Proof Systems for Retracts in Simply Typed Lambda Calculus

Colin Stirling

School of Informatics
Informatics Forum
University of Edinburgh
cps@inf.ed.ac.uk

Abstract. This paper concerns retracts in simply typed lambda calculus assuming $\beta\eta$-equality. We provide a simple tableau proof system which characterises when a type is a retract of another type and which leads to an exponential decision procedure.

1 Introduction

Type ρ is a retract of type τ if there are functions $C : \rho \to \tau$ and $D : \tau \to \rho$ with $D \circ C = \lambda x.x$. This paper concerns retracts in the case of simply typed lambda calculus [1]. Various questions can be asked. The decision problem is: given ρ and τ, is ρ a retract of τ? Is there an independent characterisation of when ρ is a retract of τ? Is there an inductive method, such as a proof system, for deriving assertions of the form "ρ is a retract of τ"? If so, can one also construct (inductively) the witness functions C and D?

Bruce and Longo [2] provide a simple proof system that solves when there are retracts in the case that $D \circ C =_\beta \lambda x.x$. The problem is considerably more difficult if β-equality is replaced with $\beta\eta$-equality. De Liguoro, Piperno and Statman [3] show that the retract relation with respect to $\beta\eta$-equality coincides with the surjection relation: ρ is a retract of τ iff for any model there is a surjection from τ to ρ. They also provide a proof system for the affine case (when each variable in C and D occurs at most once) assuming a single ground type. Regnier and Urzyczyn [9] extend this proof system to cover multiple ground types. The proof systems yield simple inductive nondeterministic algorithms belonging to NP for deciding whether ρ is an affine retract of τ. Schubert [10] shows that the problem of affine retraction is NP-complete and how to derive witnesses C and D from the proof system in [9]. Under the assumption of a single ground type, decidability of when ρ is a retract of τ is shown by Padovani [8] by explicit witness construction (rather than by a proof system) of a special form.

More generally, decidability of the retract problem follows from decidability of higher-order matching in simply typed lambda calculus [13]: ρ is a retract of τ iff the equation $\lambda z^\rho.x_1^{\tau\to\rho}(x_2^{\rho\to\tau}z) =_{\beta\eta} \lambda z^\rho.z$ has a solution (the witnesses D and C for x_1, x_2). Since the complexity of matching is non-elementary [15] this decidability result leaves open whether there is a better algorithm, or even a proof system, for the problem. In the case of β-equality matching is no guide to solvability: the retract problem is simply solvable whereas β-matching is undecidable [4].

In this paper we provide an independent solution to the retract problem. We show it is decidable by exhibiting sound and complete tableau proof systems. We develop two proof systems for retracts, one for the (slightly easier) case when there is a single ground type and the other for when there are multiple ground types. Both proof systems appeal to paths in terms. Their correctness depend on properties of such paths. We appeal to a dialogue game between witnesses of a retract to prove such properties: a similar game-theoretic characterisation of β-reduction underlies decidability of matching.

In Section 2 we introduce retracts in simply typed lambda calculus and fix some notation for terms as trees and for their paths. The two tableau proof systems for retracts are presented in Section 3 where we also briefly examine how they generate a decision procedure for the retract problem. In Section 4 we prove soundness and completeness of the proof systems. To do this, we first define the dialogue game between witnesses of a retract. We then use the game to isolate key technical properties of paths and subtrees of witnesses which are then used in the correctness proof.

2 Preliminaries

Simple types are generated from ground types using the binary function operator $\rightarrow$. We let $a, b, o, \ldots$ range over ground types and $\rho, \sigma, \tau, \ldots$ range over simple types. Assuming $\rightarrow$ associates to the right, so $\rho \rightarrow \sigma \rightarrow \tau$ is $\rho \rightarrow (\sigma \rightarrow \tau)$, if a type ρ is not a ground type then it has the form $\rho_1 \rightarrow \ldots \rightarrow \rho_n \rightarrow a$. We say that a is the *target* type of a and of any type $\rho_1 \rightarrow \ldots \rightarrow \rho_n \rightarrow a$.

Simply typed terms in Church style are generated from a countable set of typed variables x^σ using lambda abstraction and function application [1]. We write S^σ, or sometimes $S : \sigma$, to mean term S has type σ. The usual typing rules hold: if S^τ then $\lambda x^\sigma.S^\tau : \sigma \rightarrow \tau$; if $S^{\sigma\rightarrow\tau}$ and U^σ then $(S^{\sigma\rightarrow\tau}U^\sigma) : \tau$. In a sequence of unparenthesised applications we assume that application associates to the left, so $SU_1 \ldots U_k$ is $((\ldots(SU_1)\ldots)U_k)$. Another abbreviation is $\lambda z_1 \ldots z_m$ for $\lambda z_1 \ldots \lambda z_m$. Usual definitions of when a variable occurrence is free or bound and when a term is closed are assumed.

We also assume the usual dynamics of β and η-reductions and the consequent $\beta\eta$-equivalence between terms (as well as α-equivalence). Confluence and strong normalisation ensure that terms reduce to (unique) normal forms. Moreover, we assume the standard notion of η-long β-normal form (a term in normal form which is not an η-reduct of some other term) which we abbreviate to lnf. The syntax of such terms reflects their type: a lnf of type a is a variable x^a, or $x\,U_1 \ldots U_k$ where $x^{\rho_1\rightarrow\ldots\rightarrow\rho_k\rightarrow a}$ and each $U_i^{\rho_i}$ is a lnf; a lnf of type $\rho_1 \rightarrow \ldots \rightarrow \rho_n \rightarrow a$ has the form $\lambda x_1^{\rho_1} \ldots x_n^{\rho_n}.S$, where S^a is a lnf.

The following definition introduces retracts between types [2, 3].

Definition 1. *Type ρ is a* retract *of type τ, written $\models \rho \trianglelefteq \tau$, if there are terms $C : \rho \to \tau$ and $D : \tau \to \rho$ such that $D \circ C =_{\beta\eta} \lambda x^\rho.x$.*

The witnesses C and D to a retract can always be presented as lnfs. We can think of C as a "coder" and D as a "decoder" [9]. Assume $\rho = \rho_1 \to \ldots \to \rho_l \to a$ and $\tau = \tau_1 \to \ldots \to \tau_n \to a$: in a retract the types must share target type [9]. We instantiate the bound ρ_i variables in a decoder D to $D(z_1^{\rho_1}, \ldots, z_l^{\rho_l})$, often abbreviated to $D(\overline{z})$, and the bound variable of type ρ in C to $C(x^\rho)$: so, $\models \rho \trianglelefteq \tau$ if $D(z_1^{\rho_1}, \ldots, z_l^{\rho_l})(C(x^\rho)) =_{\beta\eta} xz_1 \ldots z_l$. From [9], we can restrict a decoder to be of the form $\lambda f^\tau.fS_1^{\tau_1} \ldots S_n^{\tau_n}$ with f as head variable and a coder $C(x)$ has the form $\lambda y_1^{\tau_1} \ldots y_n^{\tau_n}.H(xT_1^{\rho_1} \ldots T_l^{\rho_l})$.

Definition 2. *We say that the decoder $D(z_1, \ldots, z_l) = \lambda f^\tau.fS_1^{\tau_1} \ldots S_n^{\tau_n}$ and the coder $C(x) = \lambda y_1^{\tau_1} \ldots y_n^{\tau_n}.H(xT_1^{\rho_1} \ldots T_l^{\rho_l})$ are* canonical *witnesses for $\rho \trianglelefteq \tau$ if $D(\overline{z})(C(x)) =_{\beta\eta} xz_1 \ldots z_l$ and they obey the following properties:*

1. *variables $f, z_1, \ldots, z_l$ occur only once in $D(\overline{z})$,*
2. *x occurs only once in $C(x)$,*
3. *H is ε if ρ and τ are constructed from a single ground type,*
4. *if $T_i^{\rho_i}$ contains an occurrence of y_j then it is the head variable of $T_i^{\rho_i}$, z_i occurs in $S_j^{\tau_j}$ and $T_i^{\rho_i}$ contains no other occurrences of any y_k, $1 \le k \le n$.*

The next result follows from observations in [3, 9].

Proposition 1. *$\models \rho \trianglelefteq \tau$ iff there exist canonical witnesses for $\rho \trianglelefteq \tau$.*

So, if there is only a single ground type then $C(x)$ can be restricted to have the form $\lambda y_1^{\tau_1} \ldots y_n^{\tau_n}.xT_1^{\rho_1} \ldots T_l^{\rho_l}$ with x as head variable [3].

Example 1. From [3]. Let $\rho = \rho_1 \to \rho_2 \to o$ where $\rho_1 = \rho_2 = \sigma \to o$ and let $\tau = \tau_1 \to o$ where $\tau_1 = \sigma \to (o \to o \to o) \to o$ and σ is arbitrary. It follows that $\models \rho \trianglelefteq \tau$. A decoder $D(z_1^{\rho_1}, z_2^{\rho_2})$ is $\lambda f^\tau.f(\lambda u^\sigma v^{o\to o\to o}.v(z_1u)(z_2u))$ and a coder $C(x^\rho)$ is $\lambda y^{\tau_1}.x(\lambda w^\sigma.yw(\lambda s^o t^o.s))(\lambda w^\sigma.yw(\lambda s^o t^o.t))$; so, $(D(z_1, z_2))C(x)$ $\to_\beta^* x(\lambda w^\sigma.z_1w)(\lambda w^\sigma.z_2w) =_{\beta\eta} xz_1z_2$. □

Example 2. From [9] with multiple ground types. Let $\rho = \rho_1 \to \rho_2 \to a$ where $\rho_1 = b \to a$, $\rho_2 = a$ and let $\tau = \tau_1 \to a$ where $\tau_1 = b \to (a \to o \to a) \to a$. A decoder is $D(z_1^{\rho_1}, z_2^{\rho_2})$ is $\lambda f^\tau.f(\lambda u_1^b u_2^{a\to o\to a}.u_2(z_1u_1)z_2)$ and a coder $C(x^\rho)$ is $\lambda y^{\tau_1}.ys^b(\lambda w_1^a w_2^o.x(\lambda v^b.yv(\lambda w_1^a w_2^o.w_1))w_2)$; so, $(D(z_1, z_2))C(x) \to_\beta^* x(\lambda v^b.z_1v)z_2$ $=_{\beta\eta} xz_1z_2$. □

