representation of Ω in λ_{oxpos}, together with its evaluation, where evidently the number of $\to_{\mathsf{oe}_+}$ steps is linear in the number of $\to_{\mathsf{om}_+}$ steps:

$$\begin{aligned} & w[w{\leftarrow}(\lambda x.y[y{\leftarrow}xx])z][z{\leftarrow}\lambda x.y[y{\leftarrow}xx]] \to_{\mathsf{om}_+} w[w{\leftarrow}zz][z{\leftarrow}\lambda x.y[y{\leftarrow}xx]] \\ \to_{\mathsf{oe}_+}\ & w[w{\leftarrow}(\lambda x.y[y{\leftarrow}xx])z][z{\leftarrow}\lambda x.y[y{\leftarrow}xx]] \to_{\mathsf{om}_+} w[w{\leftarrow}zz][z{\leftarrow}\lambda x.y[y{\leftarrow}xx]] \\ \to_{\mathsf{oe}_+}\ & w[w{\leftarrow}(\lambda x.y[y{\leftarrow}xx])z][z{\leftarrow}\lambda x.y[y{\leftarrow}xx]] \cdots \end{aligned}$$

5 Dissecting λ_{ovsc}: Variable and Useful Steps

In this section, we isolate various sub-reductions of λ_{ovsc} in order to relate λ_{ovsc} and λ_{oxpos} in the next section. Essentially, some steps of λ_{ovsc} cannot be expressed in λ_{oxpos}, and some are instead *absorbed*, that is, mapped to identities rather than being simulated. We then have to partition the rewriting rules of λ_{ovsc} into sub-rules as to identify the steps that cannot be expressed, those that are absorbed, and those that are simulated by λ_{oxpos}. In particular, the partition of steps leads us to discuss useful sharing.

$$\begin{array}{rl}
\textsc{Exp. root rule for abstractions} & O\langle\!\langle x\rangle\!\rangle[x{\leftarrow}L\langle\lambda y.t\rangle] \mapsto_{\mathsf{e_{abs}}} L\langle O\langle\!\langle\lambda y.t\rangle\!\rangle[x{\leftarrow}\lambda y.t]\rangle \\
\textsc{Exp. root rule for variables} & O\langle\!\langle x\rangle\!\rangle[x{\leftarrow}L\langle y\rangle] \mapsto_{\mathsf{e_{var}}} L\langle O\langle\!\langle y\rangle\!\rangle[x{\leftarrow}y]\rangle \\
\hline
\textsc{GC root rule for abstractions} & t[x{\leftarrow}L\langle\lambda y.u\rangle] \mapsto_{\mathsf{gc_{abs}}} L\langle t\rangle \text{ if } x\notin \mathsf{fv}(t) \\
\textsc{GC root rule for variables} & t[x{\leftarrow}L\langle y\rangle] \mapsto_{\mathsf{gc_{var}}} L\langle t\rangle \text{ if } x\notin \mathsf{fv}(t) \\
\hline
\textsc{Ctx closure} & \dfrac{t\mapsto_a t'}{E\langle t\rangle \to_{\mathsf{o}a} E\langle t'\rangle}\ a\in\{\mathsf{e_{abs}},\mathsf{e_{var}},\mathsf{gc_{abs}},\mathsf{gc_{var}}\}
\end{array}$$

Fig. 5. Dissected rewriting rules for $\lambda_{\mathtt{ovsc}}$.

Variable Substitutions. In $\lambda_{\mathtt{oxpos}}$, there is no way to represent an ES $t[x{\leftarrow}y]$ containing a variable, and there is also no way of simulating a variable exponential step such as $O\langle\!\langle x\rangle\!\rangle[x{\leftarrow}y] \to_{\mathsf{e}} O\langle\!\langle y\rangle\!\rangle[x{\leftarrow}y]$ or a variable GC step $t[x{\leftarrow}y] \to_{\mathsf{e}} t$ with $x\notin\mathsf{fv}(t)$. These steps shall be *absorbed* by our translation from $\lambda_{\mathtt{ovsc}}$ to $\lambda_{\mathtt{oxpos}}$, that is, they shall be mapped to identities. To properly state this fact later on, we split now the root exponential rule $\mapsto_{\mathsf{e}}$ in two rules $\mapsto_{\mathsf{e_{abs}}}$ and $\mapsto_{\mathsf{e_{var}}}$, depending on whether the replacing value is an abstraction or a variable, and similarly for GC. The split rules are defined in Fig. 5.

Of two sub-rules $\mapsto_{\mathsf{e_{abs}}}$ and $\mapsto_{\mathsf{gc_{abs}}}$ that are not absorbed, $\mapsto_{\mathsf{gc_{abs}}}$ shall be closed by all open context and simply factored-out via the postponement of GC (Prop. 3.2). The other sub-rule $\mapsto_{\mathsf{e_{abs}}}$ is where usefulness plays a role, discussed next.

Useful Steps, First Intuitions. Another difference between the two calculi is that in $\lambda_{\mathtt{oxpos}}$ exponential steps can only be directly *useful*. Let us overview usefulness in a bit more detail than in the introduction. Replacements of variable occurrences out of ESs by abstractions amount to three *direct* cases, in the $\lambda_{\mathtt{ovsc}}$:

$$\begin{array}{rrcl}
\textsc{Directly useful} & (xt)[x{\leftarrow}\lambda y.u] & \mapsto_{\mathsf{e_{abs}}} & ((\lambda y.u)t)[x{\leftarrow}\lambda y.u] \\
\textsc{Directly non-useful 1} & (tx)[x{\leftarrow}\lambda y.u] & \mapsto_{\mathsf{e_{abs}}} & (t(\lambda y.u))[x{\leftarrow}\lambda y.u] \\
\textsc{Directly non-useful 2} & x[x{\leftarrow}\lambda y.u] & \mapsto_{\mathsf{e_{abs}}} & (\lambda y.u)[x{\leftarrow}\lambda y.u]
\end{array} \tag{5}$$

The terminology *useful* refers to the fact that the exponential step *creates* a new multiplicative redex, namely $(\lambda y.u)t$, and it is used to contrast with *non-useful*, or *useless* exponential steps (we shall prefer *non-useful* in this paper, as to denote them concisely with the letter n, given that *useful* and *useless* both start with u) that instead do not create multiplicative steps.

The reason why one considers usefulness with respect to multiplicative steps is that the number of multiplicative steps is a reasonable cost model in CbV [12,9,4], and that this fact is proved by an evaluation strategy that crucially avoids non-useful substitution steps, since non-useful substitution steps can at times add an exponential overhead, breaking the reasonability of the cost model.

In $\lambda_{\mathtt{oxpos}}$, directly useful steps can be simulated, while the two kinds of direct non-useful step cannot be expressed. The first kind because of the shape of positive applications, which can only have a variable as an argument. The second kind because abstractions cannot appear out of ESs in $\lambda_{\mathtt{oxpos}}$ (this is the relevance of the second aspect of the compactness of $\lambda_{\mathsf{pos}}/\lambda_{\mathtt{oxpos}}$ mentioned in the introduction).

There are (at least) two technical difficulties in defining the useful steps of $\lambda_{\mathtt{ovsc}}$ precisely. Before diving into them, we provide a *disclaimer*. Useful sharing is complex and takes different shapes in different settings (i.e. call-by-name/value/need). For this reason, the only two works distinguishing into useful and non-useful steps, one in call-by-name [5] and one in CbV [12], rest on slightly different definitions concerning indirectly (non-)useful steps (defined below). On purpose, later papers avoid these definitions, despite implementing useful sharing [9,4,10]. Here, we are going to define the two kinds of steps, but our definition (and terminology) shall—once more—be slightly different from [5,12]. The only thing on which all these works agree are the notions of direct useful steps, which is what $\lambda_{\mathtt{oxpos}}$ captures.

Difficulty 1: Indirect Useful Steps. In $\lambda_{\mathtt{ovsc}}$, there are also *indirect* cases to consider, given by when there is a renaming chain connecting the acting ESs and the end variable occurrence. Consider for instance

the following three indirect cases, mimicking the direct ones above via a renaming chain of length 1:

$$\begin{array}{rrcl} \textsc{Indirectly useful} & (xt)[x{\leftarrow}z][z{\leftarrow}\lambda y.u] & \mapsto_{\mathsf{e_{abs}}} & (xt)[x{\leftarrow}\lambda y.u][z{\leftarrow}\lambda y.u] \\ \textsc{Indirectly non-useful 1} & (tx)[x{\leftarrow}z][z{\leftarrow}\lambda y.u] & \mapsto_{\mathsf{e_{abs}}} & (tx)[x{\leftarrow}\lambda y.u][z{\leftarrow}\lambda y.u] \\ \textsc{Indirectly non-useful 2} & x[x{\leftarrow}z][z{\leftarrow}\lambda y.u] & \mapsto_{\mathsf{e_{abs}}} & x[x{\leftarrow}\lambda y.u][z{\leftarrow}\lambda y.u] \end{array} \tag{6}$$

The first step does not create a multiplicative redex, but it is usually considered as useful because it contributes anyway to the *future* creation of the multiplicative redex that shall happen after that step. Similarly, the other two cases are usually considered as non-useful. This indirect aspect is quite difficult to work with. In λ_{oxpos}, the indirect phenomenon disappears, because ESs such as $[x{\leftarrow}z]$, which are the cause of the indirection, do not exist. This is a considerable improvement.

It remains, however, the issue of relating indirectly useful steps in λ_{ovsc} with (directly useful) steps in λ_{oxpos}. We somewhat circumvent this issue by departing from the literature and considering indirect useful steps *as non-useful*. Therefore, *all the indirect cases above are non-useful*, in this paper. This is not cheating, as we shall explain, it is related to the core factorization theorem of the next section.

For the sake of completeness, one should also consider the replacements in which the end variable is inside an ES. The first two cases of both (5) and (6) are unaffected. The third one instead amounts to extend a chain, and then can become any of the three cases in (6).

Difficulty 2: Contextual Closure. The other difficulty is that the directly useful steps of λ_{ovsc} cannot be defined at the root level and then closed by evaluation contexts, since sometimes the useful aspect is contributed by the evaluation context itself. Consider the following root step:

$$x[x{\leftarrow}\lambda y.u] \quad \mapsto_{\mathsf{e_{abs}}} \quad (\lambda y.u)[x{\leftarrow}\lambda y.u]$$

As a root step it is not useful. But when plugged in the evaluation context $O = \langle\cdot\rangle t$, it gives rise to the following useful step, since now the steps creates a multiplicative redex in the reduct, because of the definition at a distance of these redexes:

$$x[x{\leftarrow}\lambda y.u]t \quad \mapsto_{\mathsf{e_{abs}}} \quad (\lambda y.u)[x{\leftarrow}\lambda y.u]t$$

Therefore, usefulness of a $\to_{\mathsf{oe_{abs}}}$ step depends also on the evaluation contexts surrounding the root step.

Useful Contexts and Steps. Useful exponential steps are defined via useful contexts by putting together the context used to isolate the replaced variable and the surrounding evaluation context. We also define non-useful contexts N, whose first two clauses cover the two direct cases in (5) and whose third clause cover the three indirect cases in (6).

Definition 5.1 Useful and non-useful contexts of λ_{ovsc} are defined as follows:

$$\textsc{Useful ctxs} \quad U ::= O\langle Lt\rangle \qquad\qquad \textsc{Non-useful ctxs} \quad N ::= L \mid O\langle tL\rangle \mid O\langle t[x{\leftarrow}L]\rangle$$

When taking into account the evaluation context, an $\to_{\mathsf{oe_{abs}}}$ step has the following shape:

$$O_1\langle O_2\langle\!\langle x\rangle\!\rangle[x{\leftarrow}L\langle\lambda y.t\rangle]\rangle \to_{\mathsf{oe_{abs}}} O_1\langle L\langle O_2\langle\!\langle\lambda y.t\rangle\!\rangle[x{\leftarrow}\lambda y.t]\rangle\rangle$$

Such a $\to_{\mathsf{oe_{abs}}}$ step is *useful* if $O_1\langle O_2[x{\leftarrow}L\langle\lambda y.t\rangle]\rangle$ is a useful context, and *non-useful* otherwise.

Properties of Useful Contexts. In symbols, we use predicates usef and nusef: $\mathsf{usef}(O)$ means that O is useful while $\mathsf{nusef}(O)$ means that O is non-useful. Similarly, we use the predicate sub (resp. nsub) for substitution contexts (resp. non-substitution contexts).

The following immediate lemma states how useful contexts depend on their sub-contexts, stressing that the only subtle case is the first one.

Lemma 5.2 (Useful sub-contexts)

(i) $\mathsf{usef}(Ot) \Leftrightarrow \mathsf{usef}(O) \vee \mathsf{sub}(O)$.

(ii) $\mathsf{usef}(tO) \Leftrightarrow \mathsf{usef}(O)$.

$$\text{Useful exp. rule}\quad O_1\langle O_2\langle\!\langle x\rangle\!\rangle[x{\leftarrow}L\langle\lambda y.t\rangle]\rangle \to_{\mathsf{oe_u}} O_1\langle L\langle O_2\langle\!\langle \lambda y.t\rangle\!\rangle[x{\leftarrow}\lambda y.t]\rangle\rangle \text{ if } \mathsf{usef}(O_1\langle O_2\rangle)$$
$$\text{Non-useful exp. rule}\quad O_1\langle O_2\langle\!\langle x\rangle\!\rangle[x{\leftarrow}L\langle\lambda y.t\rangle]\rangle \to_{\mathsf{oe_{nu}}} O_1\langle L\langle O_2\langle\!\langle \lambda y.t\rangle\!\rangle[x{\leftarrow}\lambda y.t]\rangle\rangle \text{ if } \mathsf{nusef}(O_1\langle O_2\rangle)$$

Fig. 6. Definition of useful and non-useful exponential variants of $\to_{\mathsf{oe_{abs}}}$, based on Lemma 5.4.1.

(iii) $\mathsf{usef}(O[x{\leftarrow}t]) \Leftrightarrow \mathsf{usef}(O)$.
(iv) $\mathsf{usef}(t[x{\leftarrow}O]) \Leftrightarrow \mathsf{usef}(O)$.

As a sanity check, we show that the definitions of useful and non-useful contexts provide a partition of open contexts. The proof is an easy induction on the open context O, the only non immediate case of which is the one for $O = O't$, as it can be guessed by Lemma 5.2.

Lemma 5.3 (Useful partitions open, proof at p. 26) *A VSC open context O is either useful or non-useful.*

The next easy lemma collects a few properties that are helpful in proofs. The first point says that in a $\to_{\mathsf{oe_{abs}}}$ step the usefulness of the context does not depend on the acting substitution $[x{\leftarrow}L\langle\lambda y.t\rangle]$ but only on the composition of the inner and outer open contexts O_1 and O_2. The definition of (non-)useful steps is then re-stated in Fig. 6.

Lemma 5.4 (Context plugging and usefulness, proof at p. 26)

(i) $\mathsf{usef}(O_1\langle L\langle O_2\rangle\rangle) \Leftrightarrow \mathsf{usef}(O_1\langle O_2\rangle)$.
(ii) $\mathsf{usef}(O_1\langle O_2\rangle) \Leftrightarrow \mathsf{usef}(O_2) \vee (\mathsf{sub}(O_2) \wedge \mathsf{usef}(O_1))$.
(iii) $\mathsf{nusef}(O_1\langle O_2\rangle) \Leftrightarrow \mathsf{nusef}(O_2) \wedge (\mathsf{sub}(O_2) \Rightarrow \mathsf{nusef}(O_1))$.

6 Core Factorization, or Postponing Non-Useful Steps

Since non-useful steps cannot be simulated by the positive calculus, the translation from $\lambda_{\mathtt{ovsc}}$ to $\lambda_{\mathtt{oxpos}}$ of the next section cannot induce a bisimulation—we need a finer approach. The idea is that any reduction sequence in $\lambda_{\mathtt{ovsc}}$ can be factored into a core part (that includes useful steps, that is, $\to_{\mathsf{oe_u}}$) and a non-useful part, and that the evaluation of a term terminates if and only if its core evaluation terminates. In this section, we prove these two facts. In the next one, we shall give a translation from $\lambda_{\mathtt{ovsc}}$ and $\lambda_{\mathtt{oxpos}}$ inducing a simulation between the core part of $\lambda_{\mathtt{ovsc}}$ and $\lambda_{\mathtt{oxpos}}$, and a termination equivalence result.

Core Reduction. The (open) core reduction $\to_{\mathtt{ocore}}$ of $\lambda_{\mathtt{ovsc}}$ is defined as $\to_{\mathtt{ocore}} := \to_{\mathsf{om}} \cup \to_{\mathsf{oe_u}} \cup \to_{\mathsf{oe_{var}}}$, and we dub *Core* $\lambda_{\mathtt{ovsc}}$ the set of VSC terms endowed with $\to_{\mathtt{ocore}}$. Beyond multiplicative and useful exponential steps, which shall be simulated by $\lambda_{\mathtt{oxpos}}$, it includes also $\to_{\mathsf{oe_{var}}}$ steps, which—on purpose—we have not been classified as useful or non-useful (in another departure from the literature) and which are going to be absorbed. As we explain next, they are crucial for our core factorization theorem.

Non-Useful Postponement. Factorization can be seen as a postponement property, in this case of non-useful steps. Similarly to how we dealt with GC, we first give a local form of postponement of $\to_{\mathsf{oe_{nu}}}$ steps. The proof of local postponement is simple but more involved than for GC steps, since it requires to check all the (many!) possible cases for a core step following a $\to_{\mathsf{oe_{nu}}}$ step, which are quite technical to list given the many contexts involved in the definition of (non-)useful rewriting steps. There is also a case of tricky local postponement diagram, where the swap postponing $\to_{\mathsf{oe_{nu}}}$ requires to do *two* core steps (this case is taken into account in Prop. 6.1.(ii) below):

$$\begin{array}{ccccc}
(xt)[x{\leftarrow}y][y{\leftarrow}\lambda z.u] & & \xrightarrow{\qquad\qquad\mathsf{oe_{nu}}\qquad\qquad} & & (xt)[x{\leftarrow}\lambda z.u][y{\leftarrow}\lambda z.u] \\
\Big\downarrow{\scriptstyle \mathsf{oe_{var}}} & & & & \Big\downarrow{\scriptstyle \mathsf{oe_u}} \\
(yt)[x{\leftarrow}y][y{\leftarrow}\lambda z.u] & \xrightarrow{\mathsf{oe_u}} & ((\lambda z.u)t)[x{\leftarrow}y][y{\leftarrow}\lambda z.u] & \xrightarrow{\mathsf{oe_{nu}}} & ((\lambda z.u)t)[x{\leftarrow}\lambda z.u][y{\leftarrow}\lambda z.u]
\end{array}$$

Translation of substitution contexts

$$[\![\langle\cdot\rangle]\!] := (\langle\cdot\rangle, \cdot) \qquad [\![L[x{\leftarrow}t]]\!] := (E'\langle E\{x{\leftarrow}y\}\rangle, \sigma\{x{\leftarrow}y\}) \quad \text{where } [\![L]\!] = (E,\sigma) \text{ and } [\![t]\!] = E'\langle y\rangle$$

Translation of terms

$$\begin{array}{rcll}
[\![x]\!] & := & x & \\
[\![\lambda x.t]\!] & := & y[y{\leftarrow}\lambda x.[\![t]\!]] & \\
[\![t[x{\leftarrow}u]]\!] & := & E\langle [\![t]\!]\{x{\leftarrow}y\}\rangle & \text{where } [\![u]\!] = E\langle y\rangle \\
[\![L\langle\lambda x.t\rangle u]\!] & := & E\langle E'\langle y[y{\leftarrow}(\lambda x.[\![t]\!]\sigma)z]\rangle\rangle & \text{where } [\![L]\!] = (E,\sigma) \text{ and } [\![u]\!] = E'\langle z\rangle \\
[\![tu]\!] & := & E\langle E'\langle y[y{\leftarrow}xz]\rangle\rangle & \text{where } [\![t]\!] = E\langle x\rangle \text{ and } [\![u]\!] = E'\langle z\rangle, \text{ if } t \text{ is not an answer}
\end{array}$$

Fig. 7. The translation from λ_{ovsc} to λ_{oxpos}.

This diagram actually justifies both our avoidance of indirect useful steps in the λ_{ovsc} and the fact that $\to_{\mathsf{oe_{var}}}$ steps are not labeled as useful/non-useful. Note that the solid lines are exactly what would usually be an indirectly useful step followed by a directly useful one, and that for us they are instead a non-useful step followed by a (directly) useful one. The point is that the sequence can always be re-arranged as shown by the diagram, using $\to_{\mathsf{oe_{var}}}$ as to turn the replacement on $[x{\leftarrow}y]$ into a directly non-useful one.

Proposition 6.1 (Local postponement of $\to_{\mathsf{oe_{nu}}}$, proof at p. 27) *Let t and u be* VSC *terms. If $t \to_{\mathsf{oe_{nu}}}\to_{\mathsf{ocore}} u$ then $t \to_{\mathsf{ocore}}\to_{\mathsf{oe_{nu}}} u$ or $t \to_{\mathsf{ocore}}\to_{\mathsf{ocore}}\to_{\mathsf{oe_{nu}}} u$. More precisely:*

(i) $\to_{\mathsf{oe_{nu}}}\to_{\mathsf{om}} \subseteq \to_{\mathsf{om}}\to_{\mathsf{oe_{nu}}}$;
(ii) $\to_{\mathsf{oe_{nu}}}\to_{\mathsf{oe_u}} \subseteq \to_{\mathsf{oe_u}}\to_{\mathsf{oe_{nu}}} \cup \to_{\mathsf{oe_{var}}}\to_{\mathsf{oe_u}}\to_{\mathsf{oe_{nu}}}$;
(iii) $\to_{\mathsf{oe_{nu}}}\to_{\mathsf{oe_{var}}} \subseteq \to_{\mathsf{oe_{var}}}\to_{\mathsf{oe_{nu}}}$.

Obtaining the global postponement property from the local one is easy because, although the number of core steps can grow with local swaps, the number of non-useful steps is preserved, and can be easily exploited for the induction lifting the local diagrams to the global property.

Theorem 6.2 (Core Factorization / Postponement of non-useful steps, proof at p. 32) *Let t and u be* VSC *terms. If $d : t \to^*_{\mathsf{o\neg gc}} u$, then $e : t \to^*_{\mathsf{ocore}}\to^*_{\mathsf{oe_{nu}}} u$ with $|e|_{\mathsf{om}} = |d|_{\mathsf{om}}$.*

Lastly, we use the postponement property to prove that the core sub-system of λ_{ovsc} is termination-equivalent to the whole of λ_{ovsc}, justifying the core terminology.

Theorem 6.3 (Termination equivalence of λ_{ovsc} and Core λ_{ovsc} , proof at p. 33)

(i) *t has a diverging $\to_{\mathsf{ovsc}}$ sequence if and only if t has a diverging $\to_{\mathsf{ocore}}$ sequence;*
(ii) *t is $\to_{\mathsf{ovsc}}$-weakly normalizing if and only if t is $\to_{\mathsf{ocore}}$-weakly normalizing.*

7 Translating λ_{ovsc} to λ_{oxpos} and Simulating Core Steps

In this section, we define a translation $[\![\cdot]\!]$ from λ_{ovsc} to λ_{oxpos} and show that it induces a simulation of the core reduction $\to_{\mathsf{ocore}}$ of λ_{ovsc} by λ_{oxpos}. There are various delicate points, concerning both the definition and the simulation, discussed all along this section.

Subtlety 1: Absorption of Variables. Since ESs in λ_{ovsc} can contain variables (as e.g. in $t[x{\leftarrow}y]$) but ESs in λ_{oxpos} cannot, the translation $[\![\cdot]\!]$ that we shall define turns these ESs of λ_{ovsc} into meta-level substitutions of λ_{oxpos}. For instance, we shall have $[\![t[x{\leftarrow}y]]\!] = [\![t]\!]\{x{\leftarrow}y\}$.

Subtlety 2: Answers. It is natural to define $[\![\cdot]\!]$ as introducing sharing points for every non-variable sub-term (as in Accattoli et al. [4]), which gives the following definition (where the meta-level substitution

$\{x{\leftarrow}y\}$ in the last case is due to the absorption of variables):

$$\begin{array}{rcl|rcll} [\![x]\!] & := & x & [\![tu]\!] & := & E\langle E'\langle x\rangle[x{\leftarrow}yz]\rangle & \text{where } [\![t]\!] := E\langle y\rangle \text{ and } [\![u]\!] = E'\langle z\rangle \\ [\![\lambda x.t]\!] & := & y[y{\leftarrow}\lambda x.[\![t]\!]] & [\![t[x{\leftarrow}u]]\!] & := & E\langle [\![t]\!]\{x{\leftarrow}y\}\rangle & \text{where } [\![u]\!] = E\langle y\rangle \end{array}$$

Unfortunately, such a definition does not induce a simulation. For instance, consider the following $\mathsf{e_u}$-step:

$$t := (xx)[x{\leftarrow}\lambda y.u] \to_{\mathsf{oe_u}} ((\lambda y.u)x)[x{\leftarrow}\lambda y.u] =: t'$$

Using the above definition of $[\![\cdot]\!]$, one would need to have:

$$[\![t]\!] = z[z{\leftarrow}xx][x{\leftarrow}\lambda y.[\![u]\!]] \to^*_{\mathsf{ox}_+} z[z{\leftarrow}wx][w{\leftarrow}\lambda y.[\![u]\!]][x{\leftarrow}\lambda y.[\![u]\!]] = [\![t']\!]$$

Note, however, that such a *duplication of ESs* cannot be performed in λ_{oxpos}. Therefore, we rather adopt a modified translation that behaves differently on applied abstractions—more precisely, on applied *answers* (answers are defined in Fig. 1, page 5). The new translation is defined in Fig. 7. In fact, the translation $[\![t]\!]$ of terms is defined by mutual induction with the translation $[\![L]\!]$ of substitution contexts, which is used in the case of applied answers. Because of the absorption of variables, the translation $[\![L]\!]$ of substitution contexts L is not simply an evaluation context of λ_{oxpos} but a *pair* of an evaluation context E *and* a renaming σ, that is, a meta-level substitution of variables for variables. For instance, $[\![\langle\cdot\rangle[x{\leftarrow}\lambda y.t][w{\leftarrow}z]]\!] = (E,\sigma)$ with $E := \langle\cdot\rangle[x{\leftarrow}\lambda y.[\![t]\!]]$ and $\sigma := \{w{\leftarrow}z\}$.

The next lemma shows that such a translation of substitution contexts is compositional. It is proved by a straightforward induction on L.

Lemma 7.1 *Let $L\langle t\rangle$ be a* VSC *term and $[\![L]\!] = (E,\sigma)$. Then $[\![L\langle t\rangle]\!] = E\langle [\![t]\!]\sigma\rangle$.*

Simulation. Core reduction $\to_{\mathsf{ocore}}$ is made out of three kinds of steps, namely $\to_{\mathsf{om}}$, $\to_{\mathsf{oe_u}}$, and $\to_{\mathsf{oe_{var}}}$. Given the special role of answers in the definition of $[\![\cdot]\!]$, the proof of the simulation becomes tricky when core steps can turn an applied non-answer into an applied answer. This can happen with $\to_{\mathsf{om}}$ and $\to_{\mathsf{oe_u}}$ steps, which are then discussed in detail in the next paragraphs. Rule $\to_{\mathsf{oe_{var}}}$, instead, does not alter whether sub-terms are answers, and so the proof that $\to_{\mathsf{oe_{var}}}$ steps are absorbed is smooth.

Lemma 7.2 (Absorption of variable exponentials, proof at p. 34) *Let t and u be* VSC *terms. If $t \to_{\mathsf{oe_{var}}} u$ then $[\![t]\!] = [\![u]\!]$.*

Subtlety 3: Simulation of Multiplicative Steps. Root multiplicative steps are simulated smoothly.

