Reducing Higher-order Recursion Scheme Equivalence to Coinductive Higher-order Constrained Horn Clauses

Jerome Jochems

Department of Computer Science
University of Bristol*
Bristol, UK

`jerome.jochems@bristol.ac.uk`

Higher-order constrained Horn clauses (HoCHC) are a semantically-invariant system of higher-order logic modulo theories. With semi-decidable unsolvability over a semi-decidable background theory, HoCHC is suitable for safety verification. Less is known about its relation to larger classes of higher-order verification problems. Motivated by program equivalence, we introduce a coinductive version of HoCHC that enjoys a greatest model property. We define an encoding of higher-order recursion schemes (HoRS) into HoCHC logic programs. Correctness of this encoding reduces decidability of the open HoRS equivalence problem – and, thus, the $\lambda \mathbf{Y}$-calculus Böhm tree equivalence problem – to semi-decidability of coinductive HoCHC over a complete and decidable theory of trees.

1   Introduction

Cathcart Burn et al. [6] have proposed a promising logical framework for higher-order safety verification. They frame the search for "safe" program invariants as a satisfiability problem for systems of *HoCHC*: these *higher-order constrained Horn clauses* – which extend constrained Horn clauses to higher-order logic with constraints from a first-order background theory – aim to act as a universal setting in which disparate verification algorithms can be compared, independent of application or programming language.

Thanks to its higher-order predicates, the HoCHC fragment expresses certain invariants of higher-order programs quite directly. Even so, it retains many of the excellent algorithmic properties to which first-order constrained Horn clauses owe their suitability for first-order model checking [3, 2]. Given a semi-decidable background theory, HoCHC unsolvability (unsatisfiability) is semi-decidable [24, 23].

We study the relation between (the logic-program presentation of) HoCHC and *higher-order recursion schemes* (HoRS). Whilst higher-order *model checking* has grown out of the decidability of HoRS model checking [21, 15] and flourished, higher-order *program equivalence* is relatively underdeveloped; decidability of HoRS equivalence is a long-standing open problem [7, 22]. Note that the HoRS model checking safety problem can be solved via a decidable higher-order Datalog fragment of HoCHC [32].

A HoRS of order n is essentially an nth-order tree grammar: the trees generated at orders 0, 1 and 2 are regular trees, algebraic trees (i.e. those generated by context-free tree grammars), and hyperalgebraic trees, respectively [9]. These potentially infinite trees generated by HoRS correspond to (abstractions of) computation trees of higher-order functional programs.

Let us consider (deterministic) HoRS $\mathcal{G}_1$ and HoRS $\mathcal{G}_2$ in Figure 1 that both generate an infinite tree with the prefix on the right (by unfolding the rewrite rules ad infinitum, starting from S_1 and S_2, respectively). To determine whether these HoRS generate the same tree, we define a HoCHC logic program that contains one predicate R_N (of arity $n+1$) for each nonterminal symbol N (of arity n) in the

---

*Most of the work was concluded while the author was a Research Associate at the University of Oxford.

$$S_1 = G\,\text{zero}$$

$$G = \lambda x.\,\text{cons}\,(\text{succ}\,x)\,(G\,(\text{succ}\,x))$$

$$S_2 = F\,\text{succ}$$

$$F = \lambda\varphi.\,\text{cons}\,(\varphi\,\text{zero})\,(F\,(B\,\varphi\,\varphi))$$

$$B = \lambda\varphi\,\psi\,x.\,\varphi\,(\psi\,x)$$

(a) Their respective rewrite rules

(b) A common prefix of the trees generated by $\mathcal{G}_1$ and $\mathcal{G}_2$

1. Eliminate divergent ⊥-labelled leaves from HoRS (Sec 2.2.2)

2. Encode HoRS into coinductive HoCHC logic program (Sec 3)

3. Solve the two HoCHC instances from Sec 4.2 concurrently (outside the scope of this paper)

(c) The "decision" procedure for HoRS equivalence, pending semi-decidability
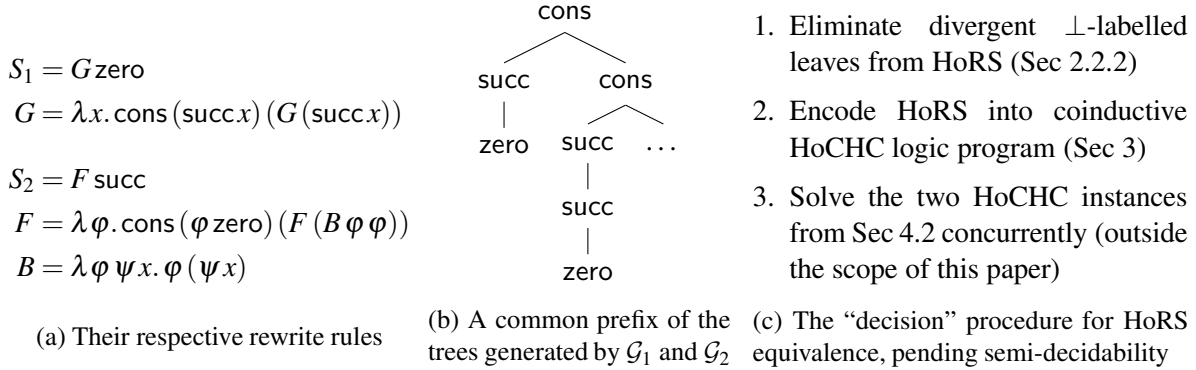
Figure 1: Example order-1 HoRS $\mathcal{G}_1$ and order-2 HoRS $\mathcal{G}_2$ over tree constructors $\{\text{cons},\text{succ},\text{zero}\}$

input HoRS. In particular, we want R_{S_1} (resp. R_{S_2}) to be the characteristic function of the tree generated from S_1 by $\mathcal{G}_1$ (resp. from S_2 by $\mathcal{G}_2$), so we can query the existence of a tree t such that $R_{S_1}\,t \wedge R_{S_2}\,t$.

Encoding HoRS into HoCHC (in "continuation-passing style") is natural, as this program shows:

$$R_{S_1} = \lambda r.\,\exists r_1.\,R_G\,r_1\,r \wedge (\text{zero} = r_1)$$

$$R_G = \lambda x'\,r.\,\exists r_1\,r_2\,r_3.\,(\text{cons}\,r_1\,r_2 = r) \wedge (\text{succ}\,x' = r_1) \wedge R_G\,r_3\,r_2 \wedge (\text{succ}\,x' = r_3)$$

$$R_{S_2} = \lambda r.\,R_F\,(\lambda y\,r'.\,\text{succ}\,y = r')\,r$$

$$R_F = \lambda\varphi'\,r.\,\exists r_1\,r_2\,r_3.\,(\text{cons}\,r_1\,r_2 = r) \wedge \varphi'\,r_3\,r_1 \wedge (\text{zero} = r_3) \wedge R_F\,(\lambda y\,r'.\,R_B\,\varphi'\,\varphi'\,y\,r')\,r_2$$

$$R_B = \lambda\varphi'\,\psi'\,x'\,r.\,\exists r_1\,r_2.\,\varphi'\,r_1\,r \wedge \psi'\,r_2\,r_1 \wedge (x' = r_2)$$

Each HoRS subterm is represented by a conjunct whose arguments are r_i bounded by a subsequent conjunct (if a tree) or are inlined (if higher type). Unfortunately though, this HoCHC logic program has no natural inductive interpretation. The empty assignment is a model (a fixpoint) of the program, because the program contains no "base cases" to break out of the recursion. In fact, there does not exist a HoCHC term such that the characteristic function of an infinite tree arises in its least fixpoint.

To tackle this disparity between HoCHC and HoRS, we define a *coinductive* HoCHC framework that enjoys a greatest model property (under the monotone interpretation). We interpret the above clauses coinductively over a complete and decidable background theory of trees first introduced by Maher [20].

In our example, R_{S_1} is assigned the characteristic function of the tree generated by $\mathcal{G}_1$ in the greatest model (and R_{S_2} of $\mathcal{G}_2$). We can (independently) query the existence of two identical and two distinct trees t_1 and t_2 such that $R_{S_1}\,t_1 \wedge R_{S_2}\,t_2$. Only two distinct such trees exist; after the common prefix in Figure 1, the trees deviate. The left children of cons have shape succ^n zero in t_1 and $\text{succ}^{2^{n-1}}$ zero in t_2, for $n \geq 1$.

This new framework allows us to characterise HoRS in HoCHC logic programs and, thus, reduce decidability of the HoRS equivalence problem to semi-decidability of coinductive HoCHC over a decidable background theory. This has implications for the $\lambda\mathbf{Y}$-calculus Böhm tree equivalence problem [7], which asks whether the Böhm trees of two given $\lambda\mathbf{Y}$-terms are equal; this problem is recursively equivalent to the HoRS equivalence problem and can also be reduced to semi-decidability of coinductive HoCHC.

Contributions. The (open) HoRS equivalence problem asks whether two given deterministic HoRS generate the same tree. We prove that decidability of this problem can be reduced to the (open) semi-decidability of coinductive HoCHC over a decidable background theory (see Figure 1c).

First, we prove that there is an algorithm that, given a HoRS (which may contain "diverging" ⊥-labelled nodes), returns its ⊥-free transform – i.e. a HoRS that generates the same tree, except that every

$\perp$-labelled node is replaced by the infinite linear tree b $(b$ $(b\cdots))$. The proof appeals to the logical reflectivity of HoRS with respect to properties definable in monadic second-order logic, in the sense of [5]. Notice that two HoRS are equivalent if and only if their respective $\perp$-free transforms are equivalent.

Next, we exhibit a natural encoding of HoRS into constrained logic programs, with the sort of individuals interpreted as the set of finite and infinite trees. Given two $\perp$-free HoRS, we define two instances of the coinductive HoCHC problem, call them positive and negative. We use Maher's first-order theory of equations of finite and infinite trees [20], which is complete and decidable, as the background theory. The positive and the negative problem instance share a logic program: the union of the respective HoRS-to-HoCHC encodings. The goal formulas of the problem instances are so designed that the two input HoRS are equivalent (resp. inequivalent) iff the positive (resp. negative) instance is solvable. Provided that the resulting coinductive HoCHC instances are semi-decidable, we obtain two semi-decision procedures, one for checking equivalence of the input HoRS and one for inequivalence. A decision procedure for the equivalence of the input HoRS could then be obtained by dovetailing the two semi-decision procedures.

Outline.   Building on Cathcart Burn et al.'s (inductive) HoCHC [6], we introduce coinductive HoCHC in Section 2.1. We define HoRS and their denotational semantics in Section 2.2, where we also prove the existence of an algorithm that generates the $\perp$-free transform of HoRS. We encode HoRS into constrained logic programs in Section 3. Section 4 shows how to use these HoRS-to-HoCHC encodings to reduce decidability of the HoRS equivalence problem to semi-decidability of coinductive HoCHC over Maher's complete and decidable theory of trees. Finally, we consider implications and related work in Section 5.

2   Preliminaries

2.1   Higher-order constrained Horn clauses

Following [6], we work in higher-order logic presented as a typed (sorted) lambda calculus. We follow their (monotone logic-program) definitions until we introduce the coinductive HoCHC decision problem.

2.1.1   Syntax

Sorts.   Given a sort ι of individuals (for example int), and a sort o of (boolean) truth values, *sorts* are just the simple types generated by $\sigma ::= \iota \mid o \mid \sigma \to \sigma$. *Relational sorts* (typically denoted by ρ) have the following restricted form: $\rho ::= o \mid \iota \to \rho \mid \rho \to \rho$.

Background theory.   Assume a fixed, first-order language over a first-order signature, consisting of distinguished subsets of first-order terms *Tm* and first-order formulas (or *constraints*) $(\varphi \in)$ *Fm*, and a first-order theory *Th* in which to interpret those. We fix a *standard model A* of *Th* we often leave implicit. We refer to this first-order language as the *constraint language*, and *Th* as the *background theory*.

Goal terms.   The class of well-sorted *goal terms* $\Delta \vdash G : \rho$ is given by the sorting judgements defined by the following rules, where σ stands for the sort of individuals ι or some relational sort.

$$(\text{GConstr}) \ \frac{}{\Delta \vdash \varphi : o} \ \Delta \vdash \varphi : o \in Fm \qquad\qquad (\text{GVar}) \ \frac{}{\Delta_1, x : \rho, \Delta_2 \vdash x : \rho}$$

$$(\text{GCst}) \ \frac{\Delta \vdash G : o \qquad \Delta \vdash H : o}{\Delta \vdash G * H : o} * \in \{\wedge, \vee\} \qquad (\text{GEx}) \ \frac{\Delta, x : \sigma \vdash G : o}{\Delta \vdash \exists x {:} \sigma.G : o} \ \sigma = \iota \text{ or } \rho$$

$$(\text{GAppI}) \ \frac{\Delta \vdash G : \iota \to \rho}{\Delta \vdash GN : \rho} \ \Delta \vdash N : \iota \in Tm \qquad (\text{GAppR}) \ \frac{\Delta \vdash G : \rho_1 \to \rho_2 \qquad \Delta \vdash H : \rho_1}{\Delta \vdash GH : \rho_2}$$

$$(\text{GAbs}) \ \frac{\Delta, x : \sigma \vdash G : \rho}{\Delta \vdash \lambda x.G : \sigma \to \rho} \ x \notin \text{dom}(\Delta)$$

From now on, we use G and H (and variants thereof) to stand for arbitrary goal terms and disambiguate as necessary, and use uppercase R to stand for *relational variables* (i.e. variables of a relational sort).

Constrained logic program. A higher-order constrained *logic program*, P, over a sort environment $\Delta = \{R_1 : \rho_1, \ldots, R_k : \rho_k\}$ is a finite system of (mutually) recursive definitions of shape $R_i : \rho_i = G_i$ for some goal term G_i. Such a program is well sorted when $\Delta \vdash G_i : \rho_i$, for each $1 \leq i \leq k$. Since each R_i is distinct, we will sometimes regard a program P as a finite map from variables to terms, defined so that $P(R_i) = G_i$. We write $\vdash P : \Delta$ to mean that P is a well-sorted program over Δ.

2.1.2 Semantics

Motivated by the fact that unlike its first-order counterpart, HoCHC has no least model property for standard semantics, Cathcart Burn et al. consider an equivalent *monotone semantics* that *does* have a least model property. This interpretation suits us too, because it also has a greatest model property.

Monotone sort frame. We define the *monotone sort frame* $\mathcal{M}[\![-]\!]$ over the domain A_ι of the background theory recursively by:

$$\mathcal{M}[\![\iota]\!] := A_\iota \qquad \mathcal{M}[\![o]\!] := \mathbb{B} := \{0 \leq 1\} \qquad \mathcal{M}[\![\rho_1 \to \rho_2]\!] := \mathcal{M}[\![\rho_1]\!] \Rightarrow_m \mathcal{M}[\![\rho_2]\!]$$

where $X \Rightarrow_m Y$ is the monotone function space between X and Y w.r.t. a partial ordering $\sqsubseteq$; this partial order is the discrete ordering on A_ι, satisfies $0 \sqsubseteq 1$, and is lifted to higher sorts in a pointwise manner. It is easy to see that each $\mathcal{M}[\![\rho]\!]$ is a complete lattice.

We extend this ordering to sort environments with $\mathcal{M}[\![\Delta]\!] := \prod x \in \text{dom}(\Delta).\mathcal{M}[\![\Delta(x)]\!]$, pointwise over its elements, i.e., for $\beta, \gamma \in \mathcal{M}[\![\Delta]\!]$, $\beta \sqsubseteq \gamma$ if and only if $\beta(x) \sqsubseteq \gamma(x)$ for all $x : \rho \in \Delta$.

Denotation of goal terms. The meaning $\mathcal{M}[\![\Delta \vdash G : \rho]\!] : \mathcal{M}[\![\Delta]\!] \to \mathcal{M}[\![\rho]\!]$ of a goal term $\Delta \vdash G : \rho$ is defined as follows, for $\beta \in \mathcal{M}[\![\Delta]\!]$:

$$\mathcal{M}[\![\Delta \vdash \varphi : o]\!](\beta) := Th[\![\varphi]\!](\beta)$$

$$\mathcal{M}[\![\Delta_1, x : \rho, \Delta_2 \vdash x : \rho]\!](\beta) := \beta(x)$$

$$\mathcal{M}[\![\Delta \vdash G \wedge H : o]\!](\beta) := \min\{\mathcal{M}[\![\Delta \vdash G : o]\!](\beta), \mathcal{M}[\![\Delta \vdash H : o]\!](\beta)\}$$

$$\mathcal{M}[\![\Delta \vdash G \vee H : o]\!](\beta) := \max\{\mathcal{M}[\![\Delta \vdash G : o]\!](\beta), \mathcal{M}[\![\Delta \vdash H : o]\!](\beta)\}$$

$$\mathcal{M}[\![\Delta \vdash \exists x : \sigma.G : o]\!](\beta) := \max\{\mathcal{M}[\![\Delta, x : \sigma \vdash G : o]\!](\beta[x \mapsto x']) \mid x' \in \mathcal{M}[\![\sigma]\!]\}$$

$$\mathcal{M}[\![\Delta \vdash \lambda x : \sigma.G : \sigma \to \rho]\!](\beta) := \lambda x' \in \mathcal{M}[\![\sigma]\!].\mathcal{M}[\![\Delta, x : \sigma \vdash G : \rho]\!](\beta[x \mapsto x'])$$

$$\mathcal{M}[\![\Delta \vdash GH : \rho_2]\!](\beta) := \mathcal{M}[\![\Delta \vdash G : \rho_1 \to \rho_2]\!](\beta)(\mathcal{M}[\![\Delta \vdash H : \rho_1]\!][\![H]\!](\beta))$$

$$\mathcal{M}[\![\Delta \vdash GN : \rho]\!](\beta) := \mathcal{M}[\![\Delta \vdash G : \iota \to \rho]\!](\beta)(Th[\![N]\!](\beta))$$

In the above, min and max denote the greatest lower bound and the least upper bound, resp., within the complete lattice of booleans $\mathcal{M}[\![o]\!]$, and $Th[\![-]\!](\beta)$ denotes the interpretation of $-$ in the (standard) model of the background theory. We write $A, \beta \vDash G : o$, i.e. A, β *satisfies* G, just if $\mathcal{M}[\![\Delta \vdash G : o]\!](\beta) = 1$.

One-step consequence operator.   Logic programs give rise to an endofunction $T_{P:\Delta}^{\mathcal{M}} : \mathcal{M}[\![\Delta]\!] \to \mathcal{M}[\![\Delta]\!]$, defined by $T_{P:\Delta}^{\mathcal{M}}(\beta)(R_i) := \mathcal{M}[\![\Delta \vdash P(R_i) : \Delta(R_i)]\!](\beta)$, called the *one-step consequence operator*. We call β a *model* of $\vdash P : \Delta$, written $A, \beta \vDash P$ for model A of the background theory, just if $\beta = T_{P:\Delta}^{\mathcal{M}}(\beta)$.

