Logic II: proofs and programs

Helmut Schwichtenberg

Mathematisches Institut der LMU, Sommersemester 2025

Contents

Preface

This is the script for the course "Logic II: proofs and programs" at LMU, held in Sommersemester 2025. It is mainly based on the textbook "Proofs and Computations" in the references. I would like to thank Valentin Herrmann for his help, and the students for their active participation.

München, 23. July 2025

Helmut Schwichtenberg

CHAPTER 1

Logic

1.1. Computational content of proofs

Mathematics differs from all other sciences by the fact that it provides proofs for its claims. In this course we study what proofs are, and what we can do with them apart from assuring us of the truth of what they state.

Let us start with simple example of a proof. Assume that we already know that $\sqrt{2}$ is irrational.

THEOREM. *There are irrational numbers x, y such that x^y is rational.*

PROOF. By cases.

Case $\sqrt{2}^{\sqrt{2}}$ is rational. Choose $x := \sqrt{2}$ and $y := \sqrt{2}$. Then x, y are irrational and by assumption x^y is rational.

Case $\sqrt{2}^{\sqrt{2}}$ is irrational. Choose $x := \sqrt{2}^{\sqrt{2}}$ and $y := \sqrt{2}$. Then by assumption x, y are irrational and

$$x^y = \left(\sqrt{2}^{\sqrt{2}}\right)^{\sqrt{2}} = \left(\sqrt{2}\right)^2 = 2$$

is rational. □

A problem with this proof is that it does not give us an example for what its statement claims to exist. Which pair of real numbers to take depends on whether $\sqrt{2}^{\sqrt{2}}$ is rational or not. As long as we do not know whether this is the case we do not have an example.

An obvious solution to this problem is to extend the standard use of the existential quantifier in mathematics by a new one written $\exists_x A(x)$, whose proof requires an explicit construction of an object x satisfying the property $A(x)$. This in *in addition* to the standard use of the existential quantifier, which we now write as $\tilde{\exists}_x A(x)$ and understand it as $\neg\forall_x \neg A(x)$. The former is called the strong (or constructive) existential quantifier, and the latter the weak (or "classical") one.

Similarly there is a strong (or constructive) disjunction written $A \vee B$, which is in addition to the standard weak (or classical) one. The latter is written $A \tilde{\vee} B$ and understood as $\neg A \to (\neg B \to \bot)$.

Derivation	Term
$u\colon A$	u^A
$\dfrac{\begin{array}{c}[u\colon A]\\ \mid M\\ B\end{array}}{A \to B} \to^+ u$	$(\lambda_{u^A} M^B)^{A\to B}$
$\dfrac{\begin{array}{cc}\mid M & \mid N\\ A \to B & A\end{array}}{B} \to^-$	$(M^{A\to B} N^A)^B$
$\dfrac{\begin{array}{c}\mid M\\ A\end{array}}{\forall_x A} \forall^+ x$ (with var.cond.)	$(\lambda_x M^A)^{\forall_x A}$ (with var.cond.)
$\dfrac{\begin{array}{cc}\mid M & \\ \forall_x A(x) & t\end{array}}{A(t)} \forall^-$	$(M^{\forall_x A(x)} t)^{A(t)}$

TABLE 1. Derivation terms for $\to$ and $\forall$

1.2. Natural deduction

Proofs are done in natural deduction style, following Gentzen (1935). Using the Curry-Howard correspondence we write them as proof terms.

We give an inductive definition of such derivation terms for the $\to, \forall$-rules in Table 1 where for clarity we have written the corresponding derivations to the left. This can be extended to the rules for $\exists$, $\vee$ and $\wedge$, but we will not do this here. The reason is that these connectives will be viewed as inductively defined (nullary) predicates with parameters.

Every derivation term carries a formula as its type. However, we shall usually leave these formulas implicit and write derivation terms without them.

Every derivation term can be written uniquely in one of the forms

$$u\vec{M} \mid \lambda_v M \mid (\lambda_v M)N\vec{L},$$

where u is an assumption variable or constant, v is an assumption or object variable, and M, N, L are derivation or object terms.

1.3. Normal form

An important property of proof terms is that they have a unique normal form. It arises as follows.

A *conversion* removes an elimination immediately following an introduction. We consider the following conversions, for derivations written in tree notation and also as derivation terms.

$\to$-conversion.

$$\dfrac{\dfrac{\begin{array}{c}[u\colon A]\\ \mid M\\ B\end{array}}{A \to B}\to^+ u \qquad \begin{array}{c}\mid N\\ A\end{array}}{B}\to^- \quad \mapsto_\beta \quad \begin{array}{c}\mid N\\ A\\ \mid M\\ B\end{array}$$

or written as derivation terms

$$(\lambda_u M(u^A)^B)^{A\to B} N^A \mapsto_\beta M(N^A)^B.$$

The reader familiar with λ-calculus should note that this is nothing other than β-conversion.

$\forall$-conversion.

$$\dfrac{\dfrac{\begin{array}{c}\mid M\\ A(x)\end{array}}{\forall_x A(x)}\forall^+ x \qquad t}{A(t)}\forall^- \quad \mapsto_\beta \quad \begin{array}{c}\mid M'\\ A(t)\end{array}$$

or written as derivation terms

$$(\lambda_x M(x)^{A(x)})^{\forall_x A(x)} t \mapsto_\beta M(t).$$

The *closure* of the conversion relation $\mapsto_\beta$ is defined by

(a) If $M \mapsto_\beta M'$, then $M \mapsto M'$.
(b) If $M \mapsto M'$, then also $MN \mapsto M'N$, $NM \mapsto NM'$, $\lambda_v M \mapsto \lambda_v M'$ (*inner reductions*).

Therefore $M \mapsto N$ means that M *reduces in one step to* N, i.e., N is obtained from M by replacement of (an occurrence of) a redex M' of M by a conversum M'' of M', i.e., by a single conversion.