Terms are represented as special kinds of tree (that we call *binding* trees in [12, 14]) with dummy lambdas and an explicit binding relation. A term of the form y^a is represented as a tree with a single node labelled y^a. In the case of $y\,U_1 \ldots U_k$, when $y^{\rho_1\to\ldots\to\rho_k\to a}$, we assume that a dummy lambda with the empty sequence of variables is placed directly above any subterm U_i in its tree representation if ρ_i is a ground type. With this understanding, the tree for $y\,U_1 \ldots U_k$ consists of a root node labelled $y^{\rho_1\to\ldots\to\rho_k\to a}$ and k-successor trees representing

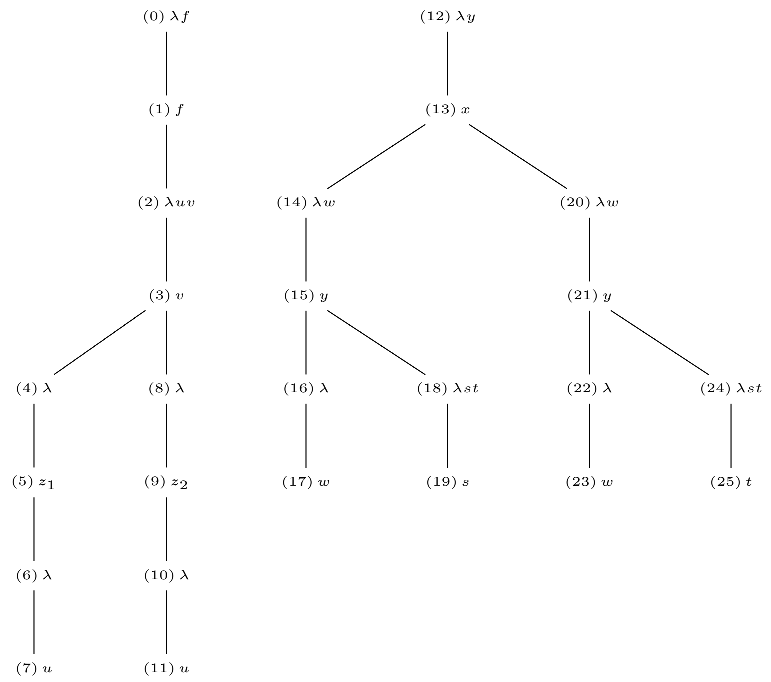

Fig. 1. $D(z_1, z_2)$ and $C(x)$ of Example 1

$U_1, \ldots, U_k$. We also use the abbreviation $\lambda\overline{y}$ for $\lambda y_1 \ldots y_m$ for $m \geq 0$, so $\overline{y}$ is possibly the empty sequence of variables in the case of a dummy lambda. The tree representation of $\lambda\overline{y}.S : \rho_1 \to \ldots \to \rho_k \to a$ consists of a root node labelled $\lambda\overline{y}$ and a single successor tree for S^a. The trees for $C(x)$ and $D(z_1, z_2)$ of Example 1, where we have omitted the types, are in Figure 1.

We say that a node is a *lambda* (*variable*) node if it is labelled with a lambda abstraction (variable). The *type* (*target type*) of a variable node is the type (target type) of the variable at that node and the *type* (*target type*) of a lambda node is the type (target type) of the subterm rooted at that node.

The other elaboration is that we assume an extra binary relation $\downarrow$ between nodes in a tree that represents *binding*; that is, between a node labelled $\lambda y_1 \ldots y_n$ and a node below it labelled y_j (that it binds). A binder $\lambda\overline{y}$ is such that either $\overline{y}$ is empty and therefore is a dummy lambda and cannot bind a variable occurrence or $\overline{y} = y_1 \ldots y_k$ and $\lambda\overline{y}$ can only then bind variable occurrences of the form y_i, $1 \leq i \leq k$. Consequently, we also employ the following abbreviation $n \downarrow_i m$ if $n \downarrow m$ and n is labelled $\lambda y_1 \ldots y_k$ and m is labelled y_i. In Figure 1 we have not included the binding relation; however, for instance, (2) $\downarrow_1$ (7).

Definition 3. *Lambda node n is a* descendant *(k-*descendant*) of m if either $m \downarrow m'$ ($m \downarrow_k m'$), n is a successor of m' for some m' and the target types of*

m, m' and n are the same, or n' is a descendant (k-descendant) of m and n is a descendant of n' for some n'.

We assume a standard presentation of nodes of a tree as sequences of integers: an initial sequence, typically ε, is the root node; if n is a node and m is the ith successor of n then $m = ni$. For the sake of brevity we have not followed this approach in Figure 1 where we have presented each node as a unique integer (i).

Definition 4. *A* path *of the tree of a term of type σ is a sequence of nodes $\overline{n} = n_1, \ldots, n_k$ where n_1 is the root of the tree, each n_{i+1} is a successor of n_i and if n_j is a variable node then for some $i < j$, $n_i \downarrow n_j$ (hence is a* closed *path).*

For paths $\overline{m} = m_1, \ldots, m_l$ and $\overline{n} = n_1, \ldots, n_k$ of type σ we write $\overline{m} \sqsubset \overline{n}$ if for some $i > 0$, for all $h \leq 2i$, $m_h = n_h$, $m_{2i+1} = m_{2i}p$, $n_{2i+1} = n_{2i}q$ and $p < q$.

A (closed) subtree *of a tree of a term of type σ is a set of paths P of type σ such that if $\overline{m}, \overline{n}$ are distinct paths in P then $\overline{m} \sqsubset \overline{n}$ or $\overline{n} \sqsubset \overline{m}$.*

A path $\overline{n} = n_1, \ldots, n_k$ is a contiguous sequence of nodes in a tree of a term starting at the root; for $i \geq 1$, each n_{2i-1} is a lambda node and each n_{2i} is a variable node (whose binder occurs earlier in the path). Path $\overline{m}$ is before $\overline{n}$, $\overline{m} \sqsubset \overline{n}$, if they have a common even length prefix and then differ as to their successors (the one in $\overline{m}$ before that in $\overline{n}$). These paths could, therefore, be in the same term: therefore, a closed subtree is a set of such paths.

Definition 5. *A path $\overline{n} = n_1, \ldots, n_l$ is* k-minimal *provided that for each binding node n_i there are at most k distinct nodes n_j, $i < j \leq l$, such that $n_i \downarrow n_j$. A subtree P is* k-minimal *if each path in P is k-minimal.*

Not every path or subtree is useful in a term. So, we define when a path or subtree is realisable meaning that their nodes are "accessible" [7] or "reachable" [6] in an applicative context.

Definition 6. *Assume $\overline{n} = n_1, \ldots, n_l$ is a path of odd length of a closed term T of type σ, m is the node below n_l in T and T' is the term T when the variable u^τ at node m is replaced with a fresh free variable z^τ. We say that $\overline{n}$ is* realisable *if there is a closed term $U = \lambda y^\sigma.yS_1 \ldots S_k$ such that $UT' =_{\beta\eta} \lambda\overline{x}.zW_1 \ldots W_q$ for some $q \geq 0$.*

Definition 7. *Assume P is a subtree of closed term T of type σ where each path has even length, $m_1, \ldots, m_q$ are the leaves of P and T_i, $1 \leq i \leq q$, is the term T when the variable $u_i^{\tau_i}$ at m_i is replaced with a fresh free variable $z_i^{\tau_i}$. We say that P is* realisable *if there is a closed term $U = \lambda y^\sigma.yS_1 \ldots S_k$ such that for each i, $UT_i =_{\beta\eta} \lambda\overline{x}.z_iW_1 \ldots W_{q_i}$ for $q_i \geq 0$.*

Next we define two useful operations on paths, restriction relative to a suffix and the subtype after a prefix.

Definition 8. *Assume that $\overline{n} = n_1, \ldots, n_p$ is a path, $\sigma = \sigma_1 \to \ldots \to \sigma_k \to a$, n_i is a lambda node of type σ and $w = n_i, \ldots, n_p$ is a suffix of $\overline{n}$.*

1. *The suffix w* admits σ_j, $1 \le j \le k$, *if either there is no n_q, $i \le q \le p$, such that $n_i \downarrow_j n_q$ or there is a j-descendant n_q of n_i whose type is $\tau_1 \to \ldots \to \tau_l \to a$ and for some r there is not a $t : q < t \le p$ such that $n_q \downarrow_r n_t$ and a is the target type of τ_r.*
2. *The* restriction of σ to w, *$\sigma \upharpoonright w$, is defined as σ_w where*
   - $a_w = a$,
   - $(\sigma_j \to \ldots \to \sigma_k \to a)_w =$ *if w admits σ_j then* $\sigma_j \to (\sigma_{j+1} \to \ldots \to \sigma_k \to a)_w$ *else* $(\sigma_{j+1} \to \ldots \to \sigma_k \to a)_w$.

Definition 9. *Assume that $\overline{n} = n_1, \ldots, n_p$ is a path of type σ. For a prefix w of $\overline{n}$ we define the* subtype of σ after w, *$w(\sigma)$:*

- *if $w = \varepsilon$ (the empty prefix) then σ,*
- *if $w = n_1, \ldots, n_q$, $q \le p$, then the type of node n_q.*

We also define a canonical presentation of a (prefix or suffix of a) path $\overline{n} = n_1, \ldots, n_k$ of type σ as a *word* w. If w is the empty prefix we write $w = \varepsilon$. Otherwise, $w = (w_1, \ldots, w_j)$, $j \le k$, where for each $i \ge 0$, $w_{2i+1} = n_{2i+1}$ and if $n_h \downarrow_m n_{2i}$ then $w_{2i} = n_h m$. Thus, we distinguish between $w = \varepsilon$ (the empty word) and $w = (\varepsilon)$ the prefix of length 1 consisting of the root node. Also, we can present a subtree as a set of words. Words will occur in our proof systems as presentations of paths. For example, $w = (\varepsilon, 1, 11, 112, 1112)$ of type τ as in Example 1 represents the path labelled $\lambda f, f, \lambda uv, v, \lambda$ of $D(z_1, z_2)$ in Figure 1 when its root is ε. To illustrate Definitions 8 and 9, for the prefix $w' = (\varepsilon, 1, 11)$ and the suffix $w'' = (11, 112, 1122)$ of w we have $w'(\tau) = \tau_1$ where $\tau_1 = \sigma \to (o \to o \to o) \to o$ as in Example 1 and $\tau_1 \upharpoonright w'' = \sigma \to o$: word w'' of type τ_1 has labelling $\lambda u^{\sigma} v^{o \to o \to o}, v, \lambda$; so, w'' admits the first component σ of τ_1 but not the second $(o \to o \to o)$. The final element of w' is the same as the first element of w''; in such a case we define their concatenation to be w.