2.1.3   Coinductive decision problem

Definition 2.1 (Coinductive HoCHC problem). *A coinductive HoCHC problem $\langle \Delta, P, G \rangle$, where Δ is a sorting of relational variables, $\vdash P : \Delta$ is a constrained logic program, and $\Delta \vdash G : o$ is a constrained goal formula, is* solvable *just if, for the standard model A of the background theory Th, there exists a valuation β of the variables in Δ such that $A, \beta \vDash P$ and $A, \beta \vDash G$.*

Note that the problem triple $\langle \Delta, P, G \rangle$ is identical to its inductive counterpart, but the definition of "solvability" differs; in the original HoCHC problem, solvability requires the existence of prefixed point β of P such that $A, \beta \nvDash G$. The background theory *Th* could be any first-order theory – in which the constraints in P and G can be interpreted – but in Section 4 we fix a specific theory of trees.

Interpreting satisfaction w.r.t. standard and monotone semantics gives rise to two distinct but equivalent HoCHC decision problems. A dual argument to Cathcart Burn et al.'s Lemma 5 for inductive HoCHC [6] shows that a coinductive HoCHC problem is *solvable under the standard interpretation* iff it is *solvable under the monotone interpretation*. We consider the monotone interpretation.

By the Knaster-Tarski theorem and $\mathcal{M}[\![\Delta]\!]$ being a complete lattice for relational Δ, the set of fixpoints of the monotone one-step consequence operator $T_{P:\Delta}^{\mathcal{M}}$ forms a complete lattice. This guarantees the existence of a greatest fixpoint of $T_{P:\Delta}^{\mathcal{M}}$. Thus, monotone HoCHC enjoys a greatest model property.

Theorem 2.2 (Greatest model property for monotone HoCHC). *Under the monotone interpretation, HoCHC definite clauses possess greatest models.*

Thus, a greatest model witnesses the solvability of a coinductive HoCHC problem in the monotone setting, like a least model witnesses solvability for traditional HoCHC. Instead of building up a least model from the least valuation, we start with the greatest valuation and work our way down. Intuitively, we are taking the *backwards closure* of our logic program.

Theorem 2.3. *A coinductive HoCHC problem $\langle \Delta, P, G \rangle$ is solvable under the monotone interpretation if and only if $\mathcal{M}[\![G]\!](M_P) = 1$ for greatest model M_P of P.*

Proof. Recall that a coinductive HoCHC problem $\langle \Delta, P, G \rangle$ is solvable iff, for the standard model A of the background theory *Th*, there exists a valuation β of the variables in Δ such that $A, \beta \vDash P$ and $A, \beta \vDash G$.

Clearly, $\mathcal{M}[\![G]\!](M_P) = 1$ for greatest model M_P of P implies solvability of $\langle \Delta, P, G \rangle$. For the converse, let $\beta \in \mathcal{M}[\![\Delta]\!]$ be a valuation such that $A, \beta \vDash P$ and $A, \beta \vDash G$, for standard model A of the background theory. By Knaster-Tarski, $\beta \sqsubseteq M_P$. By monotonicity, $A, \beta \vDash G$ implies $A, M_P \vDash G$, as required. ☐

HoCHC and coinductive HoCHC are not equivalent in the sense that the standard and the monotone interpretation are, as the following example demonstrates; it is not the case that a HoCHC problem $\langle \Delta, P, G \rangle$ is solvable if and only if the corresponding coinductive HoCHC problem $\langle \Delta, P, G \rangle$ is solvable.

Example 2.4. *Consider the HoCHC triples $\mathcal{P}_a = \langle \{R_S\}, P, R_S a^\omega \rangle$ and $\mathcal{P}_b = \langle \{R_S\}, P, R_S b^\omega \rangle$, where P consists of $R_S = \lambda r. (\exists r_1. (a r_1 = r) \wedge R_S r_1)$ and a^ω is the infinite unary tree of only as (and b^ω of bs). The logic program P has two models; relational variable R_S corresponds to the empty set in least model $M_\emptyset$, and to the singleton set $\{a^\omega\}$ in greatest model $M_{\{a^\omega\}}$.*

When we consider these HoCHC triples as inductive HoCHC problems, we find both $\mathcal{P}_a$ and $\mathcal{P}_b$ solvable, since $M_\emptyset$ is a witness to the refutation of both goal clauses. If we consider them as coinductive

problems, however, $\mathcal{P}_a$ is solvable, while $\mathcal{P}_b$ is unsolvable. This follows from $M_{\{a^\omega\}}$ witnessing the satisfiability of goal clause $R_S\,a^\omega$, while neither $M_{\{a^\omega\}}$ nor $M_\emptyset$ witnesses the satisfiability of $R_S\,b^\omega$.

Coinductive higher-order constrained Horn clauses allow us to reason about programs with datatypes inhabited by infinite objects, notably the (potentially) infinite trees generated by HoRS. These clauses are not merely an academic indulgence, though. There is a tradition of coinduction and corecursion in logic programming (see e.g. [12, 11, 28, 17]). This is hardly surprising, given that some well-formed logic programs do not have natural inductive interpretations, as we have seen in the introduction. Examples of other infinite data types that arise in practice include infinite lists and streams.

2.2 Higher-order recursion schemes

We fix a ranked alphabet Σ of tree constructors and write $\Sigma_\perp = \Sigma \cup \{\perp\}$. The set of all finite and infinite $\Sigma_\perp$-labelled trees, written $\mathcal{T}_{\Sigma_\perp}$, is a pointed poset with least element $\perp$ over the subtree ordering $\sqsubseteq$, which is the least partial order such that $C[\perp] \sqsubseteq C[t]$ for every tree context $C[\llcorner] \in \mathcal{T}_{\{\llcorner\}\cup\Sigma_\perp}$ and $t \in \mathcal{T}_{\Sigma_\perp}$.

Let $\mathcal{G} = \langle \mathcal{N}, \Sigma, \mathcal{R}, S \rangle$ be a (deterministic) higher-order recursion scheme (HoRS). That is, $\mathcal{N}$ maps a nonterminal symbol to its sort, Σ maps a terminal symbol to its sort, $S \in \mathcal{N}$ is the designated start symbol, and there exists one rewrite rule in $\mathcal{R}$ for each $F \in \mathcal{N}$ such that $\mathcal{R}(F) = \lambda x_1 \ldots x_n.t$, where $t : \iota$ is an applicative term over $\mathcal{N} \cup \Sigma \cup \{x_1, \ldots, x_n\}$ for some distinct $x_1, \ldots, x_n \in V_{\mathrm{RS}}$ drawn from a finite set of recursion scheme variables V_{RS}.

The *HoRS equivalence problem* asks whether two given HoRS generate the same tree (i.e. have the same semantics). Decidability of this problem is perhaps the best known and most challenging open problem in higher-order model checking.

2.2.1 Denotational semantics

The meaning of a HoRS can be given by a number of different formalisms. We introduce an infinite "Herbrand" interpretation that treats the rewrite rules as definitional equality in the style of a HoCHC logic program. Models are built incrementally from the smallest tree $\perp$.

Our interpretation of HoRS is Herbrandesque in that constants and function symbols are assigned very simple meanings. However, unlike typical Herbrand models, our models may contain infinite terms.

Let us define an interpretation of the sorts over ι (i.e. the sorts of HoRS terms):

$$\mathcal{H}[\![\iota]\!] := \langle \mathcal{T}_{\Sigma_\perp}, \sqsubseteq \rangle \qquad \mathcal{H}[\![\sigma_1 \to \sigma_2]\!] := \mathcal{H}[\![\sigma_1]\!] \Rightarrow_c \mathcal{H}[\![\sigma_2]\!]$$

where $X \Rightarrow_c Y$ is the continuous function space between directed-complete partial orders (dcpos) X and Y, ordered pointwise with respect to subtree ordering $\sqsubseteq$ on $\mathcal{H}[\![\iota]\!]$.

Given the environment $\Gamma = \{x_1 : \tau_1, \ldots, x_k : \tau_k\}$, let $\mathcal{N}'$ denote the extended environment $\mathcal{N}, \Gamma := \mathcal{N} \cup \Gamma$, which we view as a sort function whose domain is $\mathrm{dom}(\mathcal{N}) \cup \{x_1, \ldots, x_k\}$, mapping each symbol to its sort. Set $\mathcal{H}[\![\mathcal{N}']\!] : \prod x \in \mathrm{dom}(\mathcal{N}').\,\mathcal{H}[\![\mathcal{N}'(x)]\!]$ with typical element α. Define

$$\mathcal{H}[\![\mathcal{N}' \vdash t : \sigma]\!] : \mathcal{H}[\![\mathcal{N}']\!] \Rightarrow \mathcal{H}[\![\sigma]\!]$$

by cases and recursion on syntax:

$$\mathcal{H}[\![\mathcal{N}' \vdash x : \mathcal{N}'(x)]\!](\alpha) = \alpha(x)$$
$$\mathcal{H}[\![\mathcal{N}' \vdash f : \iota^{\mathrm{ar}(f)} \to \iota]\!](\alpha) = \widehat{F}_f$$
$$\mathcal{H}[\![\mathcal{N}' \vdash st : \tau]\!](\alpha) = \mathcal{H}[\![\mathcal{N}' \vdash s : \sigma \to \tau]\!](\alpha)(\mathcal{H}[\![\mathcal{N}' \vdash t : \sigma]\!](\alpha))$$
$$\mathcal{H}[\![\mathcal{N}' \vdash \lambda x : \sigma_1.t : \sigma_2]\!](\alpha) = \lambda v \in \mathcal{H}[\![\sigma_1]\!].\,\mathcal{H}[\![\mathcal{N}', x : \sigma_1 \vdash t]\!](\alpha[x \mapsto v])$$

for $\widehat{F}_f \in \mathcal{H}[\![\iota^{\mathrm{ar}(f)} \to \iota]\!]$ the usual Herbrand interpretation of $f : \iota^{\mathrm{ar}(f)} \to \iota \in \Sigma$. We define $\mathcal{H}[\![\mathcal{G}]\!]_{\mathcal{N}'} : \mathcal{H}[\![\mathcal{N}']\!] \Rightarrow \mathcal{H}[\![\mathcal{N}']\!]$ pointwise by $\mathcal{H}[\![\mathcal{G}]\!]_{\mathcal{N}'}(\alpha)(x) := \mathcal{H}[\![\mathcal{N}' \vdash \mathcal{R}(x) : \mathcal{N}'(x)]\!](\alpha)$.

Lemma 2.5. $\mathcal{H}[\![\mathcal{G}]\!]_{\mathcal{N}} : \mathcal{H}[\![\mathcal{N}]\!] \Rightarrow \mathcal{H}[\![\mathcal{N}]\!]$ *is continuous for all deterministic HoRS* $\mathcal{G} = \langle \mathcal{N}, \Sigma, \mathcal{R}, S \rangle$.

We define the (denotational) semantics of $\mathcal{G}$ as $\mathsf{lfp}(\mathcal{H}[\![\mathcal{G}]\!]_{\mathcal{N}})$, the least fixpoint of the continuous endo-function $\mathcal{H}[\![\mathcal{G}]\!]_{\mathcal{N}}$, which is well-defined by Kleene's Theorem. Henceforth we write $[\![\mathcal{G}]\!] := \mathsf{lfp}(\mathcal{H}[\![\mathcal{G}]\!]_{\mathcal{N}})(S)$, where $S \in \mathcal{N}$ is the start symbol of $\mathcal{G}$. Note that $[\![\mathcal{G}]\!]$ is the $\Sigma_\perp$-labelled tree generated by $\mathcal{G}$.

2.2.2   Computability of $\perp$-free transform of HoRS

Intuitively, eliminating $\perp$ from $[\![\mathcal{G}]\!]$ allow us to distinguish "unfinished" trees from "finished" (but diverging) trees in e.g. the proofs in Section 3.1.

As usual, let $\Sigma_\perp$ be a finite ranked alphabet Σ extended with (nullary) $\perp$. Let $b : \iota \to \iota \notin \Sigma_\perp$ (for "bottom") be a fresh terminal symbol.

Definition 2.6. *Given a* $\Sigma_\perp$*-labelled tree, its* $\perp$*-free conversion is obtained by replacing every* $\perp$*-labelled node by the infinite linear tree* $b\,(b\,(b\,\dots))$.

Lemma 2.7 (Computability of $\perp$-free transform of HoRS). *There is an algorithm that, given a HoRS* $\mathcal{G}$, *returns a HoRS – call it the* $\perp$*-free transform of* $\mathcal{G}$ *– that generates the* $\perp$*-free conversion of* $[\![\mathcal{G}]\!]$.

For clarity, we *convert* trees, but *transform* HoRS (their generators).

It is clear from freshness of $b : \iota \to \iota \notin \Sigma_\perp$ that the following holds.

Proposition 2.8. *HoRS are equivalent if and only if their respective* $\perp$*-free transforms are equivalent.*

Before we present a three-stage algorithm to transform a HoRS $\mathcal{G} = \langle \mathcal{N}, \Sigma, \mathcal{R}, S \rangle$ to its $\perp$-free transform and an example in Figure 2, we require some background on logical reflection.

Logical reflection of HoRS

Let $\mathfrak{R}$ be a class of generators of Σ-labelled trees, and $\mathcal{L}$ be a set of correctness properties of these trees. Define the ranked alphabet $\Sigma' := \{\underline{f} : \sigma \mid f : \sigma \in \Sigma\}$, which is a copy of Σ. Given a generator $\mathcal{G} \in \mathfrak{R}$ and property $\varphi \in \mathcal{L}$, we say that $\mathcal{G}_\varphi$ is a φ-*reflection* of $\mathcal{G}$ just if

1. $\mathcal{G}$ and $\mathcal{G}_\varphi$ generate the same underlying tree, and

2. if node α of $[\![\mathcal{G}]\!]$ has label f, then node α of $[\![\mathcal{G}_\varphi]\!]$ is labelled $\underline{f}$ if α satisfies φ and f otherwise.

We say that $\mathfrak{R}$ is *reflective w.r.t.* $\mathcal{L}$ just if there is an algorithm that transforms a given pair $\langle \mathcal{G}, \varphi \rangle$ to $\mathcal{G}_\varphi$.

Theorem 2.9 ([5]). *HoRS are reflective w.r.t. modal* μ*-calculus and monadic second-order logic.*

Stage 1: From $\Sigma_\perp$-labelling $\mathcal{G}$ to $\Sigma \cup \{b\}$-labelling $\mathcal{G}_1$.   The input HoRS $\mathcal{G}$ is first transformed to a *b-productive* counterpart $\mathcal{G}_1 := \langle \mathcal{N}, \Sigma \cup \{b\}, \mathcal{R}', S \rangle$. The idea is that in the potentially infinite process of generating the tree $[\![\mathcal{G}_1]\!]$ from the start nonterminal S by leftmost-outermost rewriting, each rewriting step is witnessed by either a terminal symbol from Σ or by b. The set $\mathcal{R}'$ of rewrite rules of $\mathcal{G}_1$ is defined as follows. For every $F : \sigma_1 \to \cdots \to \sigma_n \to \iota \in \mathcal{N}$:

   - if $\mathcal{R}(F) = \lambda x_1 \ldots x_n . f\, t_1 \ldots t_m$ for some $f \in \Sigma$, then $\mathcal{R}'(F) := \mathcal{R}(F)$

   - if $\mathcal{R}(F) = \lambda x_1 \ldots x_n . \$ t_1 \ldots t_m$ for $\$ \in \mathcal{N} \cup V_{\mathrm{RS}}$, then $\mathcal{R}'(F) := \lambda x_1 \ldots x_n . b\,(\$ t_1 \ldots t_m)$

Notice that the tree $[\![\mathcal{G}_1]\!]$, by construction, does not have any $\perp$-labelled nodes. Intuitively we can get $[\![\mathcal{G}]\!]$ back from $[\![\mathcal{G}_1]\!]$ by erasing finite b^*, and replacing infinite b^ω by $\perp$.

Stage 2: From $\Sigma \cup \{b\}$-labelling $\mathcal{G}_1$ to $\Sigma \cup \{b,s\}$-labelling $\mathcal{G}_2$. We define a modal μ-calculus formula:

$$\varphi := p_b \wedge \mu X. \left(\bigvee_{f \in \Sigma} \diamond_1 p_f \vee \diamond_1 X \right)$$

where p_b (resp. p_f for $f \in \Sigma$) is a propositional variable that denotes that a node is labelled with b (resp. $f \in \Sigma$). Refer to [4] for the syntax and semantics of the modal μ-calculus. Note that this formula holds for b-labelled nodes that are not "part of some infinite b^{ω}".

Let $s : \iota \to \iota \notin \Sigma_{\perp}$ (for "step") be another fresh arity-1 terminal symbol. Consider the following operation on $\Sigma \cup \{b\}$-labelled trees.

For every node α, if $\alpha \vDash \varphi$ then rewrite the label at α to s, otherwise do nothing.

This operation leaves exactly those occurrences of b in some infinite b^{ω} (which witnesses $\perp$) intact, while rewriting finite paths b^* to s^*. We call this operation *b-to-s conversion*.

Thanks to Theorem 2.9, the main result in [5], there is an algorithm that, given $\mathcal{G}_1$, returns a HoRS $\mathcal{G}_2$ over $\Sigma \cup \{b,s\}$ that generates the b-to-s conversion of $[\![\mathcal{G}_1]\!]$. In the language of [5], $\mathcal{G}_2$ is the φ-reflection of $\mathcal{G}_1$ where φ (above) is a property definable in the modal μ-calculus.

Stage 3: From $\Sigma \cup \{b,s\}$-labelling $\mathcal{G}_2$ to $\Sigma \cup \{b\}$-labelling $\mathcal{G}_3$. Although the tree $[\![\mathcal{G}_2]\!]$ does not have infinite paths exclusively labelled by s, it may still have nodes labelled by s. We construct a $\Sigma \cup \{b\}$-labelling HoRS $\mathcal{G}_3$ that generates the tree $[\![\mathcal{G}_2]\!]$ but with these remaining s-labelled nodes cut out, which is easily achieved by replacing every occurrence of the terminal symbol s in the rewrite rules of $\mathcal{G}_2$ by the identity nonterminal I. To be precise, if $\mathcal{R}_2$ is the set of rewrite rules of $\mathcal{G}_2$, then the resultant HoRS

$$\mathcal{G}_3 := \langle \mathcal{N} \cup \{I\}, \Sigma \cup \{b\}, \{F = \lambda \overline{x}.t[I/s] \mid F = \lambda \overline{x}.t \in \mathcal{R}_2\} \cup \{I = \lambda x.x\}, S \rangle$$

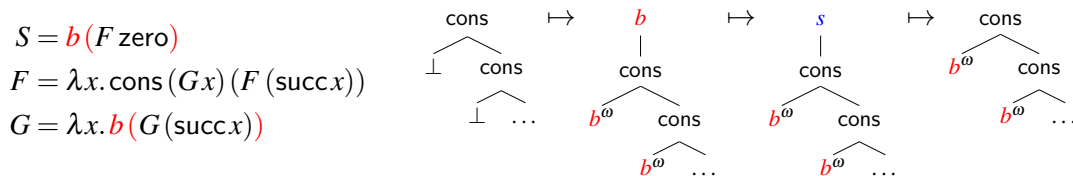is the $\perp$-free transform of the input HoRS $\mathcal{G}$.