A term M is *in normal form*, or M is *normal*, if M does not contain a redex. A *reduction sequence* is a (finite or infinite) sequence $M_0 \mapsto M_1 \mapsto$

$M_2 \ldots$ such that $M_i \mapsto M_{i+1}$, for all i. Finite reduction sequences are partially ordered under the initial part relation; the collection of finite reduction sequences starting from a term M forms a tree, the *reduction tree* of M. The branches of this tree may be identified with the collection of all infinite and all terminating finite reduction sequences. A term is *strongly normalizing* if its reduction tree is finite.

THEOREM 1.3.1. *Every derivation term is strongly normalizing, and the final element of each reduction sequence is uniquely determined.*

A proof can be found for instance in Troelstra and van Dalen (1988).

CHAPTER 2

Partial continuous functionals

The objects studied in mathematics have types, which in many cases are function types, possibly of a higher type. Such objects in most cases are infinite, and we intend to describe them in terms of their finite approximations. An appropriate framework for such an approach are the partial continuous functionals of Scott (1982) and Ershov (1977). Continuity of a function f here means that for every approximation V of the value $f(x)$ there is an approximation U of the argument x such that $f[U]$ has more information than V. We define the partial continuous functionals via Scott's information systems.

2.1. Information systems

The basic idea of information systems is to provide an axiomatic setting to describe approximations of abstract objects by concrete, finite ones. We take an arbitrary countable set A of "bits of data" or "tokens" as a basic notion to be explained axiomatically. In order to use such data to build approximations of abstract objects, we need a notion of "consistency", which determines when the elements of a finite set of tokens are consistent with each other. We also need an "entailment relation" between consistent sets U of data and single tokens a, which intuitively expresses the fact that the information contained in U is sufficient to compute the bit of information a. The axioms below are a minor modification of Scott's (1982), due to Larsen and Winskel (1991).

2.1.1. Ideals.

Definition. An *information system* is a structure $(A, \mathrm{Con}, \vdash)$ where A is an at most countable non-empty set (the *tokens*), Con is a set of finite subsets of A (the *consistent* sets) and $\vdash$ is a subset of $\mathrm{Con} \times A$ (the *entailment*

relation), which satisfy

$$
\begin{aligned}
&U \subseteq V \in \mathrm{Con} \to U \in \mathrm{Con},\\
&\{a\} \in \mathrm{Con},\\
&U \vdash a \to U \cup \{a\} \in \mathrm{Con},\\
&a \in U \in \mathrm{Con} \to U \vdash a,\\
&U \in \mathrm{Con} \to \forall_{a\in V}(U \vdash a) \to V \vdash b \to U \vdash b.
\end{aligned}
$$

The elements of Con are called *formal neighborhoods*. We use U, V, W to denote *finite* sets, and write

$$
\begin{aligned}
&U \vdash V \quad \text{for} \quad U \in \mathrm{Con} \wedge \forall_{a\in V}(U \vdash a),\\
&a \uparrow b \quad \text{for} \quad \{a, b\} \in \mathrm{Con} \qquad (a, b \text{ are } \textit{consistent}),\\
&U \uparrow V \quad \text{for} \quad \forall_{a\in U, b\in V}(a \uparrow b).
\end{aligned}
$$

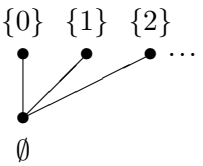
Definition. The *ideals* (also called *objects*) of an information system $\boldsymbol{A} = (A, \mathrm{Con}, \vdash)$ are defined to be those subsets x of A which satisfy

$$
\begin{aligned}
&U \subseteq x \to U \in \mathrm{Con} \quad (x \text{ is } \textit{consistent}),\\
&U \vdash a \to U \subseteq x \to a \in x \quad (x \text{ is } \textit{deductively closed}).
\end{aligned}
$$

We write $x \in |\boldsymbol{A}|$ to mean that x is an ideal of $\boldsymbol{A}$.

Examples. The *deductive closure* $\overline{U} := \{\, a \in A \mid U \vdash a \,\}$ of $U \in \mathrm{Con}$ is an ideal.

Every countable set A can be turned into a "flat" information system by letting the set of tokens be A, $\mathrm{Con} := \{\emptyset\} \cup \{\, \{a\} \mid a \in A \,\}$ and $U \vdash a$ mean $a \in U$. In this case the ideals are just the elements of Con. For $A = \mathbb{N}$ we have the following picture of the Con-sets.



A rather important example is the following, which concerns approximations of functions from a countable set A into a countable set B. The tokens are the pairs (a, b) with $a \in A$ and $b \in B$, and

$$
\begin{aligned}
&\mathrm{Con} := \{\, \{\, (a_i, b_i) \mid i < k \,\} \mid \forall_{i,j<k}(a_i = a_j \to b_i = b_j) \,\},\\
&U \vdash (a, b) := (a, b) \in U.
\end{aligned}
$$

It is easy to verify that this defines an information system whose ideals are (the graphs of) all partial functions from A to B.

2.1.2. Function spaces. We define the "function space" $\boldsymbol{A} \to \boldsymbol{B}$ between two information systems $\boldsymbol{A}$ and $\boldsymbol{B}$.

DEFINITION. Let $\boldsymbol{A} = (A, \mathrm{Con}_A, \vdash_A)$ and $\boldsymbol{B} = (B, \mathrm{Con}_B, \vdash_B)$ be information systems. Define $\boldsymbol{A} \to \boldsymbol{B} = (C, \mathrm{Con}, \vdash)$ by

$$C := \mathrm{Con}_A \times B,$$

$$\{\,(U_i, b_i) \mid i \in I\,\} \in \mathrm{Con} := \forall_{J \subseteq I}\Big(\bigcup_{j \in J} U_j \in \mathrm{Con}_A \to \{\,b_j \mid j \in J\,\} \in \mathrm{Con}_B\Big).$$

For the definition of the entailment relation $\vdash$ it is helpful to first define the notion of an *application* of $W := \{\,(U_i, b_i) \mid i \in I\,\} \in \mathrm{Con}$ to $U \in \mathrm{Con}_A$:

$$\{\,(U_i, b_i) \mid i \in I\,\}U := \{\,b_i \mid U \vdash_A U_i\,\}.$$

From the definition of Con we know that this set is in Con_B. Now define $W \vdash (U, b)$ by $WU \vdash_B b$.

