IO vs OI in Higher-Order Recursion Schemes

Axel Haddad

LIGM (Université Paris Est & CNRS),
LIAFA (Université Paris Diderot & CNRS)

Abstract. We propose a study on the modes of derivation of higher-order recursion schemes, proving that value trees obtained from schemes using innermost-outermost derivations are the same as those obtained using unrestricted derivations.

1 Introduction

Recursion schemes have been first considered as a model of computation, representing the syntactical aspects of a recursive program [15,2,3,4]. At first, (order-1) schemes used to model simple recursive programs whose functions only take values as input (and not functions). Since, higher-order versions of recursion schemes [11,5,6,7,8,9] have been studied.

More recently, recursion schemes were studied as generators of infinite ranked trees and the focus is then to decide logical properties of those infinite trees [8,1,10,13,14].

As for programming languages, the question of evaluation plays a central role. Indeed, different policies result in different evaluation trees [8,9,7]. There are two main evaluation policies for schemes: outermost-innermost derivations (oi) and inner-outermost derivations (io), respectively corresponding to call-by-need and call-by-value in programming languages.

The standardization theorem for the lambda-calculus implies that for any scheme, outermost-innermost derivations (oi) lead to the same tree as unrestricted derivation. However, this is not the case for io derivations. In this paper we establish that the trees produced using schemes with io policy are the same as those produced using schemes with oi policy. For a given scheme of order $n$, we can use a lazying transformation, to construct a new scheme of order $n + 1$ in which io derivations will be the same as oi derivations in the initial scheme (Section 3). This increase of the order is shown to be unavoidable for order 1 schemes and conjectured to be so at any order.

Conversely, in order to turn a scheme into another one in which unrestricted derivations lead to the same tree as io derivations in the initial scheme, we adapt Kobayashi’s recent results on higher order recursion scheme (HORS) model-checking [13], to compute some key properties over terms (Section 4.1). Then we embed these properties into the scheme while preserving the order obtaining a scheme in which oi and io derivations produce the same tree (Section 4.2).

Finally, we show that this technique can be used to improve efficiency of derivation, for example by preventing non-productive derivations, i.e. derivations that would not produce anything in the value tree.
2 Preliminaries

Given a set $A$ we write $A^*$ the set of words of $A$. We write $wv$ the concatenation of the words $u$ and $v$, and we say that $u$ is a prefix of $w$ if there exists $v$ such that $uv = w$. We say that a subset $S$ of $A^*$ is prefix-closed if for all $w \in S$, $S$ contains all prefixes of $w$.

Types are defined by the grammar $\tau ::= o \mid \tau \rightarrow \tau$ and $o$ is called the ground type. Considering that $\rightarrow$ is associative to the right i.e. $\tau_1 \rightarrow (\tau_2 \rightarrow \tau_3)$ can be written $\tau_1 \rightarrow \tau_2 \rightarrow \tau_3$, any type $\tau$ can be written uniquely as $\tau_1 \rightarrow \ldots \rightarrow \tau_k \rightarrow o$. The integer $k$ is called the arity of $\tau$. We inductively define the order of a type by $\text{order}(o) = 0$ and $\text{order}(\tau_1 \rightarrow \tau_2) = \max(\text{order}(\tau_1) + 1, \text{order}(\tau_2))$. For instance $o \rightarrow o \rightarrow o$ is an order 1 type of arity 3, $(o \rightarrow o) \rightarrow (o \rightarrow o)$, that can also be written $(o \rightarrow o) \rightarrow o \rightarrow o$ is a type of order 2 and arity 2. We use the notation $\tau^t \rightarrow \tau'$ as a shorthand for $\underbrace{\tau \rightarrow \ldots \rightarrow \tau}_{t \text{ times}} \rightarrow \tau'$.

Let $\Gamma$ be a finite set of typed symbols, i.e. to each symbol is associated a type, and we let $\Gamma^\tau$ denotes the set of symbols of type $\tau$. For all type $\tau$, we define the set $T^\tau(\Gamma)$ of terms of type $\tau$ as the smallest set satisfying: $\Gamma^\tau \subseteq T^\tau(\Gamma)$ and $\bigcup_{\tau} \{ t \mid t \in T^\tau(\Gamma), s \in T^s(\Gamma) \} \subseteq T^\tau(\Gamma)$. We write $T(\Gamma)$ for the set of terms of any type, and $t : \tau$ if $t$ has type $\tau$. The arity of a term $t$, $\text{arity}(t)$, is the arity of its type. Remark that any term $t$ can be uniquely written as $t = o t_1 . . . t_k$ with $o \in \Gamma$ and $t_1, ..., t_k$ some terms. We say that $o$ is the head of the term $t$.

Example 1. Let $\Gamma = \{ F : (o \rightarrow o) \rightarrow o \rightarrow o , \ G : o \rightarrow o \rightarrow o , \ H : (o \rightarrow o) , \ a : o \}$: $F H$ and $G a$ are terms of type $o \rightarrow o$; $F(G a) (H (H a))$ is a term of type $o$; $F a$ is not a term since $F$ is expecting a first argument of type $o \rightarrow o$ while $a$ is of type $o$.

Let $t : \tau$, $t' : \tau'$ be two terms, $x : \tau'$ be a symbol in $T^{\tau'}(\Gamma)$, we write $t[x \leftarrow t'] : \tau$ the term obtained by substituting all occurrences of $x$ by $t'$ in the term $t$, it is defined by induction: $x[x \leftarrow t'] = t'$, $y[y \leftarrow t'] = y$ for $y \in \Gamma$ and $y \neq x$, and $t_1 t_2[x \leftarrow t'] = t_1[x \leftarrow t'] t_2[x \leftarrow t']$. A $\tau$-context is a term $C[\bullet^\tau] \in T(\Gamma^\bullet)$ containing exactly one occurrence of $\bullet^\tau$; it can be seen as an application turning a term into another, such that for all $t : \tau$, $C[t] = C[\bullet^\tau][\bullet^\tau \rightarrow t]$. In general we will only consider ground type context where $\tau = o$ and we will omit to specify the type when it is clear. For instance, if $C[\bullet] = F \bullet (H (H a))$ and $t' = G a$ then $C[t'] = F (G a) (H (H a))$.

Let $\Sigma$ be a set of symbols of order at most 1 (i.e. each symbols has type $o$ or $o \rightarrow \ldots \rightarrow o$) and $\bot : o$ be a fresh symbol. A tree $t$ over $\Sigma \cup \{ \bot \}$ is a mapping $t : \text{dom}^t \rightarrow \Sigma^\bot$ with $\text{dom}^t$ a prefix-closed subset of $\{1, ..., m\}^*$, $m$ being the maximum arity of $\Sigma$, such that if $u \in \text{dom}^t$ and $t(u) = a$ then $\{j \mid uj \in \text{dom}^t\} = \{1, ..., \text{arity}(a)\}$. Note that there is a direct bijection between ground terms of $T^\tau(\Sigma^\bot \cup \{ \bot \})$ and finite trees. Hence we may later see ground terms over $\Sigma^\bot \cup \{ \bot \}$ as trees.
When \( F \) and \( G \) are non-terminals, \( \{ F, G \} \) is a non-terminal.

We define inductively the \textit{rewriting relation} \( \rightarrow_G \in T(\Sigma \cup \mathcal{N})^2 \) (or just \( \rightarrow \) when \( \mathcal{G} \) is clear) as \( t \rightarrow_G t' \) if and only if there exists a context \( C[\cdot] \), a rewrite rule \( F x_1 ... x_k \rightarrow e \), and a derivation \( s_0 \rightarrow s_1 \rightarrow s_2 \rightarrow ... \rightarrow s_n = e \) such that \( t = C[F t_1 ... t_k] \) and \( t' = C[t_1 ... t_k] \). Finally we define a \textit{derivation}, as a possibly infinite sequence of terms linked by the rewriting relation \( s_0 \rightarrow_G s_1 \rightarrow_G s_2 \rightarrow_G ... \) and \( \rightarrow_G^* \) to be the reflexive and transitive closure of \( \rightarrow_G \). We say that a derivation is maximal if it is either finite or it is finite and the last term cannot be rewritten.

We define inductively the \textit{\( \perp \)-transformation} \( (\cdot)^\perp : T^o(\mathcal{N} \cup \Sigma) \rightarrow T^o(\Sigma \cup \{ \perp : o \}) \), used to transform a ground typed term into a finite tree, as for all \( F \in \mathcal{N} \) \( (F t_1 \ldots t_k)^\perp = \perp \) and for all \( a \in \Sigma \) \( (a t_1 \ldots t_k)^\perp = a t_1^\perp \ldots t_k^\perp \).

Let \( t_0 = S \rightarrow_G t_1 \rightarrow_G t_2 \rightarrow_G ... \) be a derivation, then one can check that \( (t_0)^\perp \subseteq (t_1)^\perp \subseteq (t_2)^\perp \subseteq ... \), hence it admits a limit. One can prove that the set of all such limit trees has a greatest element that we denote \( \| \mathcal{G} \| \) and refer to as the \textit{value tree} of \( \mathcal{G} \). Note that \( \| \mathcal{G} \| \) is the supremum of \( \{ t^\perp : S \rightarrow t \} \). Given a term \( t : o \), we denote by \( \mathcal{G}_t \) the scheme obtained by transforming \( \mathcal{G} \) such that it starts derivations with the term \( t \), formally, \( \mathcal{G}_t = \langle \mathcal{V}, \Sigma, \mathcal{N} \cup \{ S' \}, \mathcal{R} \cup \{ S' \rightarrow t, S' \} \rangle \).

\textbf{Example 3.} Let \( \mathcal{G} = \langle \mathcal{V}, \Sigma, \mathcal{N}, \mathcal{R}, S \rangle \) be the scheme such that: \( \mathcal{V} = \{ o : o, \phi : o \rightarrow o, \psi : (o \rightarrow o) \rightarrow o \rightarrow o \} \), \( \Sigma = \{ a : o^3 \rightarrow o, b : o \rightarrow o \rightarrow o, c : o \} \). The set of non-terminals is \( \mathcal{N} = \{ F : ((o \rightarrow o) \rightarrow o \rightarrow o) \rightarrow (o \rightarrow o) \rightarrow o \rightarrow o, H : (o \rightarrow o) \rightarrow o \rightarrow o, I, J, K : o \rightarrow o, S : o \} \), and \( \mathcal{R} \) contains the following rewrite rules:

\[
\begin{align*}
F \psi \phi x & \rightarrow \psi \phi x \quad I x & \rightarrow x \\
J x & \rightarrow b (J x) (J x) \\
h \phi x & \rightarrow a (J x) (K x) (\phi x) \\
K x & \rightarrow K (K x) S \rightarrow F H I c
\end{align*}
\]
Here is an example of finite derivation:

\[
S \rightarrow F H I \ c \rightarrow H I \ c \rightarrow a \ (J \ c) \ (K \ (K \ c)) \ (I \ c) \rightarrow a \ (J \ c) \ (K \ (K \ (K \ c))) \ (I \ c)
\]

If one extends it by always rewriting a redex of head \(K\), its limit is the tree \(a \perp \perp \perp\), but this tree is not the value tree \(\|G\|\). The value tree \(\|G\|\) is depicted on the right.

### Evaluation Policies

We now put constraints on the derivations we allow. If there are no constraints, then we say that the derivations are unrestricted and we let \(\text{Acc}^G = \{t : o \mid S \rightarrow^* t\}\) be the set of accessible terms using unrestricted derivations. Intuitively an outermost-innermost rewriting is a rewriting where the redex chosen to be rewritten has no redex above it in the term (it is an outermost redex), in an innermost-outermost rewriting the redex chosen has no redex as a subterm (it is an innermost redex).

Formally given a term \(t \rightarrow t'\) such that \(t = C[s_1 ... s_k]\) and \(t' = C[e_{[\forall j \ x_j \mapsto s_j]}]\) with \(F x_1 ... x_k \rightarrow e \in R\):

- We say that \(t \rightarrow t'\) is an **outermost-innermost** (oi) rewriting (written \(t \rightarrow_{\text{oi}} t'\)) if there is no redex containing the occurrence of \(\bullet\) as a subterm of \(C[\bullet]\).
- We say that \(t \rightarrow t'\) is an **innermost-outermost** (io) rewriting (written \(t \rightarrow_{\text{io}} t'\)) if for all \(j \in \{1, ..., k\}\) there is no redex as a subterm of \(s_j\).

Let \(\text{Acc}_{\text{oi}}^G = \{t : o \mid S \rightarrow^*_{\text{oi}} t\}\) be the set of accessible terms using oi derivations and \(\text{Acc}_{\text{io}}^G = \{t : o \mid S \rightarrow^*_{\text{io}} t\}\) be the set of accessible terms using io derivations. There exists a supremum of \(\text{Acc}_{\text{oi}}\) (resp. \(\text{Acc}_{\text{io}}\)) which is the maximum of the limit trees of oi derivations (resp. io derivations). We write it \(\|G\|_{\text{oi}}\) (resp. \(\|G\|_{\text{io}}\)). For all recursive scheme \(G\), \((\text{Acc}^G)^+ = (\text{Acc}_{\text{oi}}^G)^+\) in particular \(\|G\|_{\text{oi}} = \|G\|\). But \(\|G\|_{\text{io}} \subseteq \|G\|\) and in general, the equality does not hold (see Example 4).