$S = b(F\,\mathsf{zero})$
$F = \lambda x.\mathsf{cons}\,(G\,x)\,(F\,(\mathsf{succ}\,x))$
$G = \lambda x.b\,(G\,(\mathsf{succ}\,x))$

Figure 2: Conversion $[\![\mathcal{G}]\!] \mapsto [\![\mathcal{G}_1]\!] \mapsto [\![\mathcal{G}_2]\!] \mapsto [\![\mathcal{G}_3]\!]$, with HoRS $\mathcal{G}$ on the left (w/o bs; $\mathcal{G}_1$ with bs)

3 Encoding HoRS-to-HoCHC logic program

In this section, we encode a (deterministic) HoRS $\mathcal{G} = \langle \mathcal{N}, \Sigma, \mathcal{R}, S \rangle$ into a HoCHC logic program that captures the meaning of the HoRS under the coinductive monotone interpretation. We assume $[\![\mathcal{G}]\!]$ does not contain $\perp$-labelled nodes, which is WLOG by Lemma 2.7.

We define the *HoRS-to-HoCHC encoding* $\vdash P_{\mathcal{G}} : \Delta_{\mathcal{G}}$ of HoRS $\mathcal{G}$ over the coinductive monotone HoCHC interpretation where $\mathcal{M}[\![\iota]\!]$ is interpreted as the underlying set of $\mathcal{H}[\![\iota]\!]$, which is the set of finite and infinite trees over $\Sigma_{\perp}$. The constrained logic program $\vdash P_{\mathcal{G}} : \Delta_{\mathcal{G}}$ is defined by:

$$\Delta_{\mathcal{G}} := \{R_F : \mathsf{Rel}^+(\sigma) \mid F : \sigma \in \mathcal{N}\} \qquad P_{\mathcal{G}} := \{R_F : \mathsf{Rel}^+(\sigma) = \ulcorner \mathcal{R}(F) \urcorner \mid F : \sigma \in \mathcal{N}\}$$

where

$$\mathsf{Rel}^-(\iota) := \iota \qquad \mathsf{Rel}^+(\iota) := \iota \to o \qquad \mathsf{Rel}^-(\sigma_1 \to \sigma_2) := \mathsf{Rel}^+(\sigma_1 \to \sigma_2) := \mathsf{Rel}^-(\sigma_1) \to \mathsf{Rel}^+(\sigma_2).$$

In the HoRS-to-HoCHC encoding below, we annotate variables with superscripts of not merely sorts but of *interpreted* sorts – $\mathcal{H}[\![\sigma]\!]$ or $\mathcal{M}[\![\sigma]\!]$ for sort σ – to distinguish HoRS and HoCHC variables.

Let the metavariable \$ range over $\mathcal{N} \cup \Sigma \cup V_{\mathrm{RS}}$. We define a transformation $\$ \mapsto \$'$ according to:

	\$	\$'
Variables	$x : \iota$	$D_{x'} : \mathsf{Rel}^+(\iota)$
	$x : \sigma_1 \to \sigma_2$	$x' : \mathsf{Rel}^+(\sigma_1 \to \sigma_2)$
Terminals	$f : \iota^n \to \iota$	$D_f : \mathsf{Rel}^+(\iota^n \to \iota)$
Nonterminals	$F : \sigma$	$R_F : \mathsf{Rel}^+(\sigma)$

where

$$D_f := \lambda x_1 \ldots x_{\mathrm{ar}(f)} r.\, (f\, x_1 \ldots x_{\mathrm{ar}(f)} = r).$$

Note that R_F is a relational variable, but D_f is merely a shorthand; D_f is not a symbol – and neither is $D_{x'}$. This shorthand allows us to present the relational lift $\ulcorner - \urcorner$ in a simpler way. Whenever D_f occurs in some encoded HoRS term, it occurs in a fully applied term $D_f t_1 \ldots t_{\mathrm{ar}(f)} r$, which is β-equivalent to $f t_1 \ldots t_{\mathrm{ar}(f)} = r$. It is this latter term we use in practice (similarly for $D_{x'}$).

For each $F : \sigma \in \mathcal{N}$, we define $\ulcorner \mathcal{R}(F) \urcorner : \mathsf{Rel}^+(\sigma)$, called the *relational lift*, as follows. We write $x' : \mathsf{Rel}^-(\tau)$ for the HoCHC variable that is the relational clone of HoRS variable $x : \tau \in V_{\mathrm{RS}}$, such that:

$$\ulcorner \lambda x_1^{\mathcal{H}[\![\sigma_1]\!]} \ldots x_m^{\mathcal{H}[\![\sigma_m]\!]}. e \urcorner := \lambda x_1'^{\mathcal{M}[\![\mathsf{Rel}^-(\sigma_1)]\!]} \ldots x_m'^{\mathcal{M}[\![\mathsf{Rel}^-(\sigma_m)]\!]}. \ulcorner e \urcorner$$

For HoRS term $\$ e_1 \ldots e_l$, we define the relational lift as:

$$\ulcorner \$ e_1 \ldots e_l \urcorner := \begin{array}{l} \lambda y_1^{\mathcal{M}[\![\mathsf{Rel}^-(\sigma_1)]\!]} \ldots y_n^{\mathcal{M}[\![\mathsf{Rel}^-(\sigma_n)]\!]} r. \\[4pt] \exists r_1 \ldots r_l. \left(\begin{array}{cc} & \$'\, \ulcorner\!\ulcorner (e_1, r_1) \urcorner\!\urcorner \ldots \ulcorner\!\ulcorner (e_l, r_l) \urcorner\!\urcorner y_1 \ldots y_n r \\ \wedge & \bigwedge_{i=1}^l Prop(e_i, r_i) \end{array} \right) \end{array}$$

for fresh HoCHC variables $y_1, \ldots, y_n$, where $\$: \tau_1 \to \ldots \to \tau_l \to \sigma_1 \to \ldots \to \sigma_n \to \iota$ and

$$\ulcorner\!\ulcorner (e : \sigma, r) \urcorner\!\urcorner := \begin{cases} r & \text{if } \sigma = \iota \\ \ulcorner e : \sigma \urcorner & \text{o/w} \end{cases} \qquad Prop(e : \sigma, r) := \begin{cases} \ulcorner e : \sigma \urcorner r & \text{if } \sigma = \iota \\ \text{true} & \text{o/w.} \end{cases}$$

It is worth pointing out that $\ulcorner e : \sigma \urcorner : \mathsf{Rel}^+(\sigma)$, $\ulcorner\!\ulcorner (e : \sigma, r) \urcorner\!\urcorner : \mathsf{Rel}^-(\sigma)$, and $Prop(e : \sigma, r) : o$.

3.1 Correctness

Our HoRS-to-HoCHC encoding $\vdash P_{\mathcal{G}} : \Delta_{\mathcal{G}}$ contains a relational variable for each nonterminal symbol in the original HoRS $\mathcal{G} = \langle \mathcal{N}, \Sigma, \mathcal{R}, S \rangle$. We claim that the HoCHC rational variable $R_S : \iota \to o$ corresponding to start symbol S valuates to the characteristic function of $[\![\mathcal{G}]\!]$ in the greatest model of $\vdash P_{\mathcal{G}} : \Delta_{\mathcal{G}}$:

Theorem 3.6 (Correctness). $\mathcal{M}[\![\Delta_{\mathcal{G}} \vdash R_S]\!](\mathrm{gfp}(T_{P_{\mathcal{G}} : \Delta_{\mathcal{G}}}^{\mathcal{M}})) t = 1$ *if and only if* $t = [\![\mathcal{G}]\!]$.

To prove the theorem, we establish a lockstep between iterations of the HoRS endofunction $\mathcal{H}[\![\mathcal{G}]\!]_{\mathcal{N}}$ (in the ascending Kleene chain) and (descending) iterations of the HoCHC one-step consequence operator $T_{\Delta_{\mathcal{G}} : P_{\mathcal{G}}}^{\mathcal{M}}$. The proof consists of four parts, corresponding to the respective sections of Appendix A.

First, we define two families of mappings between HoRS semantics and (coinductive) HoCHC semantics. These mappings allow us to embed HoRS semantics into HoCHC relations. Second, we show that there exists a $\perp$-free tree t such that $\mathcal{M}[\![\Delta_{\mathcal{G}} \vdash \ulcorner S \urcorner]\!](T_{P_{\mathcal{G}} : \Delta_{\mathcal{G}}}^{\mathcal{M} n}(\top_{\Delta_{\mathcal{G}}})) t = 1$, for every iteration n of the one-step consequence operator (Lemma 3.4, "nonemptiness"). Third, we show that each

$\mathcal{M}[\![\Delta_{\mathcal{G}} \vdash \ulcorner S \urcorner]\!](T_{P_{\mathcal{G}}:\Delta_{\mathcal{G}}}^{\mathcal{M}n}(\top_{\Delta_{\mathcal{G}}}))$ is included in the embedding of $\mathcal{H}[\![\mathcal{N} \vdash S]\!](\mathcal{H}[\![\mathcal{G}]\!]_{\mathcal{N}}^{n}(\bot_{\mathcal{N}}))$ into HoCHC (Corollary 3.5, "inclusion").

Finally, we prove that these "nonemptiness" and "inclusion" results suffice to show that R_S valuates to the characteristic function of $[\![\mathcal{G}]\!]$ in the greatest model of $\vdash P_{\mathcal{G}} : \Delta_{\mathcal{G}}$ (Theorem 3.6).

Appendix A details the full proof. Although our results in this section pertain to ground sort ι, we require logical relations and proofs lifted to higher-sorts to attain them. This starts with the families of mappings $\mathbf{i}_{\sigma}, \mathbf{j}_{\sigma}$ between HoRS semantics and HoCHC relations that are markedly simpler for sort ι (Definition 3.1) than the full mappings defined in Definition A.2.

Definition 3.1 (Embedding of trees into HoCHC relations)**.** *We define a function* $\mathbf{i}_{\iota} : \mathcal{H}[\![\iota]\!] \to \mathcal{M}[\![\mathrm{Rel}^{+}(\iota)]\!]$ *by* $\mathbf{i}_{\iota}(t) := \lambda s. t \sqsubseteq s$, *for all* $t \in \mathcal{H}[\![\iota]\!]$.

Note that the function $\mathbf{i}_{\iota}$ allows us to embed HoRS trees into HoCHC relations; it is antitone and injective. Because we are trying to relate a least fixpoint (HoRS semantics) to a greatest fixpoint (coinductive HoCHC), the following lemma is key. Please refer to Appendix A for the proofs.

Lemma 3.2. *For all directed sets* $D \subseteq \mathcal{H}[\![\iota]\!]$, $\mathbf{i}_{\iota}(\bigsqcup D) = \bigsqcap \{\mathbf{i}_{\iota}(d) \mid d \in D\}$.

Greatest upper bounds of chains are preserved by the semantics of relationally lifted HoRS terms.

Lemma 3.3. *For all typing judgements* $\mathcal{N} \vdash e : \sigma$ *of the HoRS* $\mathcal{G}$, *and non-increasing chains of valuations* $\mathcal{I} \subseteq \mathcal{M}[\![\Delta_{\mathcal{G}}]\!]$,

$$\mathcal{M}[\![\Delta_{\mathcal{G}} \vdash \ulcorner e : \sigma \urcorner]\!]\left(\bigsqcap \mathcal{I}\right) = \bigsqcap_{I \in \mathcal{I}} \mathcal{M}[\![\Delta_{\mathcal{G}} \vdash \ulcorner e : \sigma \urcorner]\!](I).$$

Let us write α^{n} for $\mathcal{H}[\![\mathcal{G}]\!]_{\mathcal{N}}^{n}(\bot_{\mathcal{N}})$ and β^{n} for $T_{P_{\mathcal{G}}:\Delta_{\mathcal{G}}}^{\mathcal{M}n}(\top_{\Delta_{\mathcal{G}}})$, so that the following hold.

Lemma 3.4 (Nonemptiness)**.** *There exists a* $\bot$*-free tree* $t \in \mathcal{M}[\![\iota]\!]$ *such that* $\mathcal{M}[\![\Delta_{\mathcal{G}} \vdash \ulcorner S \urcorner]\!](\bigsqcap \beta^{n}) t$.

Corollary 3.5 (Inclusion)**.** *For all* $n \geq 0$, $\mathcal{M}[\![\Delta_{\mathcal{G}} \vdash \ulcorner S \urcorner]\!](\beta^{n}) \sqsubseteq \mathbf{i}_{\iota}(\mathcal{H}[\![\mathcal{N} \vdash S]\!](\alpha^{n}))$.

Theorem 3.6 (Correctness)**.** $\mathcal{M}[\![\Delta_{\mathcal{G}} \vdash R_S]\!](\mathsf{gfp}(T_{P_{\mathcal{G}}:\Delta_{\mathcal{G}}}^{\mathcal{M}})) t = 1$ *if and only if* $t = [\![\mathcal{G}]\!]$.

Proof.
$$\begin{aligned} \mathcal{M}[\![\Delta_{\mathcal{G}} \vdash R_S]\!](\mathsf{gfp}(T_{P_{\mathcal{G}}:\Delta_{\mathcal{G}}}^{\mathcal{M}})) &= \bigsqcap\{\mathcal{M}[\![\Delta_{\mathcal{G}} \vdash \ulcorner S \urcorner]\!](\beta^{n}) \mid n \geq 0\} && \text{Lem 3.3} \\ &\sqsubseteq \bigsqcap\{\mathbf{i}_{\iota}(\mathcal{H}[\![\mathcal{N} \vdash S]\!](\alpha^{n})) \mid n \geq 0\} && \text{Cor 3.5} \\ &= \mathbf{i}_{\iota}\left(\bigsqcup\{\mathcal{H}[\![\mathcal{N} \vdash S]\!](\alpha^{n}) \mid n \geq 0\}\right) && \text{Lem 3.2} \\ &= \mathbf{i}_{\iota}\left(\mathcal{H}[\![\mathcal{N} \vdash S]\!]\left(\bigsqcup \alpha^{n}\right)\right) && \text{Lem 2.5} \\ &= \lambda r.([\![\mathcal{G}]\!] = r) && \bot\text{-freeness} \end{aligned}$$

Either $\mathcal{M}[\![\Delta_{\mathcal{G}} \vdash R_S]\!](\mathsf{gfp}(T_{P_{\mathcal{G}}:\Delta_{\mathcal{G}}}^{\mathcal{M}}))$ is the constant false function, or it is $\lambda r.([\![\mathcal{G}]\!] = r)$. By Lemma 3.4, $\mathcal{M}[\![\Delta_{\mathcal{G}} \vdash R_S]\!](\mathsf{gfp}(T_{P_{\mathcal{G}}:\Delta_{\mathcal{G}}}^{\mathcal{M}}))$ is not $\lambda r. 0$, so we conclude that it is $\lambda r.([\![\mathcal{G}]\!] = r)$, instead.

It follows that $\mathcal{M}[\![\Delta_{\mathcal{G}} \vdash R_S]\!](\mathsf{gfp}(T_{P_{\mathcal{G}}:\Delta_{\mathcal{G}}}^{\mathcal{M}})) t = 1$ if and only if $t = [\![\mathcal{G}]\!]$. $\qquad\square$

4   HoRS equivalence problem

The higher-order recursion scheme (HoRS) equivalence problem asks whether two given deterministic recursion schemes $\mathcal{G}_1, \mathcal{G}_2$ generate the same tree (i.e. whether $[\![\mathcal{G}_1]\!] = [\![\mathcal{G}_2]\!]$, see e.g. [22]).

Here, we reduce the HoRS equivalence problem to coinductive HoCHC. Our procedure formulates a positive and negative instance of the coinductive monotone HoCHC problem over a decidable background theory. We present the background theory in Section 4.1 and the HoCHC instances in Section 4.2.

If coinductive HoCHC is semi-decidable over a decidable background theory – or our HoRS-to-HoCHC encodings are semi-decidable over Maher's theory of trees in particular [20] – then these two instances can be solved concurrently for a full decision procedure for the HoRS equivalence problem.

4.1   Maher's theory of trees

A *theory T* is a set of sentences, which is *complete* if either $T \vDash \varphi$ or $T \vDash \neg\varphi$, for every sentence φ. An *axiomatisation* of an algebra $\mathfrak{A}$ is a recursive set of sentences which are true of $\mathfrak{A}$. The *theory of an algebra* $\mathfrak{A}$ is a the set of all sentences true of $\mathfrak{A}$.

Maher's (equational) theory of trees T_Σ is complete for any finite or infinite alphabet Σ [20]. It is axiomatised by three axioms:

$$\forall f \in \Sigma. \qquad \forall \overline{x}\,\overline{y}. \, f\,\overline{x} = f\,\overline{y} \leftrightarrow \overline{x} = \overline{y} \tag{1}$$

$$\forall f, g \in \Sigma. \qquad f \not\equiv g \rightarrow \forall \overline{x}\,\overline{y}. \, f\,\overline{x} \neq g\,\overline{y} \tag{2}$$

$$\forall y. \exists! x. \qquad x = t(x,y) \tag{3}$$

where $x = t(x,y)$ ranges over rational solved forms (see [20], p. 355).

In case Σ is finite, we need to add the Domain Closure Axiom to obtain completeness:

$$\forall x. \bigvee_{f \in \Sigma} \exists \overline{z}. x = f\,\overline{z} \tag{DCA}$$

Fix a ranked alphabet Σ, viewed as tree constructors. We write T_Σ for the first-order theory of equations of finite and infinite trees constructed from Σ. The theory T_Σ is complete and decidable, making it an exceedingly appropriate choice of background theory for HoCHC. The theory has several models, including the set of finite and infinite trees over Σ that we are interested in.

Djelloul et al. have presented a full first-order constraint solver for (an augmented version of) the theory T_Σ [10]. Questions of expressivity and complexity of the Maher theory are explored in [8]. Recent work by Zaiser and Ong has improved the performance of Djelloul et al.'s solver and adapted the theory to algebraic (co)datatypes [34].