REMARK. Clearly application is *monotone in the second argument*, in the sense that $U \vdash_A U'$ implies ($WU' \subseteq WU$, hence also) $WU \vdash_B WU'$. In fact, application is also *monotone in the first argument*, i.e.,

$$W \vdash W' \quad \text{implies} \quad WU \vdash_B W'U.$$

To see this let $W = \{\,(U_i, b_i) \mid i \in I\,\}$ and $W' = \{\,(U'_j, b'_j) \mid j \in J\,\}$. By definition $W'U = \{\,b'_j \mid U \vdash_A U'_j\,\}$. Now fix j such that $U \vdash_A U'_j$; we must show $WU \vdash_B b'_j$. By assumption $W \vdash (U'_j, b'_j)$, hence $WU'_j \vdash_B b'_j$. Because of $WU \supseteq WU'_j$ the claim follows.

LEMMA 2.1.1. *If $\boldsymbol{A}$ and $\boldsymbol{B}$ are information systems, then so is $\boldsymbol{A} \to \boldsymbol{B}$ defined as above.*

PROOF. Let $\boldsymbol{A} = (A, \mathrm{Con}_A, \vdash_A)$ and $\boldsymbol{B} = (B, \mathrm{Con}_B, \vdash_B)$. The first, second and fourth property of the definition are clearly satisfied. For the third, suppose

$$\{(U_1, b_1), \ldots, (U_n, b_n)\} \vdash (U, b), \quad \text{i.e.,} \quad \{\,b_j \mid U \vdash_A U_j\,\} \vdash_B b.$$

We have to show that $\{(U_1, b_1), \ldots, (U_n, b_n), (U, b)\} \in \mathrm{Con}$. So let $I \subseteq \{1, \ldots, n\}$ and suppose

$$U \cup \bigcup_{i \in I} U_i \in \mathrm{Con}_A.$$

We must show that $\{b\} \cup \{\,b_i \mid i \in I\,\} \in \mathrm{Con}_B$. Let $J \subseteq \{1, \ldots, n\}$ consist of those j with $U \vdash_A U_j$. Then also

$$U \cup \bigcup_{i \in I} U_i \cup \bigcup_{j \in J} U_j \in \mathrm{Con}_A.$$

Since

$$\bigcup_{i\in I} U_i \cup \bigcup_{j\in J} U_j \in \mathrm{Con}_A,$$

from the consistency of $\{(U_1, b_1), \ldots, (U_n, b_n)\}$ we can conclude that

$$\{\, b_i \mid i \in I \,\} \cup \{\, b_j \mid j \in J \,\} \in \mathrm{Con}_B.$$

But $\{\, b_j \mid j \in J \,\} \vdash_B b$ by assumption. Hence

$$\{\, b_i \mid i \in I \,\} \cup \{\, b_j \mid j \in J \,\} \cup \{b\} \in \mathrm{Con}_B.$$

For the final property, suppose

$$W \vdash W' \quad \text{and} \quad W' \vdash (U, b).$$

We have to show $W \vdash (U, b)$, i.e., $WU \vdash_B b$. We obtain $WU \vdash_B W'U$ by monotonicity in the first argument, and $W'U \vdash_B b$ by definition. □

We shall now give an alternative characterization of the ideals in $\boldsymbol{A} \to \boldsymbol{B}$, as "approximable maps". The basic idea for approximable maps is the desire to study "information respecting" maps from $\boldsymbol{A}$ into $\boldsymbol{B}$. Such a map is given by a relation r between Con_A and B, where $(U, b) \in r$ intuitively means that whenever we are given the information $U \in \mathrm{Con}_A$, then we know that at least the token b appears in the value.

Definition. Let $\boldsymbol{A} = (A, \mathrm{Con}_A, \vdash_A)$ and $\boldsymbol{B} = (B, \mathrm{Con}_B, \vdash_B)$ be information systems. A relation $r \subseteq \mathrm{Con}_A \times B$ is an *approximable map* if it satisfies the following:

(a) if $(U, b_1), \ldots, (U, b_n) \in r$, then $\{b_1, \ldots, b_n\} \in \mathrm{Con}_B$;
(b) if $(U, b_1), \ldots, (U, b_n) \in r$ and $\{b_1, \ldots, b_n\} \vdash_B b$, then $(U, b) \in r$;
(c) if $(U', b) \in r$ and $U \vdash_A U'$, then $(U, b) \in r$.

Theorem 2.1.2. *Let $\boldsymbol{A}$ and $\boldsymbol{B}$ be information systems. Then the ideals of $\boldsymbol{A} \to \boldsymbol{B}$ are exactly the approximable maps from $\boldsymbol{A}$ to $\boldsymbol{B}$.*

Proof. Let $\boldsymbol{A} = (A, \mathrm{Con}_A, \vdash_A)$ and $\boldsymbol{B} = (B, \mathrm{Con}_B, \vdash_B)$. Assume $r \in |\boldsymbol{A} \to \boldsymbol{B}|$. Then $r \subseteq \mathrm{Con}_A \times B$ is consistent and deductively closed. We have to show that r satisfies the axioms for approximable maps.

(a) Let $(U, b_1), \ldots, (U, b_n) \in r$. We must show that $\{b_1, \ldots, b_n\} \in \mathrm{Con}_B$. But this clearly follows from the consistency of r.

(b) Let $(U, b_1), \ldots, (U, b_n) \in r$ and $\{b_1, \ldots, b_n\} \vdash_B b$. We must show that $(U, b) \in r$. But

$$\{(U, b_1), \ldots, (U, b_n)\} \vdash (U, b)$$

by the definition of the entailment relation $\vdash$ in $\boldsymbol{A} \to \boldsymbol{B}$, hence $(U, b) \in r$ since r is deductively closed.