### 3 From OI to IO

Fix a recursion scheme \(\mathcal{G} = (\mathcal{V}, \Sigma, \mathcal{N}, R, S)\). Our goal is to define another scheme \(\mathcal{G} = (\mathcal{V}, \Sigma \cup \{k : o\}, \mathcal{N}, R, I)\) such that \(\|\mathcal{G}\|_{\text{io}} = \|\mathcal{G}\|\). The idea is to add an extra argument \((k : o)\) to each non-terminal, that will be required to rewrite it (hence the types are changed and the order is augmented by 1). We feed this argument to the outermost non-terminal, and duplicate it to subterms only if the head of the term is a terminal. Hence all derivations will be oi-derivations. This is a standard transformation.
\begin{align*}
\text{Example 4.} \quad \text{Let } G = (\mathcal{V}, \Sigma, \mathcal{N}, \mathcal{R}, S) \text{ be the order-1 recursion scheme with } \Sigma = \{a, c : o\}, \mathcal{N} = \{S : o, F : o \to o \to o, H : o \to o\}, \mathcal{V} = \{x, y : o\}, \text{ and } G &= (H a) c \\
F \ x \ y \to y \\
H \ x \to H (H x)
\end{align*}

Then we have $\|G\|_o = c$ while $\|G\|_o = 1$ (indeed, the only to derivation is the following $S \to F (H a) c \to F (H (H a)) c \to F (H (H (H a))) c \to ...$). The order-2 recursion scheme $G = (\mathcal{V}, \Sigma \cup \{k : o\}, \mathcal{N}, \mathcal{R}, I)$ is given by $\mathcal{N} = \{I : o, \overline{S}, a, \overline{c} : o \to o, F : (o \to o) \to (o \to o) \to o \to o, H : (o \to o) \to o \to o\}, \mathcal{V} = \{\ell : o, x, y : o \to o\}$ and the following rewrite rules:

\begin{align*}
I &\to \overline{S} k \\
\overline{H} \overline{x} \ell &\to \overline{H} (\overline{H} \overline{x}) \overline{c} k \\
\overline{c} \ell &\to c \\
&\quad \text{Note that in the term } \overline{F} (\overline{H} \overline{a}) \overline{c} k, \text{ the subterm } \overline{H} \overline{a} \text{ is no longer a redex since it lacks its last argument, hence it cannot be rewritten, then the only to derivation, which is the only unrestricted derivation is } I \to \overline{S} k \to \overline{F} (\overline{H} \overline{a}) \overline{c} k \to \overline{c} k \to c. \text{ Therefore } \|G\|_o = \|G\|_o = c = \|G\|_o.
\end{align*}

Formally, we define the $(\tau)$ transformation over types by $\overline{\tau} = o \to o$, and $\overline{\tau} = o \to o$. In particular, if $\tau = \tau_1 \to ... \to \tau_k \to o$ then $\overline{\tau} = \overline{\tau_1} \to ... \to \overline{\tau_k} \to o \to o$. Note that for all $\tau$, order($\overline{\tau}$) = order($\tau$) + 1 and arity($\overline{\tau}$) = arity($\tau$) + 1.

For all $x : \tau \in \mathcal{V}$ we take $\overline{x} : \overline{\tau}$ to be a fresh variable. Let $\eta_{\text{max}}$ be the maximum arity of terminals, we take $\eta_1, ..., \eta_{\text{max}} : o \to o$ and $\ell : o$ to be fresh variables, and we let $\mathcal{V} = \{x : \tau \mid x \in \mathcal{V}\} \cup \{\eta_1, ..., \eta_{\text{max}} \} \cup \{\ell : o\}$. Note that $\ell$ is the only variable of type $o$. For all $a : \tau \in \Sigma$ define $\overline{a} : \overline{\tau}$ as a fresh non-terminal for all $F : \tau \in \mathcal{N}$ define $\overline{F} : \overline{\tau}$ as a fresh non-terminal. Let $\mathcal{N} = \{\overline{a} : \overline{\tau} \mid a \in \Sigma\} \cup \{\overline{F} : \overline{\tau} \mid F \in \mathcal{N}\} \cup \{I : o\}$. Note that $I$ is the only symbol in $\mathcal{N}$ of type $o$.

Let $t : \tau \in \mathcal{T}(\mathcal{V} \cup \Sigma \cup \mathcal{N})$, we define inductively the term $\overline{t} : \overline{\tau} \in \mathcal{T}(\overline{\mathcal{V}} \cup \overline{\mathcal{N}})$:

If $t = x \in \mathcal{V}$ (resp. $t = a \in \Sigma$, $t = F \in \mathcal{N}$), we let $\overline{t} = \overline{x} \in \overline{\mathcal{V}}$ (resp. $\overline{t} = \overline{a} \in \overline{\mathcal{N}}$, $\overline{t} = \overline{F} \in \overline{\mathcal{N}}$), if $t = t_1 t_2 : \tau$ then $\overline{t} = \overline{t_1} \overline{t_2}$.

Let $F \ x_1 ... x_k \to c$ be a rewrite rule of $\mathcal{R}$. We define the (valid) rule $\overline{F} \overline{x_1} ... \overline{x_k} \ell \to \overline{c} k$ in $\overline{\mathcal{R}}$. Let $a : \Sigma$ of arity $k$, we define the rule $\pi \eta_1 ... \eta_k \ell \to a (\eta_1 k) ... (\eta_k k)$ in $\overline{\mathcal{R}}$. We also add the rule $I \to \overline{S} k$ to $\overline{\mathcal{R}}$. Finally let $\overline{G} = (\overline{\mathcal{V}}, \Sigma \cup \{k : o\}, \overline{\mathcal{N}}, \overline{\mathcal{R}}, I)$.

**Lemma 1.** Any derivation of $\overline{G}$ is in fact an OI and an IO derivation. Hence $\|\overline{G}\|_o = \|\overline{G}\|_o$.

**Proof (Sketch).** The main idea is that (1) the only redexes will be the terms that have an occurrence of $k$ as last argument of the head non-terminal, which we can prove by a study on types of the terms produced during a derivation. The scheme is constructed so that $k$ remains only on the outermost non-terminals, that is why any derivation is an OI derivation. Furthermore, we have that
t = \overline{F} t_1 \ldots t_k \kappa \text{ is a redex, then none of the } t_j \text{ contain an occurrence of } \kappa, therefore they do not contain any redex, hence (1) states that } t \text{ is an innermost redex.}

**Theorem 1 (ot vs io).** Let \( G \) be an order-n scheme. Then one can construct an order-\((n+1)\) scheme \( \overline{G} \) such that \( \|G\| = \|\overline{G}\|_{io} \).

**Proof (Sketch).** Taking into account Lemma 1, we just have to show that ot derivations in \( \overline{G} \) acts like io derivations in \( G \), hence \( \|G\| = \|\overline{G}\| \). Therefore, we construct some maximal derivations in \( G \) and in \( \overline{G} \), and we show that we can get one from the other, and that they lead to the same tree.

In Example 4 the value tree of the original scheme, was a finite tree \( c \), it could have been done with an order 0 scheme with one non-terminal: \( S \rightarrow c \). It is actually easy to show that the set of trees produced by order 0 scheme is the set of all regular trees. We show in Example 5 an order 1 scheme whose ot value tree cannot be produced by any order 1 scheme using only io derivations.

**Example 5.** Let \( G_0 = (\Sigma_0, N_0, V_0, R_0, S_0) \) be the order 1 scheme such that \( \Sigma = \{a : o \rightarrow o, b : o^2 \rightarrow o\}, N\{S_0, B : o, F : o \rightarrow o\}, V = \{x : o\}, \) and \( R_0 \) contains the following rewrite rules:

\[
S_0 \rightarrow F B \quad F x \rightarrow b \ x \ (F \ (a \ B)) \quad B \rightarrow b \ B \ B
\]

The tree \( \|G_0\| \) is depicted in Figure 1. The tree is not regular, then it cannot be generated by an order 0 scheme. We can show the following theorem.

**Theorem 2.** There exists no order 1 scheme \( G \) such that \( \|G\|_{io} = \|G_0\| \).

\[\text{Figure 1. The tree } \|G_0\|\]
4 From IO to OI

The goal of this section is to transform the scheme $G$ into a scheme $G''$ such that $\|G''\| = \|G\|_\io$. The main difference between $\io$ and $\oi$ derivations is that some redex would lead to $\bot$ in $\io$ derivation while $\oi$ derivations could be more productive. For example take $F : o \rightarrow o$ such that $F \ x \rightarrow c$, and $H : o$ such that $H \rightarrow a \ H$, with $a : o \rightarrow o$ and $c : o$ being terminal symbols. The term $F \ H$ has a unique $\oi$ derivation, $F \ H \rightarrow_\oi c$, it is finite and it leads to the value tree associated. On the other hand, the (unique) $\io$ derivation is the following $F \ H \rightarrow F (a \ H) \rightarrow F (a (a \ H)) \rightarrow ...$ which leads to the tree $\bot$. In general, take a redex $F \ t_1...t_k$, if for some $t_i$, all the $\io$ derivations from $t_i$ are infinite, then the $\io$ value tree associated to the term $F \ t_1...t_k$ will be $\bot$.

The idea of the transformation is to compute an annotation (based on a type system) that helps decide if a redex would produce $\bot$ with $\io$ derivations (Section 4.1); then we embed it into $G$ and force any such redex to produce $\bot$ even with unrestricted derivations (Section 4.2).

4.1 The Type System

Given a term $t : \tau \in T(\Sigma \uplus \mathcal{N})$, we define the following properties on $t$: $P_\bot(t) = \text{"The term } t \text{ has type } o \text{ and its associated } \io \text{ valuation tree is } \bot^\bot$, and $P_\infty(t) = \text{"the term } t \text{ has not necessarily ground type, it contains a redex } r \text{ such that any maximal } \io \text{ derivation from } r \text{ producing it's } \io \text{ valuation tree is infinite"}$ (recall that a maximal derivation is either infinite, or finite and the last term cannot be rewritten). Note that $P_\infty(t)$ is equivalent to “the term $t$ contains a redex $r$ such that $\|G_r\|_\io$ is either infinite or contains $\bot$". In this section we describe a type system, inspired from the work of Kobayashi [13], that characterises the terms verifying these properties.

Let $Q$ be the set $\{q_\bot, q_\infty\}$. Given a type $\tau$, we define inductively the sets $(\tau)^\atom$ and $(\tau)^\land$ called respectively set of atomic mappings and set of conjunctive mappings: $(o)^\atom = Q$, for all $\tau_1 \rightarrow \tau_2$, $(\tau_1 \rightarrow \tau_2)^\atom = \{q_\infty\} \uplus \{\tau_1 \rightarrow \tau_2\}^\atom$ and for all $\tau$, $(\tau \rightarrow \tau_2)^\land = \{\land \{\theta_1, ..., \theta_i\} \mid \theta_1, ..., \theta_i \in (\tau \rightarrow \tau_2)^\atom\}$.

We will usually use the letter $\theta$ to represents atomic mappings, and the letter $\sigma$ to represent conjunctive mappings. Given a conjunctive mapping $\sigma$ (resp. an atomic mapping $\theta$) and a type $\tau$, we write $\sigma :: (\tau)$ (resp. $\sigma :: \theta$) the relation $\sigma \in (\tau)^\land$ (resp. $\theta \in (\tau)^\atom$). For the sake of simplicity, we identify the atomic mapping $\theta$ with the conjunctive mapping $\land \{\theta\}$.

Given a term $t$ and a conjunctive mapping $\sigma$, we define a judgment as a tuple $\Theta \vdash t \sigma$, pronounced “from the environment $\Theta$, one can prove that $t$ matches the conjunctive mapping $\sigma$”, where the environment $\Theta$ is a partial function from $V \uplus \mathcal{N}$ to conjunctive mapping. Given an environment $\Theta$, $\alpha \in V \uplus \mathcal{N}$ and a conjunctive mapping $\sigma$, we define the environment $\Theta' = \Theta, \alpha \triangleright \sigma$ as $\text{Dom}(\Theta') = \text{Dom}(\Theta) \cup \{\alpha\}$ and $\Theta'(\alpha) = \sigma$ if $\alpha \not\in \text{Dom}(\Theta)$, $\Theta'(\alpha) = \sigma \land \Theta(\alpha)$ otherwise, and $\Theta'(\beta) = \Theta(\beta)$ if $\beta \neq \alpha$. 


We define the following judgement rules:

\[ \begin{align*}
\Theta \vdash t \triangleright_1 & \quad \ldots \quad \Theta \vdash t \triangleright_n \quad (\text{Set}) \quad \Theta, \alpha \triangleright \{t_1, \ldots, t_n\} \vdash \alpha \triangleright_i (\text{At}) \quad (\text{for all } i)
\end{align*} \]

\[ \begin{align*}
\Theta \vdash a \triangleright_1 \sigma_1 & \quad \ldots \quad \sigma_i \leq \text{arity}(a) \rightarrow q_\infty \quad (\Sigma) \quad (\text{for } a \in \Sigma \text{ and } \exists j \quad \sigma_j = q_\infty)
\end{align*} \]

\[ \begin{align*}
\Theta \vdash t_1 \triangleright \sigma & \rightarrow \Theta \vdash t_2 \triangleright \sigma \\
\Theta \vdash t_1 \triangleright t_2 \triangleright & \Theta \\
\Theta \vdash t \triangleright q_\infty & \rightarrow q_\infty \quad (\text{if } t : \tau_1 \rightarrow \tau_2) \\
\Theta \vdash t_1 \triangleright q_\infty & \rightarrow q_\infty \quad (q_\infty \rightarrow q_\infty)
\end{align*} \]

Remark that there is no rules that directly involves \( q_\bot \), but it does not mean that no term matches \( q_\bot \), since it can appear in \( \Theta \). Rules like \((\text{At})\) or \((\text{App})\) may be used to state that a term matches \( q_\bot \).