Lemma 7.3 (Simulation of root multiplicative steps, proof at p. 34) *Let t and u be* VSC *terms. If $t \mapsto_{\mathsf{m}} u$ then $[\![t]\!] \to_{\mathsf{om}_+} [\![u]\!]$.*

A complication arises for the contextual closure of multiplicative steps, because in a root step $L\langle\lambda x.t\rangle u \mapsto_{\mathsf{m}} L\langle t[x{\leftarrow}u]\rangle$ the redex is not an answer but the reduct might be one, if t is an abstraction. Thus, if the root step is applied to a further argument r, the reduction turns an applied non-answer into an applied answer, changing the clause of the translation that is used for the application to r. This phenomenon is handled by doing two additional rewriting steps in λ_{oxpos}. The simplest case is the following one, where $t = y$, $u = z$, $r = w$, and $L = \langle\cdot\rangle$ and x', y', z' are variables introduced by the translation:

$$\begin{array}{ccc} (\lambda x.\lambda y.y)zw & \xrightarrow{\quad\mathsf{om}\quad} & (\lambda y.y)[x{\leftarrow}z]w \\ {\scriptstyle[\![\cdot]\!]}\downarrow & & \downarrow{\scriptstyle[\![\cdot]\!]} \\ x'[x'{\leftarrow}y'w][y'{\leftarrow}(\lambda x.z'[z'{\leftarrow}\lambda y.y])z] & \xrightarrow{\mathsf{om}_+} x'[x'{\leftarrow}z'w][z'{\leftarrow}\lambda y.y] \xrightarrow{\mathsf{oe}_+} x'[x'{\leftarrow}(\lambda y.y)w][z'{\leftarrow}\lambda y.y] & \xrightarrow{\mathsf{ogc}_+} x'[x'{\leftarrow}(\lambda y.y)w] \end{array}$$

In general, we have the following simulation of multiplicative steps, where the first case isolates exactly when applied non-answers are turned into applied answers.

Proposition 7.4 (Simulation of $\to_{\mathsf{om}}$ steps, proof at p. 35) *Let t and u be* VSC *terms and $t \mapsto_{\mathsf{m}} u$.*

(i) *If u is an answer and $\mathsf{usef}(O)$ then $[\![O\langle t\rangle]\!] \to_{\mathsf{om}_+}\to_{\mathsf{oe}_+}\to_{\mathsf{ogc}_+} [\![O\langle u\rangle]\!]$;*

$$\textsc{Useful exp. root rule 1} \quad U\langle\!\langle x\rangle\!\rangle[x{\leftarrow}L\langle\lambda y.t\rangle] \mapsto_{\mathsf{e_{u_1}}} L\langle U\langle\!\langle \lambda y.t\rangle\!\rangle[x{\leftarrow}\lambda y.t]\rangle$$

$$\textsc{Useful exp. root rule 2} \quad L_1\langle L_2\langle\!\langle x\rangle\!\rangle[x{\leftarrow}L_3\langle\lambda y.t\rangle]\rangle u \mapsto_{\mathsf{e_{u_2}}} L_1\langle L_3\langle L_2\langle\!\langle \lambda y.t\rangle\!\rangle[x{\leftarrow}\lambda y.t]\rangle\rangle u$$

$$\textsc{Ctx closure} \quad \frac{t \mapsto_a t'}{O\langle t\rangle \to_{\mathsf{o}a} O\langle t'\rangle} \; a{\in}\{\mathsf{e_{u_1}}, \mathsf{e_{u_2}}\}$$

Fig. 8. Two root rules for $\to_{\mathsf{oe_u}}$.

(ii) *Otherwise,* $[\![O\langle t\rangle]\!] \to_{\mathsf{om}_+} [\![O\langle u\rangle]\!]$.

Subtlety 4: Simulation of Useful Exponential Steps. Exponential steps can turn applied non-answers into applied answers too: these cases actually are the very definition of useful exponential steps. In contrast to the multiplicative case, the simulation does not need extra steps, it simply maps one $\to_{\mathsf{oe_u}}$ step in λ_{ovsc} to one $\to_{\mathsf{oe}_+}$ step in λ_{oxpos}. What is tricky in this case is that the definition of useful steps needs the surrounding open context of the root step, as explained in Sect. 5, so it does not seem possible to prove the simulation for a root case and then extending the property with an induction on the context closure.

To get around this issue, in Fig. 8 we give an alternative definition of $\to_{\mathsf{oe_u}}$ resting on *two* root rules, where the second rule captures the cases when the argument of the created redex is provided by the context. With these two root rules, useful exponential steps can then be defined via a closure by *any* open context, thus bypassing the global aspect of the definition that we gave in Sect. 5. The reader probably wonders why we did not do it already in Sect. 5. The reason is that the global definition is preferable for proving local postponement (Prop. 6.1). The alternative definition is justified by the following lemma.

Lemma 7.5 (Alternative presentation of useful steps, proof at p. 37) $\to_{\mathsf{oe_u}} = \to_{\mathsf{oe_{u_1}}} \cup \to_{\mathsf{oe_{u_2}}}$.

The simulation is then proved smoothly via the alternative definition.

Proposition 7.6 (Simulation of useful exponential steps, proof at p. 38) *Let t and u be* VSC *terms. If $t \to_{\mathsf{oe_{u_1}}} u$ or $t \to_{\mathsf{oe_{u_2}}} u$ then $[\![t]\!] \to_{\mathsf{oe}_+} [\![u]\!]$.*

Summing Up. We can now put together the simulations of single core steps, and also iterate over reduction sequences. Since at times one source multiplicative step is simulated by more than one target step, the simulation does not preserve evaluation lengths. An important point to note, however, is that the number of multiplicative steps—which is the cost model of λ_{ovsc}—is preserved. More generally, the increment in length is only linear.

Theorem 7.7 (Simulation of core sequences) *Let $d : t \to^*_{\mathsf{ocore}} t'$ be a reduction sequence in λ_{ovsc}. Then there exists $e : [\![t]\!] \to^*_{\mathsf{ox}_+} [\![t']\!]$ in λ_{oxpos} such that $|e|_{\mathsf{om}_+} = |d|_{\mathsf{om}}$ and $|d|_{\mathsf{om,oe_u}} \leq |e| \leq 3 \cdot |d|$.*

8 Core Normal Forms and Termination Equivalence

Lastly, we show that the translation $[\![\cdot]\!]$ preserves and reflects termination. Reflection is a consequence of the simulation theorem: if $[\![t]\!]$ terminates then t cannot diverge, because $[\![t]\!]$ can simulate it. Preservation instead is proved by showing that $\to_{\mathsf{ocore}}$ normal forms are mapped to (non-erasing) positive normal forms, *i.e.*, $\to_{\mathsf{ox}_+\neg\mathsf{gc}}$-normal forms, which is proved via a (technical) characterization of core normal forms.

Characterization of Core Normal Forms. For characterizing core normal forms, we need a few auxiliary definitions. We start by defining two sets of variables for terms.

Definition 8.1 The set $\mathtt{ofv}(t)$ of *open free variables* of a VSC term t is the set of variables of t having occurrences out of all abstractions, formally defined as follows:

$$\mathtt{ofv}(x) := \{x\} \qquad \mathtt{ofv}(tu) := \mathtt{ofv}(t) \cup \mathtt{ofv}(u)$$
$$\mathtt{ofv}(\lambda x.t) := \emptyset \qquad \mathtt{ofv}(t[x{\leftarrow}u]) := (\mathtt{ofv}(t) \setminus \{x\}) \cup \mathtt{ofv}(u)$$

The set $\mathtt{aofv}(t)$ of *applied open free variables* of a VSC term t is the set of variables of t having applied occurrences out of all abstractions, formally defined as follows:

$$\begin{array}{rl|rll} \mathtt{aofv}(x) = \mathtt{aofv}(\lambda x.t) & \coloneqq \emptyset & \mathtt{aofv}(tu) & \coloneqq \mathtt{aofv}(t) \cup \mathtt{aofv}(u) \cup \{x\} & \text{if } t = L\langle\!\langle x \rangle\!\rangle \\ \mathtt{aofv}(t[x{\leftarrow}u]) & \coloneqq (\mathtt{aofv}(t) \setminus \{x\}) \cup \mathtt{aofv}(u) & \mathtt{aofv}(tu) & \coloneqq \mathtt{aofv}(t) \cup \mathtt{aofv}(u) & \text{otherwise} \end{array}$$

We also need a weakened notion of answer.

Definition 8.2 An *almost answer* is an answer or a VSC term of the form $L\langle L'\langle x\rangle[x{\leftarrow}t]\rangle$ where t is an answer.

Finally, we can provide a characterization of core normal forms, based on the following grammar.

Grammar of core normal terms

$$\begin{array}{rll} n = v & & \\ \quad \mid nn' & \text{with } n \text{ not an almost answer} & \\ \quad \mid n[x{\leftarrow}n'] & \text{with } n' = L\langle\lambda y.t\rangle \text{ and } x \notin \mathtt{aofv}(n) & \\ \quad \mid n[x{\leftarrow}n'] & \text{with } n' = L\langle y\rangle \text{ and } x \notin \mathtt{ofv}(n) & \\ \quad \mid n[x{\leftarrow}n'] & \text{with } n' = L\langle tu\rangle & \end{array}$$

Proposition 8.3 (Characterization of core normal forms, proof at p. 39) *Let t be a VSC term. t is $\to_{\mathtt{ocore}}$-normal if and only if it is a n term.*

Via a few technical lemmas in Appendix G (p. 39), we obtain the following preservation property.

Proposition 8.4 (Preservation of core normal forms, proof at p. 41) *Let t be a* VSC *term. If t is $\to_{\mathtt{ocore}}$-normal then $[\![t]\!]$ is $\to_{\mathtt{ox}_+\neg\mathtt{gc}}$-normal.*

Theorem 8.5 (Termination equivalence of Core $\lambda_{\mathtt{ovsc}}$ and $\lambda_{\mathtt{oxpos}}$, proof at p. 41) *Let t be a* VSC *term.*

(i) *t has a diverging $\to_{\mathtt{ocore}}$ sequence if and only if $[\![t]\!]$ has a diverging $\to_{\mathtt{ox}_+}$ sequence.*

(ii) *t is $\to_{\mathtt{ocore}}$-weakly normalizing if and only if $[\![t]\!]$ is $\to_{\mathtt{ox}_+}$-weakly normalizing.*

As a corollary, we can prove *uniform normalization* (see Sect. 2) for $\lambda_{\mathtt{ovsc}}$ and Core $\lambda_{\mathtt{ovsc}}$. Uniform normalization follows immediately from the diamond property, thus it holds for $\lambda_{\mathtt{oxpos}}$ (Thm. 4.3). Concerning $\lambda_{\mathtt{ovsc}}$ and Core $\lambda_{\mathtt{ovsc}}$, the proof is instead not immediate, because they are not diamond (see the diagram at page 6). But we can obtaining it by lifting the one of $\lambda_{\mathtt{oxpos}}$ via the proved termination equivalences.

Corollary 8.6 (Proof at p. 42) *$\lambda_{\mathtt{ovsc}}$ and Core $\lambda_{\mathtt{ovsc}}$ are uniformly normalizing.*

9 Conclusions

This paper studies Wu's positive λ-calculus $\lambda_{\mathtt{pos}}$, which at first sight looks simply as yet another call-by-value λ-calculus with sharing. It has, however, a new feature that distinguishes it among similar calculi, called here *compactness* and concerning the treatment of variables.

Our main contribution is showing that compactness allows one to elegantly capture the essence of useful sharing (in an open setting), circumventing the main technicalities of this notion. What is remarkable is that $\lambda_{\mathtt{pos}}$ has not arisen as an incremental refinement of useful sharing, but as an outcome of the completely unrelated study of focalization for minimal intuitionistic logic by Miller and Wu [28].

We believe that the positive λ-calculus is a sharp tool deserving to be studied further, in particular with respect to program transformations and optimizations, and also endowed with call-by-need evaluation.

References

[1] Accattoli, B., *Proof nets and the call-by-value λ-calculus*, Theor. Comput. Sci. **606**, pages 2–24 (2015). https://doi.org/10.1016/j.tcs.2015.08.006

[2] Accattoli, B., *Exponentials as substitutions and the cost of cut elimination in linear logic*, Log. Methods Comput. Sci. **19** (2023). https://doi.org/10.46298/LMCS-19(4:23)2023

[3] Accattoli, B., A. Condoluci, G. Guerrieri and C. Sacerdoti Coen, *Crumbling abstract machines*, in: E. Komendantskaya, editor, *Proceedings of the 21st International Symposium on Principles and Practice of Programming Languages, PPDP 2019, Porto, Portugal, October 7-9, 2019*, pages 4:1–4:15, ACM (2019). https://doi.org/10.1145/3354166.3354169

[4] Accattoli, B., A. Condoluci and C. Sacerdoti Coen, *Strong call-by-value is reasonable, implosively*, in: *36th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2021, Rome, Italy, June 29 - July 2, 2021*, pages 1–14, IEEE (2021). https://doi.org/10.1109/LICS52264.2021.9470630

[5] Accattoli, B. and U. Dal Lago, *(leftmost-outermost) beta reduction is invariant, indeed*, Log. Methods Comput. Sci. **12** (2016). https://doi.org/10.2168/LMCS-12(1:4)2016

[6] Accattoli, B., U. Dal Lago and G. Vanoni, *Reasonable space for the λ-calculus, logarithmically*, in: C. Baier and D. Fisman, editors, *LICS '22: 37th Annual ACM/IEEE Symposium on Logic in Computer Science, Haifa, Israel, August 2 - 5, 2022*, pages 47:1–47:13, ACM (2022). https://doi.org/10.1145/3531130.3533362

[7] Accattoli, B., C. Faggian and A. Lancelot, *Normal form bisimulations by value*, CoRR **abs/2303.08161** (2023). 2303.08161. https://doi.org/10.48550/arXiv.2303.08161

[8] Accattoli, B. and G. Guerrieri, *Open call-by-value*, in: A. Igarashi, editor, *Programming Languages and Systems - 14th Asian Symposium, APLAS 2016, Hanoi, Vietnam, November 21-23, 2016, Proceedings*, volume 10017 of *Lecture Notes in Computer Science*, pages 206–226 (2016). https://doi.org/10.1007/978-3-319-47958-3_12

[9] Accattoli, B. and G. Guerrieri, *Abstract machines for open call-by-value*, Sci. Comput. Program. **184** (2019). https://doi.org/10.1016/J.SCICO.2019.03.002

[10] Accattoli, B. and M. Leberle, *Useful open call-by-need*, in: F. Manea and A. Simpson, editors, *30th EACSL Annual Conference on Computer Science Logic, CSL 2022, February 14-19, 2022, Göttingen, Germany (Virtual Conference)*, volume 216 of *LIPIcs*, pages 4:1–4:21, Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2022). https://doi.org/10.4230/LIPICS.CSL.2022.4

[11] Accattoli, B. and L. Paolini, *Call-by-value solvability, revisited*, in: T. Schrijvers and P. Thiemann, editors, *Functional and Logic Programming - 11th International Symposium, FLOPS 2012, Kobe, Japan, May 23-25, 2012. Proceedings*, volume 7294 of *Lecture Notes in Computer Science*, pages 4–16, Springer (2012). https://doi.org/10.1007/978-3-642-29822-6_4

[12] Accattoli, B. and C. Sacerdoti Coen, *On the relative usefulness of fireballs*, in: *30th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2015, Kyoto, Japan, July 6-10, 2015*, pages 141–155, IEEE Computer Society (2015). https://doi.org/10.1109/LICS.2015.23

[13] Accattoli, B. and C. Sacerdoti Coen, *On the value of variables*, Information and Computation **255**, pages 224–242 (2017). https://doi.org/10.1016/j.ic.2017.01.003

[14] Andreoli, J., *Logic programming with focusing proofs in linear logic*, J. Log. Comput. **2**, pages 297–347 (1992). https://doi.org/10.1093/LOGCOM/2.3.297

[15] Brock-Nannestad, T., N. Guenot and D. Gustafsson, *Computation in focused intuitionistic logic*, in: M. Falaschi and E. Albert, editors, *Proceedings of the 17th International Symposium on Principles and Practice of Declarative Programming, Siena, Italy, July 14-16, 2015*, pages 43–54, ACM (2015). https://doi.org/10.1145/2790449.2790528

[16] Chaudhuri, K., *Focusing strategies in the sequent calculus of synthetic connectives*, in: I. Cervesato, H. Veith and A. Voronkov, editors, *Logic for Programming, Artificial Intelligence, and Reasoning, 15th International Conference, LPAR 2008, Doha, Qatar, November 22-27, 2008. Proceedings*, volume 5330 of *Lecture Notes in Computer Science*, pages 467–481, Springer (2008). https://doi.org/10.1007/978-3-540-89439-1_33

[17] Chaudhuri, K., S. Hetzl and D. Miller, *A multi-focused proof system isomorphic to expansion proofs*, J. Log. Comput. **26**, pages 577–603 (2016).
https://doi.org/10.1093/LOGCOM/EXU030

[18] Chaudhuri, K., D. Miller and A. Saurin, *Canonical sequent proofs via multi-focusing*, in: G. Ausiello, J. Karhumäki, G. Mauri and C. L. Ong, editors, *Fifth IFIP International Conference On Theoretical Computer Science - TCS 2008, IFIP 20th World Computer Congress, TC 1, Foundations of Computer Science, September 7-10, 2008, Milano, Italy*, volume 273 of *IFIP*, pages 383–396, Springer (2008).
https://doi.org/10.1007/978-0-387-09680-3_26

[19] Curien, P. and H. Herbelin, *The duality of computation*, in: M. Odersky and P. Wadler, editors, *Proceedings of the Fifth ACM SIGPLAN International Conference on Functional Programming (ICFP '00), Montreal, Canada, September 18-21, 2000*, pages 233–243, ACM (2000).
https://doi.org/10.1145/351240.351262

[20] Dal Lago, U. and S. Martini, *The weak lambda calculus as a reasonable machine*, Theor. Comput. Sci. **398**, pages 32–50 (2008).
https://doi.org/10.1016/J.TCS.2008.01.044

[21] Economou, D. J., N. Krishnaswami and J. Dunfield, *Focusing on refinement typing*, ACM Trans. Program. Lang. Syst. **45**, pages 22:1–22:62 (2023).
https://doi.org/10.1145/3610408

[22] Flanagan, C., A. Sabry, B. F. Duba and M. Felleisen, *The essence of compiling with continuations*, in: R. Cartwright, editor, *Proceedings of the ACM SIGPLAN'93 Conference on Programming Language Design and Implementation (PLDI), Albuquerque, New Mexico, USA, June 23-25, 1993*, pages 237–247, ACM (1993).
https://doi.org/10.1145/155090.155113

[23] Friedman, D. P., A. Ghuloum, J. G. Siek and O. L. Winebarger, *Improving the lazy krivine machine*, High. Order Symb. Comput. **20**, pages 271–293 (2007).
https://doi.org/10.1007/S10990-007-9014-0

[24] Krishnaswami, N. R., *Focusing on pattern matching*, in: Z. Shao and B. C. Pierce, editors, *Proceedings of the 36th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL 2009, Savannah, GA, USA, January 21-23, 2009*, pages 366–378, ACM (2009).
https://doi.org/10.1145/1480881.1480927

[25] Levy, P. B., J. Power and H. Thielecke, *Modelling environments in call-by-value programming languages*, Inf. Comput. **185**, pages 182–210 (2003).
https://doi.org/10.1016/S0890-5401(03)00088-9

[26] Liang, C. C. and D. Miller, *Focusing and polarization in linear, intuitionistic, and classical logics*, Theor. Comput. Sci. **410**, pages 4747–4768 (2009).
https://doi.org/10.1016/J.TCS.2009.07.041

[27] Marin, S., D. Miller, E. Pimentel and M. Volpe, *From axioms to synthetic inference rules via focusing*, Ann. Pure Appl. Log. **173**, page 103091 (2022).
https://doi.org/10.1016/J.APAL.2022.103091

[28] Miller, D. and J. Wu, *A positive perspective on term representation (invited talk)*, in: B. Klin and E. Pimentel, editors, *31st EACSL Annual Conference on Computer Science Logic, CSL 2023, February 13-16, 2023, Warsaw, Poland*, volume 252 of *LIPIcs*, pages 3:1–3:21, Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2023).
https://doi.org/10.4230/LIPICS.CSL.2023.3

[29] Moggi, E., *Computational λ-Calculus and Monads*, LFCS report ECS-LFCS-88-66, University of Edinburgh (1988).
http://www.lfcs.inf.ed.ac.uk/reports/88/ECS-LFCS-88-66/ECS-LFCS-88-66.pdf

[30] Moggi, E., *Computational lambda-calculus and monads*, in: *Proceedings of the Fourth Annual Symposium on Logic in Computer Science (LICS '89), Pacific Grove, California, USA, June 5-8, 1989*, pages 14–23, IEEE Computer Society (1989).
https://doi.org/10.1109/LICS.1989.39155

[31] Rioux, N. and S. Zdancewic, *Computation focusing*, Proc. ACM Program. Lang. (ICFP) **4**, pages 95:1–95:27 (2020).
https://doi.org/10.1145/3408977

[32] Sabry, A. and M. Felleisen, *Reasoning about programs in continuation-passing style*, in: J. L. White, editor, *Proceedings of the Conference on Lisp and Functional Programming, LFP 1992, San Francisco, California, USA, 22-24 June 1992*, pages 288–298, ACM (1992).
https://doi.org/10.1145/141471.141563

[33] Sabry, A. and P. Wadler, *A Reflection on Call-by-Value*, ACM Trans. Program. Lang. Syst. **19**, pages 916–941 (1997). https://doi.org/10.1145/267959.269968

[34] Sands, D., J. Gustavsson and A. Moran, *Lambda Calculi and Linear Speedups*, in: *The Essence of Computation, Complexity, Analysis, Transformation. Essays Dedicated to Neil D. Jones*, pages 60–84 (2002). https://doi.org/10.1007/3-540-36377-7_4

[35] Scherer, G., *Deciding equivalence with sums and the empty type*, in: G. Castagna and A. D. Gordon, editors, *Proceedings of the 44th ACM SIGPLAN Symposium on Principles of Programming Languages, POPL 2017, Paris, France, January 18-20, 2017*, pages 374–386, ACM (2017). https://doi.org/10.1145/3009837.3009901

[36] Scherer, G. and D. Rémy, *Which simple types have a unique inhabitant?*, in: K. Fisher and J. H. Reppy, editors, *Proceedings of the 20th ACM SIGPLAN International Conference on Functional Programming, ICFP 2015, Vancouver, BC, Canada, September 1-3, 2015*, pages 243–255, ACM (2015). https://doi.org/10.1145/2784731.2784757

[37] Sestoft, P., *Deriving a lazy abstract machine*, J. Funct. Program. **7**, pages 231–264 (1997). https://doi.org/10.1017/S0956796897002712

[38] Walker, D., *Substructural Type Systems*, in: *Advanced Topics in Types and Programming Languages*, The MIT Press (2004), ISBN 9780262281591. https://direct.mit.edu/book/chapter-pdf/186357/9780262281591_caa.pdf. https://doi.org/10.7551/mitpress/1104.003.0003

[39] Wand, M., *On the correctness of the krivine machine*, High. Order Symb. Comput. **20**, pages 231–235 (2007). https://doi.org/10.1007/S10990-007-9019-8

[40] Wu, J., *Proofs as terms, terms as graphs*, in: C. Hur, editor, *Programming Languages and Systems - 21st Asian Symposium, APLAS 2023, Taipei, Taiwan, November 26-29, 2023, Proceedings*, volume 14405 of *Lecture Notes in Computer Science*, pages 91–111, Springer (2023). https://doi.org/10.1007/978-981-99-8311-7_5

[41] Zeilberger, N., *Focusing and higher-order abstract syntax*, in: G. C. Necula and P. Wadler, editors, *Proceedings of the 35th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL 2008, San Francisco, California, USA, January 7-12, 2008*, pages 359–369, ACM (2008). https://doi.org/10.1145/1328438.1328482

A Double Contexts and A Lemma

This section develops definitions and technical tools that are used in the proofs of the following sections to manage contexts, namely the outside-in order on contexts, contexts with *two* holes (called *double contexts*), and a lemma about context predicates.

Definition A.1 [Outside-in context order, disjoint contexts] We define the partial *outside-in* order $\prec$ over contexts as follows:

$$\frac{}{\langle\cdot\rangle \prec O} \qquad \frac{O \prec O'}{O''\langle O\rangle \prec O''\langle O'\rangle}$$

And say that O is *outer* than O' if $O \prec O'$. If $O \not\prec O'$ and $O' \not\prec O$ we say that O and O' are *disjoint*, and write $O \parallel O'$.

Double Contexts. Double contexts shall be used to compare two contexts on the same term. They have as base cases binary constructors (that is, applications and ESs) having contexts replacing their sub-terms, and as inductive cases they are simply closed by an ordinary context.

Definition A.2 [Double contexts] Double contexts C are defined by the following grammar.

$$\text{Double contexts} \qquad \mathbb{O} ::= OO' \mid O[x{\leftarrow}O'] \mid O\langle\mathbb{O}\rangle$$

Some easy facts about double contexts.

- *Plugging*: the plugging $\mathbb{O}\langle t, u\rangle$ of two terms t and u into a double context $\mathbb{O}$ is defined as expected and gives a term. The two ways of plugging one term $\mathbb{O}\langle t, \langle\cdot\rangle\rangle$ and $\mathbb{O}\langle\langle\cdot\rangle, u\rangle$ into a double context give instead a context.
- *Pairs of disjoint positions and double contexts*: every pair of positions $O\langle t\rangle = O'\langle u\rangle$ which are disjoint, that is, such that $O \parallel O'$, gives rise to a double context $\mathbb{O}_O O'$ such that $\mathbb{O}_{O,O'}\langle\cdot, u\rangle = O$ and $\mathbb{O}_{O,O'}\langle t, \cdot\rangle = O'$.

Lemma A.3 *Let $\mathbb{O}$ be a double context, and t and t' be two terms. Let $O = \mathbb{O}\langle\langle\cdot\rangle, t\rangle$ (resp. $\mathbb{O}\langle t, \langle\cdot\rangle\rangle$) and $O' = \mathbb{O}\langle\langle\cdot\rangle, t'\rangle$ (resp. $\mathbb{O}\langle t', \langle\cdot\rangle\rangle$). Then:*

$$\forall \mathsf{pred} \in \{\mathsf{sub}, \mathsf{usef}, \mathsf{nusef}\}, \mathsf{pred}(O) \Leftrightarrow \mathsf{pred}(O')$$

Proof. Straightforward by induction on $\mathbb{O}$. □

B Proofs of Sect. 3 (The Open (Micro-Step) Value Substitution Calculus)

Proposition 3.1 (Local postponement of garbage collection, originally at p. 6) *For $a \in \{\mathsf{m}, \mathsf{e}\}$, If $t \to_{\mathsf{ogc}} \to_{oa} u$, then $t \to_{oa} \to_{\mathsf{ogc}} u$.*

Proof. Assume $t = O\langle t'[x{\leftarrow}L\langle v\rangle]\rangle \to_{\mathsf{ogc}} O\langle L\langle t'\rangle\rangle = r \to_{oa} u$.