We are interested in the theory $T_{\Sigma_\perp}$ over finite alphabet $\Sigma_\perp$, for input HoRS $\mathcal{G}_1 = \langle \mathcal{N}_1, \Sigma, \mathcal{R}_1, S_1 \rangle$ and $\mathcal{G}_2 = \langle \mathcal{N}_2, \Sigma, \mathcal{R}_2, S_2 \rangle$. Note that the assumption that both HoRS have the same alphabet Σ is WLOG; if they have distinct alphabets, we can take Σ to be their union. In the Maher theory $T_{\Sigma_\perp}$, the "unfinished" tree $\perp$ is treated as any other nullary terminal symbol.

4.2   Decision procedure

Let $\mathcal{G}_1 = \langle \mathcal{N}_1, \Sigma, \mathcal{R}_1, S_1 \rangle$ and $\mathcal{G}_2 = \langle \mathcal{N}_2, \Sigma, \mathcal{R}_2, S_2 \rangle$ be deterministic HoRS. Assume the trees they generate are $\perp$-free, which is WLOG due to Section 2.2.2. Consider these HoCHC goal formulas:

$$Eq_1 := \exists r_1\, r_2. \, (R_{S_1}\, r_1 \wedge R_{S_2}\, r_2) \wedge (r_1 = r_2) \qquad Eq_0 := \exists r_1\, r_2. \, (R_{S_1}\, r_1 \wedge R_{S_2}\, r_2) \wedge (r_1 \neq r_2)$$

Using the definitions from Section 3, we define HoCHC problems $\mathcal{P}_i := \langle \Delta_{\mathcal{G}_1} \cup \Delta_{\mathcal{G}_2}, P_{\mathcal{G}_1} \cup P_{\mathcal{G}_2}, Eq_i \rangle$, for $i \in \{0,1\}$, with the Maher theory $T_{\Sigma_\perp}$ as the constraint language and the set $\mathcal{T}_{\Sigma_\perp}$ of finite and infinite trees as the designated model.

Thanks to Theorem 3.6, we have: $[\![\mathcal{G}_1]\!] = [\![\mathcal{G}_2]\!]$ iff $\mathcal{P}_1$ is solvable, and $[\![\mathcal{G}_1]\!] \neq [\![\mathcal{G}_2]\!]$ iff $\mathcal{P}_0$ is solvable.

Recall that the Maher theory $T_{\Sigma_\perp}$ is decidable – to be exact, the question $T_{\Sigma_\perp} \vDash \varphi$ for first-order tree constraints φ like $r_1 = r_2$ and $r_1 \neq r_2$ above. Note, however, that $\mathcal{P}_1$ and $\mathcal{P}_0$ are coinductive HoCHC

problems. It is an open question whether coinductive HoCHC problems over a (semi-)decidable background theory – like the Maher theory $T_{\Sigma_\perp}$ – can be semi-decided via a reduction to a first-order problem, like inductive HoCHC can [24, 23]. If there exists a such semi-decision procedure for solving (monotone) coinductive HoCHC over $T_{\Sigma_\perp}$, then we can decide $[\![\mathcal{G}_1]\!] = [\![\mathcal{G}_2]\!]$ by dovetailing our two HoCHC problems.

The full "decision" procedure for the HoRS equivalence problem is outlined in Figure 1c, so that:

Theorem 4.1. *The HoRS equivalence problem is decidable if the HoRS-to-HoCHC encoding lives in a semi-decidable fragment of coinductive HoCHC over Maher's complete and decidable theory of trees.*

5 Conclusion and related work

Higher-order recursion scheme equivalence problem. To the best of our knowledge, the HoRS equivalence problem [7] remains open. We obtain a full decision procedure for this problem if: (a) coinductive HoCHC over Maher's theory of trees [20] is semi-decidable, or (b) the image of our HoRS–to-HoCHC encoding over Maher's theory lives in a semi-decidable fragment of coinductive HoCHC.

Restricted to order 1, the HoRS equivalence problem is equivalent to the DPDA equivalence problem [9]. Thus, the fundamental result of [26], and subsequent refinements by [30], [27], and [13] provide a decision procedure for the equivalence of first-order HoRS.

$\lambda\mathbf{Y}$-calculus Böhm tree equivalence problem. The HoRS equivalence problem is recursively equivalent to $\lambda\mathbf{Y}$-calculus Böhm tree equivalence problem, which asks whether the Böhm trees of two given $\lambda\mathbf{Y}$-terms are equal [7]. The question of the decidability "has been there from the beginning of the subject" [33]. Semi-decidability of coinductive HoCHC would also allow us to decide this problem.

Note that the closely related $\lambda\mathbf{Y}$-calculus word problem (are two closed $\lambda\mathbf{Y}$-terms $\beta\eta\mathbf{Y}$-equivalent?) is undecidable [29]. Although HoRS are programs of a simply-typed $\lambda\mathbf{Y}$-calculus, constructed from uninterpreted function symbols, they define a strict subsystem of the $\lambda\mathbf{Y}$-calculus: the same set of trees as ground-type $\lambda\mathbf{Y}$-terms with free variables (corresponding to terminal symbols) of order at most 1 [25].

Semi-decidability of coinductive HoCHC. Existing semi-decidability results for inductive HoCHC [24, 23] do not carry over to coinductive HoCHC, because proofs for coinductive programs may have infinite length.

For first-order Horn clauses, Coinductive Logic Programming (CoLP, [11, 28]) provides an approach to computing solutions for infinite sequences of reductions. Resolution proof systems for coinductive logic programs rely on loop detection in infinite proofs, see e.g. [19, 18] and refinements [17, 1]. Intuitively, our characterisation of HoRS in HoCHC has not necessarily made such loop detection computationally simpler.

Our best hope is that our HoRS-to-HoCHC encodings live in a semi-decidable fragment of coinductive HoCHC. There are some indications that this could be the case, e.g. Lemma 3.3 shows the semantics of encoded HoRS behaves better than (monotone) coinductive HoCHC as a whole.

Relation to HFL. In recent years, HFL model checking – where properties are expressed in higher-order modal fixpoint logic [31] – has gained traction [14, 16]. HoCHC roughly corresponds to a fragment of $HFL_{\mathbb{Z}}$ without modal operators and fixpoint alternations. HoCHC unsolvability captures HFL *non-reachability* [16]. It seems that coinductive HoCHC unsolvability corresponds to *must-reachability*. Clarifying this relation may help us understand the complexity of coinductive HoCHC.

References

[1] Henning Basold, Ekaterina Komendantskaya & Yue Li (2019): *Coinduction in Uniform: Foundations for Corecursive Proof Search with Horn Clauses*. Lecture Notes in Computer Science, pp. 783–813, doi:10.1007/978-3-030-17184-1_28.

[2] Nikolaj Bjørner, Arie Gurfinkel, Kenneth McMillan & Andrey Rybalchenko (2015): *Horn clause solvers for program verification*. In: *Fields of Logic and Computation II*, Springer, pp. 24–51, doi:10.1007/978-3-319-23534-9_2. Available at https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/nbjorner-yurifest.pdf.

[3] Nikolaj Bjørner, Kenneth McMillan & Andrey Rybalchenko (2012): *Program Verification as Satisfiability Modulo Theories*. In Pascal Fontaine & Amit Goel, editors: *SMT 2012. 10th International Workshop on Satisfiability Modulo Theories*, EPiC Series in Computing 20, EasyChair, pp. 3–11, doi:10.29007/1l7f.

[4] Julian Bradfield & Colin Stirling (2001): *CHAPTER 4 - Modal Logics and mu-Calculi: An Introduction*. In J.A. Bergstra, A. Ponse & S.A. Smolka, editors: *Handbook of Process Algebra*, Elsevier Science, Amsterdam, pp. 293–330, doi:10.1016/B978-044482830-9/50022-9. Available at http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.9.5944.

[5] Christopher H. Broadbent, Arnaud Carayol, C.-H. Luke Ong & Olivier Serre (2010): *Recursion Schemes and Logical Reflection*. In: *Proceedings of the 25th Annual IEEE Symposium on Logic in Computer Science, LICS 2010, 11-14 July 2010, Edinburgh, United Kingdom*, IEEE Computer Society, pp. 120–129, doi:10.1109/LICS.2010.40. Available at https://hal.archives-ouvertes.fr/hal-00479818.

[6] Toby Cathcart Burn, C.-H. Luke Ong & Steven J. Ramsay (2017): *Higher-order Constrained Horn Clauses for Verification*. Proc. ACM Program. Lang. 2(POPL), pp. 11:1–11:28, doi:10.1145/3158099.

[7] Pierre Clairambault & Andrzej S. Murawski (2013): *Böhm Trees as Higher-Order Recursive Schemes*. In Anil Seth & Nisheeth K. Vishnoi, editors: *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2013)*, Leibniz International Proceedings in Informatics (LIPIcs) 24, Schloss Dagstuhl–Leibniz-Zentrum für Informatik, Dagstuhl, Germany, pp. 91–102, doi:10.4230/LIPIcs.FSTTCS.2013.91.

[8] Alain Colmerauer & Thi-Bich-Hanh Dao (2003): *Expressiveness of Full First-Order Constraints in the Algebra of Finite or Infinite Trees*. Constraints 8(3), pp. 283–302, doi:10.1023/A:1025675127871. Available at https://hal.archives-ouvertes.fr/hal-00144924.

[9] Bruno Courcelle (1978): *A Representation of Trees by Languages I*. Theor. Comput. Sci. 6, pp. 255–279, doi:10.1016/0304-3975(78)90008-7. Available at https://core.ac.uk/download/pdf/82601565.pdf.

[10] Khalil Djelloul, Thi-Bich-Hanh Dao & Thom Frühwirth (2008): *Theory of finite or infinite trees revisited*. Theory Pract. Log. Program. 8(04), pp. 431–489, doi:10.1017/S1471068407003171. Available at https://arxiv.org/abs/0706.4323.

[11] Gopal Gupta, Ajay Bansal, Richard Min, Luke Simon & Ajay Mallya (2007): *Coinductive Logic Programming and Its Applications*. In: *Log. Program.*, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 27–44, doi:10.1007/978-3-540-74610-2_4. Available at https://personal.utdallas.edu/~gupta/iclp07paper.pdf.

[12] Joxan Jaffar & Peter J. Stuckey (1986): *Semantics of infinite tree logic programming*. Theor. Comput. Sci. 46, pp. 141–158, doi:10.1016/0304-3975(86)90027-7.

[13] Petr Jancar (2012): *Decidability of DPDA Language Equivalence via First-Order Grammars*. In: *Proceedings of the 27th Annual IEEE Symposium on Logic in Computer Science, LICS 2012, Dubrovnik, Croatia, June 25-28, 2012*, pp. 415–424, doi:10.1109/LICS.2012.51. Available at https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.1080.2859&rep=rep1&type=pdf.

[14] Naoki Kobayashi, Étienne Lozes & Florian Bruse (2017): *On the Relationship between Higher-Order Recursion Schemes and Higher-Order Fixpoint Logic*. In: *Proceedings of the 44th ACM SIGPLAN Symposium*

on Principles of Programming Languages, POPL 2017, Association for Computing Machinery, New York, NY, USA, pp. 246–259, doi:10.1145/3009837.3009854.

[15] Naoki Kobayashi & C.-H. Luke Ong (2009): *A Type System Equivalent to the Modal Mu-Calculus Model Checking of Higher-Order Recursion Schemes*. In: *Proceedings of the 24th Annual IEEE Symposium on Logic in Computer Science, LICS 2009, 11-14 August 2009, Los Angeles, CA, USA*, IEEE Computer Society, pp. 179–188, doi:10.1109/LICS.2009.29. Available at `https://www-kb.is.s.u-tokyo.ac.jp/~koba/papers/hors-type.pdf`.

[16] Naoki Kobayashi, Takeshi Tsukada & Keiichi Watanabe (2018): *Higher-Order Program Verification via HFL Model Checking*. In Amal Ahmed, editor: *Programming Languages and Systems*, Springer International Publishing, Cham, pp. 711–738, doi:10.1007/978-3-319-89884-1_25. Available at `https://arxiv.org/abs/1710.08614`.

[17] Ekaterina Komendantskaya & Yue Li (2017): *Productive corecursion in logic programming*. Theory and Practice of Logic Programming 17(5-6), pp. 906–923, doi:10.1017/S147106841700028X. Available at `http://arxiv.org/abs/1707.01541`.

[18] Ekaterina Komendantskaya & Yue Li (2018): *Towards Coinductive Theory Exploration in Horn Clause Logic: Position Paper*. In Temesghen Kahsai & German Vidal, editors: Proceedings 5th Workshop on *Horn Clauses for Verification and Synthesis*, Oxford, UK, 13th July 2018, *Electronic Proceedings in Theoretical Computer Science* 278, Open Publishing Association, pp. 27–33, doi:10.4204/EPTCS.278.5.

[19] Ekaterina Komendantskaya, John Power & Martin Schmidt (2016): *Coalgebraic logic programming: from Semantics to Implementation*. Journal of Logic and Computation 26(2), pp. 745–783, doi:10.1093/logcom/exu026. Available at `https://arxiv.org/abs/1312.6568`.

[20] Michael J. Maher (1988): *Complete axiomatizations of the algebras of finite, rational and infinite trees*. In: *LICS*, pp. 348–357, doi:10.1109/LICS.1988.5132. Available at `https://www.computer.org/csdl/pds/api/csdl/proceedings/download-article/12OmNyLiuB4/pdf`.

[21] C.-H. Luke Ong (2006): *On Model-Checking Trees Generated by Higher-Order Recursion Schemes*. In: *21th IEEE Symposium on Logic in Computer Science (LICS 2006), 12-15 August 2006, Seattle, WA, USA, Proceedings*, pp. 81–90, doi:10.1109/LICS.2006.38. Available at `https://www.cs.ox.ac.uk/people/luke.ong/personal/publications/lics06.pdf`.

[22] C.-H. Luke Ong (2015): *Higher-Order Model Checking: An Overview*. In: *30th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2015, Kyoto, Japan, July 6-10, 2015*, pp. 1–15, doi:10.1109/LICS.2015.9. Available at `http://www.cs.ox.ac.uk/people/luke.ong/personal/publications/LICS15.pdf`.

[23] C.-H. Luke Ong & Dominik Wagner (2019): *HoCHC: A Refutationally Complete and Semantically Invariant System of Higher-order Logic Modulo Theories*. In: *2019 34th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pp. 1–14, doi:10.1109/LICS.2019.8785784. Available at `https://arxiv.org/abs/1902.10396`.

[24] Long Pham, Steven J. Ramsay & C.-H. Luke Ong (2018): *Defunctionalization of Higher-Order Constrained Horn Clauses*. CoRR abs/1810.03598. Available at `http://arxiv.org/abs/1810.03598`.

[25] Sylvain Salvati & Igor Walukiewicz (2014): *Krivine machines and higher-order schemes*. Information and Computation 239, pp. 340–355, doi:10.1016/j.ic.2014.07.012. Available at `https://hal.inria.fr/inria-00589407/document`.

[26] Géraud Sénizergues (2001): *L(A)=L(B)? decidability results from complete formal systems*. Theor. Comput. Sci. 251(1-2), pp. 1–166, doi:10.1016/S0304-3975(00)00285-1.

[27] Géraud Sénizergues (2002): *L(A)=L(B)? A simplified decidability proof*. Theor. Comput. Sci. 281(1-2), pp. 555–608, doi:10.1016/S0304-3975(02)00027-0.

[28] Luke Simon, Ajay Bansal, Ajay Mallya & Gopal Gupta (2007): *Co-Logic Programming: Extending Logic Programming with Coinduction*. In: *Autom. Lang. Program.*, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 472–483, doi:10.1007/978-3-540-73420-8_42.

[29] Rick Statman (2004): *On the Lambda-Y calculus*. Annals of Pure and Applied Logic 130(1-3 SPEC. ISS.), pp. 325–337, doi:10.1016/j.apal.2004.04.004. Available at https://core.ac.uk/download/pdf/82358399.pdf.

[30] Colin Stirling (2001): *Decidability of DPDA equivalence*. Theor. Comput. Sci. 255(1-2), pp. 1–31, doi:10.1016/S0304-3975(00)00389-3. Available at http://homepages.inf.ed.ac.uk/cps/dpda.pdf.

[31] Mahesh Viswanathan & Ramesh Viswanathan (2004): *A Higher Order Modal Fixed Point Logic*. In Philippa Gardner & Nobuko Yoshida, editors: *CONCUR 2004 - Concurrency Theory*, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 512–528, doi:10.1007/978-3-540-28644-8_33. Available at http://vmahesh.cs.illinois.edu/papers/concur04.pdf.

[32] Dominik Wagner (2019): private communication.

[33] Igor Walukiewicz (2016): *Automata Theory and Higher-Order Model-Checking*. ACM SIGLOG News 3(4), pp. 13–31, doi:10.1145/3026744.3026745. Available at https://www.labri.fr/perso/igw/Papers/igw-siglog16.pdf.

[34] Fabian Zaiser & C.-H. Luke Ong (2020): *The Extended Theory of Trees and Algebraic (Co)datatypes*. Electronic Proceedings in Theoretical Computer Science 320, pp. 167–196, doi:10.4204/eptcs.320.14.

A Correctness proofs from Section 3

A.1 Mappings between HoRS and HoCHC semantics

We define a relaxation of the monotone HoCHC sort frame that we call *relatively monotone*. This new sort frame coincides with our trusted monotone sort frame for sorts ι and $\iota \to o$.

Definition A.1 (Relatively monotone sort frame). *For each sort σ over ι, we define*

$$\mathsf{Im}_\sigma := \{\theta \in \mathcal{D}[\![\mathsf{Rel}^-(\sigma)]\!] \mid \exists h \in \mathcal{H}[\![\sigma]\!].\,\mathbf{i}_\sigma^-(h) = \theta\}$$

where $\mathbf{i}_\sigma^-$ is defined as in Definition A.2, and $\mathcal{D}[\![-]\!]$ denotes the relatively monotone frame*:*

$$\mathcal{D}[\![\iota]\!] := \mathcal{M}[\![\iota]\!]$$
$$\mathcal{D}[\![\mathsf{Rel}^+(\iota)]\!] := \mathcal{M}[\![\iota \to o]\!]$$
$$\mathcal{D}[\![\mathsf{Rel}^+(\sigma_1 \to \sigma_2)]\!] := \left[\mathcal{D}[\![\mathsf{Rel}^-(\sigma_1)]\!] \Rightarrow_{m[\mathsf{Im}_{\sigma_1}]} \mathcal{D}[\![\mathsf{Rel}^+(\sigma_2)]\!]\right]$$

The latter denotes the space of functions that are monotone with respect to Im_{σ_1}, *i.e.* $f : \mathcal{D}[\![\mathsf{Rel}^-(\sigma_1)]\!] \to \mathcal{D}[\![\mathsf{Rel}^+(\sigma_2)]\!]$ *is an element of* $\mathcal{D}[\![\mathsf{Rel}^+(\sigma_1 \to \sigma_2)]\!]$ *just if:* $z_1 \sqsubseteq z_2$ *implies* $f z_1 \sqsubseteq f z_2$ *for all* $z_1, z_2 \in \mathsf{Im}_{\sigma_1}$.