The main property of \( \text{io} \) derivation is captured by the rules \((q_\infty)\) and \((q_\infty \rightarrow q_\infty)\), which describes the fact that if a term \( t \) contains a subterm whose maximal \( \text{io} \) derivations are infinite, then maximal \( \text{io} \) derivations of \( t \) are also infinite. This is not the case for \( \text{ot} \) derivations.

We say that \((G, t)\) matches the conjunctive mapping \( \sigma \) written \( \vdash (G, t) \triangleright \sigma \) if there exists an environment \( \Theta \), called a witness environment of \( \vdash (G, t) \triangleright \sigma \), such that \((1)\) \( \text{Dom}(\Theta) = \mathcal{N} \), \((2)\) \( \forall F : \tau \in \mathcal{N}, \Theta(F) :: \tau \), \((3)\) if \( F \ x_1 \ldots x_k \rightarrow e \in \mathcal{R} \) and \( \Theta \vdash F \triangleright \sigma_1 \rightarrow \ldots \rightarrow \sigma_i \leq k \rightarrow q \) then either there exists \( j \) such that \( q_\infty \in \sigma_j \), or \( i = k \) and \( \Theta, x_1 \triangleright \sigma_1, \ldots, x_k \triangleright \sigma_k \vdash e \triangleright q \). \((4)\) \( \Theta \vdash t \triangleright \sigma \).

The following two results state that this type system matches the properties \( \mathcal{P}_\bot \) and \( \mathcal{P}_\infty \) and furthermore we can construct an universal environment, \( \Theta^* \), that can correctly judge any term.

**Theorem 3 (Soundness and Completeness).** Let \( G \) be an HORS, and \( t \) be term (of any type), \( \vdash (G, t) \triangleright q_\infty \) (resp. \( \vdash (G, t) \triangleright q_\bot \)) if and only if \( \mathcal{P}_\infty (t) \) (resp. \( \mathcal{P}_\bot (t) \)) holds.

**Proposition 1 (Universal Witness).** There exists an environment \( \Theta^* \) such that for all term \( t \), the judgement \( \vdash (G, t) \triangleright \sigma \) holds if and only if \( \Theta^* \vdash t \triangleright \sigma \).

**Proof (Sketch).** To compute \( \Theta^* \), we start with an environment \( \Theta_0 \) satisfying Properties \((1)\) and \((2)\) (\( \text{Dom}(\Theta_0) = \mathcal{N} \) and \( \forall F : \tau \in \mathcal{N}, \Theta_0(F) :: \tau \)) that is able to judge any term \( t : \tau \) with any conjunctive mapping \( \sigma :: \tau \).

Then let \( F \) be the mapping from the set of environments to itself, such that for all \( F : \tau_1 \rightarrow \ldots \rightarrow \tau_k \rightarrow o \in \mathcal{N} \), if \( F \ x_1 \ldots x_k \rightarrow e \in \mathcal{R} \) then,

\[ F(\Theta)(F) = \{ \sigma_1 \rightarrow \ldots \rightarrow \sigma_k \rightarrow q \mid q \in Q \wedge \forall i \quad \sigma_i :: \tau_i \wedge \Theta, x_1 \triangleright \sigma_1, \ldots, x_k \triangleright \sigma_k \vdash e : q \} \]

\[ \cup \{ \sigma_1 \rightarrow \ldots \rightarrow \sigma_i \leq k \rightarrow q_\infty \mid \wedge \forall i \quad \sigma_i :: \tau_i \wedge \exists j \quad q_\infty \in \sigma_j \} \]

\[ \cup \{ \sigma_1 \rightarrow \ldots \rightarrow \sigma_k \rightarrow q_\bot \mid \forall i \quad \sigma_i :: \tau_i \wedge \exists j \quad q_\infty \in \sigma_j \} \].
We iterate $F$ until we reach a fixpoint. The environment we get is $\Theta^*$, it verifies properties (1) (2) and (3). Furthermore we can show that this is the maximum of all environments satisfying these properties, i.e. if $\vdash (G, t) \triangleright \sigma$ then $\Theta^* \vdash t \triangleright \sigma$.

4.2 Self-Correcting Scheme

For all term $t : \tau \in \mathcal{T}(\Sigma \sqcup \mathcal{N})$, we define $\llbracket t \rrbracket \in (\tau)^\wedge$, called the witnessed value of $t$, as the conjunction of all atomic mappings $\theta$ such that $\Theta^* \vdash t \triangleright \theta$ (recall that $\Theta^*$ is the environment of Proposition 1). In particular $\mathcal{P}_\perp(t)$ (resp. $\mathcal{P}_\infty(t)$) holds if and only if $q_\perp \in \llbracket t \rrbracket$ (resp. $q_\infty \in \llbracket t \rrbracket$).

Given two terms $t_1 : \tau' \rightarrow \tau$ and $t_2 : \tau' \rightarrow \tau''$ the only rules we can apply to judge $\Theta^* \vdash t_1 t_2 \triangleright \theta$ are (App), ($q_\perp \rightarrow q_\infty$) and ($q_\infty$) (recall that $\theta$ is atomic). Then, we see that $\theta$ only depends on which atomic mappings are matched by $t_1$ and $t_2$. In other words $\llbracket t_1 t_2 \rrbracket$ only depends on $\llbracket t_1 \rrbracket$ and $\llbracket t_2 \rrbracket$. Hence we can associate to $\llbracket t_1 \rrbracket$ a function $f_{\llbracket t_1 \rrbracket}$ defined on $\{ \sigma \mid \exists \theta \llbracket t_1 \rrbracket \llbracket t_2 \rrbracket = \sigma \}$ such that $f_{\llbracket t_1 \rrbracket}(\sigma) = \llbracket t_1 t_2 \rrbracket$ with $\llbracket t_2 \rrbracket = \sigma$. We use the notation $\llbracket t_1 \rrbracket \llbracket t_2 \rrbracket$ as a shortcut for $f_{\llbracket t_1 \rrbracket}(\llbracket t_2 \rrbracket)$.

In this section, given a scheme $G = (\mathcal{V}, \Sigma, \mathcal{N}, \mathcal{R}, S)$, we transform it into $G' = (\mathcal{V}', \Sigma, \mathcal{N'}, \mathcal{R'}, S)$ which is basically the same scheme except that while it is producing an IO derivation, it evaluates $[t']$ for any subterm $t'$ of the current term and label $t'$ with $[t']$. Note that if $t \rightarrow_\text{IO} t'$, then $\llbracket t \rrbracket = \llbracket t' \rrbracket$. Since we cannot syntactically label terms, we will label all symbols by the witnessed value of their arguments, for example if the term $F \ t_1 \ldots \ t_k$, we will label $F$ with the $k$-tuple $([t_1], \ldots, [t_k])$.

A problem may appear if some of the arguments are not fully applied, for example imagine we want to label $F \ H$ with $H : o \rightarrow o$. We will label $F$ with $[H]$, but since $H$ has no argument we do not know how to label it. The problem is that we cannot wait to label it because once a non-terminal is created, the derivation does not deal explicitly with it. The solution is to create one copy of $H$ per possible witnessed value for its argument (here there are four of them: $\Lambda\{\}, \Lambda\{q_\perp\}, \Lambda\{q_\infty\}, \Lambda\{q_\perp, q_\infty\}$). This means that $F[H]$ would not have the same type as $F$: $F$ has type $(o \rightarrow o) \rightarrow o$, but $F[H]$ will have type $(o \rightarrow o)^2 \rightarrow o$. Hence, $F \ H$ will be labelled the following way: $F[H] \ H \Lambda\{\} \ H \Lambda\{q_\perp\} \ H \Lambda\{q_\infty\} \ H \Lambda\{q_\perp, q_\infty\}$. Note that even if $F$ has 4 arguments, it only has to be labelled with one witnessed value since all four arguments represent different labelling of the same term. We now formalize these notions.

Let us generalize the notion of witnessed value to deal with terms containing some variables. Given an environment on the variables $\Theta^V$ such that $\text{Dom}(\Theta^V) \subseteq \mathcal{V}$ and if $x : \tau$ then $\Theta^V(x) : \tau$, and given a term $t : \tau \in \mathcal{T}(\Sigma \sqcup \mathcal{N} \sqcup \text{Dom}(\Theta^V))$, we define $\llbracket t \rrbracket_{\Theta^V} \in (\tau)^\wedge$, as the conjunction of all atomic mappings $\theta$ such that $\Theta^* \vdash t \triangleright \theta$. Given two terms $t_1 : \tau' \rightarrow \tau$ and $t_2 : \tau' \rightarrow \tau''$ we still have that $\llbracket t_1 t_2 \rrbracket_{\Theta^V}$ only depends on $\llbracket t_1 \rrbracket_{\Theta^V}$ and $\llbracket t_2 \rrbracket_{\Theta^V}$.

To a type $\tau = \tau_1 \rightarrow \ldots \rightarrow \tau_k \rightarrow o$ we associate the integer

$$n_\tau = \text{Card}(\{(\sigma_1, \ldots, \sigma_k) \mid \forall i \, \sigma_i \in (\tau_i)^\wedge\})$$
and a complete ordering of \(\{ (\sigma_1, \ldots, \sigma_k) | \forall i, \sigma_i \in (\tau_i)^{\downarrow} \} \) denoted \(vec_1, vec_2, \ldots, vec_n\). We define inductively the type \(\tau^+ = (\tau^+_1)^{\downarrow n_1} \rightarrow \cdots \rightarrow (\tau^+_k)^{\downarrow n_k} \rightarrow o\).

To a non terminal \(F : \tau_1 \rightarrow \cdots \rightarrow \tau_k \rightarrow o\) (resp. a variable \(x : \tau_1 \rightarrow \cdots \rightarrow \tau_k \rightarrow o\)) and a tuple \(\sigma_1 : \tau_1, \ldots, \sigma_k : \tau_k\), we associate the non-terminal \(F^{\sigma_1, \ldots, \sigma_k} : \tau_1^{\downarrow n_1} \rightarrow \cdots \rightarrow \tau_k^{\downarrow n_k} \rightarrow o \in N\) (resp. a variable \(x^{\sigma_1, \ldots, \sigma_k} : \tau_1^{\downarrow n_1} \rightarrow \cdots \rightarrow \tau_k^{\downarrow n_k} \rightarrow o \in V\)).

Finally, recall that...
5 Avoiding Non Productive $\text{oi}$ Derivations

The technics shown in Section 4 may seem quite heavy compared to the lazying transformation in Section 3. They are actually far more general. In this section, we show how they can be used to enhance $\text{oi}$ derivations of a scheme.

Given an infinite $\text{oi}$ derivation $t_1 \to_{\text{oi}} t_2 \to_{\text{oi}} \ldots$, we say that it is ultimately non productive if at some point the tree associated to it, does not increase, i.e. if there exists $i$ such that for all $j \geq i$, $(t_j) \perp = (t_i) \perp$.

Example 6. Take the scheme $\langle \{a : o\}, \{S : o, H : o \to o\}, \{x : o\}, S \to H \ a \ a : H \ x \to H (H \ x) \rangle$. The $\text{oi}$ derivation $S \to H \ a \ a \to_{\text{oi}} H (H \ a) \to_{\text{oi}} H (H (H \ a)) \to_{\text{oi}} \ldots$ is ultimately non productive, because its associated tree sequence is $\perp, \perp, \perp, \ldots$. In this example, this is the only $\text{oi}$ derivation from the term $H \ a$. Now add to the scheme a terminal symbol $f : o \to o$, a non terminal symbol $F : o \to \text{and}$ the rule $F x \to x (F x)$. The $\text{oi}$ derivation $f (F \ a) (H \ a) \to_{\text{oi}} f (F \ a) (H (H \ a)) \to_{\text{oi}} f (F \ a) (H (H (H \ a))) \to_{\text{oi}} \ldots$ is ultimately non productive, however there exists a productive derivation from $f (F \ a) (H \ a) : f (F \ a) (H \ a) \to_{\text{oi}} f (a (F \ a)) (H \ a) \to_{\text{oi}} f (a (a (F \ a))) (H \ a) \to_{\text{oi}} \ldots$.

Using the same tools as those in Section 4, it is possible to transform a scheme $G$ into a scheme $G''$ that produce the same valuation tree, in which no ultimately non productive $\text{oi}$ derivation is possible. The construction proceeds the same way as in section 4, first we create a type system to decide whether a term would lead to $\perp$ with $\text{oi}$ derivations, then we embed this type system into the scheme, finally we make the scheme turn to $\perp$ all the redexes that would lead to $\perp$ in the valuation tree.

6 Conclusion

We have shown that value trees obtained from schemes using innermost-outermost derivations ($\text{io}$) are the same as those obtained using unrestricted derivations. More precisely given an order-$n$ scheme $G$ we create an order-$(n + 1)$ scheme $\overline{G}$ such that $\|\overline{G}\|_{\text{IO}} = \|G\|$. However, the increase of the order seems unavoidable. We also create an order-$n$ scheme $G''$ such that $\|G''\| = \|G\|_{\text{IO}}$. In this case the order does not increase, however the size of the scheme blows up while it remains almost the same in $\overline{G}$. In the last section, we have given a glimpse on how the technics shown in Section 4 can be used to treat other problems.