(i) $r = O_1\langle L'\langle\lambda y.q\rangle p\rangle \to_{\mathsf{om}} O_1\langle L'\langle q[y{\leftarrow}p]\rangle\rangle = u$. The situation is then as follows:

$$\begin{array}{ccccc} t = O\langle t'[x{\leftarrow}L\langle v\rangle]\rangle & \xrightarrow{\mathsf{ogc}} & O\langle L\langle t'\rangle\rangle = r = O_1\langle L'\langle\lambda y.q\rangle p\rangle & & \\ & & \downarrow_{\mathsf{om}} & & \\ & & O_1\langle L'\langle q[y{\leftarrow}p]\rangle\rangle = u & & \end{array}$$

Cases of the positioning of O_1 with respect to the other shape of r, namely $O\langle L\langle t'\rangle\rangle$:

- $O_1 \parallel O$. Then there exists a double context $\mathbb{O}$ such that $\mathbb{O}\langle\langle\cdot\rangle, L'\langle\lambda y.q\rangle p\rangle = O$ and $\mathbb{O}\langle L\langle t'\rangle, \langle\cdot\rangle\rangle = O_1$. Then:

$$\begin{array}{ccc} t = \mathbb{O}\langle t'[x{\leftarrow}L\langle v\rangle], L'\langle\lambda y.q\rangle p\rangle & \xrightarrow{\mathsf{ogc}} & \mathbb{O}\langle L\langle t'\rangle, L'\langle\lambda y.q\rangle p\rangle = r \\ \Big\downarrow{\scriptstyle \mathsf{om}} & & \Big\downarrow{\scriptstyle \mathsf{om}} \\ \mathbb{O}\langle t'[x{\leftarrow}L\langle v\rangle], L'\langle q[y{\leftarrow}p]\rangle\rangle & \xrightarrow{\mathsf{ogc}} & \mathbb{O}\langle L\langle t'\rangle, L'\langle q[y{\leftarrow}p]\rangle\rangle = u \end{array}$$

- $O \prec O_1$. That is, $O_1 = O\langle O_2\rangle$ for some O_2. We have then $L\langle t'\rangle = O_2\langle L'\langle\lambda y.q\rangle p\rangle$. Sub-cases:
  - $L \parallel O_2$. Then there exists a double context $\mathbb{O}$ such that $\mathbb{O}\langle\langle\cdot\rangle, t'\rangle = O_2$ and $\mathbb{O}\langle L'\langle\lambda y.q\rangle p, \langle\cdot\rangle\rangle = L$. Let $L'' \coloneqq \mathbb{O}\langle L'\langle q[y{\leftarrow}p]\rangle, \langle\cdot\rangle\rangle$. We have:

$$\begin{array}{ccc} t = O\langle t'[x{\leftarrow}\overbrace{\mathbb{O}\langle L'\langle\lambda y.q\rangle p, v\rangle}^{=L\langle v\rangle}]\rangle & \xrightarrow{\mathsf{ogc}} & O\langle\overbrace{\mathbb{O}\langle L'\langle\lambda y.q\rangle p, t'\rangle}^{=L\langle t'\rangle}\rangle = r \\ \Big\downarrow{\scriptstyle \mathsf{om}} & & \Big\downarrow{\scriptstyle \mathsf{om}} \\ O\langle t'[x{\leftarrow}\underbrace{\mathbb{O}\langle L'\langle q[y{\leftarrow}p]\rangle, v\rangle}_{=L''\langle v\rangle}]\rangle & \xrightarrow{\mathsf{ogc}} & O\langle\underbrace{\mathbb{O}\langle L'\langle q[y{\leftarrow}p]\rangle, t'\rangle}_{=L''\langle t'\rangle}\rangle = u \end{array}$$

  - $O_2 \prec L$. That is, $L = O_2\langle O_3\rangle$ for some O_3. We have then $O_3\langle t'\rangle = L'\langle\lambda y.q\rangle p$. Since O_3 is a substitution context, $O_3\langle t'\rangle$ can only be an application when O_3 is empty, which in turn implies $L = O_2$. This shall be treated in the following case.
  - $L \prec O_2$. That is, $O_2 = L\langle O_3\rangle$ for some O_3. We have then $t' = O_3\langle L'\langle\lambda y.q\rangle p\rangle$. Then:

$$\begin{array}{ccc} t = O\langle O_3\langle L'\langle\lambda y.q\rangle p\rangle[x{\leftarrow}L\langle v\rangle]\rangle & \xrightarrow{\mathsf{ogc}} & O\langle L\langle O_3\langle L'\langle\lambda y.q\rangle p\rangle\rangle\rangle = r \\ \Big\downarrow{\scriptstyle \mathsf{om}} & & \Big\downarrow{\scriptstyle \mathsf{om}} \\ O\langle O_3\langle L'\langle q[y{\leftarrow}p]\rangle\rangle[x{\leftarrow}L\langle v\rangle]\rangle & \xrightarrow{\mathsf{ogc}} & O\langle L\langle O_3\langle L'\langle q[y{\leftarrow}p]\rangle\rangle\rangle\rangle = u \end{array}$$

- $O_1 \prec O$. That is, $O = O_1\langle O_2\rangle$ for some O_2. We have then $O_2\langle L\langle t'\rangle\rangle = L'\langle\lambda y.q\rangle p$. Note that the case where O_2 is empty has already been treated. Sub-cases:
  - $O_2 = O_3 p$ for some O_3. We have then $O_3\langle L\langle t'\rangle\rangle = L'\langle\lambda y.q\rangle$. Sub-cases:
    - $O_3 \parallel L'$. Then there exists a double context $\mathbb{O}$ such that $\mathbb{O}\langle\langle\cdot\rangle, \lambda y.q\rangle = O_3$ and $\mathbb{O}\langle L\langle t'\rangle, \langle\cdot\rangle\rangle = L'$. Let $L'' \coloneqq \mathbb{O}\langle t'[x{\leftarrow}L\langle v\rangle], \langle\cdot\rangle\rangle$. Then:

$$\begin{array}{ccc} t = O_1\langle\overbrace{\mathbb{O}\langle t'[x{\leftarrow}L\langle v\rangle], \lambda y.q\rangle}^{=L''\langle\lambda y.q\rangle} p\rangle & \xrightarrow{\mathsf{ogc}} & O_1\langle\overbrace{\mathbb{O}\langle L\langle t'\rangle, \lambda y.q\rangle}^{=L'\langle\lambda y.q\rangle} p\rangle = r \\ \Big\downarrow{\scriptstyle \mathsf{om}} & & \Big\downarrow{\scriptstyle \mathsf{om}} \\ O_1\langle\underbrace{\mathbb{O}\langle t'[x{\leftarrow}L\langle v\rangle], q[y{\leftarrow}p]\rangle}_{=L''\langle q[y{\leftarrow}p]\rangle}\rangle & \xrightarrow{\mathsf{ogc}} & O_1\langle\underbrace{\mathbb{O}\langle L\langle t'\rangle, q[y{\leftarrow}p]\rangle}_{=L'\langle q[y{\leftarrow}p]\rangle}\rangle = u \end{array}$$

    - $O_3 \prec L'$. That is, $L' = O_3\langle O_4\rangle$ for some O_4 (note that O_3 and O_4 are both substitution contexts in this case). We have $L\langle t'\rangle = O_4\langle\lambda y.q\rangle$. Sub-cases:

(a) $L \parallel O_4$. Impossible since O_4 is also a substitution context.

(b) $L \prec O_4$. That is, $O_4 = L\langle O_5\rangle$ for some O_5 (note that O_5 is a substitution context in this case). We have $t' = O_5\langle\lambda y.q\rangle$. Let $L'' \coloneqq O_3\langle O_5[x{\leftarrow}L\langle v\rangle]\rangle$. Then:

$$\begin{array}{ccc} t = O_1\langle\overbrace{O_3\langle O_5\langle\lambda y.q\rangle[x{\leftarrow}L\langle v\rangle]\rangle}^{=L''\langle\lambda y.q\rangle} p\rangle & \xrightarrow{\mathsf{ogc}} & O_1\langle\overbrace{O_3\langle L\langle O_5\langle\lambda y.q\rangle\rangle\rangle}^{=L'\langle\lambda y.q\rangle} p\rangle = r \\ \Big\downarrow{\scriptstyle \mathsf{om}} & & \Big\downarrow{\scriptstyle \mathsf{om}} \\ O_1\langle\underbrace{O_3\langle O_5\langle q[y{\leftarrow}p]\rangle[x{\leftarrow}L\langle v\rangle]\rangle}_{=L''\langle q[y{\leftarrow}p]\rangle}\rangle & \xrightarrow{\mathsf{ogc}} & O_1\langle\underbrace{O_3\langle L\langle O_5\langle q[y{\leftarrow}p]\rangle\rangle\rangle}_{=L'\langle q[y{\leftarrow}p]\rangle}\rangle = u \end{array}$$

(c) $O_4 \prec L$. That is, $L = O_4\langle O_5\rangle$ for some O_5. The case where O_5 is empty, that is, $L = O_4$, has already been treated. Assume then that O_5 is non-empty. We have $O_5\langle t'\rangle = \lambda y.q$, which is impossible since O_5 is open and non-empty.

    - $L' \prec O_3$. That is, $O_3 = L'\langle O_4\rangle$ for some O_4. Note that the case where O_4 is empty, that is, $O_3 = L'$, has already been treated. Assume then that O_4 is non-empty. We have $O_4\langle L\langle t'\rangle\rangle = \lambda y.p$, which is impossible since O_4 is open and non-empty.

· $O_2 = L'\langle \lambda y.q\rangle O_3$ for some O_3. We have then $O_3\langle L\langle t'\rangle\rangle = p$. Then:

$$\begin{array}{ccc} t = O_1\langle L'\langle \lambda y.q\rangle O_3\langle t'[x{\leftarrow}L\langle v\rangle]\rangle\rangle & \xrightarrow{\mathsf{ogc}} & O_1\langle L'\langle \lambda y.q\rangle O_3\langle L\langle t'\rangle\rangle\rangle = r \\ \Big\downarrow{\scriptstyle \mathsf{om}} & & \Big\downarrow{\scriptstyle \mathsf{om}} \\ O_1\langle q[y{\leftarrow}O_3\langle t'[x{\leftarrow}L\langle v\rangle]\rangle]\rangle & \xrightarrow{\mathsf{ogc}} & O_1\langle q[y{\leftarrow}O_3\langle L\langle t'\rangle\rangle]\rangle = u \end{array}$$

(ii) $r = O_1\langle O_2\langle y\rangle[y{\leftarrow}L'\langle v'\rangle]\rangle \to_{\mathsf{oe}} O_1\langle L'\langle O_2\langle v'\rangle[y{\leftarrow}v']\rangle\rangle = u$. The situation is then as follows:

$$\begin{array}{ccc} t = O\langle t'[x{\leftarrow}L\langle v\rangle]\rangle & \xrightarrow{\mathsf{ogc}} & O\langle L\langle t'\rangle\rangle = r = O_1\langle O_2\langle y\rangle[y{\leftarrow}L'\langle v'\rangle]\rangle \\ & & \Big\downarrow{\scriptstyle \mathsf{oe}} \\ & & O_1\langle L'\langle O_2\langle v'\rangle[y{\leftarrow}v']\rangle\rangle = u \end{array}$$

Cases of the positioning of O_1 with respect to the other shape of r, namely $O\langle L\langle t'\rangle\rangle$:

- $O_1 \parallel O$. Then there exists a double context $\mathbb{O}$ such that $\mathbb{O}\langle\langle\cdot\rangle, L\langle t'\rangle\rangle = O_1$ and $\mathbb{O}\langle O_2\langle y\rangle[y{\leftarrow}L'\langle v'\rangle], \langle\cdot\rangle\rangle = O$. Then:

$$\begin{array}{ccc} t = \mathbb{O}\langle O_2\langle y\rangle[y{\leftarrow}L'\langle v'\rangle], t'[x{\leftarrow}L\langle v\rangle]\rangle & \xrightarrow{\mathsf{ogc}} & \mathbb{O}\langle O_2\langle y\rangle[y{\leftarrow}L'\langle v'\rangle], L\langle t'\rangle\rangle = r \\ \Big\downarrow{\scriptstyle \mathsf{oe}} & & \Big\downarrow{\scriptstyle \mathsf{oe}} \\ \mathbb{O}\langle L'\langle O_2\langle v'\rangle[y{\leftarrow}v']\rangle, t'[x{\leftarrow}L\langle v\rangle]\rangle & \xrightarrow{\mathsf{ogc}} & \mathbb{O}\langle L'\langle O_2\langle v'\rangle[y{\leftarrow}v']\rangle, L\langle t'\rangle\rangle = u \end{array}$$

- $O \prec O_1$. That is, $O_1 = O\langle O_3\rangle$ for some O_3. We have then $L\langle t'\rangle = O_3\langle O_2\langle y\rangle[y{\leftarrow}L'\langle v'\rangle]\rangle$. Sub-cases:
  · $L \parallel O_3$. Then there exists a double context $\mathbb{O}$ such that $\mathbb{O}\langle\langle\cdot\rangle, O_2\langle y\rangle[y{\leftarrow}L'\langle v'\rangle]\rangle = L$ and $\mathbb{O}\langle t', \langle\cdot\rangle\rangle = O_3$. Let $L'' := \mathbb{O}\langle\langle\cdot\rangle, L'\langle O_2\langle v'\rangle[y{\leftarrow}v']\rangle\rangle$. Then:

$$\begin{array}{ccc} t = O\langle t'[x{\leftarrow}\overbrace{\mathbb{O}\langle v, O_2\langle y\rangle[y{\leftarrow}L'\langle v'\rangle]\rangle}^{=L\langle v\rangle}]\rangle & \xrightarrow{\mathsf{ogc}} & O\langle\overbrace{\mathbb{O}\langle t', O_2\langle y\rangle[y{\leftarrow}L'\langle v'\rangle]\rangle}^{=L\langle t'\rangle}\rangle = r \\ \Big\downarrow{\scriptstyle \mathsf{oe}} & & \Big\downarrow{\scriptstyle \mathsf{oe}} \\ O\langle t'[x{\leftarrow}\underbrace{\mathbb{O}\langle v, L'\langle O_2\langle v'\rangle[y{\leftarrow}v']\rangle\rangle}_{=L''\langle v\rangle}]\rangle & \xrightarrow{\mathsf{ogc}} & O\langle\underbrace{\mathbb{O}\langle t', L'\langle O_2\langle v'\rangle[y{\leftarrow}v']\rangle\rangle}_{=L''\langle t'\rangle}\rangle = u \end{array}$$

  · $O_3 \prec L$. That is, $L = O_3\langle O_4\rangle$ for some O_4 (note that O_3 and O_4 are both substitution contexts in this case). We have then $O_4\langle t'\rangle = O_2\langle y\rangle[y{\leftarrow}L'\langle v'\rangle]$. Sub-cases:
  * $O_4 \parallel \langle\cdot\rangle[y{\leftarrow}L'\langle v'\rangle]$. Impossible since O_4 is also a substitution context.
  * $\langle\cdot\rangle[y{\leftarrow}L'\langle v'\rangle] \prec O_4$. That is, $O_4 = O_5[y{\leftarrow}L'\langle v'\rangle]$ for some O_5 (note that O_5 is a substitution context in this case). We have then $O_5\langle t'\rangle = O_2\langle y\rangle$. Sub-cases:
  (a) $O_5 \parallel O_2$. Then there exists a double context $\mathbb{O}$ such that $\mathbb{O}\langle\langle\cdot\rangle, y\rangle = O_5$ and $\mathbb{O}\langle t', \langle\cdot\rangle\rangle = O_2$. Let $L'' = O_3\langle L'\langle\mathbb{O}\langle\langle\cdot\rangle, v'\rangle[y{\leftarrow}v']\rangle\rangle$. Then:

$$\begin{array}{ccc} t = O\langle t'[x{\leftarrow}\overbrace{O_3\langle\mathbb{O}\langle v, y\rangle[y{\leftarrow}L'\langle v'\rangle]\rangle}^{=L\langle v\rangle}]\rangle & \xrightarrow{\mathsf{ogc}} & O\langle\overbrace{O_3\langle\mathbb{O}\langle t', y\rangle[y{\leftarrow}L'\langle v'\rangle]\rangle}^{=L\langle t'\rangle}\rangle = r \\ \Big\downarrow{\scriptstyle \mathsf{oe}} & & \Big\downarrow{\scriptstyle \mathsf{oe}} \\ O\langle t'[x{\leftarrow}\underbrace{O_3\langle L'\langle\mathbb{O}\langle v, v'\rangle[y{\leftarrow}v']\rangle\rangle}_{=L''\langle v\rangle}]\rangle & \xrightarrow{\mathsf{ogc}} & O\langle\underbrace{O_3\langle L'\langle\mathbb{O}\langle t', v'\rangle[y{\leftarrow}v']\rangle\rangle}_{=L''\langle t'\rangle}\rangle = u \end{array}$$

  (b) $O_5 \prec O_2$. That is, $O_2 = O_5\langle O_6\rangle$ for some O_6. We have then $O_6\langle y\rangle = t'$. This is impossible since y is bound in O_4 (thus in L).
  (c) $O_2 \prec O_5$. That is, $O_5 = O_2\langle O_6\rangle$ for some O_6. Assume that O_6 is non-empty (the case where $O_6 = \langle\cdot\rangle$, that is, $O_5 = O_2$, has already been treated). We have $O_6\langle t'\rangle = y$, which is impossible since O_6 is non-empty.
  * $O_4 \prec \langle\cdot\rangle[y{\leftarrow}L'\langle v'\rangle]$. That is, $\langle\cdot\rangle[y{\leftarrow}L'\langle v'\rangle] = O_4\langle O_5\rangle$ for some O_5. The case where O_5 is empty, that is, $O_4 = \langle\cdot\rangle[y{\leftarrow}L'\langle v'\rangle]$ has already been treated in the case $\langle\cdot\rangle[y{\leftarrow}L'\langle v'\rangle] \prec O_4$ above, and the case where $O_4 = \langle\cdot\rangle$, that is, $L = O_3$, will be treated in the case $L \prec O_3$ below.
  · $L \prec O_3$. That is, $O_3 = L\langle O_4\rangle$ for some O_4. We have then $t' = O_4\langle O_2\langle y\rangle[y{\leftarrow}L'\langle v'\rangle]\rangle$. Then:

$$\begin{array}{ccc} t = O\langle O_4\langle O_2\langle y\rangle[y{\leftarrow}L'\langle v'\rangle]\rangle[x{\leftarrow}L\langle v\rangle]\rangle & \xrightarrow{\mathsf{ogc}} & O\langle L\langle O_4\langle O_2\langle y\rangle[y{\leftarrow}L'\langle v'\rangle]\rangle\rangle\rangle = r \\ \mathsf{oe}\ \vdots & & \downarrow\mathsf{oe} \\ O\langle O_4\langle L'\langle O_2\langle v'\rangle[y{\leftarrow}v']\rangle\rangle[x{\leftarrow}L\langle v\rangle]\rangle & \xrightarrow{\mathsf{ogc}} & O\langle L\langle O_4\langle L'\langle O_2\langle v'\rangle[y{\leftarrow}v']\rangle\rangle\rangle\rangle = u \end{array}$$

- $O_1 \prec O$. That is, $O = O_1\langle O_3\rangle$ for some O_3. We have then $O_3\langle L\langle t'\rangle\rangle = O_2\langle y\rangle[y{\leftarrow}L'\langle v'\rangle]$. Sub-cases:
  - $O_3 \parallel \langle\cdot\rangle[y{\leftarrow}L'\langle v'\rangle]$. Then there exists a double context $\mathbb{O}$ such that $\mathbb{O}\langle\langle\cdot\rangle, O_2\langle y\rangle\rangle = O_3$ and $\mathbb{O}\langle L\langle t'\rangle, \langle\cdot\rangle\rangle = \langle\cdot\rangle[y{\leftarrow}L'\langle v'\rangle]$. Cases of the position of $L\langle t'\rangle$ in $\langle\cdot\rangle[y{\leftarrow}L'\langle v'\rangle]$:
    - $L\langle t'\rangle$ is a subterm of L'. Then there exists a double context $\mathbb{O}'$ such that $\mathbb{O}'\langle L\langle t'\rangle, \langle\cdot\rangle\rangle = L'$ and $\mathbb{O}\langle\langle\cdot\rangle_1, \langle\cdot\rangle_2\rangle = \langle\cdot\rangle_2[y{\leftarrow}\mathbb{O}'\langle\langle\cdot\rangle_1, v'\rangle]$. Let $L'' \coloneqq \mathbb{O}'\langle t'[x{\leftarrow}L\langle v\rangle], \langle\cdot\rangle\rangle$. Then:
$$\begin{array}{ccc} t = O_1\langle \overbrace{\mathbb{O}\langle t'[x{\leftarrow}L\langle v\rangle], O_2\langle y\rangle\rangle}^{=O_2\langle y\rangle[y{\leftarrow}L''\langle v'\rangle]} \rangle & \xrightarrow{\mathsf{ogc}} & O_1\langle \overbrace{\mathbb{O}\langle L\langle t'\rangle, O_2\langle y\rangle\rangle}^{=O_2\langle y\rangle[y{\leftarrow}L'\langle v'\rangle]} \rangle = r \\ \mathsf{oe}\ \vdots & & \downarrow\mathsf{oe} \\ O_1\langle \underbrace{\mathbb{O}'\langle t'[x{\leftarrow}L\langle v\rangle], O_2\langle v'\rangle[y{\leftarrow}v']\rangle}_{=L''\langle O_2\langle v'\rangle[y{\leftarrow}v']\rangle} \rangle & \xrightarrow{\mathsf{ogc}} & O_1\langle \underbrace{\mathbb{O}'\langle L\langle t'\rangle, O_2\langle v'\rangle[y{\leftarrow}v']\rangle}_{=L'\langle O_2\langle v'\rangle[y{\leftarrow}v']\rangle} \rangle = u \end{array}$$
    - $L\langle t'\rangle$ is of the form $L_1\langle v'\rangle$ with $L' = L_2\langle L_1\rangle$ for some L_2, and we have $\mathbb{O}\langle\langle\cdot\rangle_1, \langle\cdot\rangle_2\rangle = \langle\cdot\rangle_2[y{\leftarrow}L_2\langle\langle\cdot\rangle_1\rangle]$. It is easy to see that t' is of the form $L_3\langle v'\rangle$ with $L\langle L_3\rangle = L_1$. Let $L'' = L_2\langle L_3[x{\leftarrow}L\langle v\rangle]\rangle$. Then:
$$\begin{array}{ccc} t = O_1\langle \overbrace{\mathbb{O}\langle t'[x{\leftarrow}L\langle v\rangle], O_2\langle y\rangle\rangle}^{=O_2\langle y\rangle[y{\leftarrow}L''\langle v'\rangle]} \rangle & \xrightarrow{\mathsf{ogc}} & O_1\langle \overbrace{\mathbb{O}\langle L\langle t'\rangle, O_2\langle y\rangle\rangle}^{=O_2\langle y\rangle[y{\leftarrow}L'\langle v'\rangle]} \rangle = r \\ \mathsf{oe}\ \vdots & & \downarrow\mathsf{oe} \\ O_1\langle \underbrace{L_2\langle L_3\langle O_2\langle v'\rangle[y{\leftarrow}v']\rangle[x{\leftarrow}L\langle v\rangle]\rangle}_{=L''\langle O_2\langle v'\rangle[y{\leftarrow}v']\rangle} \rangle & \xrightarrow{\mathsf{ogc}} & O_1\langle \underbrace{L_2\langle L\langle L_3\langle O_2\langle v'\rangle[y{\leftarrow}v']\rangle\rangle\rangle}_{=L'\langle O_2\langle v'\rangle[y{\leftarrow}v']\rangle} \rangle = u \end{array}$$
  - $\langle\cdot\rangle[y{\leftarrow}L'\langle v'\rangle] \prec O_3$. That is, $O_3 = O_4[y{\leftarrow}L'\langle v'\rangle]$ for some O_4. We have then $O_4\langle L\langle t'\rangle\rangle = O_2\langle y\rangle$. Sub-cases:
    - $O_4 \parallel O_2$. Then there exists a double context $\mathbb{O}$ such that $\mathbb{O}\langle\langle\cdot\rangle, y\rangle = O_4$ and $\mathbb{O}\langle L\langle t'\rangle, \langle\cdot\rangle\rangle = O_2$. Then:
$$\begin{array}{ccc} t = O_1\langle \mathbb{O}\langle t'[x{\leftarrow}L\langle v\rangle], y\rangle[y{\leftarrow}L'\langle v'\rangle]\rangle & \xrightarrow{\mathsf{ogc}} & O_1\langle \mathbb{O}\langle L\langle t'\rangle, y\rangle[y{\leftarrow}L'\langle v'\rangle]\rangle = r \\ \mathsf{oe}\ \vdots & & \downarrow\mathsf{oe} \\ O_1\langle L'\langle \mathbb{O}\langle t'[x{\leftarrow}L\langle v\rangle], v'\rangle[y{\leftarrow}v']\rangle\rangle & \xrightarrow{\mathsf{ogc}} & O_1\langle L'\langle \mathbb{O}\langle L\langle t'\rangle, v'\rangle[y{\leftarrow}v']\rangle\rangle = u \end{array}$$
    - $O_4 \prec O_2$. That is, $O_2 = O_4\langle O_5\rangle$ for some O_5. We have then $L\langle t'\rangle = O_5\langle y\rangle$. Sub-cases:

(a) y is a subterm of L. Then there exists a double context $\mathbb{O}$ such that $\mathbb{O}\langle\langle\cdot\rangle, y\rangle = L$ and $\mathbb{O}\langle t', \langle\cdot\rangle\rangle = O_5$. Then:

$$\begin{array}{ccc} t = O_1\langle O_4\langle t'[x{\leftarrow}\mathbb{O}\langle v, y\rangle]\rangle[y{\leftarrow}L'\langle v'\rangle]\rangle & \xrightarrow{\mathsf{ogc}} & O_1\langle O_4\langle \mathbb{O}\langle t', y\rangle\rangle[y{\leftarrow}L'\langle v'\rangle]\rangle = r \\ \mathsf{oe}\ \vdots & & \downarrow\mathsf{oe} \\ O_1\langle L'\langle O_4\langle t'[x{\leftarrow}\mathbb{O}\langle v, v'\rangle]\rangle[y{\leftarrow}v']\rangle\rangle & \xrightarrow{\mathsf{ogc}} & O_1\langle L'\langle O_4\langle \mathbb{O}\langle t', v'\rangle\rangle[y{\leftarrow}v']\rangle\rangle = u \end{array}$$

(b) y is a subterm t'. That is, $t' = O_6\langle y\rangle$ for some O_6, and we have $O_5 = L\langle O_6\rangle$. Then:

$$\begin{array}{ccc} t = O_1\langle O_4\langle O_6\langle y\rangle[x{\leftarrow}L\langle v\rangle]\rangle[y{\leftarrow}L'\langle v'\rangle]\rangle & \xrightarrow{\mathsf{ogc}} & O_1\langle O_4\langle L\langle O_6\langle y\rangle\rangle\rangle[y{\leftarrow}L'\langle v'\rangle]\rangle = r \\ \mathsf{oe}\ \vdots & & \downarrow\mathsf{oe} \\ O_1\langle L'\langle O_4\langle O_6\langle v'\rangle[x{\leftarrow}L\langle v\rangle]\rangle[y{\leftarrow}v']\rangle\rangle & \xrightarrow{\mathsf{ogc}} & O_1\langle L'\langle O_4\langle L\langle O_6\langle v'\rangle\rangle\rangle[y{\leftarrow}v']\rangle\rangle = u \end{array}$$

    - $O_2 \prec O_4$. That is, $O_4 = O_2\langle O_5\rangle$ for some O_5. Assume that O_5 is non-empty (the case where $O_5 = \langle\cdot\rangle$, that is, $O_2 = O_4$, has already been treated). We have $O_5\langle L\langle t'\rangle\rangle = y$, which is impossible since O_5 is non-empty.
  - $O_3 \prec \langle\cdot\rangle[y{\leftarrow}L'\langle v'\rangle]$. That is, $\langle\cdot\rangle[y{\leftarrow}L'\langle v'\rangle] = O_3\langle O_4\rangle$ for some O_4. Then either O_3 or O_4 is empty. Both cases have already been treated.

□

Proposition 3.2 (Postponement of garbage collection, originally at p. 6) *If* $d : t \to_{\mathsf{o}}^{*} u$, *then there exist reduction sequences* $e : t \to_{\mathsf{o}\neg\mathsf{gc}}^{*} u'$ *and* $f : u' \to_{\mathsf{ogc}}^{*} u$ *with* $|e|_{\mathsf{om}} = |d|_{\mathsf{om}}$, $|e|_{\mathsf{oe}} = |d|_{\mathsf{oe}}$, *and* $|f| = |d|_{\mathsf{ogc}}$.