Definition 10. *The* concatenation *of (a prefix) v and (a suffix) w, $v^\wedge w$, is: $\varepsilon^\wedge w = w$; if $v_k = w_1$ then $v_1, \ldots, v_k^\wedge w_1, \ldots, w_n = v_1, \ldots, v_k, w_2, \ldots, w_n$.*

3 Proof Systems for Retracts

We now develop goal directed tableau proof systems for showing retracts. By inverting the rules one has more classical axiomatic systems: we do it this way because it thereby provides an immediate nondeterministic decision procedure for deciding retracts. We present two such proof systems: a slightly simpler system for the restricted case when there is a single ground type and one for the general case.

3.1 Single Ground Type

Assertions in our proof system are of two kinds. First is $\rho \trianglelefteq \tau$ with meaning ρ is a retract of τ. The second has the form $[\rho_1, \ldots, \rho_k] \trianglelefteq \tau$ which is based on the "product" as defined in [3]. We follow [9] in allowing reordering of components of types since $\rho \to \sigma \to \tau$ is isomorphic to $\sigma \to \rho \to \tau$. Instead we could include explict rules for reordering (as with the axiom in [3]). Moreover, we assume that $[\rho_1, \ldots, \rho_k]$ is a multi-set and so elements can be in any order.

The proof rules are given in Figure 2. There is a single axiom I, identity,

$$I \quad \rho \trianglelefteq \rho$$

$$W \ \frac{\rho \trianglelefteq \sigma \to \tau}{\rho \trianglelefteq \tau}$$

$$C \ \frac{\delta \to \rho \trianglelefteq \sigma \to \tau}{\delta \trianglelefteq \sigma \qquad\qquad \rho \trianglelefteq \tau}$$

$$P_1 \ \frac{\rho_1 \to \ldots \to \rho_k \to \rho \trianglelefteq \sigma \to \tau}{[\rho_1, \ldots, \rho_k] \trianglelefteq \sigma \qquad\qquad \rho \trianglelefteq \tau}$$

$$P_2 \ \frac{[\rho_1, \ldots, \rho_k] \trianglelefteq \sigma}{\rho_1 \trianglelefteq \sigma \upharpoonright w_1 \quad \ldots \quad \rho_k \trianglelefteq \sigma \upharpoonright w_k} \text{ where}$$

– $w_1 \sqsubset \ldots \sqsubset w_k$ are k-minimal realisable paths of odd length of type σ

Fig. 2. Goal directed proof rules

a weakening rule W, a covariance rule C, and two product rules P_1 and P_2. The rules are goal directed: for instance, C allows one to decompose the goal $\delta \to \rho \trianglelefteq \sigma \to \tau$ into the two subgoals $\delta \trianglelefteq \sigma$ and $\rho \trianglelefteq \tau$. I, W and C (or their variants) occur in the proof systems for affine retracts (when variables in witnesses can only occur at most once) [3, 9]. The new rules are the product rules: P_2 appeals to k-minimal realisable paths (presented as words), and the restriction operator of Definition 8. The proof system does not require the axiom A4 of [3], $\sigma \trianglelefteq (\sigma \to a) \to a$: all instances are provable using W and C.

Definition 11. *A* successful *proof tree for $\rho \trianglelefteq \tau$ is a finite tree whose root is labelled with the goal $\rho \trianglelefteq \tau$, the successor nodes of a node are the result of an application of one of the rules to it, and each leaf is labelled with an axiom. We write $\vdash \rho \trianglelefteq \tau$ if there is a successful proof tree for $\rho \trianglelefteq \tau$.*

For some intuition about the product rules assume $\rho = \rho_1 \to \ldots \to \rho_l \to a$ and $\tau = \tau_1 \to \ldots \to \tau_n \to a$. Now, $\models \rho \trianglelefteq \tau$ iff there are canonical, Definition 2, witnesses $D(z_1^{\rho_1}, \ldots, z_l^{\rho_l}) = \lambda f^\tau . f S_1^{\tau_1} \ldots S_n^{\tau_n}$. Since we can reorder components of ρ and τ we can assume that z_1 is in $S_1^{\tau_1}$. Suppose $z_1, \ldots, z_k$, where $k > 1$, are in $S_1^{\tau_1}$ and so y_1 must occur in $T_1^{\rho_1}, \ldots, T_k^{\rho_k}$. Therefore, there is a common coder $S_1^{\tau_1}(x_1/z_1, \ldots, x_k/z_k)$ and k decoders $T_i(\overline{z}_i)$ where $\overline{z}_i = z_{i1}^{\rho_{i1}}, \ldots, z_{il_i}^{\rho_{il_i}}$

$$\cfrac{(\sigma \to o) \to (\sigma \to o) \to o \trianglelefteq (\sigma \to (o \to o \to o) \to o) \to o}{\cfrac{[\sigma \to o, \sigma \to o] \trianglelefteq \sigma \to (o \to o \to o) \to o}{\sigma \to o \trianglelefteq \sigma \to o \quad \sigma \to o \trianglelefteq \sigma \to o} \quad o \trianglelefteq o}$$

Fig. 3. A proof tree for Example 1

and $\rho_{i1}, \ldots, \rho_{il_i}$ are the components of ρ_i such that $T_i(\overline{z}_i)(S_1^{\tau_1}(x_1, \ldots, x_k)) =_{\beta\eta} x_i\overline{z}_i$ (which is similar to the product in [3]). In $S_1^{\tau_1}(x_1/z_1, \ldots, x_k/z_k)$ there are distinct odd length paths $w_1, \ldots, w_k$ of type τ_1 to the lambda nodes above $x_1, \ldots, x_k$. These paths are realisable, Definition 6, because each x_i belongs to the normal form of $T_i(\overline{z}_i)(S_1^{\tau_1}(x_1, \ldots, x_k))$. Using a combinatorial argument, see Proposition 3, $S_1^{\tau_1}$ can be chosen so that these words are k-minimal and by reordering ρ's components $w_1 \sqsubset \ldots \sqsubset w_k$. We may not be able to reduce to the subgoals $\rho_1 \trianglelefteq \tau_1, \ldots, \rho_k \trianglelefteq \tau_1$ as w_i may prescribe the form of $T_i(\overline{z}_i)$: if $T_i(\overline{z}_i) = \lambda f^{\tau_1}.fS_1^i \ldots S_m^i$ then path w_i may prevent S_j^i containing elements of $\overline{z}_i$; so, this may restrict the possible distribution of $\overline{z}_i$ within the subterms $S_1^i, \ldots, S_m^i$ which is captured using $\tau_1 \upharpoonright w_i$.

An example proof tree is in Figure 3 for the retract of Example 1 (which is not affine). Rule P_1 is applied to the root and then P_2 to the first subgoal where $w_1 = (\varepsilon, 2, 21)$ and $w_2 = (\varepsilon, 2, 22)$. Let $\sigma' = \sigma \to (o \to o \to o) \to o$. Now, $\sigma' \upharpoonright w_1 = \sigma \to o = \sigma' \upharpoonright w_2$; in both cases the first component of σ' is admitted unlike the second.

3.2 Multiple Ground Types

We extend the proof system to include multiple ground types. Again, assertions are of the two kinds $\rho \trianglelefteq \tau$ and $[\rho_1, \ldots, \rho_k] \trianglelefteq \tau$. However, we now assume that to be a well-formed assertion $\rho \trianglelefteq \tau$ both ρ and τ must share the same target type (which is guaranteed when there is a single ground type). The rules for this assertion are as before the axiom I, weakening W, covariance C and the product rule P_1 in Figure 2: however, C carries the requirement that the target types of δ and σ coincide. The other product rule P_2', presented in Figure 4, is different: the *arity* of $\rho_1 \to \ldots \to \rho_n \to a$ is the maximum of n and the arities of each ρ_i where a gound type a has arity 0.

In $[\rho_1, \ldots, \rho_k] \trianglelefteq \sigma$ it is not required that ρ_j and σ share the same target type. Instead rule P_2' requires that ρ_i and $v_i(\sigma)$, see Definition 9, do share target types: for the concatenation $v_i^\wedge w_i$ see Definition 10. The specialisation to the case of the single ground type is when $U = \emptyset$ and $v = \varepsilon$.

Let $\rho = \rho_1 \to \ldots \to \rho_l \to a$ and $\tau = \tau_1 \to \ldots \to \tau_n \to a$. So, $\models \rho \trianglelefteq \tau$ iff there are canonical witnesses $D(z_1^{\rho_1}, \ldots, z_l^{\rho_l}) = \lambda f^\tau.fS_1^{\tau_1} \ldots S_n^{\tau_n}$ and $C(x) = \lambda y_1^{\tau_1} \ldots y_n^{\tau_n}.H(xT_1^{\rho_1} \ldots T_l^{\rho_l})$. Assume $z_1, \ldots, z_k$, where $k \geq 1$, occur in $S_1^{\tau_1}$. There is a path v in $C(x)$ to the node above x which determines a subtree U of $S_1^{\tau_1}$. The head variable in $T_i^{\rho_i}$ bound in v has the same target type as ρ_i. There are distinct paths $v_1^\wedge w_1, \ldots, v_k^\wedge w_k$ of odd length to the lambda nodes above $z_1, \ldots, z_k$ in $S_1^{\tau_1}$: v_i is decided by the meaning of the head variable in $T_i^{\rho_i}$; so, $v_i(\tau_1)$ has the same target type as ρ_i. The rest of the path is the tail of w_i: so we need to consider whether $\models \rho_i \trianglelefteq v_i(\tau_1) \upharpoonright w_i$.