References


A From OI to IO

Complement of Definitions

A $n$ holes context is a term $C[\bullet_{1}^{i}, ..., \bullet_{n}^{i}] \in \mathcal{T}(\Gamma \cup \{\bullet_{i}^{i} : \tau_{i} \mid 1 \leq i \leq n\})$ containing exactly one occurrence of $\bullet_{i}$ for all $i$. (We will generally omit to write the type $\tau_{i}$ in the notation $\bullet_{i}^{i}$.)

For all $t_{1}, ..., t_{k} \leq n$ we are interested in the application

$$t_{1}, ..., t_{k} \mapsto (C[\bullet_{1}, ..., \bullet_{n}]\{\forall j \leq k \quad \bullet_{i} \mapsto t_{j}\})$$

with $t_{j} \in \mathcal{T}_{\tau_{j}}(\Gamma)$ for all $j$. (notice that the order of the substitution is not important). One can consider $(C[\bullet_{1}, ..., \bullet_{n}]\{\forall j \leq k \quad \bullet_{i} \mapsto t_{j}\})$ as a $n - k$ holes context. We may write $C[\bullet_{1}, ..., \bullet_{n}]$ to denote the context $C[\bullet_{1}, ..., \bullet_{n}]$ and extend this notation to $(C[\bullet_{1}, ..., \bullet_{n}]\{\forall j \leq k \quad \bullet_{i} \mapsto t_{j}\})$, for example, given the context $C[\bullet_{1}]^{[\bullet_{2}]}^{[\bullet_{3}]}$, we define $C[t_{1}]^{[\bullet_{2}]}^{[\bullet_{3}]} = (C[\bullet_{1}]^{[\bullet_{2}]}^{[\bullet_{3}]})[\bullet_{1} \mapsto t_{1}, \bullet_{2} \mapsto t_{2}]$.

Given a one hole context $C[\bullet]$, we define inductively the head symbol sequence $\text{hss}(C)$ which is a finite sequence of symbols of $\Gamma$: if $C[\bullet] = \bullet$, then $\text{hss}(C)$ is the empty sequence, if $C[\bullet] = \alpha t_{1}, ..., t_{i-1}C'[\bullet]t_{i+1}...t_{k}$, then $\text{hss}(C) = \alpha, \text{hss}(C')$.

**Proposition 2.** Given a $n$ holes context $C[\bullet_{1}]^{[\bullet_{2}]}^{[\bullet_{3}]}$, for all $i$, for all $t_{1}, ..., t_{i-1}, t_{i+1}, ..., t_{n}$ and $s_{1}, ..., s_{i-1}, s_{i+1}, ..., s_{n}$:

$$\text{hss}(C[t_{1}]^{[\bullet_{2}]}^{[\bullet_{3}]}[t_{i-1}]^{[\bullet_{2}]}^{[\bullet_{3}]}[t_{i+1}]^{[\bullet_{2}]}^{[\bullet_{3}]}[t_{n}]) = \text{hss}(C[s_{1}]^{[\bullet_{2}]}^{[\bullet_{3}]}[s_{i-1}]^{[\bullet_{2}]}^{[\bullet_{3}]}[s_{i+1}]^{[\bullet_{2}]}^{[\bullet_{3}]}[s_{n}])$$

**Proof (Proposition 2).** We prove this proposition by induction on the size of the context, for all $n$. If $C[\bullet] = \bullet$, is a 1 hole context, then the result is proven.

If $C[\bullet_{1}]^{[\bullet_{2}]}^{[\bullet_{3}]} = a t_{1}...t_{n}$ there exists exactly one $k$ such that $t_{k}$ contains one occurrence of $\bullet_{i}$, if we look all the occurrences of some $\bullet_{i}$ in $t_{k}$, we can state that $t_{k}$ is a $l$ holes context $C'[\bullet_{j_{1}}]^{[\bullet_{j_{2}}]}^{[\bullet_{j_{3}}]}$ for some $l$. Moreover

$$\text{hss}(C[t_{1}]^{[\bullet_{2}]}^{[\bullet_{3}]}[t_{i-1}]^{[\bullet_{2}]}^{[\bullet_{3}]}[t_{i+1}]^{[\bullet_{2}]}^{[\bullet_{3}]}[t_{n}]) = a, \text{hss}(C'[\bullet_{j_{1}}]^{[\bullet_{j_{2}}]}^{[\bullet_{j_{3}}]}[t_{j_{1}}])$$

and

$$\text{hss}(C[s_{1}]^{[\bullet_{2}]}^{[\bullet_{3}]}[s_{i-1}]^{[\bullet_{2}]}^{[\bullet_{3}]}[s_{i+1}]^{[\bullet_{2}]}^{[\bullet_{3}]}[s_{n}]) = a, \text{hss}(C'[\bullet_{j_{1}}]^{[\bullet_{j_{2}}]}^{[\bullet_{j_{3}}]}[s_{j_{1}}])$$

Since $\text{hss}(C'[\bullet_{j_{1}}]^{[\bullet_{j_{2}}]}^{[\bullet_{j_{3}}]}[t_{j_{1}}]) = \text{hss}(C'[\bullet_{j_{1}}]^{[\bullet_{j_{2}}]}^{[\bullet_{j_{3}}]}[s_{j_{1}}])$ by hypothesis of induction, we have

$$\text{hss}(C[t_{1}]^{[\bullet_{2}]}^{[\bullet_{3}]}[t_{i-1}]^{[\bullet_{2}]}^{[\bullet_{3}]}[t_{i+1}]^{[\bullet_{2}]}^{[\bullet_{3}]}[t_{n}]) = \text{hss}(C[s_{1}]^{[\bullet_{2}]}^{[\bullet_{3}]}[s_{i-1}]^{[\bullet_{2}]}^{[\bullet_{3}]}[s_{i+1}]^{[\bullet_{2}]}^{[\bullet_{3}]}[s_{n}])$$

**Proposition 3.** Let $t$ be a term, and let $C_{1}[\bullet]$ and $C_{2}[\bullet]$ be two contexts such that $t = C_{1}[t_{1}]$ and $t = C_{2}[t_{2}]$, then: either there exists a context $C_{1}'$ such that $C_{2}[\bullet] = C_{1}[C_{1}'[\bullet]]$ (1); Or there exists a context $C_{2}'$ such that $C_{1}[\bullet] = C_{2}[C_{2}'[\bullet]]$ (2); or there exists a two hole context $C[\bullet_{1}]^{[\bullet_{2}}$ such that $C[t_{1}]^{[\bullet_{2}] = C[\bullet][t_{2}] = C'[\bullet]$ (3).
We remark thatCorrectness of the Transformation

First, we notice that since we assume w.l.o.g. that \( i < j \) we set \( C[i][j] = a \) \( s_{i-1} C'[i][s_{i+1} \ldots s_k] \) and \( C[2][j] = a \) \( s_{j-1} C''[j][s_{j+1} \ldots s_k] \). If \( i \neq j \) (we have that \( C[t_1][i] = C_2 \) and \( C[i][t_2] = C_1 \). If \( i = j \), then \( C'_t[t_1] = C'_t[t_2] \),

by induction:

- Either there exists a two hole context \( C'[i_1][i_2] \) such that \( C'[t_1][i] = C_2'[i] \) and \( C'[i][t_2] = C_1'[i] \), in that case we set \( C'[i_1][i_2] = a \) \( s_{i-1} C''[i_1][i_2][s_{i+1} \ldots s_k] \)

Alternatively, for any subterm \( T \) has type \( \tau \) for some type \( \tau \), hence it cannot have any ground type and in particular, it is not a redex. Moreover, given two terms \( t_1, t_2 \), if the term \( t_1 \rightarrow t_2 \) is valid, then it is equal to \( t_1 \rightarrow t_2 \), in particular, it is an “overlined” term. It follows by induction, that given three terms \( t, t_1, t_2 \), if the term \( t \rightarrow_{[t_1 \rightarrow t_2]} \) is well defined, then it is equal to \( t \rightarrow_{[t_1 \rightarrow t_2]} \) which is also well defined.

Proof (Lemma 1). First We need the following claim.

Claim. (*) For all accessible term \( t \) (with unrestricted derivations), for all context \( C[\bullet] \) such that \( t = \overline{C[\text{red}]} \) with \( \text{red} \) being a redex, \( \text{hss}(C) \) only contains terminals symbols. Furthermore, the term \( \text{red} \) doesn’t contain occurrence of any terminal symbol.

Proof (Claim (*)). We prove it by induction. \( I \) satisfies Claim A, \( \overline{S} \) \( \Delta \) too. Assume that \( t = \overline{C[F[t_1 \ldots t_k]} \) satisfies Claim A, with \( k = \text{arity}(F) \) and \( F \in \mathcal{N} \).

If \( F \) \( x_{1} \ldots x_{k} \rightarrow_{\tau} \overline{\rho} \Delta \in \overline{\mathcal{N}} \), let \( t' = \overline{C[F[x_{1} ... x_{t_i}]} \). Let \( C'[\bullet] \) a context and \( \text{red} = \gamma s_{1} ... s_{\text{arity}(\gamma)} \) with \( \gamma \in \overline{\mathcal{N}} \) a redex such that \( t' = \overline{C'[\text{red}]} \).

First, we notice that since \( \overline{\rho} \overline{[x_{1} ... x_{t_i}]} \) \( \Delta \) is a ground type term only containing non-terminal symbols, it is a redex, \( \rho r_{1} \ldots r_{\text{arity}(\rho) - 1} \Delta = \overline{\rho} \overline{[x_{1} ... x_{t_i}]} \Delta \).

Using Proposition 3 we now that there are four options:

1. either \( C[\bullet] = \overline{C'[\gamma s_{1} ... s_{t_i} \overline{C''[s_{i+1} ... s_{\text{arity}(\gamma)}]}]} \) with \( \overline{C''} \) a context,
2. or \( C'[\bullet] = \overline{C[\rho r_{1} \ldots r_{t-1} \overline{C''[s_{i+1} ... s_{t_i}]}]} \) with \( \overline{C''} \) a context,
3. or \( C'[\bullet] = C[\bullet], \)
4. or there exists a two holes context \( \overline{C[\bullet][\bullet]} \) such that \( C[\bullet] = \overline{C[\bullet][\text{red}]} \) and \( C'[\bullet] = \overline{C[F[x_{1} \ldots x_{t_i}]} \).

Option 1 is impossible, otherwise \( \gamma \) would be an element of \( \text{hss}(C) \).

Option 2 would imply that \( \overline{\rho} \overline{[x_{1} ... x_{t_i}]} = \rho r_{1} \ldots r_{t-1} \overline{C''[\text{red}]} r_{t_i+1} \ldots r_{\text{arity}(\rho) - 1} \Delta \) which can’t be true, see Remark A.
If Option 3 is true, then hss(C') = hss(C) which by induction only contains terminal symbols. Since there is no terminal symbols in τ and in tᵢ for all i, there is no terminal in τΔ = red. Hence t' satisfies Claim A.

If Option 4 is true, then t = C[Δ] for all i, tᵢ = tᵢ...tₙ][red]. Then by induction, hss(C[τ₁...τₙ])[•] only contains terminal symbols. But, using Proposition 2, we know that

\[ hss(C'[•]) = hss(C[Δ]) = hss(C[Δ]) \]

Then hss(C'[•]) only contains terminal symbols. Furthermore, since red is a subterm of t, by induction it only contains non-terminal symbols, which proves that t' satisfies Claim (*)

Assume that t = C[τ₁...τₙ] satisfies Claim A with a ∈ Σ and k = arity(a), and let t' = C'[τ₁...τₙ](i). We can prove that t' satisfies Claim A in a similar way.

Let t = C[red] be an accessible term with red a redex, let exp be the rewrite expression of red, and let's look at the derivation t = C[exp].

Claim A tells us that hss(C) only contains terminals, hence there is no redex containing an occurrence of • in C, hence the derivation is OIA. Assume that red = γ τ₁...τ₉...τ₁...τ₂...τ₉(γ) with C' a context and t a term.

Then t = C[γ τ₁...τ₉...τ₁...τ₂...τ₉(γ)] then hss(C[γ τ₁...τ₉...τ₁...τ₂...τ₉(γ)]) contains a non terminal symbol γ, hence Claim A tells that t is not a redex, so no non-trivial subterm of red is a redex, so the derivation is IO.

Proof (Theorem 1). Lemma 1 shows that we only have to prove that \( \|G\|_{OI} = \|G\|_{OI} \). Concretely we will show that:

\[ \forall t ∈ \text{Acc}_G, \exists t' ∈ \text{Acc}_G : t \parallel t' \]

(1)

\[ \forall t' ∈ \text{Acc}_G, \exists t ∈ \text{Acc}_G : t' \parallel t \]

(2)

Definition 1 (\( \|\cdot\|\)-transformation). We define inductively the transformation \( \|\cdot\| : T^o(\Sigma ⊔ \overline{N}) \rightarrow T^o(\Sigma ⊔ \overline{N}) \):

\[ \|a\| t₁...t_{arity(a)} = a \|t₁\|...\|t_{arity(a)}\| \text{ for all } a ∈ \Sigma, \]

\[ \|red\| = \overline{\text{red}} Δ \text{ for red a redex} \]

Remark 1. Notice that \( t^\parallel = (\|t\|^\parallel)^\parallel \).