Proof. Straightforward proof by induction on $|d|$ using Prop. 3.1. □

Proposition 3.3 (Local termination, originally at p. 6) *Let* $a \in \{\mathsf{m}, \mathsf{e}, \mathsf{gc}\}$. *Relation* $\to_{\mathsf{o}a}$ *is strongly normalizing. Moreover,* $\to_{\mathsf{oe}} \cup \to_{\mathsf{ogc}}$ *is strongly normalizing.*

Proof. For $\to_{\mathsf{om}}$ and $\to_{\mathsf{ogc}}$ it is trivial, because the number of constructors decreases. For $\to_{\mathsf{oe}}$, one needs a standard measure, omitted here. In Accattoli [2], the interested reader can find all the details for a similar and more general case. The *moreover* part follows the fact that given a $\to_{\mathsf{oe}} \cup \to_{\mathsf{ogc}}$-sequence one can postpone $\to_{\mathsf{ogc}}$ preserving the number of steps of both $\to_{\mathsf{oe}}$ and $\to_{\mathsf{ogc}}$ (Prop. 3.2), thus reducing the strong normalization of $\to_{\mathsf{oe}} \cup \to_{\mathsf{ogc}}$ to that of $\to_{\mathsf{oe}}$ and $\to_{\mathsf{ogc}}$ separately. □

C Proofs from Sect. 4 (The (Explicit) Open Positive λ-Calculus)

Theorem 4.3 (Positive diamond, originally at p. 9) *Relation* $\to_{\mathsf{ox}_+}$ *is diamond.*

Proof. Suppose that $t_1 \ {}_{\mathsf{ox}_+}\!\leftarrow t \to_{\mathsf{ox}_+} t_2$ with $t_1 \neq t_2$. We prove that there exists t_3 such that $t_1 \to_{\mathsf{ox}_+} t_3 \ {}_{\mathsf{ox}_+}\!\leftarrow t_2$ by induction on the reduction step $t \to_{\mathsf{ox}_+} t_1$ and by case analysis on the step $t \to_{\mathsf{ox}_+} u$. The base cases:

- $t = u[x\leftarrow(\lambda y.E\langle z\rangle)w] \mapsto_{\mathsf{m}_+} E\langle u\{x\leftarrow z\}\rangle\{y\leftarrow w\} = t_1$. Since reduction is weak, the step $t \to_{\mathsf{ox}_+} t_2$ takes place entirely in u, and by Lemma 4.2 we have:
$$\begin{array}{ccc} u[x\leftarrow(\lambda y.E\langle z\rangle)w] & \xmapsto{\mathsf{m}_+} & E\langle u\{x\leftarrow z\}\rangle\{y\leftarrow w\} \\ {\scriptstyle \mathsf{ox}_+}\downarrow & & \downarrow{\scriptstyle \mathsf{ox}_+} \\ u'[x\leftarrow(\lambda y.E\langle z\rangle)w] & \xmapsto{\mathsf{m}_+} & E\langle u'\{x\leftarrow z\}\rangle\{y\leftarrow w\} \end{array}$$
- $t = E\langle u[x\leftarrow yz]\rangle[y\leftarrow\lambda w.r] \mapsto_{\mathsf{e}_+} E\langle u[x\leftarrow(\lambda w.r)z]\rangle[y\leftarrow\lambda w.r] = t_1$. Cases of $t \to_{\mathsf{ox}_+} t_2$:
  - It is a $\mapsto_{\mathsf{e}_+}$ step involving the same acting abstraction and a different receiving application. If the receiving application is in E, then $E = E_1\langle E_2[x'\leftarrow yz']\rangle$. The diagram then closes as follows:
$$\begin{array}{ccc} E_1\langle E_2\langle u[x\leftarrow yz]\rangle[x'\leftarrow yz']\rangle[y\leftarrow\lambda w.r] & \xmapsto{\mathsf{e}_+} & E_1\langle E_2\langle u[x\leftarrow(\lambda w.r)z]\rangle[x'\leftarrow yz']\rangle[y\leftarrow\lambda w.r] \\ {\scriptstyle \mathsf{e}_+}\downarrow & & \downarrow{\scriptstyle \mathsf{e}_+} \\ E_1\langle E_2\langle u[x\leftarrow yz]\rangle[x'\leftarrow(\lambda w.r)z']\rangle[y\leftarrow\lambda w.r] & \xmapsto{\mathsf{e}_+} & E_1\langle E_2\langle u[x\leftarrow(\lambda w.r)z]\rangle[x'\leftarrow(\lambda w.r)z']\rangle[y\leftarrow\lambda w.r] \end{array}$$
    If the receiving application is in u, then the diagram closes similarly.
  - It takes place entirely in E, then we have:
$$\begin{array}{ccc} E\langle u[x\leftarrow yz]\rangle[y\leftarrow\lambda w.r] & \xmapsto{\mathsf{e}_+} & E\langle u[x\leftarrow(\lambda w.r)z]\rangle[y\leftarrow\lambda w.r] \\ {\scriptstyle \mathsf{ox}_+}\downarrow & & \downarrow{\scriptstyle \mathsf{ox}_+} \\ E'\langle u[x\leftarrow yz]\rangle[y\leftarrow\lambda w.r] & \xmapsto{\mathsf{e}_+} & E'\langle u[x\leftarrow(\lambda w.r)z]\rangle[y\leftarrow\lambda w.r] \end{array}$$
  - It takes place entirely in u. The diagram is analogous to the previous one.
  - It is a $\to_{\mathsf{oe}_+}$ step where the acting abstraction is in E and the receiving application is in u. Then $E = E'\langle E''[y'\leftarrow\lambda w'.r']\rangle$ and $u = E'''\langle u'[x'\leftarrow y'z']\rangle$. The diagram then closes as follows:
$$\begin{array}{ccc} E'\langle E''\langle E'''\langle u'[x'\leftarrow y'z']\rangle[x\leftarrow yz]\rangle[y'\leftarrow\lambda w'.r']\rangle[y\leftarrow\lambda w.r] & \xmapsto{\mathsf{e}_+} & E'\langle E''\langle E'''\langle u'[x'\leftarrow y'z']\rangle[x\leftarrow(\lambda w.r)z]\rangle[y'\leftarrow\lambda w'.r']\rangle[y\leftarrow\lambda w.r] \\ {\scriptstyle \mathsf{oe}_+}\downarrow & & \downarrow{\scriptstyle \mathsf{oe}_+} \\ E'\langle E''\langle E'''\langle u'[x'\leftarrow(\lambda w'.r')z']\rangle[x\leftarrow yz]\rangle[y'\leftarrow\lambda w'.r']\rangle[y\leftarrow\lambda w.r] & \xmapsto{\mathsf{e}_+} & E'\langle E''\langle E'''\langle u'[x'\leftarrow(\lambda w'.r')z']\rangle[x\leftarrow(\lambda w.r)z]\rangle[y'\leftarrow\lambda w'.r']\rangle[y\leftarrow\lambda w.r] \end{array}$$

· It is a $\to_{\mathsf{ogc}_+}$ step with the abstraction in E and the 'body' of the step containing u. That is, Then $E = E'\langle E''[y'{\leftarrow}\lambda w'.r']\rangle$ with $y' \notin \mathsf{fv}(E''\langle u[x{\leftarrow}yz]\rangle)$. Then:

$$\begin{array}{ccc} E'\langle E''\langle u[x{\leftarrow}yz]\rangle[y'{\leftarrow}\lambda w'.r']\rangle[y{\leftarrow}\lambda w.r] & \xmapsto{\mathsf{e}_+} & E'\langle E''\langle u[x{\leftarrow}(\lambda w.r)z]\rangle[y'{\leftarrow}\lambda w'.r']\rangle[y{\leftarrow}\lambda w.r] \\ \downarrow{\scriptstyle \mathsf{ogc}_+} & & \downarrow{\scriptstyle \mathsf{ogc}_+} \\ E'\langle E''\langle u[x{\leftarrow}yz]\rangle\rangle[y{\leftarrow}\lambda w.r] & \xmapsto{\mathsf{e}_+} & E'\langle E''\langle u[x{\leftarrow}(\lambda w.r)z]\rangle\rangle[y{\leftarrow}\lambda w.r] \end{array}$$

- $t = u[x{\leftarrow}\lambda y.r] \mapsto_{\mathsf{gc}_+} u = t_1$. Then the step $t \to_{\mathsf{ox}_+} t_2$ takes place in u and we have:

$$\begin{array}{ccc} u[x{\leftarrow}\lambda y.r] & \xmapsto{\mathsf{gc}_+} & u \\ \downarrow{\scriptstyle \mathsf{ox}_+} & & \downarrow{\scriptstyle \mathsf{ox}_+} \\ u'[x{\leftarrow}\lambda y.r] & \xmapsto{\mathsf{gc}_+} & u' \end{array}$$

For the inductive cases,

- $t = u[x{\leftarrow}\lambda y.r] \to_{\mathsf{ox}_+} u_1[x{\leftarrow}\lambda y.r] = t_1$ with $u \to_{\mathsf{ox}_+} u_1$. Cases of $t \to_{\mathsf{ox}_+} t_2$:
  - · It takes place entirely in u. Then it follows by the *i.h.*
  - · It is a root step involving $[x{\leftarrow}\lambda y.r]$. Then it is an already treated root case.
- $t = u[x{\leftarrow}(\lambda y.E\langle z\rangle)w] \to_{\mathsf{ox}_+} u_1[x{\leftarrow}(\lambda y.E\langle z\rangle)w] = t_1$ with $u \to_{\mathsf{ox}_+} u_1$. Similar to the previous case.

□

Proposition 4.4 (Local postponement of garbage collection, originally at p. 9) *Let t and u be λ_{oxpos} terms and $a \in \{\mathsf{m}_+, \mathsf{e}_+\}$. If $t \to_{\mathsf{ogc}_+} \to_{oa} u$, then $t \to_{oa} \to_{\mathsf{ogc}_+} u$.*

Proof. Assume $t = E\langle t_1[x{\leftarrow}\lambda y.t_2]\rangle \to_{\mathsf{ogc}_+} E\langle t_1\rangle = r$. Cases of $r \to_{oa} u$:

- It takes place entirely in E, then we have:

$$\begin{array}{ccc} E\langle t_1[x{\leftarrow}\lambda y.t_2]\rangle & \xrightarrow{\mathsf{gc}_+} & E\langle t_1\rangle \\ \vdots{\scriptstyle a} & & \downarrow{\scriptstyle a} \\ E'\langle t_1[x{\leftarrow}\lambda y.t_2]\rangle & \xrightarrow{\mathsf{gc}_+} & E'\langle t_1\rangle \end{array}$$

- It takes place entirely in t_1, then we have:

$$\begin{array}{ccc} E\langle t_1[x{\leftarrow}\lambda y.t_2]\rangle & \xrightarrow{\mathsf{gc}_+} & E\langle t_1\rangle \\ \vdots{\scriptstyle a} & & \downarrow{\scriptstyle a} \\ E\langle t_1'[x{\leftarrow}\lambda y.t_2]\rangle & \xrightarrow{\mathsf{gc}_+} & E\langle t_1'\rangle \end{array}$$

- It is a $\to_{\mathsf{oe}_+}$ step where the acting abstraction is in E and the receiving application is in t_1. Then $E = E_1\langle E_2[z{\leftarrow}\lambda w.t_3]\rangle$ and $t_1 = E_3\langle t_1'[x'{\leftarrow}zy']\rangle$, and we have:

$$\begin{array}{ccc} E_1\langle E_2\langle E_3\langle t_1'[x'{\leftarrow}zy']\rangle[x{\leftarrow}\lambda y.t_2]\rangle[z{\leftarrow}\lambda w.t_3]\rangle & \xrightarrow{\mathsf{gc}_+} & E_1\langle E_2\langle E_3\langle t_1'[x'{\leftarrow}zy']\rangle\rangle[z{\leftarrow}\lambda w.t_3]\rangle \\ \vdots{\scriptstyle a} & & \downarrow{\scriptstyle a} \\ E_1\langle E_2\langle E_3\langle t_1'[x'{\leftarrow}(\lambda w.t_3)y']\rangle[x{\leftarrow}\lambda y.t_2]\rangle[z{\leftarrow}\lambda w.t_3]\rangle & \xrightarrow{\mathsf{gc}_+} & E_1\langle E_2\langle E_3\langle t_1'[x'{\leftarrow}(\lambda w.t_3)y']\rangle\rangle[z{\leftarrow}\lambda w.t_3]\rangle \end{array}$$

- It is a $\to_{\mathsf{om}_+}$ step with $E = E_1\langle E_2[z{\leftarrow}(\lambda w.E'\langle x'\rangle)y']\rangle$. Then we have:

$$\begin{array}{ccc} E_1\langle E_2\langle t_1[x{\leftarrow}\lambda y.t_2]\rangle[z{\leftarrow}(\lambda w.E'\langle x'\rangle)y']\rangle & \xrightarrow{\mathsf{gc}_+} & E_1\langle E_2\langle t_1\rangle[z{\leftarrow}(\lambda w.E'\langle x'\rangle)y']\rangle \\ \vdots{\scriptstyle \mathsf{m}_+} & & \downarrow{\scriptstyle \mathsf{m}_+} \\ E_1\langle E'\langle E_2\langle t_1[x{\leftarrow}\lambda y.t_2]\rangle\{z{\leftarrow}x'\}\rangle\{w{\leftarrow}y'\}\rangle & \xrightarrow{\mathsf{gc}_+} & E_1\langle E'\langle E_2\langle t_1\rangle\{z{\leftarrow}x'\}\rangle\{w{\leftarrow}y'\}\rangle \end{array}$$

□

Proposition 4.5 (Postponement of garbage collection, originally at p. 9) *Let t and u be λ_{oxpos} terms, $d : t \to^*_{\mathsf{ox}_+} u$. Then there exist reduction sequences $e : t \to^*_{\mathsf{ox}_+\neg\mathsf{gc}} u'$ and $f : u' \to^*_{\mathsf{ogc}_+} u$ with $|e|_{\mathsf{om}_+} = |d|_{\mathsf{om}_+}$, $|e|_{\mathsf{oe}_+} = |d|_{\mathsf{oe}_+}$, and $|f| = |d|_{\mathsf{ogc}_+}$.*

Proof. Straightforward proof by induction on $|d|$ using Prop. 4.4. □

Proposition 4.6 (Local termination, originally at p. 9) *Let $a \in \{\mathsf{m}_+, \mathsf{e}_+, \mathsf{gc}_+\}$. Relation $\to_{\mathsf{o}a}$ is strongly normalizing. Moreover, $\to_{\mathsf{oe}_+} \cup \to_{\mathsf{ogc}_+}$ is strongly normalizing.*

Proof. As for Prop. 3.3. □

D Proofs from Sect. 5 (Dissecting λ_{ovsc}: Variable and Useful Steps)

Lemma 5.3 (Useful partitions open, originally at p. 12) *A VSC open context O is either useful or non-useful.*

Proof. By induction on O. The empty context $\langle\cdot\rangle$ is non-useful and not useful. For the inductive cases, the only interesting one is the following:

- $O = O't$. By *i.h.*, O' is either useful or non-useful.
  - O' is useful and not non-useful, which means that it can be written as $O'_1\langle Lu\rangle$. Then we have $O = O_1\langle Lu\rangle$ with $O_1 = O'_1t$. Therefore, O is useful. We now show that it is not non-useful. Obviously, O is not a substitution context. If it is of the form $O_2\langle tL\rangle$ (resp. $O_2\langle t[x\leftarrow L]\rangle$) for some O_2, then O_2 is of the form O'_2t with $O' = O'_2\langle tL\rangle$ (resp. $O' = O'_2\langle t[x\leftarrow L]\rangle$), which contradicts the hypothesis that O' is not non-useful. Then, O is not non-useful.
  - O' is non-useful. We distinguish three cases:
    $O' = L$. Then O is useful and not non-useful.
    $O' = O_1\langle uL\rangle$. Then O is non-useful and not useful.
    $O' = O_1\langle u[x\leftarrow L]\rangle$. Then O is non-useful and not useful.
- All the remaining cases can be treated in a similar way.

□

Lemma D.1 $\mathsf{sub}(O_1\langle L\langle O_2\rangle\rangle) \Leftrightarrow \mathsf{sub}(O_1\langle O_2\rangle)$.

Proof. Straightforward by induction on O_1. □

Lemma D.2 $\mathsf{sub}(O_1\langle O_2\rangle) \Leftrightarrow \mathsf{sub}(O_1) \wedge \mathsf{sub}(O_2)$.

Proof. Straightforward by induction on O_1. □

Lemma 5.4 (Context plugging and usefulness, originally at p. 12)

(i) $\mathsf{usef}(O_1\langle L\langle O_2\rangle\rangle) \Leftrightarrow \mathsf{usef}(O_1\langle O_2\rangle)$.
(ii) $\mathsf{usef}(O_1\langle O_2\rangle) \Leftrightarrow \mathsf{usef}(O_2) \vee (\mathsf{sub}(O_2) \wedge \mathsf{usef}(O_1))$.
(iii) $\mathsf{nusef}(O_1\langle O_2\rangle) \Leftrightarrow \mathsf{nusef}(O_2) \wedge (\mathsf{sub}(O_2) \Rightarrow \mathsf{nusef}(O_1))$.

Proof.

(i) By induction on O_1. The base case is trivial by Lemma 5.2.iii. For the inductive cases:
  - $O_1 = O'_1u$. By Lemma 5.2.i, $\mathsf{usef}(O_1\langle L\langle O_2\rangle\rangle) \Leftrightarrow \mathsf{usef}(O'_1\langle L\langle O_2\rangle\rangle) \vee \mathsf{sub}(O'_1\langle L\langle O_2\rangle\rangle)$ and $\mathsf{usef}(O_1\langle O_2\rangle) \Leftrightarrow \mathsf{usef}(O'_1\langle O_2\rangle) \vee \mathsf{sub}(O'_1\langle O_2\rangle)$. We then conclude by *i.h.* and Lemma D.1.
  - The remaining cases are straightforward by *i.h.* and Lemma 5.2.

(ii) By induction on O_1. The base case is trivial. For the inductive cases:
  - $O_1 = O'_1t$. We have: $\mathsf{usef}(O'_1\langle O_2\rangle t) \overset{\text{L.5.2}}{\Longleftrightarrow} \mathsf{sub}(O'_1\langle O_2\rangle) \vee \mathsf{usef}(O'_1\langle O_2\rangle) \overset{\text{L.D.2 and } i.h.}{\Longleftrightarrow} \mathsf{usef}(O_2) \vee (\mathsf{sub}(O_2) \wedge \mathsf{usef}(O'_1)) \vee (\mathsf{sub}(O'_1) \wedge \mathsf{sub}(O_2)) \overset{\text{L.5.2}}{\Longleftrightarrow} \mathsf{usef}(O_2) \vee (\mathsf{sub}(O_2) \wedge \mathsf{usef}(O_1))$.

- $O_1 = tO_1'$. We have: $\mathsf{usef}(tO_1'\langle O_2\rangle) \overset{\text{L.5.2}}{\iff} \mathsf{usef}(O_1'\langle O_2\rangle) \overset{i.h.}{\iff} \mathsf{usef}(O_2) \vee (\mathsf{sub}(O_2) \wedge \mathsf{usef}(O_1')) \overset{\text{L.5.2}}{\iff} \mathsf{usef}(O_2) \vee (\mathsf{sub}(O_2) \wedge \mathsf{usef}(O_1))$.
- $O_1 = O_1'[x{\leftarrow}t]$. We have: $\mathsf{usef}(O_1'\langle O_2\rangle[x{\leftarrow}t]) \overset{\text{L.5.2}}{\iff} \mathsf{usef}(O_1'\langle O_2\rangle) \overset{i.h.}{\iff} \mathsf{usef}(O_2) \vee (\mathsf{sub}(O_2) \wedge \mathsf{usef}(O_1')) \overset{\text{L.5.2}}{\iff} \mathsf{usef}(O_2) \vee (\mathsf{sub}(O_2) \wedge \mathsf{sub}(O_1))$.
- $O_1 = t[x{\leftarrow}O_1']$. We have: $\mathsf{usef}(O_1\langle O_2\rangle) \Leftrightarrow \mathsf{usef}(t[x{\leftarrow}O_1'\langle O_2\rangle]) \overset{\text{L.5.2}}{\iff} \mathsf{usef}(O_1'\langle O_2\rangle) \overset{i.h.}{\iff} \mathsf{usef}(O_2) \vee (\mathsf{sub}(O_2) \wedge \mathsf{usef}(O_1')) \overset{\text{L.5.2}}{\iff} \mathsf{usef}(O_2) \vee \mathsf{sub}(O_2) \wedge \mathsf{usef}(O_1)$.

(iii) This is a consequence of Lemma 5.3 and the previous point.

□

E Proofs from Sect. 6 (Core Factorization, or Postponing Non-Useful Steps)

Proposition 6.1 (Local postponement of $\to_{\mathsf{oe_{nu}}}$, originally at p. 13) *Let t and u be* VSC *terms. If $t \to_{\mathsf{oe_{nu}}}\to_{\mathsf{ocore}} u$ then $t \to_{\mathsf{ocore}}\to_{\mathsf{oe_{nu}}} u$ or $t \to_{\mathsf{ocore}}\to_{\mathsf{ocore}}\to_{\mathsf{oe_{nu}}} u$. More precisely:*

(i) $\to_{\mathsf{oe_{nu}}}\to_{\mathsf{om}} \subseteq \to_{\mathsf{om}}\to_{\mathsf{oe_{nu}}}$;
(ii) $\to_{\mathsf{oe_{nu}}}\to_{\mathsf{oe_u}} \subseteq \to_{\mathsf{oe_u}}\to_{\mathsf{oe_{nu}}} \cup \to_{\mathsf{oe_{var}}}\to_{\mathsf{oe_u}}\to_{\mathsf{oe_{nu}}}$;
(iii) $\to_{\mathsf{oe_{nu}}}\to_{\mathsf{oe_{var}}} \subseteq \to_{\mathsf{oe_{var}}}\to_{\mathsf{oe_{nu}}}$.

Proof. Assume that $t = O_1\langle O_2\langle x\rangle[x{\leftarrow}L\langle v\rangle]\rangle \to_{\mathsf{oe_{nu}}} O_1\langle L\langle O_2\langle v\rangle[x{\leftarrow}v]\rangle\rangle = r \to_{\mathsf{ocore}} u$ where v is an abstraction. The situation is then as follows: We now consider different cases of the reduction $r \to_{\mathsf{ocore}} u$.

(i) $r = O_3\langle L'\langle\lambda y.q\rangle p\rangle \to_{\mathsf{om}} O_3\langle L'\langle q[y{\leftarrow}p]\rangle\rangle = u$. The situation is then as follows:

$$\begin{array}{ccc} t = O_1\langle O_2\langle x\rangle[x{\leftarrow}L\langle v\rangle]\rangle & \xrightarrow{\mathsf{oe_{nu}}} & O_1\langle L\langle O_2\langle v\rangle[x{\leftarrow}v]\rangle\rangle = r = O_3\langle L'\langle\lambda y.q\rangle p\rangle \\ & & \downarrow_{\mathsf{om}} \\ & & O_3\langle L'\langle q[y{\leftarrow}p]\rangle\rangle = u \end{array}$$

Cases of the positioning of O_3 with respect to the other shape of r, namely $O_1\langle L\langle O_2\langle v\rangle[x{\leftarrow}v]\rangle\rangle$:

- $O_3 \parallel O_1$. Then there exists a double context $\mathbb{O}$ such that $\mathbb{O}\langle\langle\cdot\rangle, L'\langle\lambda y.q\rangle p\rangle = O_1$ and $\mathbb{O}\langle L\langle O_2\langle v\rangle[x{\leftarrow}v]\rangle, \langle\cdot\rangle\rangle = O_3$. We have:

$$\begin{array}{ccc} t = \mathbb{O}\langle O_2\langle x\rangle[x{\leftarrow}L\langle v\rangle], L'\langle\lambda y.q\rangle p\rangle & \xrightarrow{\mathsf{oe_{nu}}} & \mathbb{O}\langle L\langle O_2\langle v\rangle[x{\leftarrow}v]\rangle, L'\langle\lambda y.q\rangle p\rangle = r \\ \vdots_{\mathsf{om}} & & \downarrow_{\mathsf{om}} \\ \mathbb{O}\langle O_2\langle x\rangle[x{\leftarrow}L\langle v\rangle], L'\langle q[y{\leftarrow}p]\rangle\rangle & \xrightarrow{\mathsf{oe_{nu}}} & \mathbb{O}\langle L\langle O_2\langle v\rangle[x{\leftarrow}v]\rangle, L'\langle q[y{\leftarrow}p]\rangle\rangle = u \end{array}$$

  Where the bottom side of the diagram is non-useful because $\mathbb{O}\langle O_2, L'\langle\lambda y.q\rangle p\rangle$ non-useful implies $\mathbb{O}\langle O_2, L'\langle q[y{\leftarrow}p]\rangle\rangle$ non-useful by Lemma A.3.
- $O_1 \prec O_3$. That is, $O_3 = O_1\langle O_3'\rangle$ for some O_3'. We have then $L\langle O_2\langle v\rangle[x{\leftarrow}v]\rangle = O_3'\langle L'\langle\lambda y.q\rangle p\rangle$. Sub-cases:
  - $O_3' \parallel L$. Then there exists a double context $\mathbb{O}$ such that $\mathbb{O}\langle\langle\cdot\rangle, L'\langle\lambda y.q\rangle p\rangle = L$ and $\mathbb{O}\langle O_2\langle v\rangle[x{\leftarrow}v], \langle\cdot\rangle\rangle = O_3'$. Let $L'' := \mathbb{O}\langle\langle\cdot\rangle, L'\langle q[y{\leftarrow}p]\rangle\rangle$. Then:

$$\begin{array}{ccc} t = O_1\langle O_2\langle x\rangle[x{\leftarrow}\overbrace{\mathbb{O}\langle v, L'\langle\lambda y.q\rangle p\rangle}^{=L\langle v\rangle}]\rangle & \xrightarrow{\mathsf{oe_{nu}}} & O_1\langle\overbrace{\mathbb{O}\langle O_2\langle v\rangle[x{\leftarrow}v], L'\langle\lambda y.q\rangle p\rangle}^{=L\langle O_2\langle v\rangle[x{\leftarrow}v]\rangle}\rangle = r \\ \vdots_{\mathsf{om}} & & \downarrow_{\mathsf{om}} \\ O_1\langle O_2\langle x\rangle[x{\leftarrow}\underbrace{\mathbb{O}\langle v, L'\langle q[y{\leftarrow}p]\rangle\rangle}_{=L''\langle v\rangle}]\rangle & \xrightarrow{\mathsf{oe_{nu}}} & O_1\langle\underbrace{\mathbb{O}\langle O_2\langle v\rangle[x{\leftarrow}v], L'\langle q[y{\leftarrow}p]\rangle\rangle}_{=L''\langle O_2\langle v\rangle[x{\leftarrow}v]\rangle}\rangle = u \end{array}$$

    and clearly the bottom step is non-useful because the top step is.
  - $L \prec O_3'$. Then in fact $L\langle\langle\cdot\rangle[x{\leftarrow}v]\rangle \prec O_3'$, that is, $O_3' = L\langle O_4[x{\leftarrow}v]\rangle$ for some O_4. We have then

$O_2\langle v\rangle = O_4\langle L'\langle\lambda y.q\rangle p\rangle$. Sub-cases:

* $O_4 \parallel O_2$. Then there exists a double context $\mathbb{O}$ such that $\mathbb{O}\langle\langle\cdot\rangle, L'\langle\lambda y.q\rangle p\rangle = O_2$ and $\mathbb{O}\langle v, \langle\cdot\rangle\rangle = O_4$. Then:

$$\begin{array}{ccc} t = O_1\langle\mathbb{O}\langle x, L'\langle\lambda y.q\rangle p\rangle[x{\leftarrow}L\langle v\rangle]\rangle & \xrightarrow{\mathsf{oe_{nu}}} & O_1\langle L\langle\mathbb{O}\langle v, L'\langle\lambda y.q\rangle p\rangle[x{\leftarrow}v]\rangle\rangle = r \\ \Big\downarrow{\scriptstyle \mathsf{om}} & & \Big\downarrow{\scriptstyle \mathsf{om}} \\ O_1\langle\mathbb{O}\langle x, L'\langle q[y{\leftarrow}p]\rangle\rangle[x{\leftarrow}L\langle v\rangle]\rangle & \xrightarrow{\mathsf{oe_{nu}}} & O_1\langle L\langle\mathbb{O}\langle v, L'\langle q[y{\leftarrow}p]\rangle\rangle[x{\leftarrow}v]\rangle\rangle = u \end{array}$$

  Where the bottom side of the diagram is non-useful because $O_1\langle\mathbb{O}\langle\langle\cdot\rangle, L'\langle\lambda y.q\rangle p\rangle\rangle$ non-useful implies $O_1\langle\mathbb{O}\langle\langle\cdot\rangle, L'\langle q[y{\leftarrow}p]\rangle\rangle\rangle$ non-useful by Lemma A.3.

* $O_2 \prec O_4$. Impossible, because then $L'\langle\lambda y.q\rangle p$ would occur in v, that is, under abstraction, against the fact that O_3 is open.

* $O_4 \prec O_2$. Sub-cases:

(a) $O_2 = O_4\langle L'\langle\lambda y.q\rangle O_5\rangle$ for some O_5. Then:

$$\begin{array}{ccc} t = O_1\langle O_4\langle L'\langle\lambda y.q\rangle O_5\langle x\rangle\rangle[x{\leftarrow}L\langle v\rangle]\rangle & \xrightarrow{\mathsf{oe_{nu}}} & O_1\langle L\langle O_4\langle L'\langle\lambda y.q\rangle O_5\langle v\rangle\rangle[x{\leftarrow}v]\rangle\rangle = r \\ \Big\downarrow{\scriptstyle \mathsf{om}} & & \Big\downarrow{\scriptstyle \mathsf{om}} \\ O_1\langle O_4\langle L'\langle q[y{\leftarrow}O_5\langle x\rangle]\rangle\rangle[x{\leftarrow}L\langle v\rangle]\rangle & \xrightarrow{\mathsf{oe_{nu}}} & O_1\langle L\langle O_4\langle L'\langle q[y{\leftarrow}O_5\langle v\rangle]\rangle\rangle[x{\leftarrow}v]\rangle\rangle = u \end{array}$$

  By Lemma 5.2, $L'\langle\lambda y.q\rangle O_5$ non-useful implies $L'\langle q[y{\leftarrow}O_5]\rangle$ non-useful. Then by Lemma 5.4, $O_1\langle O_4\langle L'\langle\lambda y.q\rangle O_5\rangle\rangle$ non-useful implies $O_1\langle O_4\langle L'\langle q[y{\leftarrow}O_5]\rangle\rangle\rangle$ non-useful, justifying the bottom step of the diagram.