For relational sorts ρ larger than $\iota \to o$, $\mathcal{D}[\![\rho]\!]$ captures a strictly larger set of functions than $\mathcal{M}[\![\rho]\!]$. The following definition extends Definition 3.1 to higher sorts.

Definition A.2. *For all sorts σ over ι, we define two pairs of mappings:*

$$\mathcal{D}[\![\mathsf{Rel}^+(\sigma)]\!] \underset{\mathbf{j}_\sigma}{\overset{\mathbf{i}_\sigma}{\longleftrightarrow}} \mathcal{H}[\![\sigma]\!] \qquad \mathcal{D}[\![\mathsf{Rel}^-(\sigma)]\!] \underset{\mathbf{j}_\sigma^-}{\overset{\mathbf{i}_\sigma^-}{\longleftrightarrow}} \mathcal{H}[\![\sigma]\!]$$

For sort ι, all $t \in \mathcal{H}[\![\iota]\!]$ and $p \in \mathcal{D}[\![\mathsf{Rel}^+(\iota)]\!]$, we define:

$$\mathbf{i}_\iota(t) := \lambda s.\, t \sqsubseteq s \qquad \mathbf{j}_\iota(p) := \begin{cases} \bot & \text{if } p = \lambda s.0 \\ \text{choice } \min\{t \mid p\, t\} & \text{otherwise} \end{cases}$$

where choice *denotes an arbitrary choice function, which exists by the Axiom of Choice.*

For $\sigma = \sigma_1 \to \cdots \to \sigma_m \to \iota$ with $m > 0$, we define the following for all $h \in \mathcal{H}[\![\sigma]\!]$:

$$\mathbf{i}_\sigma h := \lambda x_1^{\mathcal{D}[\![\mathsf{Rel}^-(\sigma_1)]\!]} \ldots x_m^{\mathcal{D}[\![\mathsf{Rel}^-(\sigma_m)]\!]}.\,\mathbf{i}_\iota\left(h\left(\mathbf{j}_{\sigma_1}^- x_1\right) \ldots \left(\mathbf{j}_{\sigma_m}^- x_m\right)\right)$$

with

$$\mathbf{j}_\sigma^- : \mathcal{D}[\![\mathsf{Rel}^-(\sigma)]\!] \to \mathcal{H}[\![\sigma]\!] := \begin{cases} \text{inclusion } \mathcal{D}[\![\iota]\!] \hookrightarrow \mathcal{H}[\![\iota]\!] & \text{if } \sigma = \iota \\ \mathbf{j}_\sigma & \text{otherwise.} \end{cases}$$

Similarly, for all $\theta \in \mathcal{D}[\![\mathsf{Rel}^+(\sigma)]\!]$:

$$\mathbf{j}_\sigma \theta := \text{choice } \max\{h \in \mathcal{H}[\![\sigma]\!] \mid \theta \sqsubseteq \mathbf{i}_\sigma h\}$$

with

$$\mathbf{i}_\sigma^- : \mathcal{H}[\![\sigma]\!] \to \mathcal{D}[\![\mathsf{Rel}^-(\sigma)]\!] := \begin{cases} \text{inclusion } \mathcal{H}[\![\iota]\!] \hookrightarrow \mathcal{D}[\![\iota]\!] & \text{if } \sigma = \iota \\ \mathbf{i}_\sigma & \text{otherwise.} \end{cases}$$

The top case of $\mathbf{j}_\iota$ will not be used in practice.

Lemma A.3. *For all sorts σ over ι,*

(1) $\mathbf{i}_\sigma\left(\perp_{\mathcal{H}[\![\sigma]\!]}\right) = \top_{\mathcal{D}[\![\mathsf{Rel}^+(\sigma)]\!]}$

(2) $\mathbf{i}_\sigma$ *is injective*

(3) $\mathbf{i}_\sigma$ *is antitone*

(4) $\mathbf{j}_\sigma \circ \mathbf{i}_\sigma = \mathrm{id}_{\mathcal{H}[\![\sigma]\!]}$

(5) $\mathbf{j}_\sigma^- \circ \mathbf{i}_\sigma^- = \mathrm{id}_{\mathcal{H}[\![\sigma]\!]}$

Lemma 3.2. *For all directed sets* $D \subseteq \mathcal{H}[\![\iota]\!]$, $\mathbf{i}_\iota\left(\bigsqcup D\right) = \bigsqcap\{\mathbf{i}_\iota(d) \mid d \in D\}$.

Proof. Recall that $\mathcal{H}[\![\iota]\!]$ is a dcpo and $\mathcal{M}[\![\mathsf{Rel}^+(\iota)]\!]$ a complete lattice, so that the bounds are defined.

$$\mathbf{i}_\iota\left(\bigsqcup D\right) = \lambda s.\left(\bigsqcup D \sqsubseteq s\right) = \lambda s. \bigsqcap\{d \sqsubseteq s \mid d \in D\} = \bigsqcap\{\lambda s.(d \sqsubseteq s) \mid d \in D\} = \bigsqcap\{\mathbf{i}_\iota(d) \mid d \in D\}$$

To see that the second equality holds: suppose that $\bigsqcup D \sqsubseteq s$ for some $s \in \mathcal{D}[\![\iota]\!] = \mathcal{H}[\![\iota]\!]$. By transitivity, $d \sqsubseteq \bigsqcup D \sqsubseteq s$ for all $d \in D$. Since the greatest lower bound on $\mathcal{M}[\![o]\!]$ is conjunction, this implies that $\bigsqcap\{d \sqsubseteq s \mid d \in D\}$. For the converse, suppose $\bigsqcap\{d \sqsubseteq s \mid d \in D\}$. This means that $d \sqsubseteq s$, for all $d \in D$. Thus, s is an upper bound on D. However, $\bigsqcup D$ is the least upper bound on this set, so $\bigsqcup D \sqsubseteq s$. □

A.2   Nonemptiness

We aim to show there exists a tree t that does not contain $\perp$ such that $\mathcal{M}[\![\Delta_{\mathcal{G}} \vdash \ulcorner S \urcorner]\!](T_{P_{\mathcal{G}}:\Delta_{\mathcal{G}}}^{\mathcal{M}n}(\top_{\Delta_{\mathcal{G}}}))t = 1$, for every $n \geq 0$ (Corollary A.7). To this end, we define a family of logical relations in Definition A.4 to capture this notion at higher sorts and for larger sort environments, as proved in Lemma A.6.

Intuitively, such a relation holds whenever a predicate maps nonempty inputs to nonempty outputs, where "nonempty" is taken to mean with respect to $\perp$-free trees.

Definition A.4. *We define a family of logical relations* $\perp\text{-free}_\sigma \subseteq \mathcal{M}[\![\sigma]\!]$:

$$\perp\text{-free}_\iota(t) := t \text{ has no } \perp\text{-labelled leaves}$$

$$\perp\text{-free}_{\mathsf{Rel}^+(\iota)}(p) := \exists s \in \mathcal{M}[\![\iota]\!]. \perp\text{-free}_\iota(s) \wedge p\,s = 1$$

$$\perp\text{-free}_{\mathsf{Rel}^+(\sigma_1 \to \sigma_2)}(p) := \forall s \in \mathcal{M}[\![\mathsf{Rel}^-(\sigma_1)]\!]. \perp\text{-free}_{\mathsf{Rel}^-(\sigma_1)}(s) \Rightarrow \perp\text{-free}_{\mathsf{Rel}^+(\sigma_2)}(p\,s)$$

$$\perp\text{-free}_{\mathsf{Rel}^-(\Gamma)}(\theta) := \mathsf{dom}(\ulcorner\Gamma\urcorner) = \mathsf{dom}(\theta) \wedge \bigwedge_{x':\mathsf{Rel}^-(\sigma)\in\ulcorner\Gamma\urcorner} \perp\text{-free}_{\mathsf{Rel}^-(\sigma)}(\theta(x'))$$

Alternatively, we can define, for all $\bar{s}$ *from the appropriate domains:*

$$\perp\text{-free}_{\mathsf{Rel}^+(\sigma_1 \to \cdots \to \sigma_m \to \iota)}(p) := \forall \bar{s}. \left(\bigwedge_{i\in[m]} \perp\text{-free}_{\mathsf{Rel}^-(\sigma_i)}(s_i)\right) \Rightarrow \perp\text{-free}_{\mathsf{Rel}^+(\iota)}(p\,s_1\ldots s_m)$$

Lemma A.5. *For all* $p_1, p_2 \in \mathcal{M}[\![\mathsf{Rel}^+(\iota)]\!]$, *if* $p_1 \sqsubseteq p_2$ *and* $\perp\text{-free}_{\mathsf{Rel}^+(\iota)}(p_1)$, *then* $\perp\text{-free}_{\mathsf{Rel}^+(\iota)}(p_2)$.

Proof. Trivial, using that witness $t \in \mathcal{M}[\![\iota]\!]$ of $\perp\text{-free}_{\mathsf{Rel}^+(\iota)}(p_1)$ also witnesses $\perp\text{-free}_{\mathsf{Rel}^+(\iota)}(p_2)$. □

Lemma A.6. *For all* $n \geq 0$, *all typing judgements* $\mathcal{N}, \Gamma \vdash e : \sigma$ *of the HoRS* $\mathcal{G}$ *where* $\Gamma = \{x_1 : \tau_1, \ldots, x_k : \tau_k\}$, *and valuations* $\theta \in \mathcal{M}[\![\ulcorner\Gamma\urcorner]\!]$,

$$\mathcal{N}, \Gamma \vdash e : \sigma \wedge \perp\text{-free}_{\mathsf{Rel}^-(\Gamma)}(\theta) \quad \Rightarrow \quad \perp\text{-free}_{\mathsf{Rel}^+(\sigma)}(\mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner\Gamma\urcorner \vdash \ulcorner e \urcorner]\!](\beta^n))$$

where

$$\ulcorner\Gamma\urcorner := \{x_1' : \mathsf{Rel}^-(\tau_1), \ldots, x_k' : \mathsf{Rel}^-(\tau_k)\}$$

$$\beta^n := \left(T_{P_{\mathcal{G}}:\Delta_{\mathcal{G}}}^{\mathcal{M}n}(\top_{\Delta_{\mathcal{G}}})\right)[\overline{x'} \mapsto \overline{\theta(x')}] \in \mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner\Gamma\urcorner]\!].$$

Notice that $\mathcal{N}, \Gamma \vdash e : \sigma$ *implies* $\Delta_{\mathcal{G}}, \ulcorner\Gamma\urcorner \vdash \ulcorner e \urcorner : \mathsf{Rel}^+(\sigma)$.

Proof. We proceed by induction on $n \geq 0$ within which (both in the base case and the induction step) we use structural induction on HoRS term e. Some parts of the proof are presented out of order to avoid duplication. Figure 3 outlines the structure of the proof. We assume WLOG that e contains no λs.

We use the following shorthand, for $e : \sigma = \sigma_1 \to \cdots \to \sigma_m \to \iota$ and $z_i \in \mathcal{M}[\![\mathsf{Rel}^-(\sigma_i)]\!]$ such that $\perp\text{-free}_{\mathsf{Rel}^-(\sigma_i)}(z_i)$, for all $i \in [m]$:

$$A = \mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner\Gamma\urcorner \vdash \ulcorner e : \sigma\urcorner]\!](\beta^n)\,\overline{z}$$

Our proof strategy is to rewrite A and provide a witness to $\perp\text{-free}_{\mathsf{Rel}^+(\iota)}(\mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner\Gamma\urcorner \vdash \ulcorner e : \sigma\urcorner]\!](\beta^n)\,\overline{z})$, which proves $\perp\text{-free}_{\mathsf{Rel}^+(\sigma)}(\mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner\Gamma\urcorner \vdash \ulcorner e : \sigma\urcorner]\!](\beta^n))$.

We present the following base cases w.r.t. the structure of e (also denoted b for *base case expression*).

Case $e = x : \iota \in V_{\mathsf{RS}}$. For all $n \geq 0$, the tree $\theta(x')$ is a witness thanks to $\perp\text{-free}_{\mathsf{Rel}^-(\Gamma)}(\theta)$:

$$A = \mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner\Gamma\urcorner \vdash \ulcorner x\urcorner]\!](\beta^n) = \mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner\Gamma\urcorner \vdash \lambda r. x' = r]\!](\beta^n) = \lambda s. \big(\beta^n(x') = s\big) = \lambda s. \big(\theta(x') = s\big)$$

Case $e = f : \sigma_1 \to \cdots \to \sigma_m \to \iota \in \Sigma$. For all $n \geq 0$, the $\perp$-free tree $\widehat{F}_f\,\overline{z}$ is a witness:

$$A = \mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner\Gamma\urcorner \vdash \ulcorner f\urcorner]\!](\beta^n)\,\overline{z} = \mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner\Gamma\urcorner \vdash \lambda \overline{y}r. f\overline{y} = r]\!](\beta^n)\,\overline{z} = \lambda s. \left(\widehat{F}_f\,\overline{z} = s\right)$$

Case $e = x : \sigma_1 \to \cdots \to \sigma_m \to \iota \in V_{\mathsf{RS}}$ and $m > 0$. For all $n \geq 0$, it holds that $\perp\text{-free}_{\mathsf{Rel}^+(\iota)}(\theta(x')\overline{z})$:

$$A = \mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner\Gamma\urcorner \vdash \ulcorner x\urcorner]\!](\beta^n)\,\overline{z} = \mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner\Gamma\urcorner \vdash \lambda \overline{y}r. x'\overline{y}r]\!](\beta^n)\,\overline{z} = \lambda s. \beta^n(x')\overline{z}s = \theta(x')\overline{z}$$

Case $e = F : \sigma_1 \to \cdots \to \sigma_m \to \iota \in \mathcal{N}$ and $n = 0$. Any $\perp$-free tree (e.g. $[\![\mathcal{G}]\!]$) is a witness:

$$A = \mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner\Gamma\urcorner \vdash \ulcorner F\urcorner]\!](\beta^0)\,\overline{z} = \mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner\Gamma\urcorner \vdash \lambda \overline{y}r. R_F\,\overline{y}r]\!](\beta^0)\,\overline{z} = \lambda s. \beta^0(R_F)\overline{z}s = \top_{\mathcal{M}[\![\mathsf{Rel}^+(\sigma)]\!]}\,\overline{z}$$

This covers $n = 0$ for all base case expressions. We distinguish three induction hypotheses, where $S(n,e)$ denotes that the claim holds for n and expression e. The proof consists of four parts (in a logical sense but not a physical, to prevent duplication) which are related as in Figure 3. Thus, we have now proved $S(0,b)$ for all base case expressions b. Next, we use IH1 to show $S(0,e)$ for all expressions e.

IH1 $S(0,e')$ for all expressions e' simpler than e

IH2 $S(n,e'')$ for n and all e''

IH3 $S(n+1,e')$ for all expressions e' simpler than e

$$\forall b.\, S(0,b) \xrightarrow{\text{IH1}} \forall e.\, S(0,e)$$
$$\xrightarrow{\text{IH2}} \quad \downarrow \text{IH2}$$
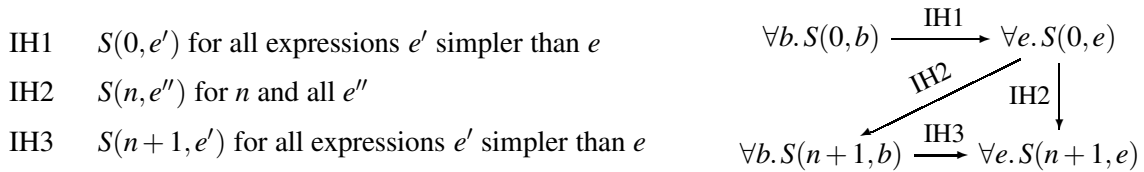$$\forall b.\, S(n+1,b) \xrightarrow{\text{IH3}} \forall e.\, S(n+1,e)$$

Figure 3: The inductive structure of the correctness proof of the HoRS-to-HoCHC encoding.

In this inductive case, we consider expressions $e = \$e_1 \ldots e_\ell : \sigma_1 \to \cdots \to \sigma_m \to \iota$ for some $\ell > 0$. As before, let $z_i \in \mathcal{M}[\![\mathsf{Rel}^-(\sigma_i)]\!]$ such that $\perp\text{-free}_{\mathsf{Rel}^-(\sigma_i)}(z_i)$, for each $i \in [m]$. We introduce some shorthands:

$$\Delta'_{\mathcal{G}} := \Delta_{\mathcal{G}}, \ulcorner\Gamma\urcorner, \overline{y}, r \qquad \Delta''_{\mathcal{G}} := \Delta_{\mathcal{G}}, \ulcorner\Gamma\urcorner, \overline{y}, r, \overline{r} \qquad \beta^{n,\overline{z},s} := \beta^n[\overline{y} \mapsto \overline{z}, r \mapsto s]$$

Note that the sort τ of $\$$ is of the form

$$\tau = \tau_1 \to \cdots \to \tau_\ell \to \sigma_1 \to \cdots \to \sigma_m \to \iota$$

where $e_1 : \tau_1, \ldots, e_\ell : \tau_\ell$, for some $\ell > 0$. Sometimes we abbreviate $\sigma_1 \to \cdots \to \sigma_m \to \iota$ to σ.