Claim (♣). If t ∈ T^o(\(\Sigma ⊔ \overline{N}\)) then \( tΔ \rightarrow \|t\| \).

Proof (Claim (♣)). The proof is done by induction. If t is a redex then \( \|t\| = tΔ \). If t = a t₁...tₙ with a ∈ \(\Sigma\) and k = arity(a), assume that for all i, \( t_i Δ \rightarrow \|t_i\| \). We have \( \overline{tΔ} = \overline{a} \overline{t_1}...\overline{t_k} \) so \( \overline{tΔ} \rightarrow \|t\| \). Hence \( \overline{tΔ} \rightarrow \|t\| \).

Claim (∨). For all t, if t ∈ \(\text{Acc}_G\), then \( \|t\| ∈ \text{Acc}_G \). This claim implies property (1).
Proof (Proof of Claim (\(\forall\))). We prove this by induction. If \(t = S\), \(\|t\| = S\), and \(I \rightarrow S \Delta\), so \(\|t\| \in Acc_{\Sigma}\).

Let \(t = C[F \ t_1...t_k] \in Acc_{\Sigma}\) with \(k = \text{arity}(F)\) and \(\text{hss}(C)\) contains only terminal symbols. Assume that \(\|t\| \in Acc_{\Sigma}\). Given that \(F \ x_1...x_k \rightarrow e \in R\), let \(t' = C[e[x_1...x_k]]\).

First, given a ground type context \(C[\bullet^*]\), we can define the associated ground type context \(\|C[\bullet^*]\|\) by adding to Definition 1 the fact \(\|\bullet^*\| = \bullet^*\). Hence we can say that \(\|t\| = \|C\|[F \ t_1...t_k \ \Delta]\).

We see that \(\|t\| \rightarrow_{\Sigma} \|C\|[e[x_1...x_k]] \ \Delta\). Claim A shows that \(\overline{t}[x_1...x_k] \ \Delta = \overline{e}[x_1...x_k] \ \Delta\), hence \(\|t\| \rightarrow_{\Sigma} \|C\|[e[x_1...x_k]] \ \Delta\) \(\rightarrow_{\Sigma} \|C\|[e[x_1...x_k]]\) = \(\|t'\|\).

Claim (\(\sigma\)). Given a term \(t' \in Acc_{\Sigma}\) there exists a term \(t \in Acc_{\Sigma}\) such that \(t' \rightarrow_{\Sigma} \|t\|\). This claim implies property (2).

Proof (Claim (\(\sigma\))). We will divide the relation \(\rightarrow_{\Sigma}\) in two relations : \(\rightarrow_{\Sigma} = \rightarrow_{\Sigma} \|\rightarrow\) depending of the head symbol of the redex we’re rewriting. Let \(t \rightarrow_{\Sigma} t'\) if the rewrite rule applied is \(r_\Sigma\) for some \(\alpha \in \Sigma\) then \(t \rightarrow_{\Sigma} t'\), if the rewrite rule is \(r_{\Sigma}\) with \(F \in N\), then \(t \rightarrow_{\Sigma} t'\).

The proof is in four steps:

1. Given a term \(t \in Acc_{\Sigma}\), there is only a finite number of derivation \(t \rightarrow_{\Sigma} t'\), furthermore, if \(t \rightarrow_{\Sigma} t_1\) and \(t \rightarrow_{\Sigma} t_2\) such that there is no \(t'\) such that \(t_1 \rightarrow_{\Sigma} t'\) or \(t_2 \rightarrow_{\Sigma} t'\), then \(t_1 = t_2\). We name this unique term \(t_{\Sigma}\), and we notice that if \(t \rightarrow_{\Sigma} t'\) then \(t' \rightarrow_{\Sigma} t_{\Sigma}\). Basically this step comes from the fact that the relation \(\rightarrow_{\Sigma}\) strictly decreases the number of occurrences of terms headed by some \(\Sigma\) with \(a \in \Sigma\).

2. Let \(t = C[F \ t_1...t_{\text{arity}(F)}]\), then \(t_{\Sigma} = C_{\Sigma}[F \ t_1...t_{\text{arity}(F)}]\), \(C_{\Sigma}\) being defined inductively: if \(C[\bullet] = \bullet\) then \(C_{\Sigma}[\bullet] = \bullet\), if \(C[\bullet] = a t_1...C'[\bullet]...t_k\), then \(C_{\Sigma}[\bullet] = a t_1...C_{\Sigma}'[\bullet]...t_k\) (Claim A shows that these are the only possibilities). This step is shown by induction.

3. If \(t \rightarrow_{\Sigma} t'\) i.e. \(t = C[F \ t_1...t_k]\) and \(t' = C[F \ x_1...x_k] \ \Delta\) with the appropriate \(\overline{t},\) let \(t'' = C_{\Sigma}[F \ x_1...x_k] \ \Delta\), then \(t_{\Sigma} = t''_{\Sigma}\).

4. Finally we prove by induction that for all \(t'\), there exists \(t \in Acc_{\Sigma}\) such that \(t_{\Sigma} = \|t\|\), which proves the claim.

B The Type System Detailed

We give here a complete proof of Theorem 3 and Proposition 1. We first recall the type system, and the definition of \(\vdash (G, t) \triangleright_\sigma\).

\[
\begin{align*}
\Theta \vdash t \triangleright \theta_1 & \quad \ldots \quad \Theta \vdash t \triangleright \theta_n \\
\Theta \vdash t \triangleright \Lambda\{\theta_1, \ldots, \theta_n\} & \quad \text{(Set)} \\
\Theta, \alpha \triangleright \Lambda\{\theta_1, \ldots, \theta_n\} & \quad \alpha \triangleright \theta_i \quad \text{(at)} & \quad \text{(for all \(i\))}
\end{align*}
\]

\[
\Theta \vdash a \triangleright \sigma_1 \rightarrow \ldots \rightarrow \sigma_{i \leq \text{arity}(a)} \rightarrow q_\infty \quad \text{(for all \(a \in \Sigma\) and \(\exists j \sigma_j = q_\infty\))}
\]
then there exists \( j \) which verifies the following properties:

\[
\text{if there exists an environment } \Theta \text{, let } t \vdash \sigma \text{ such that } t \vdash q_\infty \rightarrow q_\infty, \text{ (if } t : \tau_1 \rightarrow \tau_2) \]

\[
\frac{\Theta \vdash t \vdash \sigma \rightarrow \theta \quad \Theta \vdash t_2 \vdash \sigma}{\Theta \vdash t_1 \vdash t_2 \vdash \theta} \quad \text{(App)}
\]

\[
\Theta \vdash t \vdash q_\infty \rightarrow q_\infty \rightarrow q_\infty \quad \text{if } t : \tau_1 \rightarrow \tau_2
\]

\[
\frac{\Theta \vdash t_1 \vdash q_\infty \rightarrow q_\infty}{\Theta \vdash t_1 \vdash t_2 \vdash q_\infty} \quad \text{(q_\infty)}
\]

Remark that:

- Using rule \((Set)\) one can always prove, for any term \( t \), \( \Theta \vdash t \vdash \{\}\).
- \( \Theta \vdash t \vdash \{\theta_1, ..., \theta_k\} \) if and only if, for all \( i \), \( \Theta \vdash t \vdash \theta_i \).
- There is no rules that directly involve \( q_\perp \), but that does not mean that no term matches \( q_\perp \), since it can appear in \( \Theta \). Rules like \((At)\) or \((App)\) may be used to state that a term matches \( q_\perp \).

We say that \((G, t)\) matches the conjunctive mapping \( \sigma \) written \( \vdash (G, t) \vdash \sigma \) if there exists an environment \( \Theta \), called a witness environment of \( \vdash (G, t) \vdash \sigma \), which verifies the following properties:

1. \( \text{Dom}(\Theta) = N \).
2. \( \forall F : \tau \in N \text{, } \Theta(F) = \tau \).
3. if \( F : x_1...x_k \rightarrow e \in \mathbb{R} \text{ and } \Theta \vdash F \vdash \sigma_1 \rightarrow ... \rightarrow \sigma_{i=k} \rightarrow q \) then either there exists \( j \) such that \( q_\infty \in \sigma_j \), or \( i = k \) and \( \Theta, x_1 \vdash \sigma_1, ..., x_k \vdash \sigma_k \vdash e \vdash q \).
4. \( \Theta \vdash t \vdash \sigma \).

**Lemma 2 (Isolated Non Terminals).** Given a non terminal \( F \) that has not ground type. Then if \( \Theta \) verifies properties 1 to 3, one cannot prove \( \Theta \vdash F \vdash q_\infty \).

**Proof (Proof of lemma 2).** The proof comes from the fact that \( \Theta \) verifies property

3. Assume \( \Theta \vdash F \vdash \sigma_1 \rightarrow ... \rightarrow \sigma_i \rightarrow q_\infty \). Property 3 states that if \( i < \text{arity}(F) \) then there exists \( j \) such that \( q_\infty \in \sigma_j \) in particular \( i \neq 0 \). Then, one cannot prove \( \Theta \vdash F \vdash q_\infty \).

**Lemma 3 (Non fully-applied terminals).** Let \( \Theta \) be an environment that verifies properties 1 to 3, let \( F \) be a non terminal that has not ground type and let \( t = F \sigma_{1...1} \) with \( i < \text{arity}(F) \). If \( \Theta \vdash F \vdash t_1...t_l \vdash q_\infty \) then there exists \( j \leq i \) such that \( \Theta \vdash t_i \vdash q_\infty \).

**Proof (Proof of Lemma 3).** We prove by induction on \( l \) the following more general result: if \( \Theta \vdash F \vdash t_1...t_i \vdash \sigma_{i+1} \rightarrow ... \rightarrow \sigma_i \rightarrow q_\infty \) with \( F \in N \) and \( i < \text{arity}(F) \), then there exists \( j \leq i \) such that \( \Theta \vdash t_j \vdash q_\infty \) if \( j \leq l \) or \( q_\infty \in \sigma_j \) if \( j > l \).

If \( l = 1 \), then the rule we used to prove \( \Theta \vdash F \vdash t_1 \vdash \sigma_2 \rightarrow ... \rightarrow \sigma_i \rightarrow q_\infty \) could not be \((q_\infty)\) since one cannot prove \( \Theta \vdash F \vdash q_\infty \). If the rule we used were \((q_\infty)\), then \( q_\infty \in \sigma_2 \). If it is rule \((App)\) then \( \Theta \vdash F \vdash \sigma_1 \rightarrow \sigma_2 \rightarrow ... \rightarrow \sigma_i \rightarrow q_\infty \) and \( \Theta \vdash t_1 \vdash \sigma_1 \), and since \( i < \text{arity}(F) \), property 3 states that there exists \( j < l \) such that \( \Theta \vdash t_j \vdash q_\infty \) if \( j = 1 \), \( q_\infty \in \sigma_j \) elseweary. These are the only rules we could have applied.

If \( l > 1 \). If we applied rule \((q_\infty)\) then \( \Theta \vdash F \vdash t_1...t_{l-1} \vdash q_\infty \) by induction hypothesis there exists \( j \leq l - 1 \) such that \( \Theta \vdash t_j \vdash q_\infty \). If we applied rule
(q_\infty \rightarrow q_\infty), \text{ then } q_\infty \in \sigma_{i+1}. \text{ If we applied rule (App) then } \Theta \vdash F \ t_1...t_{l-1} \sigma_1 \rightarrow \sigma_{i+1} \rightarrow \ldots \rightarrow \sigma_i \rightarrow q_\infty \text{ and } \Theta \vdash t \sigma_1 \text{ then by induction hypothesis there exists } j \leq i \text{ such that either } \Theta \vdash t_j \vdash q_\infty \text{ if } j < l \text{ or } q_\infty \in \sigma_j \text{ if } j \geq l. \text{ If } j \leq l \text{ then either } j < l \text{ and then } \Theta \vdash t_j \vdash q_\infty, \text{ or } j = l \text{ and } q_\infty \in \sigma_j \text{ hence } \Theta \vdash t_j \vdash q_\infty, \text{ if } j > l \text{ then } q_\infty \in \sigma_j \text{ if } j \geq l.

**Lemma 4 (Redexes and q_\infty).**

(1) If \Theta \text{ verifies properties 1 to 3, then given a term } t, \text{ if } \Theta \vdash t \vdash q_\infty \text{ then either } t \text{ contains a redex } r \text{ such that } \Theta \vdash r \vdash q_\infty. \text{ In particular, if } t \text{ does not contain any redex, then one cannot prove } \Theta \vdash t \vdash q_\infty.

(2) If \Theta \text{ verifies properties 1 to 3, then given a term } t, \text{ if } t \text{ contains a redex } r \text{ such that } \Theta \vdash r \vdash q_\infty \text{ then one can prove } \Theta \vdash t \vdash q_\infty.

**Proof (Lemma 4).** We prove (1) by induction on \( t \). Assume that \( \Theta \vdash t \vdash q_\infty \).

If \( t : o \) then either \( t = F \ t_1...t_k \in \text{ in which case } t \text{ is the redex } r, \text{ or } t = a \ t_1...t_k \text{ then the only rule we could have applied to prove } \Theta \vdash t \vdash q_\infty \text{ is } (\Sigma), \text{ then there exists } t : o \text{ such that } \Theta \vdash t \vdash q_\infty, \text{ and the result comes by induction.}

If \( t : \tau \) with \( \tau \neq o \). We could not have \( t = F \in \mathcal{N} \) since one cannot prove \( \Theta \vdash F \vdash q_\infty \) if \( F \) has not ground type. If \( t = a \ t_1...t_i \) then again there exists \( t : o \) such that \( \Theta \vdash t \vdash q_\infty \), and the result comes by induction. If \( t = F \ t_1...t_i \), \( F \) has not ground type, and \( i < \text{arity}(F) \) since \( t \) has not ground type. Then Lemma 3 states that there exists \( t_j \) such that \( \Theta \vdash t_j \vdash q_\infty \) and the result comes by induction.