(b) $O_2 = O_4\langle O_5 p\rangle$ for some O_5 such that $O_5\langle v\rangle = L'\langle\lambda y.q\rangle$. Sub-cases:

- $L' \prec O_5$ or $O_5 \prec L'$. Then necessarily $L' = O_5$ and $v = \lambda y.q$, but this is impossible, because then the step:

$$t = O_1\langle O_4\langle L'\langle x\rangle p\rangle[x{\leftarrow}L\langle\lambda y.q\rangle]\rangle \to_{\mathsf{oe_{nu}}} O_1\langle O_4\langle L'\langle\lambda y.q\rangle p\rangle[x{\leftarrow}L\langle\lambda y.q\rangle]\rangle = r$$

  is useful, against hypothesis.

- $L' \parallel O_5$. Then there exists a double context $\mathbb{O}$ such that $\mathbb{O}\langle\langle\cdot\rangle, \lambda y.q\rangle = O_5$ and $\mathbb{O}\langle v, \langle\cdot\rangle\rangle = L'$. Let $L'' := \mathbb{O}\langle x, \langle\cdot\rangle\rangle$. Then:

$$\begin{array}{ccc} t = O_1\langle O_4\langle \overbrace{\mathbb{O}\langle x, \lambda y.q\rangle}^{L''\langle\lambda y.q\rangle} p\rangle[x{\leftarrow}L\langle v\rangle]\rangle & \xrightarrow{\mathsf{oe_{nu}}} & O_1\langle L\langle O_4\langle \overbrace{\mathbb{O}\langle v, \lambda y.q\rangle}^{L'\langle\lambda y.q\rangle} p\rangle[x{\leftarrow}v]\rangle\rangle = r \\ \Big\downarrow{\scriptstyle \mathsf{om}} & & \Big\downarrow{\scriptstyle \mathsf{om}} \\ O_1\langle O_4\langle \underbrace{\mathbb{O}\langle x, q[y{\leftarrow}p]\rangle}_{L''\langle q[y{\leftarrow}p]\rangle}\rangle[x{\leftarrow}L\langle v\rangle]\rangle & \xrightarrow{\mathsf{oe_{nu}}} & O_1\langle L\langle O_4\langle \underbrace{\mathbb{O}\langle v, q[y{\leftarrow}p]\rangle}_{L'\langle q[y{\leftarrow}p]\rangle}\rangle[x{\leftarrow}v]\rangle\rangle = u \end{array}$$

  Let us show that the bottom step is indeed non-useful. Since the top step is non-useful, $O_1\langle O_4\langle\mathbb{O}\langle\langle\cdot\rangle, \lambda y.q\rangle p\rangle\rangle$ is non-useful. Therefore, $\mathbb{O}\langle\langle\cdot\rangle, \lambda y.q\rangle$ is non-useful. Then by Lemmas A.3 and 5.4, $O_1\langle O_4\langle\mathbb{O}\langle\langle\cdot\rangle, q[y{\leftarrow}p]\rangle\rangle\rangle$ is non-useful.

· $O_3' \prec L$. Since inside O_3' there is the application $L'\langle\lambda y.q\rangle p$, it can only be $L = O_3'$. But then the case is impossible, because inside L there is $O_2\langle v\rangle[x{\leftarrow}v]$, which is not an application.

• $O_3 \prec O_1$. Then in fact $O_3\langle\langle\cdot\rangle p\rangle \prec O_1$, that is, $O_1 = O_3\langle O_1' p\rangle$ for some O_1'. We have then $O_1'\langle L\langle O_2\langle v\rangle[x{\leftarrow}v]\rangle\rangle = L'\langle\lambda y.q\rangle$. Sub-cases:

· $O_1' \parallel L'$. Then there exists a double context $\mathbb{O}$ such that $\mathbb{O}\langle\lambda y.q, \langle\cdot\rangle\rangle = O_1'$ and $\mathbb{O}\langle\langle\cdot\rangle, O_2\langle x\rangle[v{\leftarrow}L\langle v\rangle]\rangle = L'$. Let $L'' := \mathbb{O}\langle\langle\cdot\rangle, O_2\langle x\rangle[x{\leftarrow}L\langle v\rangle]\rangle$. The diagram closes as follows:

$$\begin{array}{ccc} t = O_3\langle \overbrace{\mathbb{O}\langle\lambda y.q, O_2\langle x\rangle[x{\leftarrow}L\langle v\rangle]\rangle}^{=L''\langle\lambda y.q\rangle} p\rangle & \xrightarrow{\mathsf{oe_{nu}}} & O_3\langle \overbrace{\mathbb{O}\langle\lambda y.q, L\langle O_2\langle v\rangle[x{\leftarrow}v]\rangle\rangle}^{=L'\langle\lambda y.q\rangle} p\rangle = r \\ \Big\downarrow{\scriptstyle \mathsf{om}} & & \Big\downarrow{\scriptstyle \mathsf{om}} \\ O_3\langle \underbrace{\mathbb{O}\langle q[y{\leftarrow}p], O_2\langle x\rangle[x{\leftarrow}L\langle v\rangle]\rangle}_{=L''\langle q[y{\leftarrow}p]\rangle}\rangle & \xrightarrow{\mathsf{oe_{nu}}} & O_3\langle \underbrace{\mathbb{O}\langle q[y{\leftarrow}p], L\langle O_2\langle v\rangle[x{\leftarrow}v]\rangle\rangle}_{=L'\langle q[y{\leftarrow}p]\rangle}\rangle = u \end{array}$$

  Let us show that the bottom step is indeed non-useful. Since the top step is non-useful, $\mathbb{O}\langle\lambda y.q, O_2\rangle$ is non-useful. Then $O := \mathbb{O}\langle q[y{\leftarrow}p], O_2\rangle$ is non-useful by Lemma A.3. Since O

is not a substitution context, plugging it in O_3 gives a non-useful context by Lemma 5.4.
- $L' \prec O_1'$. This case is impossible, because inside L' there is $\lambda y.q$ so that it must be that $O_1' = L'$ but then the content $O_2\langle v\rangle[x\leftarrow v]$ of O_1' does not coincide with the content of L'.
- $O_1' \prec L'$. Then $L' = O_1'\langle L''\rangle$ for some L''. Cases of L'' and L:
  * $L'' \prec L$. This case is impossible, for the same reason as case $L' \prec O_1'$ above.
  * $L \prec L''$. Then in fact $L\langle\langle\cdot\rangle[x\leftarrow v]\rangle \prec L''$, because L'' contains an abstraction. Then $L'' = L\langle L'''[x\leftarrow v]\rangle$ for some L'''. Now, one should analyze the various possibilities for L''' and O_2, but such an analysis is an instance of what is done above for O_4 and O_2.

(ii) $r = O_3\langle O_4\langle y\rangle[y\leftarrow L'\langle v'\rangle]\rangle \to_{\mathsf{oe_u}} O_3\langle L'\langle O_4\langle v'\rangle[y\leftarrow v']\rangle\rangle = u$ where v' is an abstraction.
The situation is then as follows:

$$\begin{array}{ccc} t = O_1\langle O_2\langle x\rangle[x\leftarrow L\langle v\rangle]\rangle & \xrightarrow{\mathsf{oe_{nu}}} & O_1\langle L\langle O_2\langle v\rangle[x\leftarrow v]\rangle\rangle = r = O_3\langle O_4\langle y\rangle[y\leftarrow L'\langle v'\rangle]\rangle \\ & & \downarrow{\scriptstyle \mathsf{oe_u}} \\ & & O_3\langle L'\langle O_4\langle v'\rangle[y\leftarrow v']\rangle\rangle = u \end{array}$$

Cases of the positioning of O_3 with respect to the other shape of r, namely $O_1\langle L\langle O_2\langle v\rangle[x\leftarrow v]\rangle\rangle$:
- $O_3 \parallel O_1$. Then there exists a double context such that $\mathbb{O}\langle\langle\cdot\rangle, L\langle O_2\langle v\rangle[x\leftarrow v]\rangle\rangle = O_3$ and $\mathbb{O}\langle O_4\langle y\rangle[y\leftarrow L'\langle v'\rangle], \langle\cdot\rangle\rangle = O_1$. We have:

$$\begin{array}{ccc} t = \mathbb{O}\langle O_4\langle y\rangle[y\leftarrow L'\langle v'\rangle], O_2\langle x\rangle[x\leftarrow L\langle v\rangle]\rangle & \xrightarrow{\mathsf{oe_{nu}}} & \mathbb{O}\langle O_4\langle y\rangle[y\leftarrow L'\langle v'\rangle], L\langle O_2\langle v\rangle[x\leftarrow v]\rangle\rangle = r \\ \downarrow{\scriptstyle \mathsf{oe_u}} & & \downarrow{\scriptstyle \mathsf{oe_u}} \\ \mathbb{O}\langle L'\langle O_4\langle v'\rangle[y\leftarrow v']\rangle, O_2\langle x\rangle[x\leftarrow L\langle v\rangle]\rangle & \xrightarrow{\mathsf{oe_{nu}}} & \mathbb{O}\langle L'\langle O_4\langle v'\rangle[y\leftarrow v']\rangle, L\langle O_2\langle v\rangle[x\leftarrow v]\rangle\rangle = u \end{array}$$

Let us show that the left step is indeed useful. Let $\mathbb{O}' = \mathbb{O}\langle O_4, \langle\cdot\rangle\rangle$. Since the right step is useful, $\mathbb{O}'\langle\langle\cdot\rangle, L\langle O_2\langle v\rangle[x\leftarrow v]\rangle\rangle$ is useful. Then by Lemma A.3, $\mathbb{O}'\langle\langle\cdot\rangle, O_2\langle x\rangle[x\leftarrow L\langle v\rangle]\rangle$ is useful. Now let us show that the bottom step is indeed non-useful. Let $\mathbb{O}'' = \mathbb{O}\langle\langle\cdot\rangle, O_2\rangle$. Since the top step is non-useful, $\mathbb{O}''\langle O_4\langle y\rangle[y\leftarrow L'\langle v'\rangle], \langle\cdot\rangle\rangle$ is non-useful. Then by Lemma A.3, $\mathbb{O}''\langle L'\langle O_4\langle v'\rangle[y\leftarrow v']\rangle, \langle\cdot\rangle\rangle$ is non-useful.
- $O_1 \prec O_3$. That is, $O_3 = O_1\langle O_5\rangle$ for some O_5. We have then $L\langle O_2\langle v\rangle[x\leftarrow v]\rangle = O_5\langle O_4\langle y\rangle[y\leftarrow L'\langle v'\rangle]\rangle$. Sub-cases:
  - $O_5 \parallel L$. Then there exists a double context $\mathbb{O}$ such that $\mathbb{O}\langle\langle\cdot\rangle, O_4\langle y\rangle[y\leftarrow L'\langle v'\rangle]\rangle = L$ and $\mathbb{O}\langle O_2\langle v\rangle[x\leftarrow v], \langle\cdot\rangle\rangle = O_5$. Let $L'' := \mathbb{O}\langle\langle\cdot\rangle, L'\langle O_4\langle v'\rangle[y\leftarrow v']\rangle\rangle$. Then:

$$\begin{array}{ccc} t = O_1\langle O_2\langle x\rangle[x\leftarrow \mathbb{O}\langle v, O_4\langle y\rangle[y\leftarrow L'\langle v'\rangle]\rangle]\rangle & \xrightarrow{\mathsf{oe_{nu}}} & O_1\langle \mathbb{O}\langle O_2\langle v\rangle[x\leftarrow v], O_4\langle y\rangle[y\leftarrow L'\langle v'\rangle]\rangle\rangle = r \\ \downarrow{\scriptstyle \mathsf{oe_u}} & & \downarrow{\scriptstyle \mathsf{oe_u}} \\ O_1\langle O_2\langle x\rangle[x\leftarrow \underbrace{\mathbb{O}\langle v, L'\langle O_4\langle v'\rangle[y\leftarrow v']\rangle\rangle}_{L''\langle v\rangle}]\rangle & \xrightarrow{\mathsf{oe_{nu}}} & O_1\langle \underbrace{\mathbb{O}\langle O_2\langle v\rangle[x\leftarrow v], L'\langle O_4\langle v'\rangle[y\leftarrow v']\rangle\rangle}_{L''\langle O_2\langle v\rangle[x\leftarrow v]\rangle}\rangle = u \end{array}$$

The bottom step is clearly non-useful as $O_1\langle O_2\rangle$ is non-useful by assumption. Let us show that the left step is indeed useful. Since the right step is useful, $O_1\langle\mathbb{O}\langle O_2\langle v\rangle[x\leftarrow v], O_4\rangle\rangle$ is useful. Then by Lemma 5.4, $\mathbb{O}\langle O_2\langle v\rangle[x\leftarrow v], O_4\rangle$ is useful (it cannot be a substitution context since $\mathbb{O}\langle\langle\cdot\rangle, O_4\langle y\rangle[y\leftarrow L'\langle v'\rangle]\rangle = L$). Then by Lemma A.3, $\mathbb{O}\langle v, O_4\rangle$ is useful, and so is $O_1\langle O_2\langle x\rangle[x\leftarrow\mathbb{O}\langle v, O_4\rangle]\rangle$.
  - $L = O_5$. We have $O_2\langle v\rangle[x\leftarrow v] = O_4\langle y\rangle[y\leftarrow L'\langle v'\rangle]$. Therefore, $v' = v$, $L' = \langle\cdot\rangle$, $x = y$, and $O_2\langle v\rangle = O_4\langle x\rangle$. Then there exists a double context $\mathbb{O}$ such that $\mathbb{O}\langle\langle\cdot\rangle, x\rangle = O_2$ and $\mathbb{O}\langle v, \langle\cdot\rangle\rangle = O_4$. Then:

$$\begin{array}{ccc} t = O_1\langle\mathbb{O}\langle x, x\rangle[x\leftarrow L\langle v\rangle]\rangle & \xrightarrow{\mathsf{oe_{nu}}} & O_1\langle L\langle\mathbb{O}\langle v, x\rangle[x\leftarrow v]\rangle\rangle = r \\ \downarrow{\scriptstyle \mathsf{oe_u}} & & \downarrow{\scriptstyle \mathsf{oe_u}} \\ O_1\langle L\langle\mathbb{O}\langle x, v\rangle[x\leftarrow v]\rangle\rangle & \xrightarrow{\mathsf{oe_{nu}}} & O_1\langle L\langle\mathbb{O}\langle v, v\rangle[x\leftarrow v]\rangle\rangle = u \end{array}$$

Let us show that the left step is indeed useful. Since the right step is useful, $O_1\langle L\langle\mathbb{O}\langle v, \langle\cdot\rangle\rangle\rangle\rangle$ is useful. Then by Lemmas A.3 and 5.4, $O_1\langle\mathbb{O}\langle x, \langle\cdot\rangle\rangle\rangle$ is useful. Now let us show that the bottom step is indeed non-useful. Since the top step is non-useful, $O_1\langle\mathbb{O}\langle\langle\cdot\rangle, x\rangle\rangle$ is non-useful. Then by Lemmas A.3 and 5.4, $O_1\langle L\langle\mathbb{O}\langle\langle\cdot\rangle, v\rangle\rangle\rangle$ is non-useful.

· $L \prec O_5$ and $L \neq O_5$. Then in fact $L\langle\langle\cdot\rangle[x{\leftarrow}v]\rangle \prec O_5$. That is, $O_5 = L\langle O_6[x{\leftarrow}v]\rangle$ for some O_6. We have $O_2\langle v\rangle = O_6\langle O_4\langle y\rangle[y{\leftarrow}L'\langle v'\rangle]\rangle$. Sub-cases:

∗ $O_6 \parallel O_2$. Then there exists a double context $\mathbb{O}$ such that $\mathbb{O}\langle\langle\cdot\rangle, v\rangle = O_6$ and $\mathbb{O}\langle O_4\langle y\rangle[y{\leftarrow}L'\langle v'\rangle], \langle\cdot\rangle\rangle = O_2$. We have:

$$\begin{array}{ccc} t = O_1\langle\mathbb{O}\langle O_4\langle y\rangle[y{\leftarrow}L'\langle v'\rangle], x\rangle[x{\leftarrow}L\langle v\rangle]\rangle & \xrightarrow{\mathsf{oe_{nu}}} & O_1\langle L\langle\mathbb{O}\langle O_4\langle y\rangle[y{\leftarrow}L'\langle v'\rangle], v\rangle[x{\leftarrow}v]\rangle\rangle = r \\ \Big\downarrow{\scriptstyle \mathsf{oe_u}} & & \Big\downarrow{\scriptstyle \mathsf{oe_u}} \\ O_1\langle\mathbb{O}\langle L'\langle O_4\langle v'\rangle[y{\leftarrow}v']\rangle, x\rangle[x{\leftarrow}L\langle v\rangle]\rangle & \xrightarrow{\mathsf{oe_{nu}}} & O_1\langle L\langle\mathbb{O}\langle L'\langle O_4\langle v'\rangle[y{\leftarrow}v']\rangle, v\rangle[x{\leftarrow}v]\rangle\rangle = u \end{array}$$

Let us show that the left step is indeed useful. Since the right step is useful, $O_1\langle L\langle\mathbb{O}\langle O_4, v\rangle\rangle[x{\leftarrow}v]\rangle$ is useful. By Lemma 5.4, $O_1\langle\mathbb{O}\langle O_4, v\rangle\rangle$ is useful. Then by Lemmas A.3 and 5.4, a $O_1\langle\mathbb{O}\langle O_4, x\rangle[x{\leftarrow}L\langle v\rangle]\rangle$ is useful. Now let us show that the bottom step is indeed non-useful. Since the top step is non-useful, $O_1\langle\mathbb{O}\langle O_4\langle y\rangle[y{\leftarrow}L'\langle v'\rangle], \langle\cdot\rangle\rangle\rangle$ is non-useful. Then by Lemmas A.3 and 5.4, $O_1\langle\mathbb{O}\langle L'\langle O_4\langle v'\rangle[y{\leftarrow}v']\rangle, \langle\cdot\rangle\rangle\rangle$ is non-useful.

∗ $O_2 \prec O_6$. Impossible, because then $O_4\langle y\rangle[y{\leftarrow}L'\langle v'\rangle]$ would occur in v, that is, under abstraction, against the fact that O_3 is open.

∗ $O_6 \prec O_2$. Then $O_2 = O_6\langle O_7\rangle$ for some O_7. We have $O_7\langle v\rangle = O_4\langle y\rangle[y{\leftarrow}L'\langle v'\rangle]$. Sub-cases:

(a) v is a subterm of O_4. Then there exists a double context $\mathbb{O}$ such that $\mathbb{O}\langle\langle\cdot\rangle, y\rangle[y{\leftarrow}L'\langle v'\rangle] = O_7$ and $\mathbb{O}\langle v, \langle\cdot\rangle\rangle = O_4$. We have:

$$\begin{array}{ccc} t = O_1\langle O_6\langle\mathbb{O}\langle x, y\rangle[y{\leftarrow}L'\langle v'\rangle]\rangle[x{\leftarrow}L\langle v\rangle]\rangle & \xrightarrow{\mathsf{oe_{nu}}} & O_1\langle L\langle O_6\langle\mathbb{O}\langle v, y\rangle[y{\leftarrow}L'\langle v'\rangle]\rangle[x{\leftarrow}v]\rangle\rangle = r \\ \Big\downarrow{\scriptstyle \mathsf{oe_u}} & & \Big\downarrow{\scriptstyle \mathsf{oe_u}} \\ O_1\langle O_6\langle L'\langle\mathbb{O}\langle x, v'\rangle[y{\leftarrow}v']\rangle\rangle[x{\leftarrow}L\langle v\rangle]\rangle & \xrightarrow{\mathsf{oe_{nu}}} & O_1\langle L\langle O_6\langle L'\langle\mathbb{O}\langle v, v'\rangle[y{\leftarrow}v']\rangle\rangle[x{\leftarrow}v]\rangle\rangle = u \end{array}$$

Let us show that the left step indeed useful. Since the right step is useful, $O_1\langle L\langle O_6\langle\mathbb{O}\langle v, \langle\cdot\rangle\rangle\rangle[x{\leftarrow}v]\rangle\rangle$ is useful. By Lemma 5.4, $O_1\langle O_6\langle\mathbb{O}\langle v, \langle\cdot\rangle\rangle\rangle\rangle$ is useful. Then by Lemmas A.3 and 5.4, $O_1\langle O_6\langle\mathbb{O}\langle x, \langle\cdot\rangle\rangle\rangle[x{\leftarrow}L\langle v\rangle]\rangle$ is useful. Now let us show that the bottom step is indeed non-useful. Since the top step is non-useful, $O_1\langle O_6\langle\mathbb{O}\langle\langle\cdot\rangle, y\rangle[y{\leftarrow}L'\langle v'\rangle]\rangle\rangle$ is non-useful. By Lemma 5.4, $O_1\langle O_6\langle\mathbb{O}\langle\langle\cdot\rangle, y\rangle\rangle\rangle$ is non-useful. Then by Lemmas A.3 and 5.4, $O_1\langle O_6\langle L'\langle\mathbb{O}\langle\langle\cdot\rangle, v'\rangle[y{\leftarrow}v']\rangle\rangle\rangle$ is non-useful.

(b) v is a subterm of L'. Then there exists a double context $\mathbb{O}$ such that $O_4\langle y\rangle[y{\leftarrow}\mathbb{O}\langle\langle\cdot\rangle, v'\rangle] = O_7$ and $\mathbb{O}\langle v, \langle\cdot\rangle\rangle = L'$. By Lemma A.3, $\mathbb{O}\langle x, \langle\cdot\rangle\rangle = L''$ for some L''. Then:

$$\begin{array}{ccc} t = O_1\langle O_6\langle O_4\langle y\rangle[y{\leftarrow}\mathbb{O}\langle x, v'\rangle]\rangle[x{\leftarrow}L\langle v\rangle]\rangle & \xrightarrow{\mathsf{oe_{nu}}} & O_1\langle L\langle O_6\langle O_4\langle y\rangle[y{\leftarrow}\mathbb{O}\langle v, v'\rangle]\rangle[x{\leftarrow}v]\rangle\rangle = r \\ \Big\downarrow{\scriptstyle \mathsf{oe_u}} & & \Big\downarrow{\scriptstyle \mathsf{oe_u}} \\ O_1\langle O_6\langle\mathbb{O}\langle x, O_4\langle v'\rangle[y{\leftarrow}v']\rangle\rangle[x{\leftarrow}L\langle v\rangle]\rangle & \xrightarrow{\mathsf{oe_{nu}}} & O_1\langle L\langle O_6\langle\mathbb{O}\langle v, O_4\langle v'\rangle[y{\leftarrow}v']\rangle\rangle[x{\leftarrow}v]\rangle\rangle = u \end{array}$$

Let us show that the left step is indeed useful. Since the right step is useful, $O_1\langle L\langle O_6\langle O_4\rangle[x{\leftarrow}v]\rangle\rangle$ is useful. By Lemma 5.4, $O_1\langle O_6\langle O_4\rangle[x{\leftarrow}L\langle v\rangle]\rangle$ is useful. Now let us show that the bottom step is indeed non-useful. Since the top step is non-useful, $O_1\langle O_6\langle O_4\langle v'\rangle[v'{\leftarrow}\mathbb{O}\langle\langle\cdot\rangle, v'\rangle]\rangle\rangle$ is non-useful. Then $\mathbb{O}\langle\langle\cdot\rangle, v'\rangle$ is non-useful. It is clear that $\mathbb{O}\langle\langle\cdot\rangle, v'\rangle$ is not a substitution context as $\mathbb{O}\langle v, \langle\cdot\rangle\rangle = L'$. Then by Lemmas A.3 and 5.4, $O_1\langle O_6\langle\mathbb{O}\langle\langle\cdot\rangle, O_4\langle v'\rangle[y{\leftarrow}v']\rangle\rangle\rangle$ is non-useful.

(c) $v = v'$ with $O_7 = O_4\langle y\rangle[y{\leftarrow}L']$. We have:

$$\begin{array}{ccc} t = O_1\langle O_6\langle O_4\langle y\rangle[y{\leftarrow}L'\langle x\rangle]\rangle[x{\leftarrow}L\langle v\rangle]\rangle & \xrightarrow{\mathsf{oe_{nu}}} & O_1\langle L\langle O_6\langle O_4\langle y\rangle[y{\leftarrow}L'\langle v\rangle]\rangle[x{\leftarrow}v]\rangle\rangle = r \\ \Big\downarrow{\scriptstyle \mathsf{oe_{var}}} & & \\ O_1\langle O_6\langle L'\langle O_4\langle x\rangle[y{\leftarrow}x]\rangle\rangle[x{\leftarrow}L\langle v\rangle]\rangle & & \Big\downarrow{\scriptstyle \mathsf{oe_u}} \\ \Big\downarrow{\scriptstyle \mathsf{oe_u}} & & \\ O_1\langle L\langle O_6\langle L'\langle O_4\langle v\rangle[y{\leftarrow}x]\rangle\rangle[x{\leftarrow}v]\rangle\rangle & \xrightarrow{\mathsf{oe_{nu}}} & O_1\langle L\langle O_6\langle L'\langle O_4\langle v\rangle[y{\leftarrow}v]\rangle\rangle[x{\leftarrow}v]\rangle\rangle = u \end{array}$$

Let us show that the second left step is indeed useful. Since the right step is useful, $O_1\langle L\langle O_6\langle O_4\rangle[x{\leftarrow}v]\rangle\rangle$ is useful. By Lemma 5.4, $O_1\langle O_6\langle O_4\rangle\rangle$ is useful. Then, by Lemma 5.4 again, $O_1\langle O_6\langle L'\langle O_4[y{\leftarrow}x]\rangle\rangle\rangle$ is useful. The bottom step is clearly non-useful by the definition of non-useful contexts.

· $O_5 \prec L$ and $L \neq O_5$. Then in fact $O_5\langle\langle\cdot\rangle[y{\leftarrow}L'\langle v'\rangle]\rangle \prec L$, that is , $L = O_5\langle O_6[y{\leftarrow}L'\langle v'\rangle]\rangle$ for

some O_6. We have $O_6\langle O_2\langle v\rangle[x{\leftarrow}v]\rangle = O_4\langle y\rangle$. Also note that O_5 and O_6 are both substitution contexts. Sub-cases:

* $O_6 \parallel O_4$. Then there exists a double context $\mathbb{O}$ such that $\mathbb{O}\langle\langle\cdot\rangle, y\rangle = O_6$ and $\mathbb{O}\langle O_2\langle v\rangle[x{\leftarrow}v], \langle\cdot\rangle\rangle = O_4$. We have:

$$\begin{array}{ccc} t = O_1\langle O_2\langle x\rangle[x{\leftarrow}O_5\langle\mathbb{O}\langle v, y\rangle[y{\leftarrow}L'\langle v'\rangle]\rangle]\rangle & \xrightarrow{\mathsf{oe_{nu}}} & O_1\langle O_5\langle\mathbb{O}\langle O_2\langle v\rangle[x{\leftarrow}v], y\rangle[y{\leftarrow}L'\langle v'\rangle]\rangle\rangle = r \\ \downarrow{\scriptstyle \mathsf{oe_u}} & & \downarrow{\scriptstyle \mathsf{oe_u}} \\ O_1\langle O_2\langle x\rangle[x{\leftarrow}O_5\langle L'\langle\mathbb{O}\langle v, v'\rangle[y{\leftarrow}v']\rangle\rangle]\rangle & \xrightarrow{\mathsf{oe_{nu}}} & O_1\langle O_5\langle L'\langle\mathbb{O}\langle O_2\langle v\rangle[x{\leftarrow}v], v'\rangle[y{\leftarrow}v']\rangle\rangle\rangle = u \end{array}$$

  Let us show that the left step is indeed useful. Since the right step is useful, $O_1\langle O_5\langle\mathbb{O}\langle O_2\langle v\rangle[x{\leftarrow}v], \langle\cdot\rangle\rangle\rangle\rangle$ is useful. By Lemmas A.3 and 5.4, $O_1\langle O_5\langle\mathbb{O}\langle v, \langle\cdot\rangle\rangle\rangle\rangle$ is useful. Note that $\mathbb{O}\langle v, \langle\cdot\rangle\rangle$ is not a substitution context since $\mathbb{O}\langle\langle\cdot\rangle, y\rangle = O_6$ is a substitution context. Then by Lemma 5.4, $\mathbb{O}\langle v, \langle\cdot\rangle\rangle$ is useful and so is $O_1\langle O_2\langle x\rangle[x{\leftarrow}O_5\langle\mathbb{O}\langle v, \langle\cdot\rangle\rangle\rangle]\rangle$. The bottom step is clearly non-useful since $O_1\langle O_2\rangle$ is non-useful by assumption.
* $O_4 \prec O_6$. Clearly impossible.
* $O_6 \prec O_4$. That is, $O_4 = O_6\langle O_7\rangle$ for some O_7. We have $O_2\langle v\rangle[x{\leftarrow}v] = O_7\langle y\rangle$, which is impossible as y is bound in L.