(c) Let $U \vdash_A U'$ and $(U', b) \in r$. We must show that $(U, b) \in r$. But

$$\{(U', b)\} \vdash (U, b)$$

since $\{(U', b)\}U = \{b\}$ (which follows from $U \vdash_A U'$), hence $(U, b) \in r$, again since r is deductively closed.

For the other direction suppose that $r \subseteq \mathrm{Con}_A \times B$ is an approximable map. We must show that $r \in |\boldsymbol{A} \to \boldsymbol{B}|$.

Consistency of r. Suppose $(U_1, b_1), \ldots, (U_n, b_n) \in r$ and $U = \bigcup\{\, U_i \mid i \in I \,\} \in \mathrm{Con}_A$ for some $I \subseteq \{1, \ldots, n\}$. We must show that $\{\, b_i \mid i \in I \,\} \in \mathrm{Con}_B$. Now from $(U_i, b_i) \in r$ and $U \vdash_A U_i$ we obtain $(U, b_i) \in r$ by axiom (c) for all $i \in I$, and hence $\{\, b_i \mid i \in I \,\} \in \mathrm{Con}_B$ by axiom (a).

Deductive closure of r. Suppose $(U_1, b_1), \ldots, (U_n, b_n) \in r$ and

$$W := \{(U_1, b_1), \ldots, (U_n, b_n)\} \vdash (U, b).$$

We must show $(U, b) \in r$. By definition of $\vdash$ for $\boldsymbol{A} \to \boldsymbol{B}$ we have $WU \vdash_B b$, which is $\{\, b_i \mid U \vdash_A U_i \,\} \vdash_B b$. Further by our assumption $(U_i, b_i) \in r$ we know $(U, b_i) \in r$ by axiom (c) for all i with $U \vdash_A U_i$. Hence $(U, b) \in r$ by axiom (b). □

2.1.3. Continuous functions. We can also characterize approximable maps in a different way, which is closer to usual characterizations of continuity[1]:

Lemma 2.1.3. *Let $\boldsymbol{A}$ and $\boldsymbol{B}$ be information systems and $f\colon |\boldsymbol{A}| \to |\boldsymbol{B}|$ monotone (i.e., $x \subseteq y$ implies $f(x) \subseteq f(y)$). Then the following are equivalent.*

(a) *f satisfies the "principle of finite support" PFS: If $b \in f(x)$, then $b \in f(\overline{U})$ for some $U \subseteq x$.*

(b) *f commutes with directed unions: for every directed $D \subseteq |\boldsymbol{A}|$ (i.e., for any $x, y \in D$ there is a $z \in D$ such that $x, y \subseteq z$)*

$$f\Big(\bigcup_{x \in D} x\Big) = \bigcup_{x \in D} f(x).$$

Note that in (b) the set $\{\, f(x) \mid x \in D \,\}$ is directed by monotonicity of f; hence its union is indeed an ideal in $|\boldsymbol{B}|$. Note also that from PFS and monotonicity of f it follows immediately that if $V \subseteq f(x)$, then $V \subseteq f(\overline{U})$ for some $U \subseteq x$.

Proof. Let f satisfy PFS, and $D \subseteq |\boldsymbol{A}|$ be directed. $f(\bigcup_{x \in D} x) \supseteq \bigcup_{x \in D} f(x)$ follows from monotonicity. For the reverse inclusion let $b \in f(\bigcup_{x \in D} x)$. Then by PFS $b \in f(\overline{U})$ for some $U \subseteq \bigcup_{x \in D} x$. From the

[1] In fact, approximable maps are exactly the continuous functions w.r.t. the so-called Scott topology. However, we will not enter this subject here.

directedness and the fact that U is finite we obtain $U \subseteq z$ for some $z \in D$. From $b \in f(\overline{U})$ and monotonicity infer $b \in f(z)$. Conversely, let f commute with directed unions, and assume $b \in f(x)$. Then

$$b \in f(x) = f(\bigcup_{U \subseteq x} \overline{U}) = \bigcup_{U \subseteq x} f(\overline{U}),$$

hence $b \in f(\overline{U})$ for some $U \subseteq x$. □

We call a function $f\colon |\boldsymbol{A}| \to |\boldsymbol{B}|$ continuous if it satifies the conditions in Lemma 2.1.3. Hence continuous maps $f\colon |\boldsymbol{A}| \to |\boldsymbol{B}|$ are those that can be completely described from the point of view of finite approximations of the abstract objects $x \in |\boldsymbol{A}|$ and $f(x) \in |\boldsymbol{B}|$: whenever we are given a finite approximation V to the value $f(x)$, then there is a finite approximation U to the argument x such that already $f(\overline{U})$ contains the information in V; note that by monotonicity $f(\overline{U}) \subseteq f(x)$.

Clearly the identity and constant functions are continuous, and also the *composition* $g \circ f$ of continuous functions $f\colon |\boldsymbol{A}| \to |\boldsymbol{B}|$ and $g\colon |\boldsymbol{B}| \to |\boldsymbol{C}|$.

THEOREM 2.1.4. *Let $\boldsymbol{A} = (A, \mathrm{Con}_A, \vdash_A)$, $\boldsymbol{B} = (B, \mathrm{Con}_B, \vdash_B)$ be information systems. Then the ideals of $\boldsymbol{A} \to \boldsymbol{B}$ are in a natural bijective correspondence with the continuous functions from $|\boldsymbol{A}|$ to $|\boldsymbol{B}|$, as follows.*

(a) *With any approximable map $r \subseteq \mathrm{Con}_A \times B$ we can associate a continuous function $|r|\colon |\boldsymbol{A}| \to |\boldsymbol{B}|$ by*

$$|r|(z) := \{\, b \in B \mid (U, b) \in r \text{ for some } U \subseteq z \,\}.$$

We call $|r|(z)$ the application *of r to z.*

(b) *Conversely, with any continuous function $f\colon |\boldsymbol{A}| \to |\boldsymbol{B}|$ we can associate an approximable map $\hat{f} \subseteq \mathrm{Con}_A \times B$ by*

$$\hat{f} := \{\, (U, b) \mid b \in f(\overline{U}) \,\}.$$

These assignments are inverse to each other, i.e., $f = |\hat{f}|$ and $r = \widehat{|r|}$.