Figure 5 is the proof tree for the retract in Example 2. There is an application of P_1 followed by P_2'. In the application of P_2' the subtree $U = \{(\varepsilon, 2)\}$, $v_1 = \varepsilon$, $w_1 = (\varepsilon, 2, 21) = v_1^\wedge w_1$, $v_2 = (\varepsilon, 2, 22) = v_2^\wedge w_2$ when $w_2 = (22)$. So, $v_1(b \rightarrow (a \rightarrow o \rightarrow a) \rightarrow a) \upharpoonright w_1 = b \rightarrow a$ as the first component is admitted (unlike the second); and $v_2(b \rightarrow (a \rightarrow o \rightarrow a) \rightarrow a) = o = o \upharpoonright w_2$.

3.3 Complexity

The proof systems provide nondeterministic decision procedures for checking retracts. Each subgoal of a proof rule has smaller size than the goal. Hence, by focussing on one subgoal at a time a proof witness can be presented in PSPACE. However, this does not take into account checking that a subgoal obeys the side conditions in the case of the product rules. Given any type σ, there are boundedly many realisable k-minimal paths (with an upper bound of k^n where n is size of σ). So, this means that overall the decision procedure requires at most EXPSPACE.

4 Soundness and Completeness

To show soundness and completeness of our proof systems, we define a dialogue game $\mathsf{G}(D(\overline{z}), C(x))$ played by a single player $\forall$ on the trees of potential witnesses for a retract that characterises when $(D(\overline{z}))C(x) =_{\beta\eta} x\overline{z}$, similar to game semantics [5].

$$P_2' \; \frac{[\rho_1, \ldots, \rho_k] \trianglelefteq \sigma}{\rho_1 \trianglelefteq v_1(\sigma) \upharpoonright w_1 \quad \ldots \quad \rho_k \trianglelefteq v_k(\sigma) \upharpoonright w_k} \text{ where}$$

- k' is the maximum of k and h^2 where h is the arity of σ
- there is a k'-minimal realisable subtree U of type σ where each path has even length (which can be $\emptyset$),
- each v_i is ε, a prefix of a path in U of odd length or the extension of a path in U with a single node,
- $v_1^\wedge w_1 \sqsubset \ldots \sqsubset v_k^\wedge w_k$ and each $v_i^\wedge w_i$ is a k'-minimal realisable path of type σ of odd length and if $U \neq \emptyset$, $v_i^\wedge w_i$ extends some path in U.

Fig. 4. Product proof rule for multiple gound types

$$\dfrac{\dfrac{b \to a \trianglelefteq b \to a \quad o \trianglelefteq o}{[b \to a, o] \trianglelefteq b \to (a \to o \to a) \to a} \quad a \trianglelefteq a}{(b \to a) \to o \to a \trianglelefteq (b \to (a \to o \to a) \to a) \to a}$$

Fig. 5. A proof tree for Example 2

4.1 Coder/Decoder Game

For the definition of the game $\mathsf{G}(D(\overline{z}), C(x))$ played by a single player $\forall$ we assume that the nodes of $D(\overline{z})$ and $C(x)$ are disjoint (as in Figure 1).

Definition 12. *Let N be the set of* nodes *of the trees of $D(\overline{z})$ and $C(x)$. We define the subset $O \subseteq N$ of* observable *nodes inductively: if n is labelled with free variable x or $z_i \in \overline{z}$ then $n \in O$; if $n \in O$, n is a variable node and m is a successor of n then $m \in O$; if $n \in O$ and $n \downarrow m$ then $m \in O$. Let N_1 be the subset of* binding nodes *in $N \backslash O$ that are not observable and assume that Σ is the set of* labels *of the nodes in N_1.*

A position in a play of $\mathsf{G}(D(\overline{z}), C(x))$ is a pair $n\,\theta$ where $n \in N$ and θ is a look-up table (similar to a closure) defined as follows [11].

Definition 13. *For each $i \geq 0$, the* set of look-up tables *Θ_i is iteratively defined:*

1. $\Theta_0 = \{\emptyset\}$
2. *Θ_{i+1} is the set of partial maps $N_1 \to (\bigcup_{\lambda y_1 \ldots y_p \in \Sigma} N^p \times \bigcup_{j \leq i} \Theta_j)$.*

Definition 14. *A* play *of $\mathsf{G}(D(\overline{z}), C(x))$ is a sequence $n_1\,\theta_1, \ldots, n_l\,\theta_l$ of positions where for each $i \geq 1$, $n_i \in N$, $\theta_i \in \Theta_i$ and $n_l \in O$ is a leaf node of $D(\overline{z})$ or $C(x)$. For the* initial position *n_1 is the root of $D(\overline{z})$ and $\theta_1 = \{((m, \emptyset)/n_1\}$ where m is the root of $C(x)$.*

Thus, θ_1 is the partial function that maps n_1 (the root of $D(\overline{z})$) labelled with a binder λf) to the pair $(m, \emptyset)$ where m is the root of $C(x)$ and $\emptyset$ is the empty partial map. Standard update notation is assumed: $\gamma\{((m_1, \ldots, m_k), \gamma')/n\}$ is the partial function γ'' similar to γ except that $\gamma''(n) = ((m_1, \ldots, m_k), \gamma')$.

Definition 15. *If the current position in a play of $\mathsf{G}(D(\overline{z}), C(x))$ is $n\,\theta$ and either $n \notin O$ or n is not a leaf of $D(\overline{z})$ or $C(x)$ then the next position is determined by a unique move according to the label at node n (and where ni is the ith successor node of n) and whether $n \in O$:*

- *$\lambda\overline{y}$ then $n1\,\theta$,*
- *y^a, $m \downarrow_j n$ and $\theta(m) = ((m_1, \ldots, m_p), \theta')$ then $m_j\,\theta'$,*
- *$y^{\sigma_1 \to \ldots \to \sigma_k \to a}$, $m \downarrow_j n$ and $\theta(m) = ((m_1, \ldots, m_p), \theta')$ then $m_j\,\theta''$ where $\theta'' = \theta'\{((n1, \ldots, nk), \theta)/m_j\}$,*

– $y^{\sigma_1 \to \ldots \to \sigma_k \to a}$, *$n \in O$ and n has successors then $\forall$ chooses $i : 1 \leq i \leq k$ and $ni\,\theta$.*

Assume play is at node n with look-up table θ. If n is a lambda node then the next position is at its successor with the same look-up table. If n is labelled with y^a, $m \downarrow_j n$ and $\theta(m) = ((m_1, \ldots, m_p), \theta')$ then play jumps to node m_j (which is labelled with a dummy lambda) and the new look-up table is θ'. If n is labelled $y^{\sigma_1 \to \ldots \to \sigma_k \to a}$, $m \downarrow_j n$ and $\theta(m) = ((m_1, \ldots, m_p), \theta')$ then m_j is labelled $\lambda v_1^{\sigma_1} \ldots v_k^{\sigma_k}$ for some v and n has k successors; so, the next position is a jump to m_j with look-up table $\theta'\{((n1, \ldots, nk), \theta)/m_j\}$: the binder m_j is associated with the successors of n; that is, if play is later at a node labelled v_r (and bound by m_j) then the next position will be at nr. In both these cases when n is labelled y, the jump from n to m_j is from a node of $D(\overline{z})$ to a node of $C(x)$ or vice-versa, hence the idea of dialogue. The other possibility is that play is at an observable variable node, $n \in O$. If n is a leaf node (of $D(\overline{z})$ or $C(x)$) then the position is final and play ends; otherwise, player $\forall$ chooses a successor ni of n and the next position is $ni\,\theta$.

Definition 16. *The* game tree *of* $\mathsf{G}(D(\overline{z}), C(x))$ *is the tree of plays where all choices of $\forall$ are included.*

The reader is invited to consider the game tree for $\mathsf{G}(D(z_1, z_2), C(x))$ when $D(z_1, z_2)$ and $C(x)$ are the term trees of Figure 1. Play is initially at the root node (0) of $D(z_1, z_2)$ with look-up table θ_1. The next position is at (1), bound by (0): play, therefore, jumps to the root node (12) of $C(x)$; any node bound by (12), such as (15) or (17), is associated with (2), the successor of (1), in $D(z_1, z_2)$ as shown by the entry for (12) in the current look-up table. The next position is at the observable node (13). $\forall$ can choose (14) or (23) as next position. Consider the left continuation. After (14), play is at (15) and so play jumps to (2): the binder (2) is associated with the successors of (15), nodes labelled u with (15) and nodes labelled v with (18). Play is next at (3), then jumps to (18): the binder (18) is asociated with nodes (4) (for s) and (8) (for t). The next position is (19); so, play jumps to (4), descends to the observable (5) and then to (6), as it is the only $\forall$ choice; then to (7) which means it jumps to (16). The next position at the observable leaf (17) is final.

Definition 17. *The* value *of* $\mathsf{G}(D(\overline{z}), C(x))$ *is the subtree of its game tree that consists of the observable nodes. The* value term *t, which we write as $t \in \mathsf{G}(D(\overline{z}), C(x))$, is the term that consists of the labels of the nodes of the value tree when dummy lambdas are omitted.*

For instance, the value of $\mathsf{G}(D(z_1, z_2), C(x))$ above is the tree in Figure 6 and the value term is $x(\lambda w.z_1 w)(\lambda w.z_2 w)$. The following is clear (from Definition 12 of observable nodes).

Fact 1 *If $t \in \mathsf{G}(D(\overline{z}), C(x))$ then t is in normal form.*

We now come to some simple properties of game playing.