• $O_3 \prec O_1$. That is, $O_1 = O_3\langle O_5\rangle$ for some O_5. We have $O_5\langle L\langle O_2\langle v\rangle[x{\leftarrow}v]\rangle\rangle = O_4\langle y\rangle[y{\leftarrow}L'\langle v'\rangle]$. Sub-cases:

· $O_5 \parallel \langle\cdot\rangle[y{\leftarrow}L'\langle v'\rangle]$. Then the subterm $L\langle O_2\langle v\rangle[x{\leftarrow}v]\rangle$ has to be a subterm of L', which implies the existence of a double context $\mathbb{O}$ such that $\mathbb{O}\langle L\langle O_2\langle v\rangle[x{\leftarrow}v]\rangle, \langle\cdot\rangle\rangle = L'$ and $O_4\langle y\rangle[y{\leftarrow}\mathbb{O}\langle\langle\cdot\rangle, v'\rangle] = O_5$. We have:

$$\begin{array}{ccc} t = O_3\langle O_4\langle y\rangle[y{\leftarrow}\mathbb{O}\langle O_2\langle x\rangle[x{\leftarrow}L\langle v\rangle], v'\rangle]\rangle & \xrightarrow{\mathsf{oe_{nu}}} & O_3\langle O_4\langle y\rangle[y{\leftarrow}\mathbb{O}\langle L\langle O_2\langle v\rangle[x{\leftarrow}v]\rangle, v'\rangle]\rangle = r \\ \downarrow{\scriptstyle \mathsf{oe_u}} & & \downarrow{\scriptstyle \mathsf{oe_u}} \\ O_3\langle\mathbb{O}\langle O_2\langle x\rangle[x{\leftarrow}L\langle v\rangle], O_4\langle v'\rangle[y{\leftarrow}v']\rangle\rangle & \xrightarrow{\mathsf{oe_{nu}}} & O_3\langle\mathbb{O}\langle L\langle O_2\langle v\rangle[x{\leftarrow}v]\rangle, O_4\langle v'\rangle[y{\leftarrow}v']\rangle\rangle = u \end{array}$$

  The left step is clearly useful as $O_3\langle O_4\rangle$ is useful by assumption. Let us show that the bottom step is indeed non-useful. Since the top step is non-useful, $\mathbb{O}\langle O_2, v'\rangle$ is non-useful. By Lemma A.3, $\mathbb{O}\langle O_2, O_4\langle v'\rangle[y{\leftarrow}v']\rangle$ is non-useful. Moreover, $\mathbb{O}\langle O_2, O_4\langle v'\rangle[y{\leftarrow}v']\rangle$ is not a substitution context since $\mathbb{O}\langle L\langle O_2\langle v\rangle[x{\leftarrow}v]\rangle, \langle\cdot\rangle\rangle = L'$. Then by Lemma 5.4, $O_3\langle\mathbb{O}\langle O_2, O_4\langle v'\rangle[y{\leftarrow}v']\rangle\rangle$ is non-useful.

· $\langle\cdot\rangle[y{\leftarrow}L'\langle v'\rangle] \prec O_5$. Then there exists O_6 such that $O_5 = O_6[y{\leftarrow}L'\langle v'\rangle]$. We have $O_6\langle L\langle O_2\langle v\rangle[x{\leftarrow}v]\rangle\rangle = O_4\langle y\rangle$. Sub-cases:

* $O_6 \parallel O_4$. Then there exists a double context $\mathbb{O}$ such that $\mathbb{O}\langle\langle\cdot\rangle, y\rangle = O_6$ and $\mathbb{O}\langle L\langle O_2\langle v\rangle[x{\leftarrow}v]\rangle, \langle\cdot\rangle\rangle = O_4$. We have:

$$\begin{array}{ccc} t = O_3\langle\mathbb{O}\langle O_2\langle x\rangle[x{\leftarrow}L\langle v\rangle], y\rangle[y{\leftarrow}L'\langle v'\rangle]\rangle & \xrightarrow{\mathsf{oe_{nu}}} & O_3\langle\mathbb{O}\langle L\langle O_2\langle v\rangle[x{\leftarrow}v]\rangle, y\rangle[y{\leftarrow}L'\langle v'\rangle]\rangle = r \\ \downarrow{\scriptstyle \mathsf{oe_u}} & & \downarrow{\scriptstyle \mathsf{oe_u}} \\ O_3\langle L'\langle\mathbb{O}\langle O_2\langle x\rangle[x{\leftarrow}L\langle v\rangle], v'\rangle[y{\leftarrow}v']\rangle\rangle & \xrightarrow{\mathsf{oe_{nu}}} & O_3\langle L'\langle\mathbb{O}\langle L\langle O_2\langle v\rangle[x{\leftarrow}v]\rangle, v'\rangle[y{\leftarrow}v']\rangle\rangle = u \end{array}$$

  Let us show that the left step is indeed useful. Since the right step is useful, $O_3\langle\mathbb{O}\langle L\langle O_2\langle v\rangle[x{\leftarrow}v]\rangle, \langle\cdot\rangle\rangle\rangle$ is useful. By Lemmas A.3 and 5.4, $O_3\langle\mathbb{O}\langle O_2\langle x\rangle[x{\leftarrow}L\langle v\rangle], \langle\cdot\rangle\rangle\rangle$ is useful. Now let us show that the bottom step is indeed non-useful. Since the top step is non-useful, $O_3\langle\mathbb{O}\langle O_2, y\rangle[y{\leftarrow}L'\langle v'\rangle]\rangle$ is non-useful. By Lemmas A.3 and 5.4, $O_3\langle L'\langle\mathbb{O}\langle O_2, v'\rangle[y{\leftarrow}v']\rangle\rangle$ is non-useful.
* $O_4 \prec O_6$. Clearly impossible.
* $O_6 \prec O_4$. That is, $O_4 = O_6\langle O_7\rangle$ for some O_7. We have $L\langle O_2\langle v\rangle[x{\leftarrow}v]\rangle = O_7\langle y\rangle$. Sub-cases:

(a) y is a subterm of L. Then there exists a double context $\mathbb{O}$ such that $\mathbb{O}\langle\langle\cdot\rangle, y\rangle = L$ and $\mathbb{O}\langle O_2\langle v\rangle[x{\leftarrow}v], \langle\cdot\rangle\rangle = O_7$. We have:

$$\begin{array}{ccc} t = O_3\langle O_6\langle O_2\langle x\rangle[x{\leftarrow}\mathbb{O}\langle v, y\rangle]\rangle[y{\leftarrow}L'\langle v'\rangle]\rangle & \xrightarrow{\mathsf{oe_{nu}}} & O_3\langle O_6\langle\mathbb{O}\langle O_2\langle v\rangle[x{\leftarrow}v], y\rangle\rangle[y{\leftarrow}L'\langle v'\rangle]\rangle \\ \downarrow{\scriptstyle \mathsf{oe_u}} & & \downarrow{\scriptstyle \mathsf{oe_u}} \\ O_3\langle L'\langle O_6\langle O_2\langle x\rangle[x{\leftarrow}\mathbb{O}\langle v, v'\rangle]\rangle[y{\leftarrow}v']\rangle\rangle & \xrightarrow{\mathsf{oe_{nu}}} & O_3\langle L'\langle O_6\langle\mathbb{O}\langle O_2\langle v\rangle[x{\leftarrow}v], v'\rangle\rangle[y{\leftarrow}v']\rangle\rangle = u \end{array}$$

  Let us show that the left step is indeed useful. Since the right step is useful, $O_3\langle O_6\langle\mathbb{O}\langle O_2\langle v\rangle[x{\leftarrow}v], \langle\cdot\rangle\rangle\rangle\rangle$ is useful. Note that $\mathbb{O}\langle O_2\langle v\rangle[x{\leftarrow}v], \langle\cdot\rangle\rangle$ is not a substitution context

since $\mathbb{O}\langle\langle\cdot\rangle, y\rangle = L$. Then by Lemma 5.4, $\mathbb{O}\langle O_2\langle v\rangle[x{\leftarrow}v], \langle\cdot\rangle\rangle$ is useful. By Lemma A.3, $\mathbb{O}\langle v, \langle\cdot\rangle\rangle$ is useful, and so is $O_3\langle O_6\langle O_2\langle x\rangle[x{\leftarrow}\mathbb{O}\langle v, \langle\cdot\rangle\rangle]\rangle\rangle$. Now let us show that the bottom step is indeed non-useful. Since the top step is non-useful, $O_3\langle O_6\langle O_2\rangle[y{\leftarrow}L'\langle v'\rangle]\rangle$ is non-useful. Then by Lemma 5.4, $O_3\langle L'\langle O_6\langle O_2\rangle[y{\leftarrow}v']\rangle\rangle$ is non-useful.

(b) y is a subterm of O_2. Then there exists a double context $\mathbb{O}$ such that $\mathbb{O}\langle\langle\cdot\rangle, y\rangle = O_2$ and $L\langle\mathbb{O}\langle v, \langle\cdot\rangle\rangle[x{\leftarrow}v]\rangle = O_7$. We have:

$$\begin{array}{ccc} t = O_3\langle O_6\langle\mathbb{O}\langle x, y\rangle[x{\leftarrow}L\langle v\rangle]\rangle[y{\leftarrow}L'\langle v'\rangle]\rangle & \overset{\mathsf{oe_{nu}}}{\to} & O_3\langle O_6\langle L\langle\mathbb{O}\langle v, y\rangle[x{\leftarrow}v]\rangle\rangle[y{\leftarrow}L'\langle v'\rangle]\rangle = r \\ \Big\downarrow{\scriptstyle \mathsf{oe_u}} & & \Big\downarrow{\scriptstyle \mathsf{oe_u}} \\ O_3\langle L'\langle O_6\langle\mathbb{O}\langle x, v'\rangle[x{\leftarrow}L\langle v\rangle]\rangle[y{\leftarrow}v']\rangle\rangle & \overset{\mathsf{oe_{nu}}}{\to} & O_3\langle L'\langle O_6\langle L\langle\mathbb{O}\langle v, v'\rangle[x{\leftarrow}v]\rangle\rangle[y{\leftarrow}v']\rangle\rangle = u \end{array}$$

Let us show that the left step is indeed useful. Since the right step is useful, $O_3\langle O_6\langle L\langle\mathbb{O}\langle v, \langle\cdot\rangle\rangle[x{\leftarrow}v]\rangle\rangle\rangle$ is useful. Then by Lemmas A.3 and 5.4, $O_3\langle O_6\langle\mathbb{O}\langle x, \langle\cdot\rangle\rangle\rangle[x{\leftarrow}L\langle v\rangle]\rangle$ is useful. Now let us show that the bottom step is indeed non-useful. Since the top step is non-useful, $O_3\langle O_6\langle\mathbb{O}\langle\langle\cdot\rangle, y\rangle\rangle[y{\leftarrow}L'\langle v'\rangle]\rangle$ is non-useful. Then by Lemmas A.3 and 5.4, $O_3\langle L'\langle O_6\langle\mathbb{O}\langle\cdot\rangle v'\rangle[y{\leftarrow}v']\rangle\rangle$ is non-useful.

· $O_5 \prec \langle\cdot\rangle[y{\leftarrow}L'\langle v'\rangle]$. Sub-cases:

* $O_5 = \langle\cdot\rangle$. Then $O_1 = O_3$ and this sub-case is treated in the case where $O_1 \prec O_3$.
* $O_5 = \langle\cdot\rangle[y{\leftarrow}L'\langle v'\rangle]$. This sub-case is treated in the case where $\langle\cdot\rangle[y{\leftarrow}L'\langle v'\rangle] \prec O_5$.

(iii) $r = O_3\langle O_4\langle y\rangle[y{\leftarrow}L'\langle z\rangle]\rangle \to_{\mathsf{oe_{var}}} O_3\langle L'\langle O_4\langle z\rangle[y{\leftarrow}z]\rangle\rangle = u$. We can proceed as in the case of $\mathsf{e_u}$. Note that the most complex form does not exist in this case.

□

Theorem 6.2 (Core Factorization / Postponement of non-useful steps, originally at p. 13) *Let t and u be* VSC *terms. If $d : t \to^*_{\mathsf{o\neg gc}} u$, then $e : t \to^*_{\mathsf{ocore}} \to^*_{\mathsf{oe_{nu}}} u$ with $|e|_{\mathsf{om}} = |d|_{\mathsf{om}}$.*

Proof. Let $|d|_{\mathsf{core}}$ and $|d|_{\mathsf{nu}}$ be the number of core and non-useful steps in d, respectively. We prove the following refined statement: there exist a reduction sequence $e : t \to^*_{\mathsf{ocore}} \to^{|d|_{\mathsf{nu}}}_{\mathsf{oe_{nu}}} u$ with $|e|_{\mathsf{om}} = |d|_{\mathsf{om}}$. By induction on the pair $(|d|_{\mathsf{nu}}, |d|_{\mathsf{core}})$ ordered lexicographically. If d is empty the statement trivially holds by taking e as the empty sequence. If d is non-empty, decompose it as follows:

$$d : \underbrace{t \to^*_{\mathsf{o\neg gc}} r}_{d'} \to_{\mathsf{o\neg gc}} u$$

Cases of $r \to_{\mathsf{o\neg gc}} u$:

(i) $r \to_{\mathsf{oe_{nu}}} u$. Then $|d'|_{\mathsf{nu}} = |d|_{\mathsf{nu}} - 1$. By *i.h.* (first component) applied to d', we obtain:

$$e : t \to^*_{\mathsf{ocore}} \to^{|d|_{\mathsf{nu}}-1}_{\mathsf{oe_{nu}}} r \to_{\mathsf{oe_{nu}}} u$$

which satisfies the statement.

(ii) $r \to_{\mathsf{ocore}} u$. Then $|d'|_{\mathsf{core}} = |d|_{\mathsf{core}} - 1$. By *i.h.* (second component) applied to d', we obtain:

$$t \to^*_{\mathsf{ocore}} \to^{|d|_{\mathsf{nu}}}_{\mathsf{oe_{nu}}} r \to_{\mathsf{ocore}} u$$

If $|d|_{\mathsf{nu}} = 0$ then the statement holds. Otherwise, we isolate the last $\to_{\mathsf{oe_{nu}}}$ step:

$$t \to^*_{\mathsf{ocore}} \to^{|d|_{\mathsf{nu}}-1}_{\mathsf{oe_{nu}}} \to_{\mathsf{oe_{nu}}} r \to_{\mathsf{ocore}} u$$

and apply the local postponement property (Prop. 6.1) to the last two steps, obtaining:

$$t \to^*_{\mathsf{ocore}} \to^{|d|_{\mathsf{nu}}-1}_{\mathsf{oe_{nu}}} \to^+_{\mathsf{ocore}} \to_{\mathsf{oe_{nu}}} u$$

Lastly, we apply the *i.h.* (first component) to the central sequence $\to^{|d|_{\mathsf{nu}}-1}_{\mathsf{oe_{nu}}} \to^+_{\mathsf{ocore}}$, obtaining a sequence that satisfies the statement:

$$e : t \to^*_{\mathsf{ocore}} \to^+_{\mathsf{ocore}} \to^{|d|_{\mathsf{nu}}-1}_{\mathsf{oe_{nu}}} r \to_{\mathsf{oe_{nu}}} u$$

The preservation of multiplicative steps follows from the two *i.h.* and the fact that local postponement

also preserves the number of multiplicative steps.

□

Theorem 6.3 (Termination equivalence of λ_{ovsc} and Core λ_{ovsc} , originally at p. 13)

(i) *t has a diverging $\to_{\mathsf{ovsc}}$ sequence if and only if t has a diverging $\to_{\mathsf{ocore}}$ sequence;*

(ii) *t is $\to_{\mathsf{ovsc}}$-weakly normalizing if and only if t is $\to_{\mathsf{ocore}}$-weakly normalizing.*

Proof.

(i) Direction $\Leftarrow$ is trivial because $\to_{\mathsf{ocore}}$ is a special case of $\to_{\mathsf{ovsc}}$.
For direction $\Rightarrow$, let t have a $\to_{\mathsf{ovsc}}$-diverging reduction sequence d. We prove that t has a $\to_{\mathsf{ocore}}$-diverging sequence e. Consider the finite prefixes $d_n : t \to^*_{\mathsf{ovsc}} u_n$ for $n \in \mathbb{N}$ of d. By local termination (Prop. 3.3), the number of multiplicative steps in d_n tends to infinity when n grows. By postponing first $\to_{\mathsf{ogc}}$ (Prop. 4.5) and then $\to_{\mathsf{oe_{nu}}}$ (Thm. 6.2), all the sequences d_n can be re-organized as sequences $e_n : t \to^*_{\mathsf{ocore}} \to^*_{\mathsf{oe_{nu}}} \to^*_{\mathsf{gc}} u_n$ in a way that preserves the number of multiplicative steps, which are all in the $\to_{\mathsf{ocore}}$-prefix. Then, t is $\to_{\mathsf{ocore}}$-diverging.

(ii) For direction $\Rightarrow$, let $d : t \to^*_{\mathsf{ovsc}} u$ be a reduction sequence to $\to_{\mathsf{ovsc}}$ normal form. By postponing first $\to_{\mathsf{ogc}}$ (Prop. 4.5) and then $\to_{\mathsf{oe_{nu}}}$ (Thm. 6.2), we obtain a reduction sequence $d : t \to^*_{\mathsf{ocore}} r \to^*_{\mathsf{oe_{nu}}} \to^*_{\mathsf{ogc}} u$ for some r. Now, it is easily seen that $\to_{\mathsf{oe_{nu}}}$ and $\to_{\mathsf{ogc}}$ cannot remove $\to_{\mathsf{ocore}}$ redexes. Thus, r is $\to_{\mathsf{ocore}}$-normal.
For direction $\Leftarrow$, let $d : t \to^*_{\mathsf{ocore}} u$ be a reduction sequence to $\to_{\mathsf{ocore}}$ normal form. By local termination, $\to_{\mathsf{oe_{nu}}} \cup \to_{\mathsf{ogc}}$ is strongly normalizing, thus $u \to^*_{\mathsf{ocore}} r$ for some r that is a $\to_{\mathsf{oe_{nu}}} \cup \to_{\mathsf{ogc}}$ normal form. It is easily seen that $\to_{\mathsf{oe_{nu}}}$ and $\to_{\mathsf{ogc}}$ cannot create $\to_{\mathsf{ocore}}$ redexes. Thus, r is $\to_{\mathsf{ovsc}}$-normal.

□

F Proofs from Sect. 7 (Translating λ_{ovsc} to λ_{oxpos} and Simulating Core Steps)

Proposition F.1 (Translation and free variables)

- *For any* VSC *term* t, $\mathsf{fv}(\llbracket t \rrbracket) \subseteq \mathsf{fv}(t)$.
- *For any* VSC *substitution context* L, *if* $\llbracket L \rrbracket = (E, \sigma)$, *then* $\mathsf{fv}(E) \subseteq \mathsf{fv}(L)$ *and* $\mathsf{range}(\sigma) \setminus (\mathsf{dom}(\sigma) \cup \mathsf{bv}(E)) \subseteq \mathsf{fv}(L)$.

Proof. By induction on the translation of terms and substitution contexts. The base cases (the empty context and variables) are trivial. For the inductive cases:

- $\llbracket L[x{\leftarrow}t] \rrbracket = (E'\langle E\{x{\leftarrow}y\}\rangle, \sigma\{x{\leftarrow}y\})$ where $\llbracket L \rrbracket = (E, \sigma)$ and $\llbracket t \rrbracket = E'\langle y \rangle$. Let $z \in \mathsf{fv}(E'\langle E\{x{\leftarrow}y\}\rangle)$. Two cases to consider:
  - $z \in \mathsf{fv}(E)$ and $z \neq x$. By *i.h.*, $z \in \mathsf{fv}(L)$. Therefore, we have $z \in \mathsf{fv}(L[x{\leftarrow}t])$.
  - $z \in \mathsf{fv}(E')$ or ($z = y$ and $y \notin \mathsf{bv}(E')$). Then $z \in \mathsf{fv}(E'\langle y \rangle) = \mathsf{fv}(\llbracket t \rrbracket)$. By *i.h.*, $z \in \mathsf{fv}(t) \subseteq \mathsf{fv}(L[x{\leftarrow}t])$.

  Now let $w \in \mathsf{range}(\sigma\{x{\leftarrow}y\}) \setminus (\mathsf{dom}(\sigma\{x{\leftarrow}y\}) \cup \mathsf{bv}(E'\langle E\{x{\leftarrow}y\}\rangle))$. Two cases to consider:
  - $w \in \mathsf{range}(\sigma)$. Then $w \in (\mathsf{range}(\sigma) \setminus (\mathsf{dom}(\sigma) \cup \mathsf{bv}(E))) \setminus \{x\}$. By *i.h.*, $w \in \mathsf{fv}(L) \setminus \{x\} \subseteq \mathsf{fv}(L[x{\leftarrow}t])$.
  - $w = y$ and $w \notin \mathsf{bv}(E')$. Then $w \in \mathsf{fv}(E'\langle y \rangle)$. By *i.h.*, $w \in \mathsf{fv}(t) \subseteq \mathsf{fv}(L[x{\leftarrow}t])$.
- $\llbracket t[x{\leftarrow}u] \rrbracket = E\langle \llbracket t \rrbracket\{x{\leftarrow}y\} \rangle$ where $\llbracket u \rrbracket = E\langle y \rangle$. Let $z \in \mathsf{fv}(E\langle \llbracket t \rrbracket\{x{\leftarrow}y\} \rangle)$. Two cases to consider:
  - $z \in \mathsf{fv}(\llbracket t \rrbracket)$ and $z \neq x$. By *i.h.*, $z \in \mathsf{fv}(t)$. Therefore, $z \in \mathsf{fv}(t[x{\leftarrow}u])$.
  - $z \in \mathsf{fv}(E)$ or ($z = y$ and $y \notin \mathsf{bv}(E)$). Then $z \in \mathsf{fv}(E\langle y \rangle)$. By *i.h.*, $z \in \mathsf{fv}(u) \subseteq \mathsf{fv}(t[x{\leftarrow}u])$.
- $\llbracket \lambda x.t \rrbracket = y[y{\leftarrow}\lambda x.\llbracket t \rrbracket]$. Then $\mathsf{fv}(\llbracket \lambda x.t \rrbracket) = \mathsf{fv}(\llbracket t \rrbracket) \setminus \{x\} \subseteq \mathsf{fv}(t) \setminus \{x\} = \mathsf{fv}(\lambda x.t)$ by *i.h.*.
- $\llbracket L\langle \lambda x.t \rangle u \rrbracket = E\langle E'\langle y[y{\leftarrow}(\lambda x.\llbracket t \rrbracket\sigma)z] \rangle \rangle$ where $\llbracket L \rrbracket = (E, \sigma)$ and $\llbracket u \rrbracket = E'\langle z \rangle$. Let $w \in \mathsf{fv}(E\langle E'\langle y[y{\leftarrow}(\lambda x.\llbracket t \rrbracket\sigma)z] \rangle \rangle)$. Four cases to consider:
  - $w \in \mathsf{fv}(E')$ or ($w = z$ and $z \notin \mathsf{bv}(E')$). Then $w \in \mathsf{fv}(E'\langle z \rangle)$. By *i.h.*, $w \in \mathsf{fv}(u) \subseteq \mathsf{fv}(L\langle \lambda x.t \rangle u)$.
  - $w \in \mathsf{fv}(E)$. By *i.h.*, $w \in \mathsf{fv}(L) \subseteq \mathsf{fv}(L\langle \lambda x.t \rangle u)$.
  - $w \in \mathsf{fv}(\llbracket t \rrbracket)$ and $w \notin \mathsf{dom}(\sigma) \cup \{x\} = \mathsf{bv}(L) \cup \{x\}$. By *i.h.*, $w \in \mathsf{fv}(t) \setminus (\mathsf{bv}(L) \cup \{x\}) \subseteq \mathsf{fv}(L\langle \lambda x.t \rangle u)$.
  - $w \in \mathsf{range}(\sigma) \setminus (\mathsf{dom}(\sigma) \cup \mathsf{bv}(E))$. By *i.h.*, $w \in \mathsf{fv}(L) \subseteq \mathsf{fv}(L\langle \lambda x.t \rangle u)$.

- $[\![L\langle t\rangle u]\!] = E\langle E'\langle y[y{\leftarrow}xz]\rangle\rangle$ where $[\![L\langle t\rangle]\!] = E\langle x\rangle$ and $[\![u]\!] = E'\langle z\rangle$. Let $w \in \mathsf{fv}(E\langle E'\langle y[y{\leftarrow}xz]\rangle\rangle)$. Two cases to consider:
  - $w \in \mathsf{fv}(E)$ or $(w = x$ and $x \notin \mathsf{bv}(E))$. Then $w \in \mathsf{fv}(E\langle x\rangle)$. By *i.h.*, $w \in \mathsf{fv}(L\langle t\rangle) \subseteq \mathsf{fv}(L\langle t\rangle u)$.
  - $w \in \mathsf{fv}(E')$ or $(w = z$ and $z \notin \mathsf{bv}(E'))$. Then $w \in \mathsf{fv}(E'\langle z\rangle)$. By *i.h.*, $w \in \mathsf{fv}(u) \subseteq \mathsf{fv}(L\langle t\rangle u)$.

□

Lemma 7.2 (Absorption of variable exponentials, originally at p. 14) *Let t and u be* VSC *terms. If $t \to_{\mathsf{oe_{var}}} u$ then $[\![t]\!] = [\![u]\!]$.*

Proof. By induction on $t \to_{\mathsf{oe_{var}}} u$. Cases:

- *Root step*: $O\langle\!\langle x\rangle\!\rangle[x{\leftarrow}L\langle y\rangle] \mapsto_{\mathsf{e_{var}}} L\langle O\langle\!\langle y\rangle\!\rangle[x{\leftarrow}y]\rangle$. Let $[\![L]\!] = (E, \sigma)$. By Lemma 7.1, $[\![L\langle y\rangle]\!] = E\langle y\sigma\rangle$. Then:

$$\begin{aligned}
[\![O\langle\!\langle x\rangle\!\rangle[x{\leftarrow}L\langle y\rangle]]\!] &= E\langle [\![O\langle\!\langle x\rangle\!\rangle]\!]\{x{\leftarrow}y\sigma\}\rangle \\
&= E\langle [\![O\langle\!\langle x\rangle\!\rangle\{x{\leftarrow}y\sigma\}]\!]\rangle \\
&= E\langle [\![O\langle\!\langle y\sigma\rangle\!\rangle\{x{\leftarrow}y\sigma\}]\!]\rangle \\
&= E\langle [\![O\langle\!\langle y\sigma\rangle\!\rangle]\!]\{x{\leftarrow}y\sigma\}\rangle \\
&= E\langle [\![O\langle\!\langle y\rangle\!\rangle]\!]\{x{\leftarrow}y\}\sigma\rangle \\
&= E\langle [\![O\langle\!\langle y\rangle\!\rangle[x{\leftarrow}y]]\!]\sigma\rangle \\
&=_{L.7.1} [\![L\langle O\langle\!\langle y\rangle\!\rangle[x{\leftarrow}y]\rangle]\!]
\end{aligned}$$

- *Inductive cases*: the statement follows immediately from the *i.h.* and the definition of the translation.