To prove (2), assume that there is a redex \( r \) such that \( \Theta \vdash r \vdash q_\infty \) and \( t = C[r] \). We prove the result by induction on \( C[\bullet] \). If \( C[\bullet] = \bullet \), then \( t = r \) therefore \( \Theta \vdash r \vdash q_\infty \). Assume \( t = t_1 t_2 \) with \( t_1 = C'[r] \) or \( t_2 = C'[r] \). If \( t_1 = C'[r] \) then by induction hypothesis, one can prove \( \Theta \vdash t_1 \vdash q_\infty \) and then, using rule \( q_\infty \dashv l, \Theta \vdash t_1 t_2 \vdash q_\infty \). If \( t_2 = C'[r] \), by induction hypothesis, one can prove \( \Theta \vdash t_2 \vdash q_\infty \), using rule \( (q_\infty \rightarrow q_\infty) \), we have \( \Theta \vdash t_1 \vdash q_\infty \rightarrow q_\infty \) and then, rule (App) gives us \( \Theta \vdash t_1 t_2 \vdash q_\infty \).

**Lemma 5 (Ground type terms and q_\bot).** if \( \Theta \) verifies properties 1 to 3, then if \( t : \tau \) and \( \Theta \vdash t : \sigma, \sigma : \tau \). In particular, if \( \Theta \vdash t \vdash q_\bot \), then \( t : o \).

**Proof (Lemma 5).** We can assume, without loss of generality that \( \sigma = \{\emptyset\} \) for some atomic mapping \( \emptyset \). We prove this by induction on the structure of \( t \).

If \( t = \alpha \) with \( \alpha \in \Sigma \cup \mathcal{N} \), then the only rules we can apply are (At), (\Sigma) and \( (q_\infty \rightarrow q_\infty) \) and they all satisfy the property.

If \( t = t_1 t_2 \) with \( t_1 : \tau_1 \rightarrow \tau \) and \( t_2 : \tau_2 \), then the rules we can apply are either \( (q_\infty) \) or (App). If it is \( (q_\infty) \) then we have proven \( \Theta \vdash t_1 \vdash q_\infty \) and \( q_\infty : \tau_2 \). If it is (App) it means that we have proven \( \Theta \vdash t_1 \vdash \sigma' \rightarrow \theta \) and \( \Theta \vdash t_2 \vdash \sigma', \) and by induction hypothesis, \( \sigma' : \tau_2 \) and \( \emptyset : \tau \).

**Theorem 6 (Soundness).** Let \( G \) be an HORS, and \( t \) be term (of any type), if \( \vdash (G,t) \vdash q_\infty \) (resp. \( \vdash (G,t) \vdash q_\bot \)) then \( P_\infty(t) \) (resp. \( P_\bot(t) \)) holds.

**Proof (Theorem 6).**
Lemma 6 (Type Preservation). Let $t : \tau$ be a term. If $\vdash (G, t) \triangleright \sigma$ and $t \rightarrow_{IO} t'$ then $\vdash (G, t') \triangleright \sigma$.

Proof (Lemma 6). Assume that $\vdash (G, t) \triangleright \sigma$ and $t \rightarrow_{IO} t'$. Let $\Theta$ be a witness environment of $\vdash (G, t) \triangleright \sigma$, we will prove that it is also a witness environment of $\vdash (G, t') \triangleright \sigma$ (we only have to check that $\Theta \vdash t' \triangleright \sigma$).

We know that $t = C[F \ s_1...s_k]$ and $t' = C[e[\forall i \ x_i \rightarrow s_i]]$ for some context $C[\bullet : \sigma] : \tau$. We proceed by induction on $C[\bullet]$.

If $C[\bullet] = \bullet$, we can assume without loss of generality that $\sigma = q \in Q$. We look at the proof of $\vdash (G, t') \triangleright \sigma$ and remark that either (1) the proof contains $\Theta \vdash F \triangleright \sigma_1 \rightarrow ... \rightarrow \sigma_k \rightarrow \sigma$ and $\Theta \vdash s_i \triangleright \sigma_i$ for all $i$ and the last steps are using the rule (App), or (2) the proving tree contains $\Theta \vdash F \ s_1...s_i \triangleright q_{∞} \rightarrow q_{∞}$ and $\Theta \vdash s_{i+1} \triangleright q_{∞}$, the last steps are using the rule (App) once and then only rule $(q_{∞}I)$. The former case is impossible: since $t \rightarrow_{IO} t'$ is an IO derivation there’s no redex in $s_i$ for all $i$, hence Lemma 4 shows that one cannot have $\Theta \vdash s_{i+1} \triangleright q_{∞}$.

Hence the proving tree contains $\Theta \vdash F \triangleright \sigma_1 \rightarrow ... \rightarrow \sigma_k \rightarrow \sigma$ and $\Theta \vdash s_i \triangleright \sigma_i$ for all $i$, then $\Theta, x_1 \triangleright \sigma_1, ... , x_k \triangleright \sigma_k \vdash e \triangleright q$, and if we replace all statements of $x_i \triangleright \sigma_i$ by the proof of $\Theta \vdash s_i \triangleright \sigma_i$, we obtain a proof of $\Theta \vdash e[\forall i \ x_i \rightarrow s_i] \triangleright \sigma$.

Now we proceed by induction step. Assume that $C = C'[\bullet] t_2$ or $C = t_1 C'[\bullet]$, then $t_1 = t_1 t_2$ with either $t_1 = C'[F \ s_1...s_k]$ or $t_2 = C'[F \ s_1...s_k]$. Then $t' = t_1 t_2'$ with respectively, either $t_1' = C'[e[\forall i \ x_i \rightarrow s_i]]$ and $t_2' = t_2$, or $t_1' = t_1$ and $t_2' = C'[e[\forall i \ x_i \rightarrow s_i]]$. Either way, by induction hypothesis, if $\Theta \vdash t_1 \triangleright \sigma_1$ (resp. $\Theta \vdash t_2 \triangleright \sigma_2$), then $\Theta \vdash t_1 \triangleright \sigma_1$ (resp. $\Theta \vdash t_2 \triangleright \sigma_2$). Assume we have proven $\Theta \vdash t \triangleright \sigma$. In order to do it, either we use rule $(q_{∞}I)$ or rule (App), either way we could use the same rule to prove $\Theta \vdash t' \triangleright \sigma$.

We extend in an intuitive way the properties $P_\bot$ and $P_{∞}$ to trees: if $t$ is a tree then $P_\bot(t) = \text{"t is \text{\"} a redex, hence } t^1 \text{ contains } \bot, \text{ therefore } P_{∞}(t^1) \text{ holds}$.

Lemma 7 (Weak Soundness). Given a term $t : \alpha$, (1) if $\vdash (G, t) \triangleright q_{∞}$, then $P_\bot(t^1)$ holds, (2) if $\vdash (G, t) \triangleright q_{∞}$, then $P_{∞}(t^1)$ holds.

Proof (Lemma 7). We can use Lemma 4 to prove (2): if $\vdash (G, t) \triangleright q_{∞}$ then $t$ contains a redex, hence $t^1$ contains $\bot$, therefore $P_{∞}(t^1)$ holds.

We prove (1) by induction on the structure of $t^1$. If $t^1 = \bot$ then $P_\bot(t^1)$ is true hence (1) holds. If $t^1 = \alpha$ with $\alpha \in \Sigma$, then $\alpha = \alpha$ and there is no rule that we can apply to state $\vdash (G, t) \triangleright q_{∞}$, hence (1) and (2) holds. If $t^1 = a t_1...t_k$ with $k > 0$, then $t = a t_1...t_k$ with $\alpha \in \Sigma$ and $t_i = t_i'$ for all $i$. For all environment $\Theta$, we show by induction that for all $i$, if $\Theta \vdash a t_1...t_i \triangleright \sigma'$ then $\sigma' = \sigma_1 \rightarrow ... \rightarrow \sigma_i \rightarrow q_{∞}$. The term $\alpha$ can only be judge by the rule $(\Sigma)$ hence it is true if $i = 0$, the term $(a t_1...t_i) t_{i+1}$ can be judge by rules $(q_{∞})$, $(q_{∞} \rightarrow q_{∞})$ and (App) and by induction hypothesis, in all three cases, we get $\Theta \vdash a t_1...t_i \triangleright q_{∞}$ for some $l$. In particular, we don’t have $\vdash (G, t) \triangleright q_{∞}$, hence (1) holds.

Using Lemma 1 and 2, in order to prove Theorem 6 we can assume that $t : \alpha$. We prove it by contradiction. Assume that $\vdash (G, t) \triangleright q_{∞}$ but $P_{∞}(t)$ doesn’t hold. Then it means that $||G_t||$ is finite and contains only terminals. Since it’s
finite, there exists a finite IO derivation from $t$ that leads to $|G_t|: t \rightarrow^*_{IO} |G_t|$, hence using Lemmas 6 and 7 we can prove $\mathcal{P}_\infty((|G_t|)^\perp)$, but since $|G_t|$ is a tree, $|G_t| = (|G_t|)^\perp$, hence $|G_t|$ is infinite or contains $\bot$ which raises a contradiction.

We treat the case $\vdash (G, t) \triangleright q_{\bot}$ the same way: Assume that $\vdash (G, t) \triangleright q_{\bot}$ but $\mathcal{P}_\bot(t)$ doesn’t hold. Then it means that $|G_t|$ contains some terminals. Then there exists a finite IO derivation from $t$ that leads to a term $t'$ such that $t'_{\bot} \neq \bot$: $t \rightarrow^*_{IO} t'$, hence using Lemmas 6 and 7 we can prove $\mathcal{P}_\bot((t')_{\bot})$ which is false.

**Theorem 7** (Completeness). Let $G$ be an HORS, if $\mathcal{P}_\infty(t)$ (resp. $\mathcal{P}_\bot(t)$) holds then $\vdash (G, t) \triangleright q_{\infty}$ (resp. $\vdash (G, t) \triangleright q_{\bot}$).

**Proof** (Proof of Theorem 7).

Using Lemma 4 we can assume without loss of generality that $t$ has ground type.

We recall the properties that an environment $\Theta$ has to satisfy in order to be a witness of $\vdash (G, t) \triangleright \sigma$.

1. $Dom(\Theta) = \mathcal{N}$,
2. $\forall F : \tau \in \mathcal{N} \Theta(F) :: \tau$,
3. if “$F \ x_1...x_k \rightarrow e'' \in \mathcal{R}$ and $\Theta \vdash F \triangleright \sigma_1 \rightarrow ... \rightarrow \sigma_{i\leq k} \rightarrow q$ then either there exists $j$ such that $q_{\infty} \in \sigma_j$, or $i = k$ and $\Theta \vdash \sigma_1,...,\sigma_k \triangleright e \triangleright q$,
4. $\Theta \vdash t \triangleright \sigma$.

Let $\mathcal{E}$ be the set of environment that matches properties 1 and 2. Let $F : \mathcal{E} \rightarrow \mathcal{E}$ be a mapping such that for all $F : \tau_1 \rightarrow ... \rightarrow \tau_k \rightarrow o \in \mathcal{N}$, if $F \ x_1...x_k \rightarrow e \in \mathcal{R}$ then,

$$
\mathcal{F}(\Theta)(F) = \{ \sigma_1 \rightarrow ... \rightarrow \sigma_k \rightarrow q \mid q \in Q \land \forall i \sigma_i :: \tau_i \land \Theta, x_1 \triangleright \sigma_1,...,x_k \triangleright \sigma_k \vdash e \triangleright q \} \\
\cup \{ \sigma_1 \rightarrow ... \rightarrow \sigma_{i\leq k} \rightarrow q_{\infty} \land \forall i \sigma_i :: \tau_i \land \exists j q_{\infty} \in \sigma_j \} \\
\cup \{ \sigma_1 \rightarrow ... \rightarrow \sigma_k \rightarrow q_{\bot} \mid \forall i \sigma_i :: \tau_i \land \exists j q_{\infty} \in \sigma_j \}.
$$

Let $\Theta_0 \in \mathcal{E}$ be the environment such that, for all $F : \tau = \tau_1 \rightarrow ... \rightarrow \tau_k \rightarrow o \in \mathcal{N}$, $\Theta_0(F)$ is defined and contains all atomic mappings $\theta :: \tau$. Notice that:

$$
\Theta_0(F) = \{ \sigma_1 \rightarrow ... \rightarrow \sigma_k \rightarrow q \mid q \in Q \land \forall i \sigma_i :: \tau_i \} \cup \{ \sigma_1 \rightarrow ... \rightarrow \sigma_{i\leq k} \rightarrow q_{\infty} \land \forall j \sigma_j :: \tau_j \}.
$$

**Lemma 8** (Universal Witness). There exists $m \in \mathbb{N}$ such that the judgment $\vdash (G, t) \triangleright \sigma$ holds if and only if $\mathcal{F}^m(\Theta_0) \vdash t \triangleright \sigma$ (This is Proposition 1 with $\Theta^* = \mathcal{F}^m(\Theta_0)$).