PROOF. Let r be an ideal of $\boldsymbol{A} \to \boldsymbol{B}$; then by Theorem 2.1.2 we know that r is an approximable map. We first show that $|r|$ is well-defined. So let $z \in |\boldsymbol{A}|$.

$|r|(z)$ is consistent: let $b_1, \ldots, b_n \in |r|(z)$. Then there are $U_1, \ldots, U_n \subseteq z$ such that $(U_i, b_i) \in r$. Hence $U := U_1 \cup \cdots \cup U_n \subseteq z$ and $(U, b_i) \in r$ by axiom (c) of approximable maps. Now from axiom (a) we can conclude that $\{b_1, \ldots, b_n\} \in \mathrm{Con}_B$.

$|r|(z)$ is deductively closed: let $b_1, \ldots, b_n \in |r|(z)$ and $\{b_1, \ldots, b_n\} \vdash_B b$. We must show $b \in |r|(z)$. As before we find $U \subseteq z$ such that $(U, b_i) \in r$. Now from axiom (b) we can conclude $(U, b) \in r$ and hence $b \in |r|(z)$.

Continuity of $|r|$ follows immediately from part (a) of Lemma 2.1.3 above, since by definition $|r|$ is monotone and satisfies PFS.

Now let $f\colon |\boldsymbol{A}| \to |\boldsymbol{B}|$ be continuous. It is easy to verify that $\hat{f}$ is indeed an approximable map. Furthermore one can easily show

$$b \in |\hat{f}|(z) \leftrightarrow b \in f(z)$$

Furthermore

$$\begin{aligned} b \in |\hat{f}|(z) &\leftrightarrow (U,b) \in \hat{f} && \text{for some } U \subseteq z \\ &\leftrightarrow b \in f(\overline{U}) && \text{for some } U \subseteq z \\ &\leftrightarrow b \in f(z) && \text{by monotonicity and PFS.} \end{aligned}$$

Finally, for any approximable map $r \subseteq \mathrm{Con}_A \times B$ we have

$$\begin{aligned} (U,b) \in r &\leftrightarrow \exists_{V \subseteq \overline{U}} (V,b) \in r && \text{by axiom (c) for approximable maps} \\ &\leftrightarrow b \in |r|(\overline{U}) \\ &\leftrightarrow (U,b) \in \widehat{|r|}, \end{aligned}$$

hence $r = \widehat{|r|}$. □

Consequently we can (and will) view approximable maps $r \subseteq \mathrm{Con}_A \times B$ as continuous functions from $|\boldsymbol{A}|$ to $|\boldsymbol{B}|$.

Equality of two subsets $r, s \subseteq \mathrm{Con}_A \times B$ means that they consist of the same tokens (U, b). We can characterize equality $r = s$ by extensional equality of the associated functions $|r|, |s|$. It even suffices that $|r|$ and $|s|$ coincide on all compact elements $\overline{U}$ for $U \in \mathrm{Con}_A$.

Lemma 2.1.5 (Extensionality). *Assume that $\boldsymbol{A} = (A, \mathrm{Con}_A, \vdash_A)$ and $\boldsymbol{B} = (B, \mathrm{Con}_B, \vdash_B)$ are information systems and $r, s \subseteq \mathrm{Con}_A \times B$ approximable maps. Then the following are equivalent.*

(a) $r = s$,
(b) $|r|(z) = |s|(z)$ *for all* $z \in |\boldsymbol{A}|$,
(c) $|r|(\overline{U}) = |s|(\overline{U})$ *for all* $U \in \mathrm{Con}_A$.

Proof. It suffices to prove (c) → (a). As above this follows from

$$\begin{aligned} (U,b) \in r &\leftrightarrow \exists_{V \subseteq \overline{U}} (V,b) \in r && \text{by axiom (c) for approximable maps} \\ &\leftrightarrow b \in |r|(\overline{U}). \end{aligned}$$

□

Moreover, one can easily check that

$$s \circ r := \{\, (U,c) \mid \exists_V ((V,c) \in s \wedge (U,V) \subseteq r) \,\}$$

is an approximable map (where $(U,V) := \{\, (U,b) \mid b \in V \,\}$), and

$$|s \circ r| = |s| \circ |r|, \quad \widehat{g \circ f} = \hat{g} \circ \hat{f}.$$

We usually write $r(z)$ for $|r|(z)$, and similarly $(U,b) \in f$ for $(U,b) \in \hat{f}$. It should always be clear from the context where the mods and hats should be inserted.

2.2. Objects of a given type

We now use information systems to define the objects of the Scott-Ershov model of partial continuous functionals, each of a given "type".

2.2.1. Types. If τ and σ are types, we clearly want that $\tau \to \sigma$ is a type as well, to be called "function type". But we have to start somewhere. The basic idea is that we consider finite lists of (named) "constructor types".

Types may involve type variables $\alpha, \beta, \gamma, \xi, \zeta$. Iterated arrows are understood as associated to the right. For example, $\alpha \to \beta \to \gamma$ means $\alpha \to (\beta \to \gamma)$, not $(\alpha \to \beta) \to \gamma$.

Definition. *Constructor types* κ have the form

$$\vec{\alpha} \to (\xi)_{i<n} \to \xi$$

with all type variables α_i distinct from each other and from ξ. An argument type of a constructor type is called a *parameter* argument type if it is different from ξ, and a *recursive* argument type otherwise. A constructor type is *recursive* if it has a recursive argument type. Each list of named constructor types with all of its parameter argument types distinct determines a *base type* $\iota_{\vec{\kappa}}$. Base types given by a list of named constructors are called *algebras*.