□

Lemma 7.3 (Simulation of root multiplicative steps, originally at p. 14) *Let t and u be* VSC *terms. If $t \mapsto_{\mathsf{m}} u$ then $[\![t]\!] \to_{\mathsf{om}_+} [\![u]\!]$.*

Proof. Let $t = L\langle\lambda x.r\rangle q \mapsto_{\mathsf{m}} L\langle r[x{\leftarrow}q]\rangle = u$ and let the translations of L and the sub-terms be $[\![L]\!] = (E, \sigma)$, $[\![q]\!] = E'\langle y\rangle$, and $[\![r]\!] = E''\langle w\rangle$. By Lemma 7.1,

$$[\![L\langle r[x{\leftarrow}q]\rangle]\!] = E\langle [\![r[x{\leftarrow}q]]\!]\sigma\rangle = E\langle E'\langle [\![r]\!]\{x{\leftarrow}y\}\rangle\sigma\rangle = E\langle E'\langle [\![r]\!]\sigma\{x{\leftarrow}y\}\rangle\rangle$$

since $\mathsf{dom}(\sigma) \cap \mathsf{fv}([\![q]\!]) \subseteq \mathsf{bv}(L) \cap \mathsf{fv}(q) = \emptyset$ by Prop. F.1. Then:

$$\begin{aligned}
[\![L\langle\lambda x.r\rangle q]\!] &= E\langle E'\langle z[z{\leftarrow}(\lambda x.[\![r]\!]\sigma)y]\rangle\rangle \\
&= E\langle E'\langle z[z{\leftarrow}(\lambda x.E''\langle w\rangle\sigma)y]\rangle\rangle \\
&= E\langle E'\langle z[z{\leftarrow}(\lambda x.E''\sigma\langle w\sigma\rangle)y]\rangle\rangle \\
&\to_{\mathsf{om}_+} E\langle E'\langle E''\sigma\langle z\{z{\leftarrow}w\sigma\}\rangle\{x{\leftarrow}y\}\rangle\rangle \\
&= E\langle E'\langle E''\sigma\langle w\sigma\rangle\{x{\leftarrow}y\}\rangle\rangle \\
&= E\langle E'\langle E''\langle w\rangle\sigma\{x{\leftarrow}y\}\rangle\rangle \\
&= E\langle E'\langle [\![r]\!]\sigma\{x{\leftarrow}y\}\rangle\rangle \\
&= E\langle E'\langle [\![r]\!]\{x{\leftarrow}y\}\rangle\sigma\rangle \\
&= E\langle [\![r[x{\leftarrow}q]]\!]\sigma\rangle \\
&=_{L.7.1} [\![L\langle r[x{\leftarrow}q]\rangle]\!]
\end{aligned}$$

□

To extend the simulation of root steps to general steps we shall need the following lemma that guarantees that simulating steps in λ_{oxpos} that are obtained by *i.h.* can be extended to the translation of the larger term, despite the term re-arrangement done by the definition of the translation $[\![\,\cdot\,]\!]$.

Lemma F.2 (Contextual lifting of positive rewriting steps) *Let E be such that it does not capture variables of E'', v'', and z, and let $a \in \{\mathsf{m}_+, \mathsf{e}_+, \mathsf{gc}_+\}$.*

(i) *If $E\langle x\rangle \to_{\mathsf{oa}} E'\langle x'\rangle$, then:*
   (a) $E\langle E''\langle y[y\leftarrow xz]\rangle\rangle \to_{\mathsf{oa}} E'\langle E''\langle y[y\leftarrow x'z]\rangle\rangle$,
   (b) $E''\langle E\langle y[y\leftarrow v''x]\rangle\rangle \to_{\mathsf{oa}} E''\langle E'\langle y[y\leftarrow v''x']\rangle\rangle$,
   (c) $E\langle E''\langle z\rangle\{w\leftarrow x\}\rangle \to_{\mathsf{oa}} E'\langle E''\langle z\rangle\{w\leftarrow x'\}\rangle$, *and*
   (d) $E''\langle E\langle x\rangle\{w\leftarrow z\}\rangle \to_{\mathsf{oa}} E''\langle E'\langle x'\rangle\{w\leftarrow z\}\rangle$.

(ii) *If $E\langle x[x\leftarrow v]\rangle \to_{\mathsf{oa}} E'\langle x[x\leftarrow v']\rangle$, then $E\langle E''\langle y[y\leftarrow vz]\rangle\rangle \to_{\mathsf{oa}} E'\langle E''\langle y[y\leftarrow v'z]\rangle\rangle$.*

Proof. Trivial for e_+ and gc_+ as they take place entirely in E. For $a = \mathsf{m}_+$, we prove the second point here (the first point can be treated similarly). It is clear that E is of the form $E_1\langle E_2[w\leftarrow(\lambda x'.E_3\langle y'\rangle)z']\rangle$ and we have

$$E_1\langle E_2\langle x[x\leftarrow v]\rangle[w\leftarrow(\lambda x'.E_3\langle y'\rangle)z']\rangle \to_{\mathsf{om}_+} E_1\langle E_3\langle E_2\langle x[x\leftarrow v]\rangle\{w\leftarrow y'\}\rangle\{x'\leftarrow z'\}\rangle.$$

Therefore, $E' = E_1\langle E_3\langle E_2\{w\leftarrow y'\}\rangle\{x'\leftarrow z'\}\rangle$ and $v' = v\{w\leftarrow y'\}\{x'\leftarrow z'\}$. Then we have:

$$\begin{aligned}
E\langle E''\langle y[y\leftarrow vz]\rangle\rangle &= E_1\langle E_2\langle E''\langle y[y\leftarrow vz]\rangle\rangle[w\leftarrow(\lambda x'.E_3\langle y'\rangle)z']\rangle \\
&\to_{\mathsf{om}_+} E_1\langle E_3\langle E_2\langle E''\langle y[y\leftarrow vz]\rangle\rangle\{w\leftarrow y'\}\rangle\{x'\leftarrow z'\}\rangle \\
&= E'\langle E''\langle y[y\leftarrow vz]\rangle\{w\leftarrow y'\}\{x'\leftarrow z'\}\rangle \\
&= E'\langle E''\langle y[y\leftarrow v'z]\rangle\rangle
\end{aligned}$$

The last equality holds since both w and x' are not free in E'' or z. □

Proposition 7.4 (Simulation of $\to_{\mathsf{om}}$ steps, originally at p. 14) *Let t and u be* VSC *terms and $t \mapsto_{\mathsf{m}} u$.*

(i) *If u is an answer and $\mathsf{usef}(O)$ then $[\![O\langle t\rangle]\!] \to_{\mathsf{om}_+}\to_{\mathsf{oe}_+}\to_{\mathsf{ogc}_+} [\![O\langle u\rangle]\!]$;*

(ii) *Otherwise, $[\![O\langle t\rangle]\!] \to_{\mathsf{om}_+} [\![O\langle u\rangle]\!]$.*

Proof. By induction on O. The base case, for which $O = \langle\cdot\rangle$ and thus O is non-useful and case (*ii*) should hold, is treated in Lemma 7.3. Note that if $t \mapsto_{\mathsf{m}} u$ then t is not an answer. Cases (the first is the interesting / difficult one):

- $O = O't'$. This case is the difficult one because the shape of the translations of $O\langle t\rangle$ and $O\langle u\rangle$ depends on O' and whether u is an answer.
  (i) Let u be an answer and $\mathsf{usef}(O)$. By Lemma 5.2, we have $\mathsf{usef}(O')$ or $\mathsf{sub}(O')$:
  (a) If $\mathsf{usef}(O')$ then $[\![O'\langle t\rangle]\!] \to_{\mathsf{om}_+}\to_{\mathsf{oe}_+}\to_{\mathsf{ogc}_+} [\![O'\langle u\rangle]\!]$ by *i.h.* Let $[\![O'\langle t\rangle]\!] = E_1\langle x\rangle$, $[\![O'\langle u\rangle]\!] = E_4\langle x'\rangle$. Note that $\mathsf{usef}(O')$ implies $\neg\mathsf{sub}(O')$. Thus, whether $O'\langle u\rangle$ is an answer is independent of u and depends only on O', that is, $O'\langle u\rangle$ is an answer if and only if $O'\langle t'\rangle$ is an answer for every t'. Therefore, $O'\langle t\rangle$ and $O'\langle u\rangle$ are either both answers or both non-answers. In both cases, then the statement easily follows from the *i.h.*, lifting step using Lemma F.2.2 if both are answers, and using Lemma F.2.1.(a) if instead they are not.
  (b) If $\mathsf{sub}(O')$ then let us use L for O'. Since $t \mapsto_{\mathsf{m}} u$, for some r and q we have:
  $$t = L'\langle\lambda x.r\rangle q \mapsto_{\mathsf{m}} L'\langle r[x\leftarrow q]\rangle = u$$
  Since u is an answer, we also have $r = L''\langle\lambda y.p\rangle$ for some p, so that:
  $$t = L'\langle\lambda x.L''\langle\lambda y.p\rangle\rangle q \mapsto_{\mathsf{m}} L'\langle L''\langle\lambda y.p\rangle[x\leftarrow q]\rangle = u.$$
  Note that $L\langle u\rangle$ is an answer while $L\langle t\rangle$ is not. This is the tricky case of this proof. Let
  · $[\![L]\!] = (E, \sigma_L)$;
  · $[\![L']\!] = (E', \sigma')$;
  · $[\![L'']\!] = (E'', \sigma'')$ and $[\![r]\!] = [\![L''\langle\lambda y.p\rangle]\!] = E''\langle z'[z'\leftarrow\lambda y.[\![p]\!]\sigma'']\rangle$;
  · $[\![q]\!] = E'''\langle z\rangle$;
  We then have:
  $$\begin{aligned}
  [\![t]\!] &= [\![L'\langle\lambda x.r\rangle q]\!] \\
  &= E'\langle E'''\langle w[w\leftarrow(\lambda x.[\![r]\!]\sigma')z]\rangle\rangle
  \end{aligned}$$

and

$$\begin{aligned}\llbracket L\langle t\rangle\rrbracket &= \llbracket L\langle L'\langle\lambda x.r\rangle q\rangle\rrbracket\\ &= E\langle\llbracket L'\langle\lambda x.r\rangle q\rrbracket\sigma\rangle\\ &= E\langle E'\langle E'''\langle w[w{\leftarrow}(\lambda x.\llbracket r\rrbracket\sigma')z]\rangle\rangle\sigma\rangle\\ &= E\langle E'\sigma\langle E'''\sigma\langle w[w{\leftarrow}(\lambda x.\llbracket r\rrbracket\sigma'\sigma)z]\rangle\rangle\rangle\end{aligned}$$

As well as:

$$\begin{aligned}\llbracket u\rrbracket &= \llbracket L'\langle L''\langle\lambda y.p\rangle[x{\leftarrow}q]\rangle\rrbracket\\ &= E'\langle\llbracket L''\langle\lambda y.p\rangle[x{\leftarrow}q]\rrbracket\sigma'\rangle\\ &= E'\langle E'''\langle\llbracket L''\langle\lambda y.p\rangle\rrbracket\{x{\leftarrow}z\}\rangle\sigma'\rangle\\ &= E'\langle E'''\langle E''\langle y'[y'{\leftarrow}\lambda y.\llbracket p\rrbracket\sigma'']\rangle\{x{\leftarrow}z\}\rangle\sigma'\rangle\\ &= E'\langle E'''\sigma'\langle E''\{x{\leftarrow}z\}\sigma'\langle y'[y'{\leftarrow}\lambda y.\llbracket p\rrbracket\sigma''\{x{\leftarrow}z\}\sigma']\rangle\rangle\rangle\end{aligned}$$

and

$$\begin{aligned}\llbracket L\langle u\rangle\rrbracket &= E\langle\llbracket L'\langle L''\langle\lambda y.p\rangle[x{\leftarrow}q]\rangle\rrbracket\sigma\rangle\\ &= E\langle E'\langle E'''\sigma'\langle E''\{x{\leftarrow}z\}\sigma'\langle y'[y'{\leftarrow}\lambda y.\llbracket p\rrbracket\sigma''\{x{\leftarrow}z\}\sigma']\rangle\rangle\rangle\sigma\rangle\\ &= E\langle E'\sigma\langle E'''\sigma'\sigma\langle E''\{x{\leftarrow}z\}\sigma'\sigma\langle y'[y'{\leftarrow}\lambda y.\llbracket p\rrbracket\sigma''\{x{\leftarrow}z\}\sigma'\sigma]\rangle\rangle\rangle\rangle\end{aligned}$$

Let $\llbracket t'\rrbracket = E^{t'}\langle w'\rangle$. We have:

$$\begin{aligned}\llbracket O'\langle t\rangle t'\rrbracket &= \llbracket L\langle L'\langle\lambda x.r\rangle q\rangle t'\rrbracket\\ &= E\langle E'\sigma\langle E'''\sigma\langle E^{t'}\langle y'[y'{\leftarrow}ww']\rangle[w{\leftarrow}(\lambda x.\llbracket r\rrbracket\sigma'\sigma)z]\rangle\rangle\rangle\\ &= E\langle E'\sigma\langle E'''\sigma\langle E^{t'}\langle y'[y'{\leftarrow}ww']\rangle[w{\leftarrow}(\lambda x.E''\langle z'[z'{\leftarrow}\lambda y.\llbracket p\rrbracket\sigma'']\rangle\sigma'\sigma)z]\rangle\rangle\rangle\\ &= E\langle E'\sigma\langle E'''\sigma\langle E^{t'}\langle y'[y'{\leftarrow}ww']\rangle[w{\leftarrow}(\lambda x.E''\sigma'\sigma\langle z'[z'{\leftarrow}\lambda y.\llbracket p\rrbracket\sigma''\sigma'\sigma]\rangle)z]\rangle\rangle\rangle\\ &\to_{\mathsf{om}_+} E\langle E'\sigma\langle E'''\sigma\langle E''\sigma'\sigma\langle E^{t'}\langle y'[y'{\leftarrow}z'w']\rangle[z'{\leftarrow}\lambda y.\llbracket p\rrbracket\sigma''\sigma'\sigma]\rangle\{x{\leftarrow}z\}\rangle\rangle\rangle\\ &= E\langle E'\sigma\langle E'''\sigma\langle E''\sigma'\sigma\{x{\leftarrow}z\}\langle E^{t'}\langle y'[y'{\leftarrow}z'w']\rangle[z'{\leftarrow}\lambda y.\llbracket p\rrbracket\sigma''\sigma'\sigma\{x{\leftarrow}z\}]\rangle\rangle\rangle\rangle\\ &\to_{\mathsf{oe}_+} E\langle E'\sigma\langle E'''\sigma\langle E''\sigma'\sigma\{x{\leftarrow}z\}\langle E^{t'}\langle y'[y'{\leftarrow}(\lambda y.\llbracket p\rrbracket\sigma''\sigma'\sigma\{x{\leftarrow}z\})w']\rangle[z'{\leftarrow}\lambda y.\llbracket p\rrbracket\sigma''\sigma'\sigma\{x{\leftarrow}z\}]\rangle\rangle\rangle\rangle\\ &\to_{\mathsf{ogc}_+} E\langle E'\sigma\langle E'''\sigma\langle E''\sigma'\sigma\{x{\leftarrow}z\}\langle E^{t'}\langle y'[y'{\leftarrow}(\lambda y.\llbracket p\rrbracket\sigma''\sigma'\sigma\{x{\leftarrow}z\})w']\rangle\rangle\rangle\rangle\rangle\\ &= E\langle E'\sigma\langle E'''\sigma\langle E''\{x{\leftarrow}z\}\sigma'\sigma\langle E^{t'}\langle y'[y'{\leftarrow}(\lambda y.\llbracket p\rrbracket\sigma''\{x{\leftarrow}z\}\sigma'\sigma)w']\rangle\rangle\rangle\rangle\rangle\\ &= \llbracket L\langle L'\langle L''\langle\lambda y.p\rangle[x{\leftarrow}q]\rangle\rangle t'\rrbracket\\ &= \llbracket L\langle u\rangle t'\rrbracket\\ &= \llbracket O'\langle u\rangle t'\rrbracket\end{aligned}$$

(ii) $\mathsf{nusef}(O)$. By Lemma 5.2.i, we have $\mathsf{nusef}(O')$ and $\mathsf{nsub}(O')$. Note that $\mathsf{nsub}(O')$ implies that whether $O'\langle u\rangle$ is an answer is independent of u and depends only on O', that is, $O'\langle u\rangle$ is an answer if and only if $O'\langle t'\rangle$ is an answer for every t'. Therefore, $O'\langle t\rangle$ and $O'\langle u\rangle$ are either both answers or both non-answers. By *i.h.*, $\llbracket O'\langle t\rangle\rrbracket \to_{\mathsf{om}_+} \llbracket O'\langle u\rangle\rrbracket$. In both cases, then the statement easily follows from the *i.h.*, lifting step using Lemma F.2.2 if both are answers, and using Lemma F.2.1.(a) if instead they are not.

(iii) *u is not an answer.* We have that both t and u are not answers, so $O'\langle t\rangle$ is an answer if and only if $O'\langle u\rangle$ is, and the details go as in the previous case.

- $O = t'O'$.
  (i) $\mathsf{usef}(O)$ and u is an answer. By Lemma 5.2.ii, we have $\mathsf{usef}(O')$ and then $\llbracket O'\langle t\rangle\rrbracket \to_{\mathsf{om}_+}\to_{\mathsf{oe}_+}\to_{\mathsf{ogc}_+} \llbracket O'\langle u\rangle\rrbracket$ by *i.h.* The statement easily follows from the *i.h.*, lifting step using Lemma F.2.1.(b).

  (ii) $\mathsf{nusef}(O)$ or u is not an answer. If u is not an answer we can apply the *i.h.* If u is an answer note that then $\mathsf{nusef}(O)$ holds, which implies $\mathsf{nusef}(O')$ by Lemma 5.2.ii, so that we can apply the *i.h.* anyway. Therefore, $\llbracket O'\langle t\rangle\rrbracket \to_{\mathsf{om}_+} \llbracket O'\langle u\rangle\rrbracket$ by *i.h.* Then it goes exactly as Point 1, except that only

one rewriting step (instead of three) is transported by the extension of the evaluation context.

- $O = O'[x\leftarrow t']$.
  (i) $\mathsf{usef}(O)$ and u is an answer. By Lemma 5.2.iii, we have $\mathsf{usef}(O')$ and then $[\![O'\langle t\rangle]\!] \to_{\mathsf{om}_+} \to_{\mathsf{oe}_+} \to_{\mathsf{ogc}_+} [\![O'\langle u\rangle]\!]$ by *i.h.* The statement easily follows from the *i.h.*, lifting step using Lemma F.2.1.(d).
  (ii) $\mathsf{nusef}(O)$ or u is not an answer. If u is not an answer we can apply the *i.h.* If u is an answer note that then $\mathsf{nusef}(O)$ holds, which implies $\mathsf{nusef}(O')$ by Lemma 5.2.iii, so that we can apply the *i.h.* anyway. Therefore, $[\![O'\langle t\rangle]\!] \to_{\mathsf{om}_+} [\![O'\langle u\rangle]\!]$ by *i.h.* Then it goes exactly as Point 1, except that only one rewriting step (instead of three) is transported by the extension of the evaluation context.
- $O = t'[x\leftarrow O']$. As the previous case, except that the use of Lemma 5.2.iii is replaced by Lemma 5.2.iv, and the use of Lemma F.2.1.(d) is replaced by Lemma F.2.1.(c).

□

Simulation of Useful Exponential Steps.

Lemma 7.5 (Alternative presentation of useful steps, originally at p. 15) $\to_{\mathsf{oe_u}} = \to_{\mathsf{oe}_{\mathsf{u}_1}} \cup \to_{\mathsf{oe}_{\mathsf{u}_2}}$.

Proof. It is clear that $\to_{\mathsf{oe}_{\mathsf{u}_1}} \cup \to_{\mathsf{oe}_{\mathsf{u}_2}} \subseteq \to_{\mathsf{oe_u}}$. It suffices to prove the other inclusion. An $\mathsf{e_u}$ step has the form:

$$O_1\langle O_2\langle x\rangle[x\leftarrow L\langle\lambda y.t\rangle]\rangle \to_{\mathsf{oe_u}} O_1\langle L\langle O_2\langle\lambda y.t\rangle[x\leftarrow\lambda y.t]\rangle\rangle$$

with $O_1\langle O_2\rangle$ useful. By Lemma 5.4.2, there are two cases:

- O_2 is useful. In this case, we have $O_1\langle O_2\langle x\rangle[x\leftarrow L\langle\lambda y.t\rangle]\rangle \to_{\mathsf{oe}_{\mathsf{u}_1}} O_1\langle L\langle O_2\langle\lambda y.t\rangle[x\leftarrow\lambda y.t]\rangle\rangle$.
- O_2 is left and O_1 is useful. Since O_1 is useful, it can be written as $O\langle L'u\rangle$ for some O, L', and u. Then we have $O_1\langle O_2\langle x\rangle[x\leftarrow L\langle\lambda y.t\rangle]\rangle = O\langle L'\langle O_2\langle x\rangle[x\leftarrow L\langle\lambda y.t\rangle]\rangle u\rangle \to_{\mathsf{oe}_{\mathsf{u}_2}} O\langle L'\langle L\langle O_2\langle\lambda y.t\rangle[x\leftarrow\lambda y.t]\rangle\rangle t\rangle = O_1\langle L\langle O_2\langle\lambda y.t\rangle[x\leftarrow\lambda y.t]\rangle\rangle$

□

For the simulation of $\to_{\mathsf{oe}_{\mathsf{u}_1}}$, we need the following auxiliary lemma, which characterizes the shape of the translation of values in useful contexts.

Proposition F.3 (Translation of useful contexts surrounding values) *Let U be a useful* VSC *context. Then there exist E, t, and z such that for all values v satisfying $\mathsf{fv}(v) \cap \mathsf{bv}(U) = \emptyset$ the translation verifies $[\![U\langle v\rangle]\!] = E\langle t[y\leftarrow[\![v]\!]z]\rangle$.*

Proof. By definition $U = O\langle Lt\rangle$. The proof is by induction on O.

- *Base case*, that is, $O = \langle\cdot\rangle$ and $U = Lt$. Let $[\![L]\!] = (E,\sigma)$ and $[\![t]\!] = E'\langle z\rangle$. Cases of v:
  · *Variable*, that is, $v = x \notin \mathsf{bv}(U) \supseteq \mathsf{bv}(L)$. Then $[\![L\langle x\rangle]\!] = E\langle x\sigma\rangle = E\langle x\rangle$. Therefore, $[\![U\langle x\rangle]\!] = E\langle E'\langle y[y\leftarrow xz]\rangle\rangle$.
  · *Abstraction*, that is, $v = \lambda w.u$. Note that the hypothesis $\mathsf{fv}(\lambda w.u) \cap \mathsf{bv}(U) = \emptyset$ and Prop. F.1 imply that $[\![u]\!]\sigma = [\![u]\!]$. Then: $[\![U\langle\lambda w.u\rangle]\!] = [\![L\langle\lambda w.u\rangle t]\!] = E\langle E'\langle y[y\leftarrow(\lambda w.[\![u]\!]\sigma)z]\rangle\rangle = E\langle E'\langle y[y\leftarrow(\lambda w.[\![u]\!])z]\rangle\rangle$.
- $U = U'r$. By *i.h.*, there exist E, t, and z such that for all v satisfying $\mathsf{fv}(v) \cap \mathsf{bv}(U') = \emptyset$ the translation verifies $[\![U'\langle v\rangle]\!] = E\langle t[y\leftarrow[\![v]\!]z]\rangle$. Let $[\![r]\!] = E'\langle x'\rangle$. There are two cases to consider, depending on whether $U'\langle v\rangle$ is an answer. Note that U' by definition is not a substitution context, so that $U'\langle v\rangle$ being an answer is independent from whether v is a variable or an abstraction. Cases:
  · $U'\langle v\rangle$ is an answer $L\langle\lambda y'.p\rangle$. Let $[\![L]\!] = (E'',\sigma)$. We know that $E\langle t[y\leftarrow[\![v]\!]z]\rangle = [\![U'\langle v\rangle]\!] = [\![L\langle\lambda y'.p\rangle]\!] =_{L.7.1} E''\langle z'[z'\leftarrow\lambda y'.[\![p]\!]\sigma]\rangle$ for some E'' and z'. Therefore, t is of the form $E'''\langle z'[z'\leftarrow\lambda y'.[\![p]\!]\sigma]\rangle$ for some E''' satisfying $E'' = E\langle E'''[y\leftarrow[\![v]\!]z]\rangle$. Then the statement holds:

$$\begin{aligned}[\![U'\langle v\rangle r]\!] &= E''\langle E'\langle z'[z'\leftarrow(\lambda y'.[\![p]\!]\sigma)x']\rangle\rangle\\ &= E\langle E'''\langle E'\langle z'[z'\leftarrow(\lambda y'.[\![p]\!]\sigma)x']\rangle\rangle[y\leftarrow[\![v]\!]z]\rangle.\end{aligned}$$

  · $t = U'\langle v\rangle$ is not an answer. We have $E\langle t[y\leftarrow[\![v]\!]z]\rangle = [\![U'\langle v\rangle]\!] = E''\langle z'\rangle$ for some E'' and z'. Therefore, t is of the form $E'''\langle z'\rangle$ for some E''' satisfying $E'' = E\langle E'''[y\leftarrow[\![v]\!]z]\rangle$. Then the statement holds:

$$\begin{aligned}\llbracket U'\langle v\rangle r\rrbracket &= E''\langle E'\langle w[w{\leftarrow}z'x']\rangle\rangle \\ &= E\langle E'''\langle E'\langle w[w{\leftarrow}z'x']\rangle\rangle[y{\leftarrow}\llbracket v\rrbracket z]\rangle.\end{aligned}$$

- $U = rU'$. By *i.h.*, there exist E, t, and z such that for all v satisfying $\mathsf{fv}(v)\cap\mathsf{bv}(U') = \emptyset$ the translation verifies $\llbracket U'\langle v\rangle\rrbracket = E\langle t[y{\leftarrow}\llbracket v\rrbracket z]\rangle$. There are two cases to consider, depending on whether r is an answer. Cases:
  - r is an answer $L\langle\lambda w.q\rangle$. Let $\llbracket L\rrbracket = (E',\sigma)$. Let $t = E''\langle x'\rangle$. Then the statement holds:

$$\llbracket rU'\langle v\rangle\rrbracket = E'\langle E\langle E''\langle y'[y'{\leftarrow}(\lambda w.\llbracket q\rrbracket\sigma)x']\rangle[y{\leftarrow}\llbracket v\rrbracket z]\rangle\rangle.$$

  - r is not an answer. Let $\llbracket r\rrbracket = E'\langle w\rangle$ and $t = E''\langle x'\rangle$. Then the statement holds:

$$\llbracket rU'\langle v\rangle\rrbracket = E'\langle E\langle E''\langle y'[y'{\leftarrow}wx']\rangle[y{\leftarrow}\llbracket v\rrbracket z]\rangle\rangle.$$

- $U = r[w{\leftarrow}U']$. By *i.h.*, there exist E, t, and z such that for all v satisfying $\mathsf{fv}(v)\cap\mathsf{bv}(U') = \emptyset$ the translation verifies $\llbracket U'\langle v\rangle\rrbracket = E\langle t[y{\leftarrow}\llbracket v\rrbracket z]\rangle$. Let $t = E'\langle x'\rangle$. Then the statement holds:

$$\llbracket r[w{\leftarrow}U'\langle v\rangle]\rrbracket = E\langle E'\langle\llbracket r\rrbracket\{w{\leftarrow}x'\}\rangle[y{\leftarrow}\llbracket v\rrbracket z]\rangle.$$

- $U = U'[w{\leftarrow}r]$. By *i.h.*, there exist E, t, and z such that for all v satisfying $\mathsf{fv}(v)\cap\mathsf{bv}(U') = \emptyset$ the translation verifies $\llbracket U'\langle v\rangle\rrbracket = E\langle t[y{\leftarrow}\llbracket v\rrbracket z]\rangle$. Let $\llbracket r\rrbracket = E'\langle x'\rangle$. Since $w\in\mathsf{bv}(U)$, $w\notin\mathsf{fv}(v)\supset\mathsf{fv}(\llbracket v\rrbracket)$ by Prop. F.1. Then the statement holds:

$$\begin{aligned}\llbracket U'\langle v\rangle[w{\leftarrow}r]\rrbracket &= E'\langle E\langle t[y{\leftarrow}\llbracket v\rrbracket z]\rangle\{w{\leftarrow}x'\}\rangle \\ &= E'\langle E\sigma\langle t\sigma[y{\leftarrow}\llbracket v\rrbracket(z\sigma)]\rangle\rangle\end{aligned}$$

  where $\sigma = \{w{\leftarrow}x'\}$.