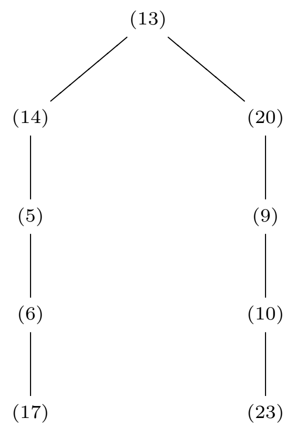

Fig. 6. The value subtree of $\mathsf{G}(D(z_1, z_2), C(x))$

Fact 2 *Assume* $n_1\,\theta_1, \ldots, n_l\,\theta_l$ *is a play of* $\mathsf{G}(D(\overline{z}), C(x))$*.*

1. *If* $i = 2k + 1$ *for* $k \geq 0$ *then* n_i *is a lambda node.*
2. *If* $i = 2k$ *for* $k \geq 1$ *then* n_i *is a variable node.*
3. n_l *is a variable node.*

The same look-up table may occur multiple times in a play. To be precise, two look-up tables θ, θ' are *equal*, $\theta = \theta'$, if they have the same entries. That is, they are defined for the same nodes and for each m, $\theta(m) = ((n_1, \ldots, n_p), \theta_i)$ iff $\theta'(m) = ((n_1, \ldots, n_p), \theta'_i)$ and $\theta_i = \theta'_i$: this is well-defined (and not "circular") because for each play position j, if $\theta_j(m) = ((n_1, \ldots, n_p), \theta')$ then θ' is $\emptyset$ or the look-up table at some earlier position $i < j$ (as we now show).

Proposition 2. *Assume* $n_1\,\theta_1, \ldots, n_l\,\theta_l$ *is a play of* $\mathsf{G}(D(\overline{z}), C(x))$ *and* $\theta_j(m) = ((m_1, \ldots, m_p), \theta')$ *for* $1 \leq j \leq l$*.*

1. $\theta_j(n)$ *is defined iff* n *is a binding node above or equal to* n_j *and* $n \notin O$*.*
2. $\theta'(n)$ *is defined iff* n *is a binding node above each* m_k*,* $1 \leq k \leq p$ *and* $n \notin O$*.*
3. *For some* u*,* m *is labelled* $\lambda u_1 \ldots u_p$ *and each* u_k *has the same type as* m_k*,* $1 \leq k \leq p$*.*
4. *Either* $\theta' = \emptyset$ *or there is* $i < j$ *with* $\theta' = \theta_i$ *and* $n_i \notin O$ *is a variable node and each* $m_k = n_i k$*,* $1 \leq k \leq p$*.*

Proof. We prove the four properties by simultaneous induction on the position j. For the base case $j = 1$, n_1 is the root of $D(\overline{z})$ labelled λf where f has the same type as $C(x)$ and $\theta_1 = \{(m, \emptyset)/n_1\}$ where m is the root of $C(x)$. Therefore, $n_1 \notin O$. So, all four properties hold. For the general case assume that the four properties hold for all positions before $n_s\,\theta_s$. We show that they hold at $n_s\,\theta_s$ by cases on the label at node $n = n_{s-1}$ and whether $n \in O$.

$\lambda\overline{y}$: so $n_s = n1$ and $\theta_s = \theta_{s-1}$. We know that $n1$ is not a binding node and so all four hold by the induction hypothesis.

y: and $m \downarrow_q n$ and $\theta_{s-1}(m) = ((m_1, \ldots, m_p), \theta')$ for some $p \geq q$. Therefore, $n_s = m_q$ and $\theta_s = \theta'$ when y^a and so by the induction hypothesis m_q is then labelled with a dummy lambda (as it has type a too). Otherwise, $\theta_s = \theta'\{((n1, \ldots, nk), \theta_{s-1})/m_q\}$ and $y^{\sigma_1 \to \ldots \to \sigma_k \to a}$ and by the induction hypothesis m_q has the same type (so, is labelled $\lambda u_1 \ldots u_k$ for some u). As $n_s = m_q$ we note that the binding nodes above m_q are those strictly above $m_1, \ldots, m_p$ (which by the induction hypothesis on 4 have the form $n'1, \ldots n'p$) and so are above m_q when y_q is not of base type. Thus, 1 holds for θ_s using the induction hypothesis for 2 (for θ') and the entry for m_q in θ_s when y has higher type. For the remaining cases assume $\theta_s(m') = ((m'_1, \ldots, m'_{p'}), \theta'')$. So, either this is also an entry for θ' and therefore 2, 3 and 4 follow by the induction hypothesis on θ' (which is θ_i for $i < s$) or $m' = m_q$, $\theta'' = \theta_{s-1}$ and $m'_1, \ldots, m'_{p'}$ are $n1, \ldots, nr$; and so 2, 3 and 4 hold.

y: and $n \in O$. If n is a leaf of $D(\overline{z})$ or $C(x)$ then position $s-1$ is final. Otherwise $\forall$ chooses i and $n_s = ni$, and $\theta_s = \theta_{s-1}$: as $ni \in O$, all 4 properties hold by the induction hypothesis.

□

We now come to the characterisation theorem which appeals to the standard β-reduction, $\longrightarrow_\beta$, and its reflexive and transitive closure, $\longrightarrow_\beta^*$.

Lemma 1. *$t \in \mathsf{G}(D(\overline{z}), C(x))$ iff $D(\overline{z})(C(x)) \longrightarrow_\beta^* t$.*

Proof. We define leftmost reduction sequences using the following two rules:

1. $(\lambda y_1 \ldots y_p.U)S_1 \ldots S_p \longrightarrow U\{S_1/y_1, \ldots, S_p/y_p\}$,
2. $yS_1 \ldots S_p \longrightarrow S'_j$ for $j : 1 \leq j \leq p$ where $S'_j = S''_j$ if $S_j = \lambda\overline{y}.S''_j$ and $S'_j = S_j$ otherwise.

A term U is *terminal* if neither rule applies to it. We slightly enlarge reduction sequences by allowing a third reduction rule for padding: $S \longrightarrow S$ if S is not terminal. A *padding* of $U_1 \longrightarrow \ldots \longrightarrow U_m$ includes 0 or more applications of the padding rule as well as applications of the other rules.

Consider the game $\mathsf{G}(D(\overline{z}), C(x))$. The tree representations of $D(\overline{z})$ and $C(x)$ introduce extra dummy lambdas which we assume are not present in the reduction sequences. Given a node $n \in N$ and a look-up table θ we define the *associated* term as $\theta[n]$ as follows by cases on the label at n:

- λ: $\theta[n1]$,
- $\lambda\overline{y}$: $\lambda\overline{y}.\theta[n1]$, $\overline{y}$ not empty
- y_j^a: if $m \downarrow n$ and $\theta(m) = ((m_1, \ldots, m_p), \theta')$ then $\theta'[m_j]$ else y_j,
- $y_j^{\sigma_1 \to \ldots \to \sigma_k \to a}$: if $m \downarrow n$ and $\theta(m) = ((m_1, \ldots, m_k), \theta')$ then $\theta'[m_j]\,\theta[n1] \ldots \theta[nk]$ else $y_j\,\theta[n1] \ldots \theta[nk]$.

We shall show that there is a correspondence between terms in reduction sequences and associated terms in game plays.

Assume a reduction sequence $\overline{U}$ that starts with $D(\overline{z})(C(x))$. We show that there is a play $n_1\,\theta_1,\ldots,n_l\,\theta_l$ of $\mathsf{G}(D(\overline{z}),C(x))$ and a padding of $\overline{U}$ of the form $U_1 \longrightarrow \ldots \longrightarrow U_m$ such that $U_{j+1} = \theta_{2j}[n_{2j}]$ for all $j : 1 \leq j < m$. The proof is by induction on i equal to the length of $\overline{U}$.

The base case is $i = 1$. The first position of any play of $\mathsf{G}(D(\overline{z}),C(x))$ is $n_1\theta_1$ where n_1 is the root of $D(\overline{z})$ which has the form $\lambda f^{\tau}.fS_1\ldots S_p$ and $\theta_1(n_1) = (m,\emptyset)$ where m is the root of $C(x)$: we can assume that f does not occur in any S_i. The second position is $n_2\theta_2$ where $\theta_2 = \theta_1$ and $n_2 = n_1 1$; so $\theta_2[n_2] = \emptyset[m]\theta_2[n_2 1]\ldots\theta_2[n_2 p]$. As $\emptyset$ is empty, and the definition of $\theta[n]$ removes dummy lambdas $\emptyset[m] = C(x)$ and, moreover since f does not occur in S_i, $\theta_2[ni] = S_i$. And by definition of the reduction sequence also $U_2 = C(x)S_1\ldots S_p$.

Assume the result holds for $i \leq q$: there is a play $n_1\,\theta_1,\ldots,n_l\,\theta_l$ of $\mathsf{G}(D(\overline{z}),C(x))$ and a padding of $\overline{U}$, $U_1 \longrightarrow \ldots \longrightarrow U_m$ such that $U_{j+1} = \theta_{2j}[n_{2j}]$ for all $j < q$. This means that if $U_i = yT_1\ldots T_p$, $T_k = \lambda\overline{y}.T''_k$ and $U_{i+1} = T''_k$, or T^a_k and $U_{i+1} = T_k$ then $\theta_{2(i-1)}[n_{2(i-1)}] = yT_1\ldots T_p$ and $\theta_{2i}[n_{2i}] = T''_k$, or $\theta_{2i}[n_{2i}] = T_k$; so, the play follows the same branch of observable nodes as the reduction sequence. We consider $i = q+1$. We know that $U_q = \theta_{2(q-1)}[n_{2(q-1)}]$. We proceed by cases on the variable y at $n_{2(q-1)}$ (which cannot be a lambda node by Fact 2).