For some common lists of named constructor types there are standard names for the corresponding base types:

$\mathrm{Dummy}\colon \xi$	$\mathbb{U}$ (unit),
$\mathtt{tt}\colon \xi, \mathtt{ff}\colon \xi$	$\mathbb{B}$ (booleans),
$\mathrm{SdL}\colon \xi, \mathrm{SdM}\colon \xi, \mathrm{SdR}\colon \xi$	$\mathbb{D}$ (signed digits),
$\mathrm{Zero}\colon \xi, \mathrm{Succ}\colon \xi \to \xi$	$\mathbb{N}$ (natural numbers, unary),
$\mathrm{One}\colon \xi, \mathrm{S}_0\colon \xi \to \xi, \mathrm{S}_1\colon \xi \to \xi$	$\mathbb{P}$ (positive numbers, binary),
$\mathrm{L}\colon \xi, \mathrm{B}\colon \xi \to \xi \to \xi$	$\mathbb{Y}$ (binary trees)

and with parameter types

$\mathrm{Id}\colon \alpha \to \xi$	$\mathbb{I}(\alpha)$ (identity),
$\mathrm{Nil}\colon \xi, \mathrm{Cons}\colon \alpha \to \xi \to \xi$	$\mathbb{L}(\alpha)$ (lists),
$\mathrm{SCons}\colon \alpha \to \xi \to \xi$	$\mathbb{S}(\alpha)$ (streams),
$\mathrm{Pair}\colon \alpha \to \beta \to \xi$	$\alpha \times \beta$ (product),
$\mathrm{InL}\colon \alpha \to \xi, \mathrm{InR}\colon \beta \to \xi$	$\alpha + \beta$ (sum),
$\mathrm{DummyL}\colon \xi, \mathrm{Inr}\colon \alpha \to \xi$	$\mathrm{uysum}(\alpha)$ (for $\mathbb{U} + \alpha$),
$\mathrm{Inl}\colon \alpha \to \xi, \mathrm{DummyR}\colon \xi$	$\mathrm{ysumu}(\alpha)$ (for $\alpha + \mathbb{U}$).

DEFINITION. *Types* are inductively defined by

(a) Every type variable α is a type.
(b) If $\vec{\kappa}(\vec{\alpha})$ is a list of named constructor types and $\vec{\tau}$ are types where the length of $\vec{\tau}$ is the number of parameters in $\vec{\kappa}$, then $\iota_{\vec{\kappa}(\vec{\tau})}$ is a type.
(c) It τ and σ are types, then so is $\tau \to \sigma$.

Types of the form $\tau \to \sigma$ are called *function types*, and types of the form $\iota_{\vec{\kappa}(\vec{\tau})}$ *base types*.

If the base type corresponding to a list of named constructor types has a standard name, then we use this name to denote the base type.

A type is *closed* it it has no parameters. Let $\tau(\vec{\alpha})$ be a type with $\vec{\alpha}$ its parameters, and let $\vec{\rho}$ be closed types. We define the *level* of $\tau(\vec{\rho})$ by

$$\begin{aligned}\mathrm{lev}(\iota_{\vec{\kappa}(\vec{\rho})}) &:= \max(\mathrm{lev}(\vec{\rho})),\\ &\qquad \text{where the length of } \vec{\rho} \text{ is the number of parameters in } \vec{\kappa}(\vec{\alpha}),\\ \mathrm{lev}(\tau \to \sigma) &:= \max(\mathrm{lev}(\sigma), 1 + \mathrm{lev}(\tau)).\end{aligned}$$

Examples of base types:

- $\mathbb{L}(\alpha)$, $\mathbb{L}(\mathbb{L}(\alpha))$, $\alpha \times \beta$ are base types of level 0.
- $\mathbb{L}(\mathbb{L}(\mathbb{N}))$, $\mathbb{N} + \mathbb{B}$, $\mathbb{Z} := \mathbb{P} + \mathbb{U} + \mathbb{P}$, $\mathbb{Q} := \mathbb{Z} \times \mathbb{P}$ are closed base types of level 0.
- $\mathbb{R} := (\mathbb{N} \to \mathbb{Q}) \times (\mathbb{P} \to \mathbb{N})$ is a closed base type of level 1.

2.2.2. The information system of a given type. For every closed type τ we define the information system $\boldsymbol{A}_\tau = (A_\tau, \mathrm{Con}_\tau, \vdash_\tau)$. The ideals $x \in |\boldsymbol{A}_\tau|$ are the *partial continuous functionals* of type τ. Since we will have $\boldsymbol{A}_{\tau\to\sigma} = \boldsymbol{A}_\tau \to \boldsymbol{A}_\sigma$, the partial continuous functionals of type $\tau \to \sigma$ will correspond to the continuous functions from $|\boldsymbol{A}_\tau|$ to $|\boldsymbol{A}_\sigma|$.

DEFINITION (Information system of a closed type τ). We simultaneously define $A_{\iota_{\vec{\kappa}(\vec{\tau})}}$, $A_{\tau\to\sigma}$, $\mathrm{Con}_{\iota_{\vec{\kappa}(\vec{\tau})}}$ and $\mathrm{Con}_{\tau\to\sigma}$.

(a) The *tokens* $a \in A_{\iota_{\vec{\kappa}(\vec{\tau})}}$ are the type correct constructor expressions

$$\mathrm{C}U_1 \dots U_m a_1^* \dots a_n^*$$

with C the name of a constructor type $\vec{\alpha} \to (\xi)_{i<n} \to \xi$ from $\vec{\kappa}$, all U_j $(1 \le j \le m)$ from Con_{τ_j} and each a_i^* $(1 \le i \le n)$ an *extended token*, i.e., a token or the special symbol $*$ which carries no information.

(b) The tokens in $A_{\tau\to\sigma}$ are the pairs (U, b) with $U \in \mathrm{Con}_\tau$ and $b \in A_\sigma$.