**Proof** (Proof of Lemma 8). We define the partial order $\sqsubseteq$ on $\mathcal{E}$ such that $\Theta_1 \sqsubseteq \Theta_2$ if and only if, for all $F \in \mathcal{N}$, $\Theta_1(F) \subseteq \Theta_2(F)$. Note that if $\Theta_1 \sqsubseteq \Theta_2$ and $\Theta_1 \vdash t \triangleright \sigma$ then $\Theta_2 \vdash t \triangleright \sigma$. $\Theta_0(F)$ contains all atomic mappings $\theta :: \tau$, hence $\Theta_0$ is a maximum of $\mathcal{E}$ with respect to $\sqsubseteq$. Note that the mapping $\mathcal{F}$ is monotonic with respect to $\sqsubseteq$ (i.e. if $\Theta \sqsubseteq \Theta'$ then $\mathcal{F}(\Theta) \sqsubseteq \mathcal{F}(\Theta')$). Given $\Theta \in \mathcal{E}$, we say that $\Theta$ is a post-fixpoint of $\mathcal{F}$ if and only if $\Theta \subseteq \mathcal{F}(\Theta)$. Remark that being a post-fixpoint of $\mathcal{F}$ is the same as verifying property 3.
Since $\Theta_0$ is a maximum of $\mathcal{E}$, and $\mathcal{F}(\Theta_0) \in \mathcal{E}$, then $\Theta_0 \sqsupseteq \mathcal{F}(\Theta_0)$, therefore, since $\mathcal{F}$ is monotonic, $\Theta_0 \sqsupseteq \mathcal{F}(\Theta_0) \sqsupseteq \mathcal{F}^2(\Theta_0) \sqsupseteq \ldots$. Because $\mathcal{E}$ is finite, there exists $m$ such that $\mathcal{F}^m(\Theta_0) = \mathcal{F}^{m+1}(\Theta_0)$, in particular $\mathcal{F}^m(\Theta_0)$ is a post-fixpoint of $\mathcal{F}$, hence it verifies properties 1, 2, and 3.

Take a witness $\Theta$ of $\vdash (G, t) \triangleright \sigma$. $\Theta$ is a post-fixpoint of $\mathcal{F}$, and since $\mathcal{F}^m(\Theta_0)$ is the greatest post-fixpoint, $\mathcal{F}^m(\Theta_0) \sqsupseteq \Theta$, hence $\mathcal{F}^m(\Theta_0) \vdash t \triangleright \sigma$, thus $\mathcal{F}^m(\Theta_0)$ is a witness of $\vdash (G, t) \triangleright \sigma$.

Let $G^{(m)} = (\mathcal{V}, \Sigma, \mathcal{N}^{(m)} \cup \{ \text{Void} : \alpha \}, \mathcal{R}^{(m)}, I)$ be the scheme such that $\mathcal{N}^{(m)} = \bigcup_{0 \leq i \leq m} \{ F_i \mid F \in \mathcal{N} \}$. For all $F$ $x_1 \ldots x_k \rightarrow e \in \mathcal{R}$, $\mathcal{R}^{(m)}$ contains the following rewrite rules:

$$F_i \ x_1 \ldots x_k \rightarrow e_{[\forall H \in \mathcal{N} \ H \rightarrow H_{i-1}]} \quad \text{for } i > 0$$

$$F_0 \ x_1 \ldots x_k \rightarrow e_{[\forall H \in \mathcal{N} \ H \rightarrow H_0]}$$

$$F_0 \ x_1 \ldots x_k \rightarrow \text{Void}$$

$$\text{Void} \rightarrow \text{Void}$$

Notice that $\text{Void}$ here is a non-terminal of order 0 that produce itself. Hence applying its rewrite rule to a term would produce the same term. In the following we forbid this rule to be applied. $G^{(m)}$ with this restriction is said to be recursion free, i.e. the graph whose vertices are the non terminals and where there is an edge from $F$ to $G$ if and only if there exist an allowed rewrite rule $F \ x_1 \ldots x_k \rightarrow e$ such that $e$ contains an occurrence of $G$, has no loop. Such non-recursive schemes are known to be strongly normalizing, i.e. for any term $t$ all derivations using only allowed rewrite rules are finite. In particular, there exists a finite IO derivation $t \rightarrow^{*\text{IO}} t'$ such that $(t')^\dagger = \|t\|_{1\text{O}}$.

We define the environment $\Theta^{(m)}$ on $\mathcal{N}^{(m)} \cup \{ \text{Void} : \alpha \}$: for all $F \in \mathcal{N}$, for all $i \leq j$, $\Theta^{(m)}(F_i) = \mathcal{F}^i(\Theta_0)(F)$ and $\Theta^{(m)}(\text{Void}) = \bigwedge \{ q_\infty, q_\bot \}$.

**Lemma 9.** Given two terms $t, t' \in T(\Sigma \cup \mathcal{N}^{(m)})$ such that $t \rightarrow^{*\text{IO}} t'$ is allowed in $G^{(m)}$. If $\Theta^{(m)} \vdash t \triangleright \sigma$, then $\Theta^{(m)} \vdash t \triangleright q$.

**Proof (Lemma 9).** We proceed by induction on the structure of $t$. We prove the initial case: $t = F_i \ s_1 \ldots s_k$ and $t' = e_{[\forall i \ x_i \rightarrow s_i]}$, with $F_i \ x_1 \ldots x_k \rightarrow e \in \mathcal{R}$. We assume without loss of generality that $\sigma$ is atomic, hence $\sigma = q \in \mathcal{L}$. Let $s_i$ be the union of all mappings assigned to $s_i$ in the proof of $\Theta^{(m)} \vdash e_{[\forall i \ x_i \rightarrow s_i]} \triangleright q$. Then we have $\Theta^{(m)}$, $x_1 \triangleright \sigma_1$, ..., $x_k \triangleright \sigma_k \vdash e \triangleright q$. Let $\Theta' = \Theta^{(m)}$, $F_i \triangleright \sigma_1 \rightarrow \ldots \rightarrow \sigma_k \rightarrow q$. Since $\Theta' \vdash F_i \triangleright \sigma_1 \rightarrow \ldots \rightarrow \sigma_k \rightarrow q$, and $\Theta' \vdash F_i \triangleright \sigma_1 \rightarrow \ldots \rightarrow \sigma_k \rightarrow q$ (indeed, $\Theta^{(m)} \subseteq \Theta'$), we can prove $\Theta' \vdash t \triangleright q$. If $l = 0$, by definition, $F \triangleright \sigma_1 \rightarrow \ldots \rightarrow \sigma_k \rightarrow q \in \Theta_0(F)$, and since $\mathcal{G}^{(m)}(F_0) = \Theta_0(F)$, we have $F \triangleright \sigma_1 \rightarrow \ldots \rightarrow \sigma_k \rightarrow q \in \Theta^{(m)}(F_0)$, hence $\Theta' = \Theta^{(m)}$. If $l > 0$, $e$ only contains terminals of the form $G_{l-1}$, then we can transform the proof of $\Theta^{(m)}$, $x_1 \triangleright \sigma_1$, ..., $x_k \triangleright \sigma_k \vdash e \triangleright q$ to obtain a proof of $\mathcal{G}^{l-1}(\Theta_0), x_1 \triangleright \sigma_1, \ldots, x_k \triangleright \sigma_k \vdash e_{[\forall G \in \mathcal{N} \ G_{l-1} \rightarrow G]} \triangleright q$. Then by definition of $\mathcal{G}$, $F \triangleright \sigma_1 \rightarrow \ldots \rightarrow \sigma_k \rightarrow q \in \mathcal{G}^{l}(\Theta_0)(F)$, and since $\mathcal{G}^{(m)}(F_i) = \mathcal{G}^{l}(\Theta_0)(F)$, we have $\sigma_1 \rightarrow \ldots \rightarrow \sigma_k \rightarrow q \in \Theta^{(m)}(F_i)$, hence $\Theta' = \Theta^{(m)}$. Thus $\Theta^{(m)} \vdash t \triangleright q$. 
Lemma 10 (Terms that contains Void). Given a term \( t \) that contains the non terminal Void, one can prove \( \Theta^{(m)} \vdash t \triangleright q_\infty \).

Proof (Lemma 10). We prove the result by induction on the structure of \( t \) : if \( t = \text{Void} \) then we use rule (At) , if \( t = G_j t_1...t_k \) with \( G_j \in \mathcal{N}^{(m)} \) and \( t_k \) contains \( \text{Void} \) for some \( k \leq i \), then by induction hypothesis, \( \Theta^{(m)} \vdash t_k \triangleright q_\infty \). Using rule (Set) we have, for all \( k' \neq k \), \( \Theta^{(m)} \vdash t_{k'} \triangleright \{ \} \). We have by construction \( \Theta^{(m)} \vdash a \triangleright \sigma_1 \rightarrow ... \rightarrow \sigma_i \rightarrow q_\infty \) for \( \sigma_j = q_\infty \) and \( \sigma_i = \{ \} \) for all \( i \neq j \). Then, if we apply k times rule (App) we can prove \( \Theta^{(m)} \vdash a t_1...t_k \triangleright q_\infty \). By definition, if \( \sigma_{i\neq k} = \{ \} \) and \( \sigma_k = q_\infty \), we have \( \sigma_1 \rightarrow ... \rightarrow \sigma_i \rightarrow q_\infty \) with \( \sigma_{i\neq k} = \{ \} \) and \( \sigma_k = q_\infty \).

Lemma 11 (Weak Completeness). Given a term \( t : o \in \mathcal{T}(\Sigma \cup \mathcal{N}^{(m)}) \), if \( \mathcal{P}_\perp(t^+) \) (resp. \( \mathcal{P}_\infty(t^+) \)) holds and if there exists no IO allowed rewrite rule we can apply in \( t \), then \( \Theta^{(m)} \vdash t \triangleright q_\perp \) (resp. \( \emptyset \vdash t^+ \triangleright q_\infty \)).

Proof (Lemma 11).

We prove both results simultaneously by induction on the structure of \( t \). If \( t = \text{Void} \) then one can directly prove both result using rule (At). We know that \( t \neq a \) with \( a \in \Sigma \) since \( \mathcal{P}_\perp(a) \) (resp. \( \mathcal{P}_\infty(a) \)) do not hold. If \( t = a t_1...t_k \) we know that \( \mathcal{P}_\perp(t^-) \) doesn’t hold, assume that \( \mathcal{P}_\infty(t^-) \) holds. Since \( t_k \) contains \( \perp \), there exists \( j \) such that \( t_j \) contains \( \perp \), i.e. \( \mathcal{P}_\perp(t_j^+) \). Furthermore there exists no allowed rewrite rule we can apply on \( t_j \), elsewey we could apply it in \( t \). Therefore, by induction hypothesis, \( \Theta^{(m)} \vdash t_j \triangleright q_\infty \). Using rule (Set) we have, for all \( i \neq j \), \( \Theta^{(m)} \vdash t_i \triangleright \{ \} \). Rule (Σ) gives \( \Theta^{(m)} \vdash a \triangleright \sigma_1 \rightarrow ... \rightarrow \sigma_k \rightarrow q_\infty \) for \( \sigma_j = q_\infty \) and \( \sigma_i = \{ \} \) for all \( i \neq j \). Then, if we apply k times rule (App) we can prove \( \Theta^{(m)} \vdash a t_1...t_k \triangleright q_\infty \).

Assume now that \( t = F_i t_1...t_k \) with \( F_i \in \mathcal{N}^{(m)} \). Since there exists no IO allowed rewrite rule we can apply in \( t \) it means that there exists \( i \) such that \( t_i \) contains a redex, but this redex can’t be applied, in other words, \( t_i \) contains \( \text{Void} \). Using Lemma 10 we have \( \Theta^{(m)} \vdash t_i \triangleright q_\infty \). By definition, if \( \sigma_{i\neq i} = \{ \} \) and \( \sigma_i = q_\infty \), we have \( \sigma_1 \rightarrow ... \rightarrow \sigma_k \rightarrow q \in \Theta^{(m)}(F_i) \) for \( q \in \mathcal{Q} \). Hence one can prove \( \Theta^{(m)} \vdash t \triangleright q \).

Now, we can prove the Theorem. Given a term \( t : o \in \mathcal{T}(\Sigma \cup \mathcal{N}) \), assume that \( \mathcal{P}_\perp(t) \) (resp. \( \mathcal{P}_\infty(t) \)) holds. We define the term \( t^{(m)} : o = t_{\{F \in \mathcal{N} \mid F \triangleright F_i \}} \in \mathcal{T}(\Sigma \cup \mathcal{N}) \).
$N^{(m)}$. Notice that $\|G_i\|_{IO}$ is obtained by turning some subtrees of $\|G_i\|_{IO}$ into $\bot$. Hence, $P_\bot(t^{(m)})$ (resp. $P_\infty(t^{(m)})$) holds. Let $t': o \in T(\Sigma \uplus N^{(m)} \uplus \{\bot\})$ such that $t^{(m)} \Rightarrow t'$ and $(t')^\bot = \|G_i^{(m)}\|_{IO}$ (we have seen previously that such $t'$ exists). Lemma 11 states that $\Theta^{(m)} \vdash t' \triangleright q_\bot$ (resp. $\Theta^{(m)} \vdash t' \triangleright q_\infty$). then, Lemma 9 shows that $\Theta^{(m)} \vdash t^{(m)} \triangleright q_\bot$ (resp. $\Theta^{(m)} \vdash t^{(m)} \triangleright q_\infty$). Since non terminals in $t^{(m)}$ have the form $F_m$, if we restrict the domain of $\Theta^{(m)}$ only to $\{F_m \mid F \in N\}$ the proof still holds, furthermore in this proof, if we remove all “$m$” subscripts, we get $F^m(\Theta_0) \vdash t \triangleright q_\bot$ (resp. $F^m(\Theta_0) \vdash t \triangleright q_\infty$). Lemma 8 allows us to conclude: $\vdash (G, t) \triangleright q_\bot$ (resp. $\vdash (G, t) \triangleright q_\infty$).