In the sequel, steps marked with † use IH1 for $n = 0$, and IH3 for $n > 0$.
For all $n \geq 0$ and expressions $e = \$\overline{e}$, we can rewrite A:

$$A = \mathcal{M}[\![\Delta_\mathcal{G}, \ulcorner\Gamma\urcorner \vdash \ulcorner\$\overline{e}\urcorner]\!](\beta^n)\overline{z}$$

$$= \mathcal{M}[\![\Delta_\mathcal{G}, \ulcorner\Gamma\urcorner \vdash \lambda\overline{y}r.\exists\overline{r}.\$'\ulcorner\top\urcorner(e_1, r_1)\ulcorner\top\urcorner \ldots \ulcorner\top\urcorner(e_\ell, r_\ell)\ulcorner\top\urcorner\overline{y}r \wedge \bigwedge_{i \in [\ell]} Prop(e_i, r_i)]\!](\beta^n)\overline{z}$$

$$= \lambda s.\mathcal{M}[\![\Delta'_\mathcal{G} \vdash \exists\overline{r}.\$'\ulcorner\top\urcorner(e_1, r_1)\ulcorner\top\urcorner \ldots \ulcorner\top\urcorner(e_\ell, r_\ell)\ulcorner\top\urcorner\overline{y}r \wedge \bigwedge_{i \in [\ell]} Prop(e_i, r_i)]\!](\beta^{n,\overline{z},s})$$

$$= \lambda s.\max \Big\{$$
$$\min\{$$
$$\mathcal{M}[\![\Delta''_\mathcal{G} \vdash \$']\!](\beta^{n,\overline{z},s}[\overline{r} \mapsto \overline{r'}])(\mathcal{M}[\![\Delta''_\mathcal{G} \vdash \ulcorner\top\urcorner(e_1, r_1)\ulcorner\top\urcorner]\!](\beta^{n,\overline{z},s}[\overline{r} \mapsto \overline{r'}]))$$
$$\ldots (\mathcal{M}[\![\Delta''_\mathcal{G} \vdash \ulcorner\top\urcorner(e_\ell, r_\ell)\ulcorner\top\urcorner]\!](\beta^{n,\overline{z},s}[\overline{r} \mapsto \overline{r'}]))\overline{z}s,$$
$$\min\{\mathcal{M}[\![\Delta''_\mathcal{G} \vdash Prop(e_i, r_i)]\!](\beta^{n,\overline{z},s}[\overline{r} \mapsto \overline{r'}]) \mid i \in [\ell]\}$$
$$\}$$
$$\mid \forall i \in [\ell].r'_i \in \mathcal{M}[\![\mathsf{Rel}^-(\tau_i)]\!]\Big\}$$

We now distinguish two cases for each subexpression $e_i : \tau_i$, namely $\tau_i = \iota$ and $\tau_i = \tau'_1 \to \tau'_2$.
   If e_i is of sort ι, then the following holds.

$$\mathcal{M}[\![\Delta''_\mathcal{G} \vdash Prop(e_i : \iota, r_i)]\!](\beta^{n,\overline{z},s}[\overline{r} \mapsto \overline{r'}]) = \mathcal{M}[\![\Delta_\mathcal{G}, \ulcorner\Gamma\urcorner \vdash \ulcorner e_i\urcorner]\!](\beta^n)r'_i$$
$$\mathcal{M}[\![\Delta''_\mathcal{G} \vdash \ulcorner\top\urcorner(e_i : \iota, r_i)\ulcorner\top\urcorner]\!](\beta^{n,\overline{z},s}[\overline{r} \mapsto \overline{r'}]) = r'_i$$

We know from † that $\bot\text{-free}_{\mathsf{Rel}^+(\iota)}(\mathcal{M}[\![\Delta_\mathcal{G}, \ulcorner\Gamma\urcorner \vdash \ulcorner e_i\urcorner]\!](\beta^n))$. This means that there exists $r'' \in \mathcal{M}[\![\iota]\!]$
such that $\bot\text{-free}_\iota(r'')$ and $\mathcal{M}[\![\Delta_\mathcal{G}, \ulcorner\Gamma\urcorner \vdash \ulcorner e_i\urcorner]\!](\beta^n)r'' = 1$.
   Otherwise, in case $e_i : \tau_i = \tau'_1 \to \tau'_2$, the following holds:

$$\mathcal{M}[\![\Delta''_\mathcal{G} \vdash Prop(e_i : \tau_i, r_i)]\!](\beta^{n,\overline{z},s}[\overline{r} \mapsto \overline{r'}]) = 1$$
$$\mathcal{M}[\![\Delta''_\mathcal{G} \vdash \ulcorner\top\urcorner(e_i : \tau_i, r_i)\ulcorner\top\urcorner]\!](\beta^{n,\overline{z},s}[\overline{r} \mapsto \overline{r'}]) = \mathcal{M}[\![\Delta_\mathcal{G}, \ulcorner\Gamma\urcorner \vdash \ulcorner e_i\urcorner]\!](\beta^n)$$

We know from † that $\bot\text{-free}_{\mathsf{Rel}^+(\tau_i)}(\mathcal{M}[\![\Delta_\mathcal{G}, \ulcorner\Gamma\urcorner \vdash \ulcorner e_i\urcorner]\!](\beta^n))$.
   As "semantic equivalents" of the above terms, let us write

$$P_i : o := \begin{cases} \mathcal{M}[\![\Delta_\mathcal{G}, \ulcorner\Gamma\urcorner \vdash \ulcorner e_i\urcorner]\!](\beta^n)r'_i & \text{if } \tau_i = \iota \\ 1 & \text{if } \tau_i = \tau'_1 \to \tau'_2 \end{cases}$$

and

$$T_i : \mathsf{Rel}^-(\tau_i) := \begin{cases} r'_i & \text{if } \tau_i = \iota \\ \mathcal{M}[\![\Delta_\mathcal{G}, \ulcorner\Gamma\urcorner \vdash \ulcorner e_i\urcorner]\!](\beta^n) & \text{if } \tau_i = \tau'_1 \to \tau'_2 \end{cases}$$

for all $i \in [\ell]$. Additionally, we define:

$$S_i : \mathsf{Rel}^-(\tau_i) := \begin{cases} r''_i & \text{if } \tau_i = \iota \\ \mathcal{M}[\![\Delta_\mathcal{G}, \ulcorner\Gamma\urcorner \vdash \ulcorner e_i\urcorner]\!](\beta^n) & \text{if } \tau_i = \tau'_1 \to \tau'_2 \end{cases}$$

where r''_i is an arbitrary ($\bot$-free) witness to $\bot\text{-free}_{\mathsf{Rel}^+(\iota)}(\mathcal{M}[\![\Delta_\mathcal{G}, \ulcorner\Gamma\urcorner \vdash \ulcorner e_i\urcorner]\!](\beta^n))$, which exists by †.
This gives us $\bot\text{-free}_{\mathsf{Rel}^-(\tau_i)}(S_i)$ for all $i \in [m]$.

We derive by abuse of notation, using the above:

$$A = \lambda s. \exists \overline{r'}. \left(\mathcal{M}[\![\Delta''_{\mathcal{G}} \vdash \$']\!](\beta^{n,\overline{z},s}[\overline{r} \mapsto \overline{r'}]) \overline{T} \, \overline{z} s \wedge \bigwedge_{i \in [\ell]} P_i \right)$$

We continue by case analysis on $\$$.

Case $e = f e_1 \ldots e_\ell$ with $f \in \Sigma$. For all $n \geq 0$, the $\perp$-free tree $\widehat{F}_f \, \overline{r''} \, \overline{z}$ is a witness:

$$A = \lambda s. \exists \overline{r'}. \left(\mathcal{M}[\![\Delta''_{\mathcal{G}} \vdash D_f]\!](\beta^{n,\overline{z},s}[\overline{r} \mapsto \overline{r'}]) \overline{r'} \, \overline{z} s \wedge \bigwedge_{i \in [\ell]} \mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner \Gamma \urcorner \vdash \ulcorner e_i \urcorner]\!](\beta^n) r'_i \right)$$

$$= \lambda s. \exists \overline{r'}. \left(\widehat{F}_f \, \overline{r'} \, \overline{z} = s \wedge \bigwedge_{i \in [\ell]} \mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner \Gamma \urcorner \vdash \ulcorner e_i \urcorner]\!](\beta^n) r'_i \right)$$

$$\sqsupseteq \lambda s. \left(\widehat{F}_f \, \overline{r''} \, \overline{z} = s \right) \qquad \dagger$$

Case $e = x e_1 \ldots e_\ell$ with $x \in V_{RS}$. For all $n \geq 0$, it holds that $\perp\text{-free}_{\mathsf{Rel}^+(\iota)}(\theta(x') \, \overline{S} \overline{z})$ and:

$$A = \lambda s. \exists \overline{r'}. \left(\mathcal{M}[\![\Delta''_{\mathcal{G}} \vdash x']\!](\beta^{n,\overline{z},s}[\overline{r} \mapsto \overline{r'}]) \overline{T} \, \overline{z} s \wedge \bigwedge_{i \in [\ell]} P_i \right)$$

$$= \lambda s. \exists \overline{r'}. \left(\beta^{n,\overline{z},s}[\overline{r} \mapsto \overline{r'}](x') \overline{T} \, \overline{z} s \wedge \bigwedge_{i \in [\ell]} P_i \right)$$

$$= \lambda s. \exists \overline{r'}. \left(\theta(x') \overline{T} \, \overline{z} s \wedge \bigwedge_{i \in [\ell]} P_i \right)$$

$$\sqsupseteq \lambda s. \theta(x') \overline{S} \, \overline{z} s \qquad \dagger$$

$$= \theta(x') \overline{S} \, \overline{z}$$

Case $e = F e_1 \ldots e_\ell$ with $F : \sigma \in \mathcal{N}$, and $n = 0$. Any $\perp$-free tree (e.g. $[\![\mathcal{G}]\!]$) is a witness:

$$A = \lambda s. \exists \overline{r'}. \left(\mathcal{M}[\![\Delta''_{\mathcal{G}} \vdash R_F]\!](\beta^{0,\overline{z},s}[\overline{r} \mapsto \overline{r'}]) \overline{T} \, \overline{z} s \wedge \bigwedge_{i \in [\ell]} P_i \right)$$

$$= \lambda s. \exists \overline{r'}. \left(\beta^{0,\overline{z},s}[\overline{r} \mapsto \overline{r'}](R_F) \overline{T} \, \overline{z} s \wedge \bigwedge_{i \in [\ell]} P_i \right)$$

$$= \lambda s. \exists \overline{r'}. \left(\top_{\mathcal{M}[\![\mathsf{Rel}^+(\tau)]\!]} \overline{T} \, \overline{z} s \wedge \bigwedge_{i \in [\ell]} P_i \right)$$

$$= \lambda s. \exists \overline{r'}. \left(1 \wedge \bigwedge_{i \in [\ell]} P_i \right)$$

$$= \lambda s. 1 \qquad \text{IH1}$$

We have now established that $S(0,e'')$ holds for expressions all e''. The following case is the last remaining case to prove $S(n+1,b)$ for all base case expressions b:

Case $e = F : \sigma \in \mathcal{N}$, for $n+1$. Thanks to IH2, $\perp$-free$_{\text{Rel}^+(\iota)}(\mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner\Gamma\urcorner \vdash \ulcorner\mathcal{R}(F)\urcorner]\!](\beta^n)\bar{z})$ and:

$$
\begin{aligned}
A &= \mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner\Gamma\urcorner \vdash \ulcorner F\urcorner]\!](\beta^{n+1})\bar{z} \\
&= \mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner\Gamma\urcorner \vdash \lambda \bar{y}\, r. R_F\, \bar{y}\, r]\!](\beta^{n+1})\bar{z} \\
&= \lambda s.\, \beta^{n+1,\bar{z},s}(R_F)\bar{z}s \\
&= \lambda s.\, \beta^{n+1}(R_F)\bar{z}s \\
&= \lambda s.\, \mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner\Gamma\urcorner \vdash \ulcorner\mathcal{R}(F)\urcorner]\!](\beta^n)\bar{z}s \\
&= \mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner\Gamma\urcorner \vdash \ulcorner\mathcal{R}(F)\urcorner]\!](\beta^n)\bar{z}
\end{aligned}
$$

Finally, we present the remaining case to prove that $S(n',e'')$ for all $n' \geq 0$ and all expressions e''.
Case $e = F\, e_1 \ldots e_\ell$ with $F \in \mathcal{N}$ and $\ell > 0$, for $n+1$.

$$
\begin{aligned}
A &= \lambda s. \exists \bar{r'}.\ \left(\mathcal{M}[\![\Delta_{\mathcal{G}}'' \vdash R_F]\!](\beta^{n+1,\bar{z},s}[\bar{r} \mapsto \bar{r'}])\,\overline{T}\,\bar{z}s \wedge \bigwedge_{i \in [\ell]} P_i \right) \\
&= \lambda s. \exists \bar{r'}.\ \left(\beta^{n+1,\bar{z},s}[\bar{r} \mapsto \bar{r'}](R_F)\,\overline{T}\,\bar{z}s \wedge \bigwedge_{i \in [\ell]} P_i \right) \\
&= \lambda s. \exists \bar{r'}.\ \left(\beta^{n+1}(R_F)\,\overline{T}\,\bar{z}s \wedge \bigwedge_{i \in [\ell]} P_i \right) \\
&= \lambda s. \exists \bar{r'}.\ \left(\mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner\Gamma\urcorner \vdash \ulcorner\mathcal{R}(F)\urcorner]\!](\beta^n)\,\overline{T}\,\bar{z}s \wedge \bigwedge_{i \in [\ell]} P_i \right) \\
&\sqsupseteq \lambda s.\, \mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner\Gamma\urcorner \vdash \ulcorner\mathcal{R}(F)\urcorner]\!](\beta^n)\,\overline{S}\,\bar{z}s \qquad\qquad \text{IH3} \\
&= \mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner\Gamma\urcorner \vdash \ulcorner\mathcal{R}(F)\urcorner]\!](\beta^n)\,\overline{S}\,\bar{z}
\end{aligned}
$$

By IH2, $\perp$-free$_{\text{Rel}^+(\tau)}(\mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner\Gamma\urcorner \vdash \ulcorner\mathcal{R}(F)\urcorner]\!](\beta^n))$. All arguments $\overline{S}$ and $\bar{z}$ are also $\perp$-free, so it follows that $\perp$-free$_{\text{Rel}^+(\iota)}(\mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner\Gamma\urcorner \vdash \ulcorner\mathcal{R}(F)\urcorner]\!](\beta^n)\,\overline{S}\,\bar{z})$, as required. $\qquad\qquad\square$

Corollary A.7. *For all $n \geq 0$ and HoRS $\mathcal{G}$, $\perp$-free$_{\text{Rel}^+(\iota)}(\mathcal{M}[\![\Delta_{\mathcal{G}} \vdash \ulcorner S\urcorner]\!](\beta^n))$. I.e. there exists a $\perp$-free tree $t \in \mathcal{M}[\![\iota]\!]$ such that $\mathcal{M}[\![\Delta_{\mathcal{G}} \vdash \ulcorner S\urcorner]\!](\beta^n)\, t$.*

Lemma 3.4 (Nonemptiness). *There exists a $\perp$-free tree $t \in \mathcal{M}[\![\iota]\!]$ such that $\mathcal{M}[\![\Delta_{\mathcal{G}} \vdash \ulcorner S\urcorner]\!](\bigsqcap \beta^n)\, t$.*

Proof sketch. Note that the constructed HoCHC logic program $\vdash P_{\mathcal{G}} : \Delta_{\mathcal{G}}$ is *incremental* in the sense that each iteration of the one-step consequence operator (further) constrains a finite prefix of the trees it generates. For sort $\iota \to o$, this means that either a contradiction occurs in finite time (e.g. $\exists r_1.\, a\, r_1 = r \wedge b\, r_1 = r$ where a, b are distinct unary alphabet symbols) or no contradiction occurs and the program is strictly incremental.

By Corollary A.7, no contradiction occurs after finite time. This means that no contradiction occurs at all and $\vdash P_{\mathcal{G}} : \Delta_{\mathcal{G}}$ is strictly incremental. It follows that there exists a $\perp$-free tree $t \in \mathcal{M}[\![\iota]\!]$ such that $\mathcal{M}[\![\Delta_{\mathcal{G}} \vdash \ulcorner S\urcorner]\!](\bigsqcap \beta^n)\, t$. $\qquad\qquad\square$

A.3   Inclusion

We aim to show that $\mathcal{M}[\![\Delta_{\mathcal{G}} \vdash \ulcorner S \urcorner]\!](T_{P_{\mathcal{G}}:\Delta_{\mathcal{G}}}^{\mathcal{M} n}(\top_{\Delta_{\mathcal{G}}}))$ is included in $\mathbf{i}_\iota(\mathcal{H}[\![\mathcal{N} \vdash S]\!]((\mathcal{H}[\![\mathcal{G}]\!]_{\mathcal{N}}^n(\bot_{\mathcal{N}})))$, for every $n \geq 0$ (Corollary 3.5). To this end, we define a family of logical relations in Definition A.8 to capture this notion at higher sorts and for larger sort environments, as proved in Lemma A.10.

The intuition is that the relation comprises pairs that preserve order on order-preserving arguments.

Definition A.8. *We define a family of logical relations* $\mathsf{Incl}_\sigma \subseteq \mathcal{M}[\![\sigma]\!] \times \mathcal{D}[\![\sigma]\!]$:

$$\mathsf{Incl}_\iota(t_1, t_2) := t_1 = t_2$$

$$\mathsf{Incl}_{\mathsf{Rel}^+(\iota)}(p_1, p_2) := p_1 \sqsubseteq p_2$$

$$\mathsf{Incl}_{\mathsf{Rel}^+(\sigma_1 \to \sigma_2)}(p_1, p_2) := \forall w \in \mathcal{M}[\![\mathsf{Rel}^-(\sigma_1)]\!]. \forall z \in \mathcal{H}[\![\sigma_1]\!].$$

$$\mathsf{Incl}_{\mathsf{Rel}^-(\sigma_1)}(w, \mathbf{i}_{\sigma_1}^-(z)) \Rightarrow \mathsf{Incl}_{\mathsf{Rel}^+(\sigma_2)}(p_1\, w, p_2\, \mathbf{i}_{\sigma_1}^-(z))$$

$$\mathsf{Incl}_\Gamma(\theta_\beta, \theta_\alpha) := \mathrm{dom}(\ulcorner\Gamma\urcorner) = \mathrm{dom}(\theta_\beta) \wedge \mathrm{dom}(\Gamma) = \mathrm{dom}(\theta_\alpha) \wedge$$

$$\bigwedge_{x:\sigma \in \Gamma} \mathsf{Incl}_{\mathsf{Rel}^-(\sigma)}(\theta_\beta(x'), \mathbf{i}_\sigma^-(\theta_\alpha(x)))$$

Alternatively, $\mathsf{Incl}_{\mathsf{Rel}^+(\sigma_1 \to \cdots \to \sigma_m \to \iota)}(p_1, p_2)$ *can be defined as*

$$\forall \overline{w}. \forall \overline{z}. \left(\bigwedge_{i \in [m]} \mathsf{Incl}_{\mathsf{Rel}^-(\sigma_i)}(w_i, \mathbf{i}_{\sigma_i}^-(z_i)) \right) \Rightarrow \mathsf{Incl}_{\mathsf{Rel}^+(\iota)}(p\, w_1 \ldots w_m, p_2\, \mathbf{i}_{\sigma_1}^-(z_1) \ldots \mathbf{i}_{\sigma_m}^-(z_m))$$

for all $\overline{w}$ *and* $\overline{z}$ *from the appropriate domains.*

Note that in general $\mathcal{M}[\![\sigma]\!]$ differs from $\mathcal{D}[\![\sigma]\!]$, and the arguments of Incl_σ do not necessarily live in the same set. However, for the sort we are interested in, namely $\mathsf{Rel}^+(\iota) = \iota \to o$, the denotations $\mathcal{M}[\![\iota \to o]\!]$ and $\mathcal{D}[\![\iota \to o]\!]$ coincide (idem for ι), so that the relations are well-defined.