(c) A finite set U of tokens in $A_{\iota_{\vec{\kappa}(\vec{\tau})}}$ is *consistent* (i.e., $U \in \mathrm{Con}_{\iota_{\vec{\kappa}(\vec{\tau})}}$) if
   (i) all its elements start with the same constructor C, say of arity $\vec{\tau} \to (\iota_{\vec{\kappa}(\vec{\tau})})_{i<n} \to \iota_{\vec{\kappa}(\vec{\tau})}$,
   (ii) the union V_j of all Con-sets at the j-th $(1 \le j \le m)$ argument position of some token in U is in Con_{τ_j}, and
   (iii) all $U_i \in \mathrm{Con}_{\iota_{\vec{\kappa}(\vec{\tau})}}$ $(1 \le i \le n)$, where U_i consists of all (proper) tokens at the $(m+i)$-th argument position of some token in U.

(d) $\{\,(U_i, b_i) \mid i \in I\,\} \in \mathrm{Con}_{\tau\to\sigma}$ is defined to mean

$$\forall_{J\subseteq I}\Big(\bigcup_{j\in J} U_j \in \mathrm{Con}_\tau \to \{\,b_j \mid j \in J\,\} \in \mathrm{Con}_\sigma\Big).$$

Building on this definition, we define $U \vdash_\tau a$ for $U \in \mathrm{Con}_\tau$ and $a \in A_\tau$.

(e) $\{\mathrm{C}\vec{U}_1\vec{a}_1^*, \dots, \mathrm{C}\vec{U}_l\vec{a}_l^*\} \vdash_{\iota_{\vec{\kappa}(\vec{\tau})}} \mathrm{C}'\vec{W}\vec{a^*}$ is defined to mean $\mathrm{C} = \mathrm{C}'$, $l \ge 1$, $V_j \vdash W_j$ with V_j as in (c) above and $U_i \vdash a_i^*$ with U_i as in (c) above (and $U \vdash *$ defined to be true).

(f) $W \vdash_{\tau\to\sigma} (U, b)$ is defined to mean $WU \vdash_\sigma b$, where application WU of $W = \{\,(U_i, b_i) \mid i \in I\,\} \in \mathrm{Con}_{\tau\to\sigma}$ to $U \in \mathrm{Con}_\tau$ is defined to be $\{\,b_i \mid U \vdash_\tau U_i\,\}$.

Note that the present definition is by recursion on the *height* of the syntactic expressions involved, defined by

$$\begin{aligned}
|\alpha| &:= 0,\\
|\iota_{\vec{\kappa}(\vec{\tau})}| &:= 1 + \max\{\,|\tau_i| \mid \tau_i \in \vec{\tau}\,\},\\
|\tau \to \sigma| &:= \max\{1 + |\tau|, |\sigma|\},\\
|\mathrm{C}U_1 \dots U_m a_1^* \dots a_n^*| &:= 1 + \max(\{\,|U_j| \mid 1 \le j \le m\,\} \cup \{\,|a_i^*| \mid 1 \le i \le n\,\}),\\
|*| &:= 0,\\
|(U, b)| &:= 1 + \max\{|U|, |b|\},\\
|\{\,a_i \mid i \in I\,\}| &:= 1 + \max\{\,|a_i| \mid i \in I\,\},\\
|U \vdash a| &:= 1 + \max\{|U|, |a|\}.
\end{aligned}$$

It is easy to see that $(A_\tau, \mathrm{Con}_\tau, \vdash_\tau)$ is an information system. Observe that all the notions involved are computable: $a \in A_\tau$, $U \in \mathrm{Con}_\tau$ and $U \vdash_\tau a$.

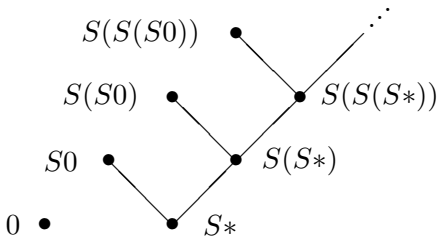


FIGURE 1. Tokens and entailment for $\mathbb{N}$

DEFINITION (Partial continuous functionals). For every closed type τ let $\boldsymbol{A}_\tau$ be the information system $(A_\tau, \mathrm{Con}_\tau, \vdash_\tau)$. The set $|\boldsymbol{A}_\tau|$ of ideals in $\boldsymbol{A}_\tau$ is the set of *partial continuous functionals* of type τ. A partial continuous functional $x \in |\boldsymbol{A}_\tau|$ is *computable* if it is recursively enumerable when viewed as a set of tokens.

Notice that $\boldsymbol{A}_{\tau\to\sigma} = \boldsymbol{A}_\tau \to \boldsymbol{A}_\sigma$ as defined generally for information systems.

For example, the tokens for the base type $\mathbb{N}$ are shown in Figure 1 (with 0 for Zero and S for Succ). For tokens a, b we have $\{a\} \vdash b$ if and only if there is a path from a (up) to b (down). As another example, consider the base type $\mathbb{Y}$ of binary trees with a nullary constructor L (for Leaf) and a binary B (for Branch). Then $\{B(L,*), B(*,L)\}$ is consistent, and it entails $B(L,L)$.

2.2.3. Cototal and total ideals of a closed base type. Let τ be a closed base type, for simplicity without parameters. An example is the type $\mathbb{Y}$ of binary trees. We want to take a closer look at the elements of $|\boldsymbol{A}_\tau|$, i.e., the ideals in $\boldsymbol{A}_\tau$. Among them it seems natural to single out those with the following property, to be called "cototality":

DEFINITION (Cototal ideal). Consider a token in the ideal, and in this token (a constructor expression) a position occupied by the symbol $*$ (indicating "no information"). Then it must be possible to further analyze the ideal, in the following sense. There must be another token in the ideal where this symbol $*$ is replaced by a constructor expression $\mathrm{C}\vec{*}$ with C the name of a constructor of the underlying base type.