Assume $y^{\sigma_1\to\ldots\to\sigma_r\to a}$ and $m \downarrow_k n_{2(q-1)}$. If $\theta_{2(q-1)}[m] = ((m_1,\ldots,m_s),\theta')$ then $\theta_{2(q-1)}[n_{2(q-1)}] = \theta'[m_k]\theta_{2(q-1)}[n_{2(q-1)}1]\ldots\theta_{2(q-1)}[n_{2(q-1)}r] = U_q$ which has the form $(\lambda y_1\ldots y_r.S')T'_1\ldots T'_r$. So, there is the reduction $U_q \longrightarrow U_{q+1}$ where $U_{q+1} = S'\{T'_1/y_1,\ldots,T'_r/y_r\}$, and, in the game play, $n_{2q-1} = m_k$ and $\theta_{2q-1} = \theta'\{((n_{2(q-1)}1,\ldots,n_{2(q-1)}r),\theta_{2(q-1)})/m_k\}$, and $n_{2q} = m_k 1$ and $\theta_{2q} = \theta_{2q-1}$ and so $\theta_{2q}[m_k 1] = S'\{T'_1/y_1,\ldots,T'_r/y_r\}$ as required. Otherwise, $\theta_{2(q-1)}[m]$ is not defined. So, $\theta_{2(q-1)}[n_{2(q-1)}] = y\theta_{2(q-1)}[n_{2(q-1)}1]\ldots\theta_{2(q-1)}[n_{2(q-1)}r] = yT'_1\ldots T'_r$ and there is the reduction $U_q \longrightarrow U_{q+1}$ where $U_{q+1} = T''_l$ for some l where $T''_l = U'$ if $T'_l = \lambda\overline{y}.U'$ and $T''_l = T'_l$ otherwise. So, we consider the next position in the play when $\forall$ chooses l, $n_{2q-1} = n_{2(q-1)}l$ and $\theta_{2q-1} = \theta_{2(q-1)}$. If n_{2q-1} is labelled with a dummy lambda then $T'_l = \theta_{2q}[n_{2q}]$ otherwise n_{2q-1} is labelled $\lambda\overline{y}$ and $T'_k = \lambda\overline{y}.\theta_{2q}[n_{2q}]$, so $\theta_{2q}[n_{2q}] = U_{q+1}$.

Next assume y^a, $m \downarrow_k n_{2(q-1)}$ and $\theta_{2(q-1)}[m] = ((m_1,\ldots,m_s),\theta')$. It is only this case that introduces padding, so $U_{q+1} = U_q$. So, $\theta_{2(q-1)}[n_{2(q-1)}] = \theta'[m_k] = U_q$ which has the form $S' : a$. Node $n_{2q-1} = m_k$ which is labelled with a dummy lambda and $\theta'[m_k] = \theta'[m_k 1]$: $n_{2q} = m_k 1$ and $\theta_{2q} = \theta'$. The case where there is no entry $\theta_{2(q-1)}[m]$ cannot arise as then U_q would be terminal.

An almost identical argument shows that given any *prefix* $n_1\theta_1,\ldots,n_l\theta_l$ of a play in $\mathsf{G}(D(\overline{z}),C(x))$ there is a reduction sequence $\overline{U}$ and a padding of $\overline{U}$ of the form $U_1 \longrightarrow \ldots \longrightarrow U_m$ such that $U_{j+1} = \theta_{2j}[n_{2j}]$ for all $j : 1 \leq j < m$: the proof proceeds by induction on the length of the prefix play. One imediate corollary of this argument is that because of strong normalisation there cannot be an infinite length play in $\mathsf{G}(D(\overline{z}),C(x))$.

Now the main theorem follows. A *maximal* reduction sequence is one whose last term is terminal. If $D(\overline{z})(C(x)) \longrightarrow^*_\beta t$ where t is in normal form then $t \in \mathsf{G}(D(\overline{z}),C(x))$: every play passes through observable nodes in a branch of t

and there cannot be any other plays (as there would be corresponding reduction sequences). □

In the following, paths in terms are paths in their tree representation.

Fact 3 *Assume* $\mathsf{G}(D(\overline{z}), C(x))$ *is a game,* $D(\overline{z}) = \lambda f.fS_1 \dots S_n$ *and* $C(x) = \lambda y_1 \dots y_n.H(xT_1 \dots T_l)$*,* $w = n_1, \dots, n_p$ *is the path in* S_i *to the lambda node above the single occurrence of* z_j*,* $v = m_1, \dots, m_q$ *is the path from the root of* $C(x)$ *to the lambda node above the single occurrence of* x*.*

1. z_j *is in the normal form of* $D(\overline{z})(C(x))$ *iff there is a play in* $\mathsf{G}(D(\overline{z}), C(x))$ *such that* $n_1\theta_{i_1}, \dots, n_p\theta_{i_p}$ *is a subsequence of positions for some* $\theta_{i_1}, \dots, \theta_{i_p}$*.*
2. x *is in the normal form of* $D(\overline{z})(C(x))$ *iff there is a play in* $\mathsf{G}(D(\overline{z}), C(x))$ *such that* $m_1\theta_{j_1}, \dots, m_q\theta_{j_q}$ *is a subsequence of positions for some* $\theta_{j_1}, \dots, \theta_{j_q}$*.*

Definition 18. *Assume* $D(\overline{z}) = \lambda f.fS_1 \dots S_n$ *and* $C(x) = \lambda y_1 \dots y_n.H(xT_1 \dots T_l)$*,* w *is the path in* S_i *to the lambda node above the single occurrence of* z_j*,* v *is the path from the root of* $C(x)$ *to the lambda node above the single occurrence of* x *and both* x *and* z_j *are in the normal form of* $D(\overline{z})(C(x))$*.*

1. *The* subtree *of* S_i associated with v *is the smallest subtree* S'_i *such that if* $D'(\overline{z}) = D(\overline{z})(S'_i/S_i)$ *then* x *is in the normal form of* $D'(\overline{z})(C(x))$*.*
2. *The* subtree *of* T_i associated with w *is the smallest subtree* T'_i *such that if* $C'(x) = C(x)(T'_i/T_i)$ *then* x *is in the normal form of* $D'(\overline{z})(C(x))$*.*

Definition 19. *Assume* w *is a path in a term* $\lambda x_1 \dots x_k.S'$*. The* subpath of w generated by x_i *is the subsequence of* w' *of* w *such that*

– *if* $n \in w$ *is labelled* x_i *then* $n \in w'$*,*
– *if* $n \in w'$ *is a variable node and* n' *is a successor of* $n \in w$ *then* $n' \in w'$*,*
– *if* $n \in w'$ *and* $n \downarrow n' \in w$ *then* $n' \in w'$*.*

Proposition 3. *Assume* $\models \rho \trianglelefteq \tau$ *and* $k' = k$ *if there is a single ground type and otherwise is the maximum of* k *and the square of the arity of* τ_1*. There are canonical witnesses* $D(\overline{z}) = \lambda f^{\tau}.fS_1^{\tau_1} \dots S_n^{\tau_n}$ *and* $C(x) = \lambda y_1^{\tau_1} \dots y_n^{\tau_n}.H(xT_1^{\rho_1} \dots T_l^{\rho_l})$ *for* $\rho \trianglelefteq \tau$ *such that if* $z_1, \dots, z_k$ *occur in* S_1 *and* v *is the path from the root of* $C(x)$ *to the node above* x

1. *for each* i*, there is a* k'*-minimal path* w_i *from the root of* S_1 *to the node above* z_i*, and*
2. *the subpath of* v *generated by* y_1 *is* k'*-minimal.*

Proof. Assume $D(\overline{z}) = \lambda f^{\tau}.fS_1^{\tau_1} \dots S_n^{\tau_n}$ and $C(x) = \lambda y_1^{\tau_1} \dots y_n^{\tau_n}.H(xT_1^{\rho_1} \dots T_l^{\rho_l})$ are canonical witnesses for $\rho \trianglelefteq \tau$, $z_1, \dots, z_k$ occur in S_1 and v is the path from the root of $C(x)$ to the node above x and w_i is the path from the root of S_1 to the node above z_i. Consider the subtree S'_1 of S_1 given as the set of paths $\{w_1, \dots, w_k\}$. Let V be the subtree of S'_1 associated with v (which is the subtree associated with the subpath of v generated by y_1). So, each path $w_i = v_i^{\wedge}w'_i$ where either $v_i = \varepsilon$, v_i is an odd length prefix of a path in V or an extension of a path in V with a single node. The upward closure of a set of nodes X is defined as X^u as expected: if $n \in X$ then $n \in X^u$; if $n \in X^u$ and $m \downarrow n$ then $m \in X^u$; if $n \in X^u$ is a lambda node and m is above n then $m \in X^u$.

We now pick out important kinds of node in the tree S'_1. A (variable) node is a separator if it has more than one successor. The others only occur when there is multiple ground types. If $v_i \neq \varepsilon$ then its final node is a root if its target type is different from that of the target type of τ_1. A lambda node labelled $\lambda\overline{s}$ which is an ith successor of a node labelled y' is a change node if there is not a node above which is also an ith successor of a node labelled y', and there is $s_j \in \overline{s}$ whose result type is different from that of y'.

A node in S'_1 is essential if it belongs to the upward closure of the set of successors of separators, of roots and of change nodes. Otherwise a node is inessential. The argument is completed by showing that we can discard all inessential nodes from S'_1 (and associated nodes from v). The resulting S'_1 and subpath generated by y_1 are therefore k'-minimal.

First, consider the case when there is a single ground type. To get the result as $H = \varepsilon$ we can simply transform S'_1: if w'_i is the subsequence of essential nodes in w_i and w''_i is the subsequence of inesssential nodes then we replace w_i by w'_i and add w''_i beneath the root node of each successor of z_i. A simple argument (using the dialogue game) shows that this preserves normal form. For the general case, the argument is similar but more involved; we delete inessential nodes from any v_i and their corresponding nodes in v. The transformation on S'_1 is similar to the ground type case. If a variable node and its successor m in w_i share the same result type and m only binds variable nodes below some successors of z_i then this pair can be removed from w_i and placed below the roots of those successors of z_i; this is then iterated. Otherwise, we delete other inessential nodes which may require that nodes are also deleted from T_i, and, in the case, where a variable node and its successor recurs, any node bound by that successor is rebound to the earlier successor. Again these transformations are justified by the dialogue game. □

4.2 Soundness of Tableau Proof Systems

We now show soundness of the proof systems.

Theorem 1. *(Soundness) If* $\vdash \rho \trianglelefteq \tau$ *then* $\models \rho \trianglelefteq \tau$.