Lemma A.9. *For* $F \in \mathcal{H}[\![\sigma_1 \to \cdots \to \sigma_k \to \tau_1 \to \cdots \to \tau_\ell \to \iota]\!]$, $t_i \in \mathcal{H}[\![\sigma_i]\!]$ *for all* $i \in [k]$, *and* $z_j \in \mathcal{H}[\![\tau_j]\!]$ *for all* $j \in [\ell]$,

$$\left(\lambda s. \exists \overline{r}. F\, \overline{r}\, \overline{z} = s \wedge \bigwedge_{i \in [k]} t_i \sqsubseteq r_i \right) \sqsubseteq \left(\lambda s. \exists \overline{r}. F\, \overline{r}\, \overline{z} \sqsubseteq s \wedge \bigwedge_{i \in [k]} t_i \sqsubseteq r_i \right) \sqsubseteq (\lambda s. F\, \overline{t}\, \overline{z} \sqsubseteq s)$$

where $\overline{r} = r_1 \ldots r_k$, $\overline{t} = t_1 \ldots t_k$, *and* $\overline{z} = z_1 \ldots z_\ell$.

Lemma A.10. *For all* $n \geq 0$, *all typing judgements* $\mathcal{N}, \Gamma \vdash e : \sigma$ *of the HoRS* $\mathcal{G}$ *where* $\Gamma = \{x_1 : \tau_1, \ldots, x_k : \tau_k\}$, *and valuations* $\theta_\alpha \in \mathcal{H}[\![\Gamma]\!]$ *and* $\theta_\beta \in \mathcal{M}[\![\ulcorner\Gamma\urcorner]\!]$,

$$\mathcal{N}, \Gamma \vdash e : \sigma \wedge \mathsf{Incl}_\Gamma(\theta_\beta, \theta_\alpha) \quad \Rightarrow$$
$$\mathsf{Incl}_{\mathsf{Rel}^+(\sigma)}(\mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner\Gamma\urcorner \vdash \ulcorner e \urcorner]\!](\beta^n), \mathbf{i}_\sigma(\mathcal{H}[\![\mathcal{N}, \Gamma \vdash e]\!](\alpha^n)))$$

where

$$\ulcorner\Gamma\urcorner := \{x_1' : \mathsf{Rel}^-(\tau_1), \ldots, x_k' : \mathsf{Rel}^-(\tau_k)\}$$
$$\alpha^n := (\mathcal{H}[\![\mathcal{G}]\!]_{\mathcal{N}}^n(\bot_{\mathcal{N}}))[\overline{x} \mapsto \theta_\alpha(x)] \in \mathcal{H}[\![\mathcal{N}, \Gamma]\!]$$
$$\beta^n := \left(T_{P_{\mathcal{G}}:\Delta_{\mathcal{G}}}^{\mathcal{M} n}(\top_{\Delta_{\mathcal{G}}}) \right)[\overline{x'} \mapsto \overline{\theta_\beta(x')}] \in \mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner\Gamma\urcorner]\!].$$

Notice that $\mathcal{N}, \Gamma \vdash e : \sigma$ *implies* $\Delta_{\mathcal{G}}, \ulcorner\Gamma\urcorner \vdash \ulcorner e \urcorner : \mathsf{Rel}^+(\sigma)$.

Proof. We proceed by induction on $n \geq 0$ within which (both in the base case and the induction step) we use structural induction on HoRS term e. Some parts of the proof are presented out of order to avoid duplication. In fact, the structure of this proof and order of presentation correspond to the proof of Lemma A.6, the structure of which is outlined in Figure 3. We again assume WLOG e contains no λs.

We use the following shorthands for $e : \sigma = \sigma_1 \to \cdots \to \sigma_m \to \iota$, $w_i \in \mathcal{M}[\![\mathsf{Rel}^-(\sigma_i)]\!]$, and $z_i \in \mathcal{H}[\![\sigma_i]\!]$ such that $\mathsf{Incl}_{\mathsf{Rel}^-(\sigma_i)}(w_i, \mathbf{i}^-_{\sigma_i}(z_i))$, for all $i \in [m]$:

$$B = \mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner\Gamma\urcorner \vdash \ulcorner e\urcorner]\!](\beta^n)\overline{w} \qquad\qquad C = \mathbf{i}_\sigma(\mathcal{H}[\![\mathcal{N}, \Gamma \vdash e]\!](\alpha^n))\overline{\mathbf{i}^-(z)}$$

Thus, both B and C are both elements of $\mathcal{M}[\![\iota \to o]\!] = \mathcal{D}[\![\iota \to o]\!]$, and it suffices to show that $B \sqsubseteq C$.

We present the following base cases w.r.t. the structure of e (also denoted b for *base case expression*).

Case $e = x : \iota \in V_{\mathrm{RS}}$. For all $n \geq 0$, $\mathsf{Incl}_\Gamma(\theta_\beta, \theta_\alpha)$ implies $\beta^n(x') = \alpha^n(x)$ and:

$$\begin{aligned}
B &= \mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner\Gamma\urcorner \vdash \ulcorner x\urcorner]\!](\beta^n) \\
&= \mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner\Gamma\urcorner \vdash \lambda r.x' = r]\!](\beta^n) \\
&= \lambda s \in \mathcal{M}[\![\iota]\!].\left(\beta^n(x') = s\right) \\
&= \lambda s \in \mathcal{M}[\![\iota]\!].\left(\alpha^n(x) = s\right) \\
&\sqsubseteq \mathbf{i}_\iota(\alpha^n(x)) \\
&= \mathbf{i}_\iota(\mathcal{H}[\![\mathcal{N}, \Gamma \vdash x]\!](\alpha^n)) \\
&= C
\end{aligned}$$

Case $e = \$: \sigma_1 \to \cdots \to \sigma_m \to \iota \in \Sigma \cup V_{\mathrm{RS}}$. For all $n \geq 0$, we rely on Lemma A.3. If $\$ \in \mathrm{dom}(V_{\mathrm{RS}})$, let $m > 0$. Then, $\mathsf{Incl}_{\mathsf{Rel}^-(\sigma)}(\beta^n(x'), \mathbf{i}^-_\sigma(\alpha^n(x)))$ and $\mathsf{Incl}_{\mathsf{Rel}^-(\sigma_j)}(w_j, \mathbf{i}^-_{\sigma_j}(z_j))$, for all $j \in [m]$, such that:

$$\begin{aligned}
B &= \mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner\Gamma\urcorner \vdash \ulcorner f\urcorner]\!](\beta^n)\overline{w} & B &= \mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner\Gamma\urcorner \vdash \ulcorner x\urcorner]\!](\beta^n)\overline{w} \\
&= \mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner\Gamma\urcorner \vdash \ulcorner f\urcorner]\!](\beta^n)\overline{z} & &= \mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner\Gamma\urcorner \vdash \lambda \overline{y}r.x' \, \overline{y}r]\!](\beta^n)\overline{w} \\
&= \mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner\Gamma\urcorner \vdash \lambda \overline{y}r.f\,\overline{y} = r]\!](\beta^n)\overline{z} & &= \lambda s \in \mathcal{M}[\![\iota]\!].\beta^n(x')\overline{w}s \\
&= \lambda s \in \mathcal{M}[\![\iota]\!].\left(\widehat{F_f}\,\overline{z} = s\right) & &= \beta^n(x')\overline{w} \\
&\sqsubseteq \mathbf{i}_\iota(\widehat{F_f}\,\overline{z}) & &\sqsubseteq \mathbf{i}^-_\sigma(\alpha^n(x))\overline{\mathbf{i}^-(z)} \\
&= \mathbf{i}_\iota(\mathcal{H}[\![\mathcal{N}, \Gamma \vdash f]\!](\alpha^n)\overline{z}) & &= \mathbf{i}_\sigma(\mathcal{H}[\![\mathcal{N}, \Gamma \vdash x]\!](\alpha^n))\overline{\mathbf{i}^-(z)} \\
&= \mathbf{i}_\sigma(\mathcal{H}[\![\mathcal{N}, \Gamma \vdash f]\!](\alpha^n))\overline{\mathbf{i}^-(z)} & &= C \\
&= C
\end{aligned}$$

This remaining base case expression is where we start needing induction on $n \geq 0$.

Case $e = F : \sigma \in \mathcal{N}$, and $n = 0$. We rely on Lemma A.3 to prove B and C are the universal relation:

$$\begin{aligned}
B &= \mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner\Gamma\urcorner \vdash \ulcorner F\urcorner]\!](\beta^0)\overline{w} \\
&= \mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner\Gamma\urcorner \vdash \lambda \overline{y}r.R_F\,\overline{y}r]\!](\beta^0)\overline{w} \\
&= \lambda s \in \mathcal{M}[\![\iota]\!].\beta^0(R_F)\overline{w}s \\
&= \top_{\mathcal{M}[\![\mathsf{Rel}^+(\iota)]\!]} \\
&= \top_{\mathcal{D}[\![\mathsf{Rel}^+(\sigma)]\!]}\overline{\mathbf{i}^-(z)} \\
&= \mathbf{i}_\sigma(\bot_{\mathcal{H}[\![\sigma]\!]})\overline{\mathbf{i}^-(z)} \\
&= \mathbf{i}_\sigma(\mathcal{H}[\![\mathcal{N}, \Gamma \vdash F]\!](\alpha^0))\overline{\mathbf{i}^-(z)} \\
&= C
\end{aligned}$$

This covers $n = 0$ for base case expressions. Recall our proof follows the structure of Lemma A.6, which is outlined in Figure 3. We distinguish three induction hypotheses, where $S(n,e)$ denotes that the claim holds for n and expression e. Thus, we have now proved $S(0,b)$ for all base case expressions b.

In this inductive case, we consider expressions $e = \$e_1 \dots e_\ell : \sigma_1 \to \cdots \to \sigma_m \to \iota$ for some $\ell > 0$. For this, we introduce some more shorthands:

$$\Delta'_{\mathcal{G}} := \Delta_{\mathcal{G}}, \ulcorner \Gamma \urcorner, \bar{y}, r \qquad \Delta''_{\mathcal{G}} := \Delta_{\mathcal{G}}, \ulcorner \Gamma \urcorner, \bar{y}, r, \bar{r} \qquad \beta^{n,\bar{w},s} := \beta^n[\bar{y} \mapsto \bar{w}, r \mapsto s]$$

Note that the sort τ of \$ is of the form

$$\tau = \tau_1 \to \cdots \to \tau_\ell \to \sigma_1 \to \cdots \to \sigma_m \to \iota$$

where $e_1 : \tau_1, \dots, e_\ell : \tau_\ell$, for some $\ell > 0$. Sometimes we abbreviate $\sigma_1 \to \cdots \to \sigma_m \to \iota$ to σ.

In the sequel, steps marked with † use IH1 for $n = 0$, and IH3 for $n > 0$.

Because B is A from Lemma A.6 with $\bar{w}$ substituted for (their) $\bar{z}$, we simply rewrite B like A is rewritten there; for all $n \geq 0$ and expressions $e = \$\bar{e}$, we can rewrite B to obtain:

$$B = \lambda s. \max \Bigg\{$$
$$\min\{$$
$$\mathcal{M}[\![\Delta''_{\mathcal{G}} \vdash \$']\!](\beta^{n,\bar{w},s}[\bar{r} \mapsto \bar{r'}])(\mathcal{M}[\![\Delta''_{\mathcal{G}} \vdash \ulcorner(e_1,r_1)\urcorner]\!](\beta^{n,\bar{w},s}[\bar{r} \mapsto \bar{r'}]))$$
$$\dots (\mathcal{M}[\![\Delta''_{\mathcal{G}} \vdash \ulcorner(e_\ell,r_\ell)\urcorner]\!](\beta^{n,\bar{w},s}[\bar{r} \mapsto \bar{r'}])) \bar{w} s,$$
$$\min\{\mathcal{M}[\![\Delta''_{\mathcal{G}} \vdash Prop(e_i,r_i)]\!](\beta^{n,\bar{w},s}[\bar{r} \mapsto \bar{r'}]) \mid i \in [\ell]\}$$
$$\}$$
$$\mid \forall i \in [\ell]. r'_i \in \mathcal{M}[\![\mathsf{Rel}^-(\tau_i)]\!]\Bigg\}$$

We now distinguish two cases for each subexpression $e_i : \tau_i$, namely $\tau_i = \iota$ and $\tau_i = \tau'_1 \to \tau'_2$.

If e_i is of sort ι, then the following holds:

$$\mathcal{M}[\![\Delta''_{\mathcal{G}} \vdash Prop(e_i : \iota, r_i)]\!](\beta^{n,\bar{w},s}[\bar{r} \mapsto \bar{r'}]) = \mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner \Gamma \urcorner \vdash \ulcorner e_i \urcorner]\!](\beta^n) r'_i$$
$$\mathcal{M}[\![\Delta''_{\mathcal{G}} \vdash \ulcorner(e_i : \iota, r_i)\urcorner]\!](\beta^{n,\bar{w},s}[\bar{r} \mapsto \bar{r'}]) = r'_i$$

We know from † that $\mathsf{Incl}_{\mathsf{Rel}^+(\iota)}(\mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner \Gamma \urcorner \vdash \ulcorner e_i \urcorner]\!](\beta^n), \mathbf{i}_\iota(\mathcal{H}[\![\mathcal{N}, \Gamma \vdash e_i]\!](\alpha^n)))$.

Otherwise, in case $e_i : \tau_i = \tau'_1 \to \tau'_2$, the following holds:

$$\mathcal{M}[\![\Delta''_{\mathcal{G}} \vdash Prop(e_i : \tau_i, r_i)]\!](\beta^{n,\bar{w},s}[\bar{r} \mapsto \bar{r'}]) = 1$$
$$\mathcal{M}[\![\Delta''_{\mathcal{G}} \vdash \ulcorner(e_i : \tau_i, r_i)\urcorner]\!](\beta^{n,\bar{w},s}[\bar{r} \mapsto \bar{r'}]) = \mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner \Gamma \urcorner \vdash \ulcorner e_i \urcorner]\!](\beta^n)$$

We know from † that $\mathsf{Incl}_{\mathsf{Rel}^+(\tau_i)}(\mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner \Gamma \urcorner \vdash \ulcorner e_i \urcorner]\!](\beta^n), \mathbf{i}_{\tau_i}(\mathcal{H}[\![\mathcal{N}, \Gamma \vdash e_i]\!](\alpha^n)))$.

As "semantic equivalents" of the above terms, let us write

$$P_i : o := \begin{cases} \mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner \Gamma \urcorner \vdash \ulcorner e_i \urcorner]\!](\beta^n) r'_i & \text{if } \tau_i = \iota \\ 1 & \text{if } \tau_i = \tau'_1 \to \tau'_2 \end{cases}$$

and

$$T_i : \mathsf{Rel}^-(\tau_i) := \begin{cases} r'_i & \text{if } \tau_i = \iota \\ \mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner \Gamma \urcorner \vdash \ulcorner e_i \urcorner]\!](\beta^n) & \text{if } \tau_i = \tau'_1 \to \tau'_2 \end{cases}$$

for all $i \in [\ell]$. Additionally, we define

$$P_i' : o := \begin{cases} \mathbf{i}_\iota(\mathcal{H}[\![\mathcal{N}, \Gamma \vdash e_i]\!](\alpha^n))\, r_i' & \text{if } \tau_i = \iota \\ 1 & \text{if } \tau_i = \tau_1' \to \tau_2' \end{cases}$$

and

$$T_i' : \mathsf{Rel}^-(\tau_i) := \begin{cases} r_i' & \text{if } \tau_i = \iota \\ \mathbf{i}_{\tau_i}(\mathcal{H}[\![\mathcal{N}, \Gamma \vdash e_i]\!](\alpha^n)) & \text{if } \tau_i = \tau_1' \to \tau_2' \end{cases}$$

for all $i \in [\ell]$, to be used after applying the induction hypothesis †. And finally, for all $i \in [\ell]$,

$$S_i : \mathsf{Rel}^-(\tau_i) := \begin{cases} r_i' & \text{if } \tau_i = \iota \\ \mathcal{H}[\![\mathcal{N}, \Gamma \vdash e_i]\!](\alpha^n) & \text{if } \tau_i = \tau_1' \to \tau_2' \end{cases}$$

We derive by abuse of notation, using the above:

$$B = \lambda s. \exists \overline{r'}. \left(\mathcal{M}[\![\Delta_\mathcal{G}, \ulcorner \Gamma \urcorner \vdash \$']\!](\beta^n)\, \overline{T}\, \overline{w}\, s \wedge \bigwedge_{i \in [\ell]} P_i \right)$$

We continue by case analysis on $\$$.