C Selfcorrecting Scheme

Proof (Theorem 4).

Lemma 12 (Equality of Trees). Let $t : o \in T(\mathcal{V} \uplus N)$ be a term, then $t^\bot = (t^\bot)^\bot$.

Proof (Lemma 12). We prove it by induction on the structure of $t : o$. If $t = F_{t_1...t_k}$ with $F \in N$ then $t^\bot = \bot$ and $t^\bot = F[t_1]^\bot...[t_k]^\bot$, then $(t^\bot)^\bot = \bot = t^\bot$.

If $t = a_{t_1...t_k}$ with $a : o^k \rightarrow o \in \Sigma$ and $t_i : o$ for all $i$, then $t^\bot = a[t_1]^\bot...[t_k]^\bot$, $t^\bot = a^{t_1^\bot...t_k^\bot}$ and $(t^\bot)^\bot = a^{(t_1^\bot)^\bot...((t_k^\bot)^\bot)}$. By induction hypothesis, for all $i$, $(t_i^\bot)^\bot = t_i$, then $(t^\bot)^\bot = a^{t_1^\bot...t_k^\bot}$.

Lemma 13 (Label Conservation in Rewrite Rules). Given a term $t : \tau = \tau_1 \rightarrow ... \rightarrow \tau_k \rightarrow o \in T(\Sigma \uplus N)$, such that $t = F_{s_1...s_k}$ with $F \in N$, and $t$ is an $IO$-relevant redex. Note that $t^\bot = F[s_1]^\bot...[s_k]^\bot$, if $F \rightarrow e \in \mathcal{R}$, let $t' = e[\sigma_i^j \rightarrow s_i^j]$ and $s = e^\Theta[\sigma_i^j \rightarrow s_i^j]$ with $\Theta^\sigma_{\{\sigma_i^j \rightarrow s_i^j\}}[\{s_i\}]$ for all $i$ (in particular, $t \rightarrow t'$ and $t^\bot \rightarrow s$).

We have $s = (t')^\bot$.

Proof (Lemma 13). Besides the labeling, by construction, $s$ matches $t'$. Take a subterm $e'$ of $e$, if one can prove $\Theta^\sigma \vdash e'[\sigma_i^j \rightarrow s_i^j]$, then one can prove $\Theta^\sigma \vdash e'[\sigma_i^j \rightarrow s_i^j]$, hence $s$ is well labeled, therefore $s = (t')^\bot$. Then $t^\bot \Rightarrow t^\bot (t')^\bot$.

Given two terms $t, t'$, we write $t \Rightarrow_{IO} t'$ if $t'$ is obtained by applying in parallel all $IO$ rewrite availables in $t$. Formally, we define it inductively: if $t$ is an $IO$-relevant redex and $t'$ is the term obtained by rewriting this redex then $t \Rightarrow_{IO} t'$. If $t$ is not an $IO$ redex and $t = t_1 t_2$ then $t \Rightarrow_{IO} t'$ if and only if:

- either there exists $t_1', t_2'$ such that $t_1 \Rightarrow_{IO} t_1'$ and $t_2 \Rightarrow_{IO} t_2'$, and $t' = t_1 t_2'$,
- or there exists $t_1'$ such that $t_1 \Rightarrow_{IO} t_1'$ but no $t_2'$ such that $t_2 \Rightarrow_{IO} t_2'$ and $t' = t_1' t_2$,
- or there exists $t_2'$ such that $t_2 \Rightarrow_{IO} t_2'$ but no $t_1'$ such that $t_1 \Rightarrow_{IO} t_1'$ and $t' = t_1 t_2'$.
Notice that if such \( t' \) exists then it is unique, and it exists if and only if \( t \) contains a redex. The \( \cdot \Rightarrow_{IO} \cdot \) relation is known as parallel rewrite, and from a term \( t : \sigma \) the unique associated parallel derivation \( t \Rightarrow_{IO} t_1 \Rightarrow_{IO} t_2 \Rightarrow_{IO} \ldots \) leads to the tree \( ||G_t|| \).

**Lemma 14 (Coincidence of Parallel Derivation).** Given a terms \( t \in T(\Sigma \uplus N) \), and some conjunctive mappings \( \sigma_1, \ldots, \sigma_k \). There exists \( t' \in T(\Sigma \uplus N) \) such that \( t \Rightarrow_{IO} t' \), if and only if there exists \( s' \in T(\Sigma' \uplus N') \) such that \( t'^{+}_{\sigma_1 \ldots \sigma_k} \Rightarrow_{IO} s' \). Furthermore, if it is true, then \( s' = (t')^{+}_{\sigma_1 \ldots \sigma_k} \).

**Proof (Lemma 14).** The first part of the result comes from the observation that \( t \) contains a redex if and only if \( t^{+}_{\sigma_1 \ldots \sigma_k} \) contains a redex. We prove the second part by induction. If \( t \) is an IO-relevant redex, \( t^{+}_{\sigma_1 \ldots \sigma_k} \) is too, and Lemma 13 proves the result. If \( t = t_1 t_2 \), \( t \) is not an IO-relevant redex a then \( t^{+}_{\sigma_1 \ldots \sigma_k} = t_1^{+}_{\sigma_1 \ldots \sigma_k} \uplus t_2^{+}_{\sigma_1 \ldots \sigma_k} \) and \( t^{+}_{\sigma_1 \ldots \sigma_k} \) is not an IO-relevant redex. Assume that \( t \Rightarrow_{IO} t' \) then,

- either there exists \( t'_1, t'_2 \) such that \( t_1 \Rightarrow_{IO} t'_1 \) and \( t_2 \Rightarrow_{IO} t'_2 \), and \( t' = t'_1 t'_2 \),
- or there exists \( t'_1 \) such that \( t_1 \Rightarrow_{IO} t'_1 \) but no \( t'_2 \) such that \( t_2 \Rightarrow_{IO} t'_2 \) and \( t' = t'_1 \),
- or there exists \( t'_2 \) such that \( t_2 \Rightarrow_{IO} t'_2 \) but no \( t'_1 \) such that \( t_1 \Rightarrow_{IO} t'_1 \) and \( t' = t'_2 \).

By induction hypothesis, \( t_i \Rightarrow_{IO} t'_i \) if and only if \( t'_i \Rightarrow_{IO} t^{+}_i \) for \( i \in \{1, 2\} \), hence

- either there exists \( t'_1, t'_2 \) such that \( t_1^{+}_{\sigma_1 \ldots \sigma_k} \Rightarrow_{IO} t'_1^{+} \) and \( t_2^{+}_{\sigma_1 \ldots \sigma_k} \Rightarrow_{IO} t'_2^{+} \) for all \( j \), and \( (t')^{+}_{\sigma_1 \ldots \sigma_k} = (t'_1)^{+}_{\sigma_1 \ldots \sigma_k} \uplus (t'_2)^{+}_{\sigma_1 \ldots \sigma_k} \),
- or there exists \( t'_1 \) such that \( t_1^{+}_{\sigma_1 \ldots \sigma_k} \Rightarrow_{IO} t'_1^{+} \) but no \( t'_2 \) such that \( t_2^{+}_{\sigma_1 \ldots \sigma_k} \Rightarrow_{IO} t'_2^{+} \) for all \( j \), and \( (t')^{+}_{\sigma_1 \ldots \sigma_k} = (t'_1)^{+}_{\sigma_1 \ldots \sigma_k} \uplus (t'_2)^{+}_{\sigma_1 \ldots \sigma_k} \),
- or there exists \( t'_2 \) such that \( t_2^{+}_{\sigma_1 \ldots \sigma_k} \Rightarrow_{IO} t'_2^{+} \) but no \( t'_1 \) such that \( t_1^{+}_{\sigma_1 \ldots \sigma_k} \Rightarrow_{IO} t'_1^{+} \) for all \( j \), and \( (t')^{+}_{\sigma_1 \ldots \sigma_k} = (t'_1)^{+}_{\sigma_1 \ldots \sigma_k} \uplus (t'_2)^{+}_{\sigma_1 \ldots \sigma_k} \). Therefore, \( t^{+}_{\sigma_1 \ldots \sigma_k} \Rightarrow_{IO} (t')^{+}_{\sigma_1 \ldots \sigma_k} \).

Given a term \( t : \sigma \) let \( t \Rightarrow_{IO} t_1 \Rightarrow_{IO} t_2 \Rightarrow_{IO} \ldots \) be the parallel derivation associated to it. Thanks to Lemma 14 we know that the parallel derivation associated to \( t^{+} \) is \( t^{+} \Rightarrow_{IO} t^{+}_1 \Rightarrow_{IO} t^{+}_2 \Rightarrow_{IO} \ldots \), then \( \|G'_t\|_{IO} \) is the limit of \( (t^{+}_i)^{\perp} \) then \( (\|G'_t\|_{IO})^{\perp} \) is the limit of \( (t^{+}_i)^{\perp} = (t_i)^{\perp} \). Then \( \|G'_t\|_{IO} = \|G_t\|_{IO} \).

**Proof (Lemma 14).** Take a term \( t \in T(\Sigma \uplus N) \) we define \( \text{void}(t) \in T(\Sigma \uplus N \uplus \{\text{Void}\}) \) as the set of terms obtained by substituting some redex \( r \) in \( t \) such that \( \|G'_t\|_{IO} = \perp \) by \( \text{Void} \). From the definition comes that if \( t' \in \text{void}(t) \) then \( (t')^{\perp} = t^{\perp} \).

Given a term \( t \in T(\Sigma \uplus N) \) and an IO derivation associated \( t = t_1 \Rightarrow_{IO} t_2 \Rightarrow_{IO} \ldots \) in \( G' \) we construct by induction an IO derivation in \( G'' \) \( t = t'_1 \Rightarrow_{IO} t'_2 \Rightarrow_{IO} \ldots \) such that for all \( i t'_i \in \text{void}(t_i) \). The initial step is straightforward:
$t \in \text{void}(t)$, Assume that $t'_i \in \text{void}(t_i)$, and assume that $t_i = C[F_{t_1...t_k}]$ and $t_{i+1} = C[e_{[x_i \mapsto t_i]}]$ with $F_{x_1...x_k} \to e \in \mathcal{R}'$. If this redex is a subterm of another one that is transformed by Void in $t'_i$ then we just rewrite this void obtaining $t'_{i+1} = t'_i$ by induction hypothesis, we still have $t'_{i+1} \in \text{void}(t_{i+1})$. If this redex is not transformed in $t'_i$ then we rewrite this redex, and either $\parallel F_{t_1...t_k} \parallel_{IO} = \bot$, in which case the semantics associated contains $q_\bot$ thanks to Theorem ??, and then $e_{[x_i \mapsto t_i]}$ is still a redex and is transformed to Void in $t'_{i+1}$ or $\parallel F_{t_1...t_k} \parallel_{IO} \neq \bot$ and no more transformation is added to create $t'_{i+1}$, in both cases $t'_{i+1} \in \text{void}(t_{i+1})$.

This result gives that $\parallel G' \parallel_{IO} \sqsubseteq \parallel G'' \parallel_{IO}$. Since $G''$ is obtained by only changing some redex into other that will produce $\bot$, it is clear that $\parallel G'' \parallel_{IO} \sqsubseteq \parallel G' \parallel_{IO}$, then $\parallel G' \parallel_{IO} = \parallel G'' \parallel_{IO}$.

**Proof ($\parallel G'' \parallel_{IO} = \parallel G'' \parallel_{IO}$).** We already know that $\parallel G'' \parallel_{IO} \sqsubseteq \parallel G'' \parallel_{IO}$, we just have to show that for all $t$, if there is $\bot$ at node $u$ in $\parallel G'' \parallel_{IO}$ then there is $\bot$ at node $u$ in $\parallel G'' \parallel_{IO}$. We show this by induction on the size of $u$.

If $\parallel G'' \parallel_{IO} = \bot$. Then $\parallel G'' \parallel_{IO} = \parallel G' \parallel_{IO} = \bot$, hence $[t] = \bot$, hence the only derivation in $G''$ from $t$ is $t \to \text{Void} \to \text{Void} \to \ldots$, therefore $\parallel G'' \parallel = \bot$. If $u = ju'$ then let $a$ be the terminal at the root of $\parallel G'' \parallel_{IO}$ then there exists an IO derivation $t \to a_{t_1...t_k}$, and $\parallel G'' \parallel_{IO}$ is equal to the subtree of $\parallel G'' \parallel_{IO}$ rooted at node $j$. Since $t \to a_{t_1...t_k}$, $\parallel G''_{t_j} \parallel$ is equal to the subtree of $\parallel G'' \parallel_{IO}$ rooted at node $j$ and by induction hypothesis, since there is $\bot$ at node $u'$ in $\parallel G'' \parallel_{IO}$, there is $\bot$ at node $u'$ in $\parallel G''_{t_j} \parallel$ hence there is $\bot$ at node $u$ in $\parallel G'' \parallel_{IO}$. 