Proof. By induction on the depth of a proof. For the base case, the result is clear for a proof that uses the axiom I. So, assume the result for all proofs of depth $< d$. Consider now a proof of depth d. We proceed by examining the first rule that is applied to show $\vdash \rho \trianglelefteq \tau$. If it is W or C the result follows using the same arguments as in [3]. Assume the rule is W and suppose $\models \rho \trianglelefteq \tau$. Therefore there are terms D_1 and C_1 such that $D_1^{\tau\to\rho}(C_1^{\rho\to\tau x}) =_{\beta\eta} x$. Now $D^{(\sigma\to\tau)\to\rho} = \lambda f^{\sigma\to\tau} y^{\sigma}.D_1(fy)$ and $C^{\rho\to(\sigma\to\tau)}x = \lambda s^{\sigma}.C_1(x)$ are witnesses for $\models \rho \trianglelefteq \sigma \to \tau$. Assume that the rule is C, so $\models \delta \trianglelefteq \sigma$ and $\models \rho \trianglelefteq \tau$. So there are terms D_1, C_1, D_2, C_2 such that $D_1^{\sigma\to\delta}(C_1^{\delta\to\sigma}x) =_{\beta\eta} x$ and $D_2^{\tau\to\rho}(C_2^{\rho\to\tau}x) =_{\beta\eta} x$.

Now $D^{(\sigma\to\tau)\to(\delta\to\rho)} = \lambda xy.C_2(x(D_1y))$ and $C^{(\delta\to\rho)\to(\sigma\to\tau)} = \lambda uz.D_2(u(C_1z))$ are witnesses for $\models \delta \to \rho \trianglelefteq \sigma \to \tau$.

Consider next that the first rule is P_1. So after P_1 there is either an application of P_2 or P_2': in the former case, there are k-minimal realisable paths $w_1 \sqsubset \ldots \sqsubset w_k$ of odd length of type σ such that $\vdash \rho_i \trianglelefteq \sigma \upharpoonright w_i$; in the latter case, there is a k'-minimal realisable subtree U of type σ where each path has even length; and there are paths $v_1^\wedge w_1 \sqsubset \ldots \sqsubset v_k^\wedge w_k$ where each element is a k'-minimal realisable path of type σ of odd length and if $U \neq \emptyset$, it extends some path in U and where each v_i is ε, a prefix of a path in U of odd length path or an extension of a path in U with a single node and $\vdash \rho_i \trianglelefteq v_i(\sigma) \upharpoonright w_i$; where k' is the maximum of k and the square of the arity of σ. So, by the induction hypothesis there are terms $D_i(\overline{z}_i)$ and $C_i(x_i)$ such that $D_i(\overline{z}_i)(C_i(x_i)) =_{\beta\eta} x_i\overline{z}_i$, witnesses for $\rho_i \trianglelefteq \sigma \upharpoonright w_i$ or $\rho_i \trianglelefteq v_i(\sigma) \upharpoonright w_i$, and terms $D'(z_{k+1}, \ldots, z_l)$ and $C'(x')$ such that $D'(z_{k+1}, \ldots, z_l)(C'(x')) =_{\beta\eta} x'z_{k+1}\ldots z_l$, witnesses for $\rho_{k+1} \to \ldots \to \rho_l \to a \trianglelefteq \tau'$ where $\tau' = \tau_2 \to \ldots \to \tau_n \to a$. We assume that all these terms are canonical witnesses. The term $D'(z_{k+1}, \ldots, z_l)$ is $\lambda f^{\tau'}.fS_2^{\tau_2}\ldots S_n^{\tau_n}$ and $C'(x')$ is $\lambda y_2^{\tau_2}\ldots y_n^{\tau_n}.H'(x'T_{k+1}^{\rho_{k+1}}\ldots T_l^{\rho_l})$ where $H' = \varepsilon$ if the rule applied was P_2.

We need to show that there are terms $D(z_1, \ldots, z_l)$ and $C(x)$ that are witnesses for $\models \rho \trianglelefteq \tau$. $D(\overline{z})$ will have the form $\lambda f^\tau.fS_1^{\tau_1}\ldots S_n^{\tau_n}$ and $C(x)$ the form $\lambda y_1^{\tau_1}\ldots y_n^{\tau_n}.H(xT_1^{\rho_1}\ldots T_l^{\rho_l})$ where $H = \varepsilon$ in the case of a single ground type. All that remains is to define $S_1^{\tau_1}$ so it contains $z_1, \ldots, z_k$, $T_1^{\rho_1}, \ldots, T_k^{\rho_k}$ and H (as an extension of H'). If $U = \emptyset$ then $H = H'$. Otherwise, let u be an odd length path such that U is associated with (so, its head variable is $y_1^{\tau_1}$) Definition 18. H consists of the suffix of u followed by the subtree H'. The head variable of each $T_i^{\rho_i}$ is y_1 in the case of the single ground type and $g_i^{v_i(\sigma)}$ in the general case (which is either y_1 or bound in u). We assume that S_i' is the subterm of S_1^σ that is rooted at the initial vertex of the path w_i: which is S_1^σ itself in the single ground type. To complete these terms we require that $T_i^{\rho_i}(S_1^\sigma(z_1, \ldots, z_k)) =_{\beta\eta} z_i$. Therefore, removing lambda abstraction over variables z_{ij} and changing z_i to x_i, we require that $T_i(\overline{z}_i)(S_i'(x_1, \ldots, x_k)) =_{\beta\eta} x_i\overline{z}_i$. We construct a term $C''(x_i)$ that occurs after the path w_i in S_i' (and which has root x_i when there is a single ground type). We also complete $T_i(\overline{z}_i)$ whose initial part is the tree U_i associated with the path w_i, Definition 18.

First, we examine the single ground type case. So, S_1^σ will have the form $\lambda u_1\ldots u_m.S_1'$, $C''(x_i)$ the form $x_iC_{i1}''\ldots C_{ip}''$ and $T_i(\overline{z}_i)$ the form $\lambda f_i^\sigma.f_iV_1^i\ldots V_m^i$. Assume $D_i(\overline{z}_i)$ is $\lambda g_i^{\sigma\upharpoonright w_i}.g_iW_{i_1}^i\ldots W_{i_l}^i$ and $C_i(x_i)$ is $\lambda u_{i_1}\ldots u_{i_l}.x_iC_1^i\ldots C_p^i$. Assume w_i admits σ_{i_j}: therefore, for some $r : 1 \leq r \leq m$, $i_j = r$ (so, W_r^i may contain occurrences of variables in $\overline{z}_i$). If u_r does not occur in the path w_i then we set $V_r^i = W_r^i$. Otherwise, there is a non-empty subpath w_{ir} of w_i generated by u_r, Definition 19 and a subtree U_r^i of V_r^i associated with w_{ir}. Each C_j^i contains a single u_{i_k} (as head variable). Assume C_s^i contains u_r. Assume that the path in W_r^i to the lambda node above z_{is} is w_s'. If we can build the same path in V_r^i (by copying nodes of C_s^i to C_{is}'') then we are done (letting V_r^i include this path followed by the subterm of W_r^i rooted at z_{is}). Otherwise, we initially include w_{ir}

in C''_{is} and then try to build w'_s in V^i_r by copying nodes of C^i_s to C''_{is}: in V^i_r and, therefore in U^i_r, there is a path whose prefix except for its final variable vertex is the same as a prefix of w'_s and then differ. In the game $\mathsf{G}(C''_{is}, V^i_r)$, play jumps from that variable in V^i_r to a lambda node in w_{ir}. By definition of admits, there is a binder n' labelled $\lambda\overline{v}$ in w_{ir} such that for some q not($n' \downarrow_q n'_i$) for all nodes n'_i after n' in w_i (and in w_{ir}). Therefore, we add a variable node labelled v_q to the end of w_{ir} in C''_{is}; so play jumps to a lambda node in V^i_r which is a successor of a leaf of U^i_r; below this node, we build the path w'_s except for its root node (by adding further nodes to C''_{is} and add the subtree rooted at z_{is} in W^i_r to V^i_r).

For the general case, assume $v_i(\sigma) = \sigma'_1 \rightarrow \ldots \rightarrow \sigma'_m \rightarrow b$. So, $S_i^{v_i(\sigma)}$ will have the form $\lambda u_1 \ldots u_m.S'_1$, $C''(x_i)$ the form $H_i(x_i C''_{i1} \ldots C''_{ip})$ and $T_i(\overline{z}_i)$ the form $\lambda f_i^{\sigma}.f_i V^i_1 \ldots V^i_m$. Assume $D_i(\overline{z}_i)$ is $\lambda g_i^{v_i(\sigma)\restriction w_i}.g_i W^i_{i_1} \ldots W^i_{i_l}$ and $C_i(x_i)$ is $\lambda u'_{i_1} \ldots u'_{i_l}.H'_i(x_i C^i_1 \ldots C^i_p)$. We set $H_i = H'_i$. Then we proceed in a similar fashion to the single base type case. If some u'_r does not occur in the path w_i then $V^i_r = W^i_r$; otherwise we need to build similar paths to z_{is} in W^i_r in V^i_r (by copying vertices from C^i_s to C''_{is} and using that w_i admits $(v_i(\sigma))_r$. □

4.3 Completeness of Tableau Proof Systems

In the following proof assume $\rho = \rho_1 \rightarrow \ldots \rightarrow \rho_l \rightarrow a$, $\tau = \tau_1 \rightarrow \ldots \rightarrow \tau_n \rightarrow a$ and $\tau_1 = \sigma = \sigma_1 \rightarrow \ldots \rightarrow \sigma_m \rightarrow b$.

Theorem 2. *(Completeness) If* $\models \rho \trianglelefteq \tau$ *then* $\vdash \rho \trianglelefteq \tau$.