□

Proposition 7.6 (Simulation of useful exponential steps, originally at p. 15) *Let t and u be* VSC *terms. If $t\to_{\mathsf{oe}_{\mathsf{u}_1}} u$ or $t\to_{\mathsf{oe}_{\mathsf{u}_2}} u$ then $\llbracket t\rrbracket\to_{\mathsf{oe}_+}\llbracket u\rrbracket$.*

Proof. The root cases:

- *Useful exponential root rule 1*: $U\langle\!\langle x\rangle\!\rangle[x{\leftarrow}L\langle\lambda y.t\rangle]\mapsto_{\mathsf{e}_\mathsf{u}} L\langle U\langle\!\langle\lambda y.t\rangle\!\rangle[x{\leftarrow}\lambda y.t]\rangle$. Let $\llbracket L\rrbracket = (E,\sigma)$. By Lemma 7.1, $\llbracket L\langle\lambda y.t\rangle\rrbracket = E\langle z[z{\leftarrow}\lambda y.\llbracket t\rrbracket\sigma]\rangle$. By Prop. F.3, there exist E'', u, and x' such that $\llbracket U\langle\!\langle x\rangle\!\rangle\rrbracket = E''\langle u[w{\leftarrow}xx']\rangle$ and $\llbracket U\langle\!\langle(\lambda y.t)\rangle\!\rangle\rrbracket = E''\langle u[w{\leftarrow}(\lambda y.\llbracket t\rrbracket)x']\rangle$. Then:

$$\begin{aligned}
\llbracket U\langle\!\langle x\rangle\!\rangle[x{\leftarrow}L\langle\lambda y.t\rangle]\rrbracket &=_{L.7.1} && E\langle\llbracket U\langle\!\langle x\rangle\!\rangle\rrbracket\{x{\leftarrow}z\}[z{\leftarrow}\lambda y.\llbracket t\rrbracket\sigma]\rangle \\
&=_{Pr.\ F.3} && E\langle E''\langle u[w{\leftarrow}xx']\rangle\{x{\leftarrow}z\}[z{\leftarrow}\lambda y.\llbracket t\rrbracket\sigma]\rangle \\
&= && E\langle E''\langle u[w{\leftarrow}zx']\rangle\{x{\leftarrow}z\}[z{\leftarrow}\lambda y.\llbracket t\rrbracket\sigma]\rangle \\
&\to_{\mathsf{oe}_+} && E\langle E''\langle u[w{\leftarrow}(\lambda y.\llbracket t\rrbracket\sigma)x']\rangle\{x{\leftarrow}z\}[z{\leftarrow}\lambda y.\llbracket t\rrbracket\sigma]\rangle \\
&= && E\langle E''\langle u[w{\leftarrow}(\lambda y.\llbracket t\rrbracket)x']\rangle\{x{\leftarrow}z\}[z{\leftarrow}\lambda y.\llbracket t\rrbracket]\sigma\rangle \\
&=_{Pr.\ F.3} && E\langle\llbracket U\langle\!\langle\lambda y.t\rangle\!\rangle\rrbracket\{x{\leftarrow}z\}[z{\leftarrow}\lambda y.\llbracket t\rrbracket]\sigma\rangle \\
&= && E\langle\llbracket U\langle\lambda y.t\rangle[x{\leftarrow}\lambda y.t]\rrbracket\sigma\rangle \\
&=_{L.7.1} && \llbracket L\langle U\langle\lambda y.t\rangle[x{\leftarrow}\lambda y.t]\rangle\rrbracket
\end{aligned}$$

- *Useful exponential root rule 2*: $L_1\langle L_2\langle\!\langle x\rangle\!\rangle[x{\leftarrow}L_3\langle\lambda y.t\rangle]\rangle u\mapsto_{\mathsf{e}_{\mathsf{u}_2}} L_1\langle L_3\langle L_2\langle\!\langle\lambda y.t\rangle\!\rangle[x{\leftarrow}\lambda y.t]\rangle\rangle u$. Let $\llbracket L_1\rrbracket = (E_1,\sigma_1)$, $\llbracket L_2\rrbracket = (E_2,\sigma_2)$, $\llbracket L_3\rrbracket = (E_3,\sigma_3)$, $\llbracket t\rrbracket = E\langle z\rangle$, and $\llbracket u\rrbracket = E'\langle w\rangle$.

By Lemma 7.1, $[\![L_3\langle\lambda y.t\rangle]\!] = E_3\langle x'[x'\leftarrow\lambda y.[\![t]\!]\sigma_3]\rangle$. Then:

$$\begin{array}{rll}
[\![L_1\langle L_2\langle\!\langle x\rangle\!\rangle[x\leftarrow L_3\langle\lambda y.t\rangle]\rangle]\!] & = & E_1\langle[\![L_2\langle\!\langle x\rangle\!\rangle[x\leftarrow L_3\langle\lambda y.t\rangle]]\!]\sigma_1\rangle \\
& = & E_1\langle E_3\langle[\![L_2\langle\!\langle x\rangle\!\rangle]\!]\{x\leftarrow x'\}[x\leftarrow\lambda y.[\![t]\!]\sigma_3]\rangle\sigma_1\rangle \\
& =_{\alpha} & E_1\langle E_3\langle[\![L_2\langle\!\langle x\rangle\!\rangle]\!][x\leftarrow\lambda y.[\![t]\!]\sigma_3]\rangle\sigma_1\rangle \\
& =_{x\notin\mathsf{dom}(\sigma_2)} & E_1\langle E_3\langle E_2\langle\!\langle x\rangle\!\rangle[x\leftarrow\lambda y.[\![t]\!]\sigma_3]\rangle\sigma_1\rangle \\
& = & E_1\langle E_3\sigma_1\langle E_2\sigma_1\langle\!\langle x\rangle\!\rangle[x\leftarrow\lambda y.[\![t]\!]\sigma_3\sigma_1]\rangle\rangle
\end{array}$$

and since $L_1\langle L_2\langle\!\langle x\rangle\!\rangle[x\leftarrow L_3\langle\lambda y.t\rangle]\rangle$ is not an answer,

$$[\![L_1\langle L_2\langle\!\langle x\rangle\!\rangle[x\leftarrow L_3\langle\lambda y.t\rangle]\rangle u]\!] = E_1\langle E_3\sigma_1\langle E_2\sigma_1\langle E'\langle x'[x'\leftarrow xw]\rangle\rangle[x\leftarrow\lambda y.[\![t]\!]\sigma_3\sigma_1]\rangle\rangle.$$

Similarly, keeping in mind that L_2 does not capture any variable of $\lambda y.t$, we have:

$$\begin{array}{rl}
[\![L_1\langle L_3\langle L_2\langle\!\langle\lambda y.t\rangle\!\rangle[x\leftarrow\lambda y.t]\rangle\rangle]\!] & = E_1\langle[\![L_3\langle L_2\langle\!\langle\lambda y.t\rangle\!\rangle[x\leftarrow\lambda y.t]\rangle]\!]\sigma_1\rangle \\
& = E_1\langle E_3\langle[\![L_2\langle\!\langle\lambda y.t\rangle\!\rangle[x\leftarrow\lambda y.t]]\!]\sigma_3\rangle\sigma_1\rangle \\
& = E_1\langle E_3\langle[\![L_2\langle\!\langle\lambda y.t\rangle\!\rangle]\!][x\leftarrow\lambda y.[\![t]\!]]\sigma_3\rangle\sigma_1\rangle \\
& = E_1\langle E_3\langle E_2\langle x'[x'\leftarrow\lambda y.[\![t]\!]]\rangle[x\leftarrow\lambda y.[\![t]\!]]\sigma_3\rangle\sigma_1\rangle \\
& = E_1\langle E_3\sigma_1\langle E_2\sigma_1\langle x'[x'\leftarrow\lambda y.[\![t]\!]\sigma_3\sigma_1]\rangle[x\leftarrow\lambda y.[\![t]\!]\sigma_3\sigma_1]\rangle\rangle
\end{array}$$

and, since $L_1\langle L_3\langle L_2\langle\!\langle\lambda y.t\rangle\!\rangle[x\leftarrow\lambda y.t]\rangle\rangle$ is an answer, we have:

$$[\![L_1\langle L_3\langle L_2\langle\!\langle\lambda y.t\rangle\!\rangle[x\leftarrow\lambda y.t]\rangle\rangle u]\!] = E_1\langle E_3\sigma_1\langle E_2\sigma_1\langle E'\langle x'[x'\leftarrow(\lambda y.[\![t]\!]\sigma_3\sigma_1)w]\rangle\rangle[x\leftarrow\lambda y.[\![t]\!]\sigma_3\sigma_1]\rangle\rangle.$$

Then, it is clear that $[\![L_1\langle L_2\langle\!\langle x\rangle\!\rangle[x\leftarrow L_3\langle\lambda y.t\rangle]\rangle u]\!] \to_{\mathsf{oe}_+} [\![L_1\langle L_3\langle L_2\langle\!\langle\lambda y.t\rangle\!\rangle[x\leftarrow\lambda y.t]\rangle\rangle u]\!]$.

For the inductive cases, note that $\mapsto_{\mathsf{e}_{\mathsf{u}_1}}$ and $\mapsto_{\mathsf{e}_{\mathsf{u}_2}}$ cannot alter the shape of answers and non-answers. Then, the statement follows immediately from the *i.h.*, the definition of the translation, and the lifting given by Lemma F.2. □

G Proofs from Sect. 8 (Core Normal Forms and Termination Equivalence)

Proposition 8.3 (Characterization of core normal forms, originally at p. 16) *Let t be a VSC term. t is $\to_{\mathsf{ocore}}$-normal if and only if it is a n term.*

Proof. Direction $\Rightarrow$: by induction on t. Cases:

- *Value*, *i.e.* $t = v$. v is $\to_{\mathsf{ocore}}$-normal and a n term.
- *Application*, *i.e.* $t = ur$. By *i.h.*, u and r are n terms. Note that u cannot be an almost answer, otherwise t would have a root multiplicative redex or an $\to_{\mathsf{oe}_{\mathsf{u}}}$-redex. Then t is a n term.
- *ES*, *i.e.* $t = u[x\leftarrow r]$. By *i.h.*, u and r are n terms. Cases of r:
  - r *is an answer* $L\langle\lambda y.r'\rangle$. Then $x \notin \mathtt{aofv}(u)$, otherwise there would be a $\to_{\mathsf{oe}_{\mathsf{u}}}$ step. Then t is a n term.
  - r *has shape* $L\langle y\rangle$. Then $x \notin \mathtt{ofv}(u)$, otherwise there would be a $\to_{\mathsf{oe}_{\mathsf{var}}}$ step. Then t is a n term.
  - r *has shape* $L\langle r_1 r_2\rangle$. Then t is a n term.

Direction $\Leftarrow$: by induction on t. Cases:

- *Value*, *i.e.* $t = v$. v is a n term and $\to_{\mathsf{ocore}}$-normal.
- *Application*, *i.e.* $t = ur$ with u not an almost answer. By *i.h.*, u and r are $\to_{\mathsf{ocore}}$-normal. Thus, the only possible core redex of t must be at the root. Since u is not an almost answer, there is neither root multiplicative redex nor $\to_{\mathsf{oe}_{\mathsf{u}}}$-redex. Then t is $\to_{\mathsf{ocore}}$-normal.
- *ES*, *i.e.* $t = u[x\leftarrow r]$. By *i.h.*, u and r are $\to_{\mathsf{ocore}}$-normal. Thus, the only possible core redexes of t must involve the ES at the root. Cases of t:

- r *is an answer* $L\langle\lambda y.r'\rangle$ *and* $x \notin \mathtt{aofv}(u)$. Then the root ES is not involved in any $\to_{\mathtt{oe_u}}$ redex (because $x \notin \mathtt{aofv}(u)$) nor any $\to_{\mathtt{oe_{var}}}$ redex (because $r \neq L\langle z\rangle$). Then t is $\to_{\mathtt{ocore}}$-normal.
- r *has shape* $L\langle y\rangle$ *and* $x \notin \mathtt{ofv}(u)$. Then the root ES is not involved in any $\to_{\mathtt{oe_u}}$ redex (because r is not an answer) nor any $\to_{\mathtt{oe_{var}}}$ redex (because $x \notin \mathtt{ofv}(u)$). Then t is $\to_{\mathtt{ocore}}$-normal.
- r *has shape* $L\langle r_1 r_2\rangle$. Then the root ES is not involved in any $\to_{\mathtt{oe_u}}$ redex (because r is not an answer) nor any $\to_{\mathtt{oe_{var}}}$ redex (because $r \neq L\langle z\rangle$). Then t is $\to_{\mathtt{ocore}}$-normal.

□

Now we need some lemmas that shall be used to prove that $\to_{\mathtt{ocore}}$ normal forms are mapped on $\to_{\mathtt{ox_+\neg gc}}$ normal forms by $[\![\cdot]\!]$.

Lemma G.1

- $\mathtt{ofv}([\![tu]\!]) = \mathtt{ofv}([\![t]\!]) \cup \mathtt{ofv}([\![u]\!])$.
- $\mathtt{aofv}([\![tu]\!]) = \begin{cases} \mathtt{aofv}([\![t]\!]) \cup \mathtt{aofv}([\![u]\!]) \cup \{x\} & \text{if } [\![t]\!] = E\langle\!\langle x\rangle\!\rangle \\ \mathtt{aofv}([\![t]\!]) \cup \mathtt{aofv}([\![u]\!]) & \text{otherwise} \end{cases}$
- $\mathtt{ofv}([\![t[x\leftarrow u]]\!]) = \begin{cases} (\mathtt{ofv}([\![t]\!]) \setminus \{x\}) \cup \mathtt{ofv}([\![u]\!]) & \text{if } x \in \mathtt{ofv}([\![t]\!]) \\ \mathtt{ofv}([\![t]\!]) \cup (\mathtt{ofv}([\![u]\!]) \setminus \{y\}) & \text{if } x \notin \mathtt{ofv}([\![t]\!]) \text{ and } [\![u]\!] = E\langle y\rangle \text{ with } y \notin \mathtt{ofv}(E) \\ \mathtt{ofv}([\![t]\!]) \cup \mathtt{ofv}([\![u]\!]) & \text{otherwise} \end{cases}$
- $\mathtt{aofv}([\![t[x\leftarrow u]]\!]) = \begin{cases} \mathtt{aofv}([\![t]\!]) \cup \mathtt{aofv}([\![u]\!]) & \text{if } x \notin \mathtt{aofv}([\![t]\!]) \\ (\mathtt{aofv}([\![t]\!]) \setminus \{x\}) \cup \mathtt{aofv}([\![u]\!]) \cup \{y\} & \text{if } x \in \mathtt{aofv}([\![t]\!]) \text{ and } [\![u]\!] = E\langle\!\langle y\rangle\!\rangle \\ (\mathtt{aofv}([\![t]\!]) \setminus \{x\}) \cup \mathtt{aofv}([\![u]\!]) & \text{otherwise} \end{cases}$

Proof. Straightforward by definition. □

Proposition G.2

(i) *For any* VSC *term* t, $\mathtt{ofv}([\![t]\!]) \subseteq \mathtt{ofv}(t)$.

(ii) *For any* n *term* t, $\mathtt{aofv}([\![t]\!]) \subseteq \mathtt{aofv}(t)$.

Proof.

(i) By induction on t. Cases:
- $t = x$. Then $\mathtt{ofv}([\![t]\!]) = \{x\} = \mathtt{ofv}(t)$.
- $t = \lambda x.u$. Then $\mathtt{ofv}([\![t]\!]) = \mathtt{ofv}(y[y\leftarrow\lambda x.[\![u]\!]]) = \emptyset = \mathtt{ofv}(t)$.
- $t = ur$. Then by Lemma G.1, $\mathtt{ofv}([\![t]\!]) = \mathtt{ofv}([\![u]\!]) \cup \mathtt{ofv}([\![r]\!]) \subseteq_{i.h.} \mathtt{ofv}(u) \cup \mathtt{ofv}(r) = \mathtt{ofv}(t)$.
- $t = u[x\leftarrow r]$. Then by Lemma G.1, $\mathtt{ofv}([\![t]\!]) \subseteq (\mathtt{ofv}([\![u]\!]) \setminus \{x\}) \cup \mathtt{ofv}([\![r]\!]) \subseteq_{i.h.} (\mathtt{ofv}(u) \setminus \{x\}) \cup \mathtt{ofv}(r) = \mathtt{ofv}(t)$.

(ii) By induction on t. Cases:
- $t = v$. Then $\mathtt{aofv}([\![t]\!]) = \emptyset = \mathtt{aofv}(t)$.
- $t = nn'$ with n not an almost answer. By *i.h.*, $\mathtt{aofv}([\![n]\!]) \subseteq \mathtt{aofv}(n)$ and $\mathtt{aofv}([\![n']\!]) \subseteq \mathtt{aofv}(n')$. Sub-cases:
  - n is of the form $L\langle\!\langle x\rangle\!\rangle$. Then $[\![n]\!]$ is of the form $E\langle\!\langle x\rangle\!\rangle$ by Lemma G.3. By Lemma G.1, we have $\mathtt{aofv}([\![t]\!]) = \mathtt{aofv}([\![n]\!]) \cup \mathtt{aofv}([\![n']\!]) \cup \{x\} \subseteq \mathtt{aofv}(n) \cup \mathtt{aofv}(n') \cup \{x\} = \mathtt{aofv}(t)$.
  - Otherwise, $[\![n]\!]$ is not of the form $E\langle x\rangle$ by Lemma G.3. By Lemma G.1, we have $\mathtt{aofv}([\![t]\!]) = \mathtt{aofv}([\![n]\!]) \cup \mathtt{aofv}([\![n']\!]) \subseteq \mathtt{aofv}(n) \cup \mathtt{aofv}(n') = \mathtt{aofv}(t)$.
- $t = n[x\leftarrow n']$. By *i.h.*, $\mathtt{aofv}([\![n]\!]) \subseteq \mathtt{aofv}(n)$ and $\mathtt{aofv}([\![n']\!]) \subseteq \mathtt{aofv}(n')$. Sub-cases:
  - $n' = L\langle\lambda y.u\rangle$ and $x \notin \mathtt{aofv}(n)$. Then $x \notin \mathtt{aofv}(n)$ and by Lemma G.1, we have $\mathtt{aofv}([\![n[x\leftarrow n']]\!]) = \mathtt{aofv}([\![n]\!]) \cup \mathtt{aofv}([\![n']\!]) \subseteq \mathtt{aofv}(n) \cup \mathtt{aofv}(n') = \mathtt{aofv}(t)$.
  - $n' = L\langle y\rangle$ and $x \notin \mathtt{ofv}(n)$. Then $x \notin \mathtt{aofv}(n) \supseteq \mathtt{aofv}([\![n]\!])$ and by Lemma G.1, we have $\mathtt{aofv}([\![t]\!]) = \mathtt{aofv}([\![n]\!]) \cup \mathtt{aofv}([\![n']\!]) \subseteq \mathtt{aofv}(n) \cup \mathtt{aofv}(n') = \mathtt{aofv}(t)$.
  - $n' = L\langle t_1 t_2\rangle$. Let $[\![n']\!] = E\langle y\rangle$. It is clear, by definition, that y is bound in E. Then by Lemma G.1, we have $\mathtt{aofv}([\![t]\!]) = (\mathtt{aofv}([\![n]\!]) \setminus \{x\}) \cup \mathtt{aofv}([\![n']\!]) \subseteq (\mathtt{aofv}(n) \setminus \{x\}) \cup \mathtt{aofv}(n') = \mathtt{aofv}(t)$.

□

Lemma G.3 *If $[\![t]\!]$ is of the form $E\langle\!\langle x\rangle\!\rangle$ then t is of the form $L\langle\!\langle x\rangle\!\rangle$.*

Proof. Straightforward by induction on t. □

Lemma G.4 *$[\![t]\!]$ is of the form $E\langle E'\langle\!\langle x\rangle\!\rangle[x\leftarrow\lambda y.u]\rangle$ if and only if there exists an almost answer t' such that $t \to^*_{\mathsf{oe_{var}}} t'$.*

Proof. Direction $\Rightarrow$: by induction on t. Cases:

- *Variables*, *i.e.* $t = z$. Trivial because $[\![z]\!]$ is not in the desired form.
- *Abstraction*, *i.e.* $t = \lambda y.r$. Then $[\![t]\!]$ is in the desired form and $t' := t$ satisfies the statement.
- *Application*, *i.e.* $t = rq$. Then $[\![t]\!]$ is not in the desired form.
- *ESs*, *i.e.* $t = r[x\leftarrow q]$. Let $[\![q]\!] = E\langle y\rangle$. Then $[\![t]\!] = E\langle [\![r]\!]\{x\leftarrow y\}\rangle$. If $[\![t]\!]$ is in the desired form, we have one of the following cases:
  - $[\![r]\!]$ is in the desired form. Then by *i.h.*, $r \to^*_{\mathsf{oe_{var}}} r'$ for some almost answer r'. We have $t = r[x\leftarrow q] \to^*_{\mathsf{oe_{var}}} r'[x\leftarrow q]$ which is an almost answer.
  - $[\![q]\!]$ is in the desired form and $[\![r]\!]$ is of the form $E'\langle\!\langle x\rangle\!\rangle$. By Lemma G.3, r is of the form $L\langle\!\langle x\rangle\!\rangle$. By *i.h.*, $q \to^*_{\mathsf{oe_{var}}} q'$ for some almost answer q'. Cases of q':

  (i) *q' is an answer.* Then $t = L\langle\!\langle x\rangle\!\rangle[x\leftarrow q] \to^*_{\mathsf{oe_{var}}} L\langle\!\langle x\rangle\!\rangle[x\leftarrow q']$ which is an almost answer.

  (ii) *q' is an almost answer $L'\langle L''\langle\!\langle z\rangle\!\rangle[z\leftarrow q'']\rangle$ with q'' an answer.* Then:
  $$t = L\langle\!\langle x\rangle\!\rangle[x\leftarrow q] \to^*_{\mathsf{oe_{var}}} L\langle\!\langle x\rangle\!\rangle[x\leftarrow L'\langle L''\langle\!\langle z\rangle\!\rangle[z\leftarrow q'']\rangle] \to_{\mathsf{oe_{var}}} L'\langle L''\langle L\langle\!\langle z\rangle\!\rangle\rangle[z\leftarrow q'']\rangle$$
  which is an almost answer.

□

Proposition 8.4 (Preservation of core normal forms, originally at p. 16) *Let t be a* VSC *term. If t is $\to_{\mathsf{ocore}}$-normal then $[\![t]\!]$ is $\to_{\mathsf{ox_+\neg gc}}$-normal.*

Proof. By induction on the grammar of t. Cases:

- $t = v$. Trivial.
- $t = nn'$ with n not an almost answer. By *i.h.*, $[\![n]\!]$ and $[\![n']\!]$ are $\to_{\mathsf{ox_+\neg gc}}$-normal. Let $[\![n]\!] = E\langle x\rangle$ and $[\![n']\!] = E'\langle z\rangle$. Since n is not an answer, $[\![t]\!] = E\langle E'\langle y[y\leftarrow xz]\rangle\rangle$. Knowing that there is no $\to_{\mathsf{ox_+\neg gc}}$-redex within E and E', the only possible $\to_{\mathsf{ox_+\neg gc}}$-redex in $[\![t]\!]$ would be an $\to_{\mathsf{oe_u}}$-redex corresponding to the applicative occurrence of x, in the case where $[\![n]\!]$ is in the desired form of Lemma G.4. This would imply that n is an almost answer (since it is $\to_{\mathsf{ocore}}$-normal), which is not the case.
- $t = n[x\leftarrow n']$ with $n' = L\langle\lambda y.u\rangle$ and $x \notin \mathtt{aofv}(n)$. By *i.h.*, $[\![n]\!]$ and $[\![n']\!]$ are $\to_{\mathsf{ox_+\neg gc}}$-normal. Let $[\![L]\!] = (E, \sigma)$. Then $[\![t]\!] = E\langle [\![n]\!][x\leftarrow\lambda y.[\![u]\!]\sigma]\rangle$, which is $\to_{\mathsf{ox_+\neg gc}}$-normal since $x \notin \mathtt{aofv}([\![n]\!])$ by Prop. G.2.
- $t = n[x\leftarrow n']$ with $n' = L\langle y\rangle$ and $x \notin \mathtt{ofv}(n)$. By *i.h.*, $[\![n]\!]$ and $[\![n']\!]$ are $\to_{\mathsf{ox_+\neg gc}}$-normal. Let $[\![L]\!] = (E, \sigma)$. Then $[\![t]\!] = E\langle [\![n]\!]\{x\leftarrow y\sigma\}\rangle$, which is $\to_{\mathsf{ox_+\neg gc}}$-normal since $x \notin \mathtt{ofv}([\![n]\!])$ by Prop. G.2.
- $t = n[x\leftarrow n']$ with $n' = L\langle tu\rangle$. This case is straightforward by *i.h.*.

□

Theorem 8.5 (Termination equivalence of Core λ_{ovsc} and λ_{oxpos}, originally at p. 16) *Let t be a* VSC *term.*

(i) *t has a diverging $\to_{\mathsf{ocore}}$ sequence if and only if $[\![t]\!]$ has a diverging $\to_{\mathsf{ox_+}}$ sequence.*

(ii) *t is $\to_{\mathsf{ocore}}$-weakly normalizing if and only if $[\![t]\!]$ is $\to_{\mathsf{ox_+}}$-weakly normalizing.*

Proof. We split each of the two statements in its two directions and shuffle the order, since one of the directions of Point 2 is used to prove one of the directions of Point 1.

1$\Rightarrow$ *If t is $\to_{\mathsf{ocore}}$-diverging then $[\![t]\!]$ is $\to_{\mathsf{ox_+}}$-diverging.* If t has an infinite $\to_{\mathsf{ocore}}$ reduction sequence d then by local termination (Prop. 3.3) there is an infinity of multiplicative steps in d. By the simulation of core sequences (Thm. 7.7), $[\![t]\!]$ also has a diverging reduction sequence.

2⇒ *If t is $\to_{\mathtt{ocore}}$-weakly normalizing then $[\![t]\!]$ is $\to_{\mathtt{ox}_+}$-weakly normalizing.* If t has an $\to_{\mathtt{ocore}}$ reduction sequence $d : t \to^*_{\mathtt{ocore}} u$ with u $\to_{\mathtt{ocore}}$-normal, then by the simulation of core sequences (Thm. 7.7) there is a reduction sequence $e : [\![t]\!] \to^*_{\mathtt{ox}_+} [\![u]\!]$. By Prop. 8.4, $[\![u]\!]$ is $\to_{\mathtt{ox}_+\neg\mathtt{gc}}$-normal. Since $\to_{\mathtt{ogc}_+}$ is strongly normalizing (by local termination Prop. 4.6), $[\![u]\!] \to^*_{\mathtt{ogc}_+} r$ with r $\to_{\mathtt{ox}_+}$-normal.

1⇐ *If $[\![t]\!]$ is $\to_{\mathtt{ox}_+}$-diverging then t is $\to_{\mathtt{ocore}}$-diverging.* If t is $\to_{\mathtt{ocore}}$ weakly normalizing then, by direction 2⇒, $[\![t]\!]$ is $\to_{\mathtt{ox}_+}$ weakly normalizing, which is absurd because $\to_{\mathtt{ox}_+}$ is diamond (Thm. 4.3) and thus uniformly normalizing. Then t is $\to_{\mathtt{ocore}}$ diverging.

2⇐ *If $[\![t]\!]$ is $\to_{\mathtt{ox}_+}$-weakly normalizing then t is $\to_{\mathtt{ocore}}$-weakly normalizing.* If t has a diverging $\to_{\mathtt{ocore}}$ sequence then, by direction 1⇒, $[\![t]\!]$ has a diverging $\to_{\mathtt{ox}_+}$ sequence, which is absurd because $\to_{\mathtt{ox}_+}$ is diamond (Thm. 4.3) and thus uniformly normalizing. Then t has no diverging $\to_{\mathtt{ocore}}$ sequences, *i.e.* it is strongly normalizing.

□

Corollary 8.6 (Originally at p. 16) *$\lambda_{\mathtt{ovsc}}$ and Core $\lambda_{\mathtt{ovsc}}$ are uniformly normalizing.*

Proof. By Thm. 8.5.2, if t has a $\to_{\mathtt{ocore}}$ normalizing reduction then $[\![t]\!]$ has a $\to_{\mathtt{ox}_+}$ normalizing reduction. Now, t cannot have a $\to_{\mathtt{ocore}}$-diverging reduction, otherwise $[\![t]\!]$ has a $\to_{\mathtt{ox}_+}$ diverging reduction by Thm. 8.5.1, against uniform normalization of $\to_{\mathtt{ox}_+}$. Then t is $\to_{\mathtt{ocore}}$ strongly normalizing.

For $\lambda_{\mathtt{ovsc}}$, one repeats the same argument now using $\to_{\mathtt{ocore}}$ as the uniformly normalizing reduction (instead of $\to_{\mathtt{ox}_+}$) and Thm. 6.3 (instead of Thm. 8.5) to transfer reduction sequences between $\lambda_{\mathtt{ovsc}}$ and Core $\lambda_{\mathtt{ovsc}}$. □