Case $e = f\, e_1 \ldots e_\ell$ with $f \in \Sigma$. For all $n \geq 0$:

$$B = \lambda s. \exists \overline{r'}. \left(\mathcal{M}[\![\Delta_\mathcal{G}, \ulcorner \Gamma \urcorner \vdash D_f]\!](\beta^n)\, \overline{r'}\, \overline{w}\, s \wedge \bigwedge_{i \in [\ell]} \mathcal{M}[\![\Delta_\mathcal{G}, \ulcorner \Gamma \urcorner \vdash \ulcorner e_i \urcorner]\!](\beta^n)\, r_i' \right)$$

$$= \lambda s. \exists \overline{r'}. \left(\widehat{F}_f\, \overline{r'}\, \overline{w} = s \wedge \bigwedge_{i \in [\ell]} \mathcal{M}[\![\Delta_\mathcal{G}, \ulcorner \Gamma \urcorner \vdash \ulcorner e_i \urcorner]\!](\beta^n)\, r_i' \right)$$

$$\sqsubseteq \lambda s. \exists \overline{r'}. \left(\widehat{F}_f\, \overline{r'}\, \overline{w} = s \wedge \bigwedge_{i \in [\ell]} \mathbf{i}_\iota(\mathcal{H}[\![\mathcal{N}, \Gamma \vdash e_i]\!](\alpha^n))\, r_i' \right) \qquad \dagger$$

$$= \lambda s. \exists \overline{r'}. \left(\widehat{F}_f\, \overline{r'}\, \overline{w} = s \wedge \bigwedge_{i \in [\ell]} \left(\mathcal{H}[\![\mathcal{N}, \Gamma \vdash e_i]\!](\alpha^n) \sqsubseteq r_i' \right) \right)$$

$$= \lambda s. \exists \overline{r'}. \left(\widehat{F}_f\, \overline{r'}\, \overline{z} = s \wedge \bigwedge_{i \in [\ell]} \left(\mathcal{H}[\![\mathcal{N}, \Gamma \vdash e_i]\!](\alpha^n) \sqsubseteq r_i' \right) \right)$$

$$\sqsubseteq \lambda s. \left(\widehat{F}_f\, \mathcal{H}[\![\mathcal{N}, \Gamma \vdash e_1]\!](\alpha^n) \ldots \mathcal{H}[\![\mathcal{N}, \Gamma \vdash e_\ell]\!](\alpha^n)\, \overline{z} \sqsubseteq s \right) \qquad \text{Lem A.9}$$

$$= \mathbf{i}_\iota(\widehat{F}_f\, \mathcal{H}[\![\mathcal{N}, \Gamma \vdash e_1]\!](\alpha^n) \ldots \mathcal{H}[\![\mathcal{N}, \Gamma \vdash e_\ell]\!](\alpha^n)\, \overline{z})$$

$$= \mathbf{i}_\sigma(\mathcal{H}[\![\mathcal{N}, \Gamma \vdash f]\!](\alpha^n))\, \mathbf{i}_\iota^-(\mathcal{H}[\![\mathcal{N}, \Gamma \vdash e_1]\!](\alpha^n)) \ldots \mathbf{i}_\iota^-(\mathcal{H}[\![\mathcal{N}, \Gamma \vdash e_\ell]\!](\alpha^n))\, \overline{z}$$

$$= C$$

Case $e = x e_1 \ldots e_\ell$ with $x \in V_{\mathsf{RS}}$ of sort $\tau = \tau_1 \to \tau_2$. For all $n \geq 0$:

$$B = \lambda s. \exists \overline{r'}. \left(\mathcal{M}[\![\Delta_\mathcal{G}, \ulcorner \Gamma \urcorner \vdash x']\!](\beta^n) \overline{T} \, \overline{w} s \wedge \bigwedge_{i \in [\ell]} P_i \right)$$

$$= \lambda s. \exists \overline{r'}. \left(\beta^n(x') \overline{T} \, \overline{w} s \wedge \bigwedge_{i \in [\ell]} P_i \right)$$

$$\sqsubseteq \lambda s. \exists \overline{r'}. \left(\mathbf{i}_\tau(\alpha^n(x)) \overline{T'} \, \overline{\mathbf{i}^-(z)} s \wedge \bigwedge_{i \in [\ell]} P_i \right) \qquad \dagger$$

$$\sqsubseteq \lambda s. \exists \overline{r'}. \left(\mathbf{i}_\tau(\alpha^n(x)) \overline{T'} \, \overline{\mathbf{i}^-(z)} s \wedge \bigwedge_{i \in [\ell]} P_i' \right) \qquad \dagger$$

$$= \lambda s. \exists \overline{r'}. \left(\mathbf{i}_\iota(\alpha^n(x) \overline{S} \, \overline{z}) s \wedge \bigwedge_{i \in [\ell]} P_i' \right) \qquad \text{Lem A.3}$$

$$= \lambda s. \exists \overline{r'}. \left(\mathcal{H}[\![\mathcal{N}, \Gamma \vdash x]\!](\alpha^n) \overline{S} \, \overline{z} \sqsubseteq s \wedge \bigwedge_{i \in [\ell]} P_i' \right)$$

$$\sqsubseteq \lambda s. (\mathcal{H}[\![\mathcal{N}, \Gamma \vdash x\overline{e}]\!](\alpha^n) \overline{z} \sqsubseteq s) \qquad \text{Lem A.9}$$

$$= \mathbf{i}_\iota(\mathcal{H}[\![\mathcal{N}, \Gamma \vdash x\overline{e}]\!](\alpha^n) \overline{z})$$

$$= \mathbf{i}_\tau(\mathcal{H}[\![\mathcal{N}, \Gamma \vdash x\overline{e}]\!](\alpha^n)) \overline{\mathbf{i}^-(z)} \qquad \text{Lem A.3}$$

$$= C$$

Recall that $\mathsf{Incl}_{\mathsf{Rel}^-(\tau)}(\beta^n(x'), \mathbf{i}_\tau^-(\alpha^n(x)))$. The IH $\dagger$ gives us $\mathsf{Incl}_{\mathsf{Rel}^-(\tau_i)}(T_i, T_i')$. Because we also have $\mathsf{Incl}_{\mathsf{Rel}^-(\sigma_j)}(w_j, \mathbf{i}_{\sigma_j}^-(z_j))$, we derive the first inclusion.

Case $e = F e_1 \ldots e_\ell$ with $F : \sigma \in \mathcal{N}$, and $n = 0$.

$$B = \lambda s. \exists \overline{r'}. \left(\mathcal{M}[\![\Delta_\mathcal{G}, \ulcorner \Gamma \urcorner \vdash R_F]\!](\beta^n) \overline{T} \, \overline{w} s \wedge \bigwedge_{i \in [\ell]} P_i \right)$$

$$\sqsubseteq \lambda s. 1$$

$$= \mathbf{i}_\iota(\bot) \qquad \text{Lem A.3}$$

$$= \mathbf{i}_\iota(\bot_{\mathcal{H}[\![\sigma]\!]} \mathcal{H}[\![\mathcal{N}, \Gamma \vdash e_1]\!](\alpha^0) \ldots \mathcal{H}[\![\mathcal{N}, \Gamma \vdash e_\ell]\!](\alpha^0) \overline{z})$$

$$= \mathbf{i}_\iota(\alpha^0(F) \mathcal{H}[\![\mathcal{N}, \Gamma \vdash e_1]\!](\alpha^0) \ldots \mathcal{H}[\![\mathcal{N}, \Gamma \vdash e_\ell]\!](\alpha^0) \overline{z})$$

$$= \mathbf{i}_\iota(\mathcal{H}[\![\mathcal{N}, \Gamma \vdash F]\!](\alpha^0) \mathcal{H}[\![\mathcal{N}, \Gamma \vdash e_1]\!](\alpha^0) \ldots \mathcal{H}[\![\mathcal{N}, \Gamma \vdash e_\ell]\!](\alpha^0) \overline{z})$$

$$= \mathbf{i}_\sigma(\mathcal{H}[\![\mathcal{N}, \Gamma \vdash F]\!](\alpha^0)) \mathbf{i}_{\tau_1}^-(\mathcal{H}[\![\mathcal{N}, \Gamma \vdash e_1]\!](\alpha^0)) \ldots \mathbf{i}_{\tau_\ell}^-(\mathcal{H}[\![\mathcal{N}, \Gamma \vdash e_\ell]\!](\alpha^0)) \overline{\mathbf{i}^-(z)}$$

$$= C$$

We have now established that $S(0, e'')$ holds for expressions all e''. The following case is the last remaining case to prove that $S(n+1, b)$ for all base case expressions b:

Case $e = F : \sigma \in \mathcal{N}$, for $n + 1$.

$$\begin{aligned}
B &= \mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner\Gamma\urcorner \vdash \ulcorner F\urcorner]\!](\beta^{n+1})\,\overline{w} \\
&= \mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner\Gamma\urcorner \vdash \lambda\overline{y}r.\,R_F\,\overline{y}\,r]\!](\beta^{n+1})\,\overline{w} \\
&= \lambda s.\,\beta^{n+1}(R_F)\,\overline{w}\,s \\
&= \lambda s.\,\mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner\Gamma\urcorner \vdash \ulcorner\mathcal{R}(F)\urcorner]\!](\beta^n)\,\overline{w}\,s \\
&\sqsubseteq \lambda s.\,\mathbf{i}_\sigma(\mathcal{H}[\![\mathcal{N}, \Gamma \vdash \mathcal{R}(F)]\!](\alpha^n))\,\overline{\mathbf{i}^-(z)}\,s \quad \text{IH2} \\
&= \lambda s.\,\mathbf{i}_\sigma(\alpha^{n+1}(F))\,\overline{\mathbf{i}^-(z)}\,s \\
&= \mathbf{i}_\sigma(\mathcal{H}[\![\mathcal{N}, \Gamma \vdash F]\!](\alpha^{n+1}))\,\overline{\mathbf{i}^-(z)} \\
&= C
\end{aligned}$$

Finally, we present the remaining case to prove that $S(n', e'')$ for all $n' \geq 0$ and all expressions e''.

Case $e = F\,e_1 \ldots e_\ell$ with $F \in \mathcal{N}$ and $\ell > 0$, for $n + 1$.

$$\begin{aligned}
B &= \lambda s.\,\exists\overline{r'}.\left(\mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner\Gamma\urcorner \vdash R_F]\!](\beta^{n+1})\,\overline{T}\,\overline{w}\,s \wedge \bigwedge_{i\in[\ell]} P_i\right) \\
&= \lambda s.\,\exists\overline{r'}.\left(\beta^{n+1}(R_F)\,\overline{T}\,\overline{w}\,s \wedge \bigwedge_{i\in[\ell]} P_i\right) \\
&= \lambda s.\,\exists\overline{r'}.\left(\mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner\Gamma\urcorner \vdash \ulcorner\mathcal{R}(F)\urcorner]\!](\beta^n)\,\overline{T}\,\overline{w}\,s \wedge \bigwedge_{i\in[\ell]} P_i\right) \\
&\sqsubseteq \lambda s.\,\exists\overline{r'}.\left(\mathbf{i}_\tau(\mathcal{H}[\![\mathcal{N}, \Gamma \vdash \mathcal{R}(F)]\!](\alpha^n))\,\overline{T'}\,\overline{\mathbf{i}^-(z)}\,s \wedge \bigwedge_{i\in[\ell]} P_i\right) \quad \text{IH2, †} \\
&\sqsubseteq \lambda s.\,\exists\overline{r'}.\left(\mathbf{i}_\tau(\mathcal{H}[\![\mathcal{N}, \Gamma \vdash \mathcal{R}(F)]\!](\alpha^n))\,\overline{T'}\,\overline{\mathbf{i}^-(z)}\,s \wedge \bigwedge_{i\in[\ell]} P_i'\right) \quad \text{†} \\
&= \lambda s.\,\exists\overline{r'}.\left(\mathbf{i}_\iota(\mathcal{H}[\![\mathcal{N}, \Gamma \vdash \mathcal{R}(F)]\!](\alpha^n)\,\overline{S}\,\overline{z})\,s \wedge \bigwedge_{i\in[\ell]} P_i'\right) \quad \text{Lem A.3} \\
&= \lambda s.\,\exists\overline{r'}.\left(\alpha^{n+1}(F)\,\overline{S}\,\overline{z} \sqsubseteq s \wedge \bigwedge_{i\in[\ell]} P_i'\right) \\
&= \lambda s.\,\exists\overline{r'}.\left(\mathcal{H}[\![\mathcal{N}, \Gamma \vdash F]\!](\alpha^{n+1})\,\overline{S}\,\overline{z} \sqsubseteq s \wedge \bigwedge_{i\in[\ell]} P_i'\right) \\
&\sqsubseteq \mathbf{i}_\iota(\mathcal{H}[\![\mathcal{N}, \Gamma \vdash F\,\overline{e}]\!](\alpha^{n+1})\,\overline{z}) \quad \text{Lem A.9} \\
&= \mathbf{i}_\tau(\mathcal{H}[\![\mathcal{N}, \Gamma \vdash F\,\overline{e}]\!](\alpha^{n+1}))\,\overline{\mathbf{i}^-(z)} \quad \text{Lem A.3} \\
&= C
\end{aligned}$$

$\square$

Corollary 3.5 (Inclusion). *For all $n \geq 0$, $\mathcal{M}[\![\Delta_{\mathcal{G}} \vdash \ulcorner S \urcorner]\!](\beta^n) \sqsubseteq \mathbf{i}_\iota(\mathcal{H}[\![\mathcal{N} \vdash S]\!](\alpha^n))$.*

A.4 Main result: equality

Lemma 3.3. *For all typing judgements $\mathcal{N} \vdash e : \sigma$ of the HoRS $\mathcal{G}$, and non-increasing chains of valuations $\mathcal{I} \subseteq \mathcal{M}[\![\Delta_{\mathcal{G}}]\!]$,*

$$\mathcal{M}[\![\Delta_{\mathcal{G}} \vdash \ulcorner e : \sigma \urcorner]\!]\left(\bigsqcap \mathcal{I}\right) = \bigsqcap_{I \in \mathcal{I}} \mathcal{M}[\![\Delta_{\mathcal{G}} \vdash \ulcorner e : \sigma \urcorner]\!](I).$$

Proof. Recall that $\mathcal{M}[\![\Delta_{\mathcal{G}}]\!]$ and $\mathcal{M}[\![\rho]\!]$ are complete lattices for each relational sort environment $\Delta_{\mathcal{G}}$ and relational sort ρ. Thus, we know that the greatest lower bounds exist.

To perform induction on the structure of $e : \sigma$, we strengthen the claim to the following.

For all typing judgements $\mathcal{N}, \Gamma \vdash e : \sigma$ of the HoRS $\mathcal{G}$ where $\Gamma = \{x_1 : \tau_1, \ldots, x_k : \tau_k\}$, for all descending chains of valuations $\mathcal{I} \subseteq \mathcal{M}[\![\Delta_{\mathcal{G}}]\!]$, and valuations $\theta \in \mathcal{M}[\![\ulcorner \Gamma \urcorner]\!]$,

$$\mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner \Gamma \urcorner \vdash \ulcorner e : \sigma \urcorner]\!]\left(\left(\bigsqcap \mathcal{I}\right)[\overline{x'} \mapsto \overline{\theta(x')}]\right)$$
$$= \bigsqcap_{I \in \mathcal{I}} \mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner \Gamma \urcorner \vdash \ulcorner e : \sigma \urcorner]\!](I[\overline{x'} \mapsto \overline{\theta(x')}]).$$

where $\ulcorner \Gamma \urcorner = \{x'_1 : \mathsf{Rel}^-(\tau_1), \ldots, x'_k : \mathsf{Rel}^-(\tau_k)\}$. We abbreviate $I[\overline{x'} \mapsto \overline{\theta(x')}]$ to I'. Note that

$$\left(\bigsqcap \mathcal{I}\right)[\overline{x'} \mapsto \overline{\theta(x')}] = \bigsqcap_{I \in \mathcal{I}} I[\overline{x'} \mapsto \overline{\theta(x')}] = \bigsqcap_{I \in \mathcal{I}} I',$$

so that we shorten the above equation to

$$\mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner \Gamma \urcorner \vdash \ulcorner e : \sigma \urcorner]\!]\left(\bigsqcap_{I \in \mathcal{I}} I'\right) = \bigsqcap_{I \in \mathcal{I}} \mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner \Gamma \urcorner \vdash \ulcorner e : \sigma \urcorner]\!](I').$$

Case $e = f : \iota^k \to \iota \in \Sigma$. The meaning of $\ulcorner f \urcorner$ is independent of the valuation, as demonstrated by:

$$\ulcorner f \urcorner = \lambda y_1 \ldots y_k\, r.\, (f\, y_1 \ldots y_k = r)$$

Case $x : \sigma \in V_{\mathrm{RS}}$. The meaning of $\ulcorner x \urcorner$ relies only on the θ part of the valuation, as evident from:

$$\ulcorner x : \sigma \urcorner = \begin{cases} \lambda r.\, (x' = r) & \text{if } \sigma = \iota \\ \lambda y_1 \ldots y_k\, r.\, x'\, y_1 \ldots y_k\, r & \text{if } \sigma = \sigma_1 \to \cdots \to \sigma_k \to \iota \text{ for } k > 0 \end{cases}$$

Case $F\sigma_1 \to \cdots \to \sigma_k \to \iota \in \mathcal{N}$.

$$\mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner\Gamma\urcorner \vdash \ulcorner F\urcorner]\!]\left(\prod_{I\in\mathcal{I}}I'\right) = \mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner\Gamma\urcorner \vdash \lambda y_1 \ldots y_k\, r. R_F\, y_1 \ldots y_k\, r]\!]\left(\prod_{I\in\mathcal{I}}I'\right)$$

$$= \lambda y_1 \ldots y_k\, r. \left(\prod_{I\in\mathcal{I}}I'\right)(R_F)\, y_1 \ldots y_k\, r$$

$$= \lambda y_1 \ldots y_k\, r. \left(\prod_{I\in\mathcal{I}}I\right)(R_F)\, y_1 \ldots y_k\, r$$

$$= \lambda y_1 \ldots y_k\, r. \left(\prod_{I\in\mathcal{I}}I(R_F)\right) y_1 \ldots y_k\, r$$

$$= \prod_{I\in\mathcal{I}}I(R_F)$$

$$= \prod_{I\in\mathcal{I}}I'(R_F)$$

$$= \prod_{I\in\mathcal{I}}\mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner\Gamma\urcorner \vdash R_F]\!](I')$$

$$= \prod_{I\in\mathcal{I}}\mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner\Gamma\urcorner \vdash \lambda y_1 \ldots y_k\, r. R_F\, y_1 \ldots y_k\, r]\!](I')$$

$$= \prod_{I\in\mathcal{I}}\mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner\Gamma\urcorner \vdash \ulcorner F\urcorner]\!](I')$$

For the fourth equality, we rely on the codomain of $\mathcal{I}$ being a complete lattice (namely, a finite product of complete lattices $\mathcal{M}[\![\rho]\!]$).

   Case $e = \$\overline{e}$ *with* $\overline{e} = e_1 \ldots e_\ell$ *for* $\ell > 0$. This case follows from applying the induction hypothesis in a straightforward though laborious unfolding of the relational lift and the semantics. Recall that:

$$\mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner\Gamma\urcorner \vdash \ulcorner\$\overline{e}\urcorner]\!]\left(\prod_{I\in\mathcal{I}}I'\right)$$

$$= \mathcal{M}[\![\Delta_{\mathcal{G}}, \ulcorner\Gamma\urcorner \vdash \lambda\overline{y}\, r. \exists\overline{r}. \$'^{\ulcorner\top}(e_1,r_1)^{\urcorner\top} \ldots ^{\ulcorner\top}(e_\ell,r_\ell)^{\urcorner\top}\overline{y}\, r \wedge \bigwedge_{i\in[\ell]} Prop(e_i,r_i)]\!]\left(\prod_{I\in\mathcal{I}}I'\right)$$

The previous cases show that the greatest lower bound is preserved by $\mathcal{M}[\![\$']\!]$. Observe that $^{\ulcorner\top}(e_i,r_i)^{\urcorner\top}$ is r_i or $\ulcorner e_i \urcorner$. Either way, the greatest lower bound is preserved by $\mathcal{M}[\![^{\ulcorner\top}(e_i,r_i)^{\urcorner\top}]\!]$. Similarly, $Prop(e_i,r_i)$ is either $\ulcorner e_i \urcorner r_i$ or true, and the greatest lower bound is thus preserved by $\mathcal{M}[\![Prop(e_i,r_i)]\!]$. This concludes the proof. $\qquad\square$

Theorem 3.6 (Correctness). $\mathcal{M}[\![\Delta_{\mathcal{G}} \vdash R_S]\!](\mathsf{gfp}(T^{\mathcal{M}}_{P_{\mathcal{G}}:\Delta_{\mathcal{G}}}))\, t = 1$ *if and only if* $t = [\![\mathcal{G}]\!]$.