Proof. We proceed by induction on the size of ρ. The base case is when $\rho = a$ and $\models a \trianglelefteq \tau$. Using the weakening rule W repeatedly we obtain a successful proof tree whose leaf is the axiom $a \trianglelefteq a$. Now assume the result for ρ' smaller than ρ and that $D(\overline{z})$ and $C(x)$ are canonical witnesses in lnf for $\models \rho \trianglelefteq \tau$. So, $D(\overline{z})$ has the form $\lambda f^{\tau}.f S_1^{\tau_1} \ldots S_n^{\tau_n}$ and $C(x)$ is $\lambda y_1^{\tau_1} \ldots y_n^{\tau_n}.H(x T_1^{\rho_1} \ldots T_l^{\rho_l})$ where $H = \varepsilon$ when there is a single ground type and $D(\overline{z})(C(x)) =_{\beta\eta} x\overline{z}$. We can reorder the arguments of both ρ and of τ, as discussed previously. Each z_i occurs only once in $D(z_1, \ldots, z_l)$. Therefore, for the multiple ground type case, there is a realisable odd length path v from the root of $C(x)$ to the lambda node above x. Associated with this path are subtrees U_i of $S_i^{\tau_i}$. The target type of each node in v (and, therefore, in each U_i) is a.

First, if only z_1 is in $S_1^{\tau_1}$ and the target types of ρ_1 and τ_1 are the same then we show that $\models \rho_1 \trianglelefteq \tau_1$ and $\models \rho' \trianglelefteq \tau'$ when $\rho' = \rho_2 \rightarrow \ldots \rightarrow \rho_l \rightarrow a$ and $\tau' = \tau_2 \rightarrow \ldots \rightarrow \tau_n \rightarrow a$. Assume there is just one ground type. Then $D_1(\overline{z}_1) = \lambda y_1^{\tau_1}.T_1^{\rho_1}(\overline{z}_1)$ and $C(x) = S_1^{\tau_1}(x/z_1)$ are the witnesses for $\models \rho_1 \trianglelefteq \tau_1$ as $\lambda y_1^{\tau_1}.T_1^{\rho_1}(\overline{z}_1)(S_1^{\tau}(x/z_1)) =_{\beta\eta} x\overline{z}_1$. For $\models \rho' \trianglelefteq \tau'$, the decoder $D'(z_2, \ldots, z_l)$ is $\lambda f^{\tau'}.f S_2^{\tau_2} \ldots S_n^{\tau_n}$ and the coder $C'(x)$ is $\lambda y_2^{\tau_2} \ldots y_n^{\tau_n}.x T_2^{\rho_2} \ldots T_l^{\rho_l}$. For the multiple ground type case, assume v' is the subpath of v generated by y_1, Definition 19. Now $D_1(\overline{z}_1) = \lambda y_1.v'(T_1(\overline{z}_1))$ and $C(x) = S_1(x/z_1)$ are witnesses for $\models \rho_1 \trianglelefteq \tau_1$; also $\lambda f^{\tau'}.f S_2^{\tau_2} \ldots S_n^{\tau_n}$ and $\lambda y_2^{\tau_2} \ldots y_n^{\tau_n}.H'(x T_2^{\rho_2} \ldots T_l^{\rho_l})$ where H' is the result of removing v' from v are witnesses for $\models \rho' \trianglelefteq \tau'$. Therefore, by the induction hypothesis $\vdash \rho_1 \trianglelefteq \tau_1$ and $\vdash \rho' \trianglelefteq \tau'$; by the covariance rule it follows that $\vdash \rho \trianglelefteq \tau$.

Next we assume that $z_1, \ldots, z_k$, $k \geq 1$, occur in $S_1^{\tau_1}$ such that if $k = 1$ then the target types of ρ_1 and τ_1 differ. (As we can reorder arguments we can guarantee that $S_1^{\tau_1}$ contains a prefix of $\overline{z}$). We will now show that $[\rho_1, \ldots, \rho_k] \trianglelefteq \tau_1$ and $\models \rho_{k+1} \to \ldots \to \rho_l \to a \trianglelefteq \tau'$ where τ' is as above; the witnesses for the latter are $\lambda f^{\tau'}.f S_2^{\tau_2} \ldots S_n^{\tau_n}$ and $\lambda y_2^{\tau_2} \ldots y_n^{\tau_n}.H'(x T_{k+1}^{\rho_{k+1}} \ldots T_l^{\rho_l})$ where H' is either ε or defined as the result of removing v' from v in H.

For the single ground type case, we show that there are k-minimal realisable paths $w_1 \sqsubset \ldots \sqsubset w_k$ such that $\models \rho_i \trianglelefteq \tau_1 \upharpoonright w_i$. So, $\vdash \rho \trianglelefteq \tau$ using the induction hypothesis and the rules P_2 and P_1. Consider the realisable subpath v' of v defined above; we can assume it is k'-minimal by Proposition 3 (where k' is maximum of k and the square of the arity of σ). Associated with it is a k'-minimal subtree U of type $\sigma = \tau_1$ where each path has even length. Therefore, in $S_1^{\tau_1}$ there are distinct paths $v_i^\wedge w_i$ of odd length from its root to the lambda node above z_i, $1 \leq i \leq k$, where either $v_i = \varepsilon$ or v_i is a prefix of a path in U or the extension of a path in U with a single node. These v_is are chosen so that the first element of w_i is the lambda node to which play jumps after the head variable of $T_i^{\rho_i}$: so, $v_i(\sigma)$ and ρ_i share the same target type. The rest of the proof that for each i, $\models \rho_i \trianglelefteq v_i(\sigma) \upharpoonright w_i$ is similar to the single ground type case to which we now turn. So, $\vdash \rho \trianglelefteq \tau$ follows by the induction hypothesis using P_2' and P_1.

For the single ground type case, y_1 occurs once as the head variable in each of $T_1^{\rho_1}, \ldots, T_k^{\rho_k}$ and S_1^{σ} contains one occurrence of each z_i, $1 \leq i \leq k$, where, clearly, no z_i occurs below a z_j. Therefore, in S_1^{σ} there are distinct paths w_i of odd length from the root to the lambda node above z_i, $1 \leq i \leq k$. By definition each w_i is realisable (as z_i occurs in the normal form). By Proposition 3 we know that each w_i can be chosen to be k-minimal. So, S_1^{σ} has the form $\lambda u_1 \ldots u_m.S'$. Instantiating the ρ_i variables, each $T_i(\overline{z}_i)$ when y_1 is abstracted has the form $\lambda y_1^{\tau_1}.y_1 V_1^i \ldots V_m^i$. Consider the subtree U_r^i of V_r^i associated with w_i. Assume that some element in $\overline{z}_i$ occurs in V_r^i. We show that w_i admits σ_r (so, it is a component of $\sigma \upharpoonright w_i$). If u_r does not occur in w_i then w_i admits σ_r. Otherwise consider a closed path from the root of V_r^i to the lambda node above an element in $\overline{z}_i$. Consider the longest even length prefix of this path in U_r^i: its final node is a variable node which cannot be of ground type (as w_i cannot contain a variable node of ground type) and, therefore, for which the next position in the play is at lambda node in w_i that obeys the property that w_i admits σ_r. Let $i_1, \ldots, i_l$ be the indices of components of σ that belong to $\sigma \upharpoonright w_i$ and let $j_1, \ldots, j_p$ be the remaining indices. Let $D_i(\overline{z}_i) = \lambda y_1^{\sigma \upharpoonright w_i}.y_1 V_{i_1}^i \ldots V_{i_l}^i$ and $C_i(x) = \lambda u_{i_1} \ldots u_{i_l}.S'(V_{j_1}^i/u_{j_1}, \ldots, V_{j_p}^i/u_{j_p})$: these are witnesses for $\models \rho_i \trianglelefteq \sigma \upharpoonright w_i$. $\Box$

5 Conclusion

We have provided tableau proof systems that characterise when a type is a retract of another type in simply typed lambda calculus (with respect to $\beta\eta$-equality). The proof systems are goal directed and appeal to finite paths in terms. They offer a nondeterministic decision procedure for the retract problem in EXPSPACE:

however, it may be possible to improve on the rather crude k-minimality bounds used on paths within the proof systems by further minimisation. Given the constructive proof of correctness, we also expect to be able to extract witnesses for a retract from a successful tableau proof tree (similar in spirit to [10]).

References

1. Barendregt, H. Lambda calculi with types. In *Handbook of Logic in Computer Science, Vol 2*, ed. Abramsky, S., Gabbay, D. and Maibaum, T., Oxford University Press, 118-309, (1992).
2. Bruce, K. and Longo, G. Provable isomorphisms and domain equations in models of typed languages. *Proc. 17th Symposium on Theory of Computing*, ACM, 263-272, (1985).
3. de'Liguoro, U., Piperno, A. and Statman, R. Retracts in simply typed $\lambda\beta\eta$-calculus. *Procs. LICS 1992*, 461-469, (1992).
4. Loader, R. Higher-order β-matching is undecidable, *Logic Journal of the IGPL*, 11(1), 51-68, (2003).
5. Ong, C.-H. L. On model-checking trees generated by higher-order recursion schemes, *Procs LICS 2006*, 81-90.
6. Ong and Tzevelekos. Functional Reachability. *Procs LICS 2009*, 286-295, (2009).
7. Padovani, V. Decidability of fourth-order matching. *Mathematical Structures in Computer Science*, 10(3), 361-372, (2000).
8. Padovani, V. Retracts in simple types. In: Abramsky, S. (ed.) TLCA 2001. *LNCS*, 2044, 376-384, (2001)
9. Regnier, L. and Urzyczyn, P. Retractions of types with many atoms. At http://arxiv.org/abs/cs/0212005, pp1-16.
10. Schubert, A. On the building of affine retractions. *Math. Struct. in Comp. Science*, 18, 753-793, (2008).
11. Stirling, C. Higher-order matching, games and automata. *Procs LICS 2007*, 326-335, (2007).
12. Stirling, C. Dependency tree automata. In: de Alfaro, L. (ed.) FOSSACS 2009. *LNCS*, 5504, 92-106, (2009).
13. Stirling, C. Decidability of higher-order matching *Logical Methods in Computer Science*, 5(3:2), 1-52, (2009).
14. Stirling, C. An introduction to decidability of higher-order matching. *Submitted for publication.* Available at author's website, (2012).
15. Vorobyov, S. The "hardest" natural decidable theory. *Procs LICS 1997*, 294-305, (1997).