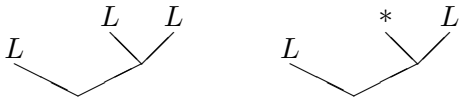
Clearly cototal ideals may be infinite. However, they can be analyzed (or "destructed") up to an arbitrary depth. It may also happen that a cototal ideal is finite. In this case it is called "total".

Hence in our model of partial continuous functionals already at base types we have ideals (i.e., objects) which are either

- cototal and infinite, or

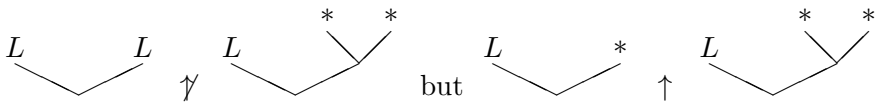
- total (i.e., cototal and finite), or
- neither.

We give some examples for the base type $\mathbb{Y}$, with a more pictorial representation. Tokens in $\boldsymbol{A}_{\mathbb{Y}}$ (omitting B):

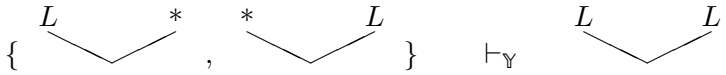


Consistency in $\boldsymbol{A}_{\mathbb{Y}}$:

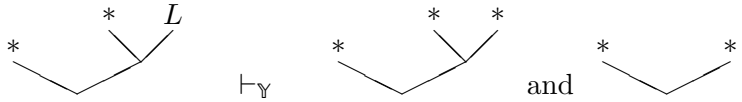
Moreover



Entailment in $\boldsymbol{A}_{\mathbb{Y}}$:

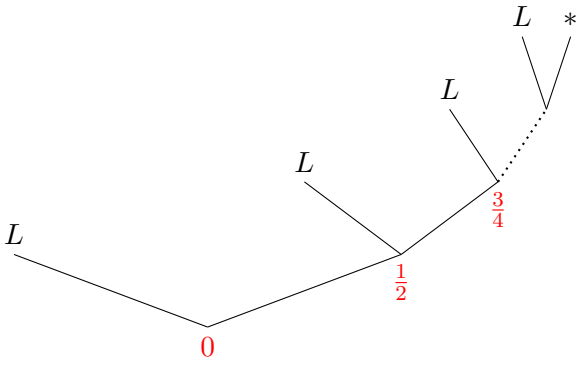


and also



Ideals in $\boldsymbol{A}_{\mathbb{Y}}$:

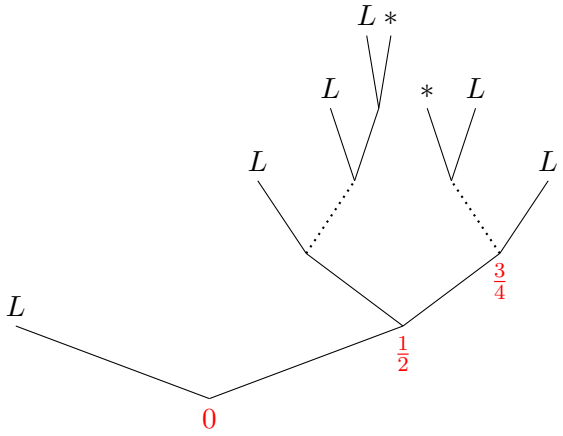
(i) $1 :=$ closure of all



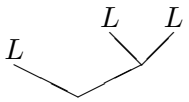
(ii) -1 is defined similarly

(iii) $-1 \cup 1$

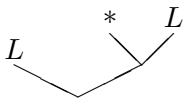
(iv) $\frac{1}{2}$:= closure of all



(v) Closure of



(vi) Closure of



Among these

(i) – (iv)	are infinite cototal ideals,
(v)	is a total ideal, and
(vi)	is an ideal but not cototal.

Cototality and totality of ideals in $\boldsymbol{A}_{\mathbb{Y}}$ can also be characterized differently. We can represent the set of total ideals in $\boldsymbol{A}_{\mathbb{Y}}$ as a predicate "inductively defined" by the two "clauses"

$$\begin{aligned}&(T_{\mathbb{Y}})^+_0 \colon L \in T_{\mathbb{Y}},\\ &(T_{\mathbb{Y}})^+_1 \colon \forall_{x_1,x_2}(x_1, x_2 \in T_{\mathbb{Y}} \to Bx_1x_2 \in T_{\mathbb{Y}})\end{aligned} \tag{1}$$

To state that $T_{\mathbb{Y}}$ is the *least* set satisfying the two clauses we use the *elimination* or *least fixed point* property

$$T^-_{\mathbb{Y}} \colon L \in X \to \forall_{x_1,x_2}(x_1, x_2 \in T_{\mathbb{Y}} \cap X \to Bx_1x_2 \in X) \to T_{\mathbb{Y}} \subseteq X. \tag{2}$$

It says that every "competitor" X satisfying the same clauses contains $T_{\mathbb{Y}}$.

We now want to represent the set of *cototal* binary trees by a "coinductively" defined predicate ${}^{\mathrm{co}}T_{\mathbb{Y}}$. Note that the conjunction of the two clauses of $T_{\mathbb{Y}}$ is equivalent to

$$\forall_x((x \equiv L) \vee \exists_{x_1,x_2}(x_1, x_2 \in T_{\mathbb{Y}} \wedge x \equiv Bx_1x_2) \to x \in T_{\mathbb{Y}}).$$

Since $T_{\mathbb{Y}}$ is the least predicate with this property we even have the equivalence

$$\forall_x(x \in T_{\mathbb{Y}} \leftrightarrow (x \equiv L) \vee \exists_{x_1,x_2}(x_1, x_2 \in T_{\mathbb{Y}} \wedge x \equiv Bx_1x_2))$$

(called inversion property). How can we formally represent cototality of a binary tree? The idea is to define ${}^{\mathrm{co}}T_{\mathbb{Y}}$ as the *largest* set satisfying the equivalence. Formulated differently, cototal ideals are not built from initial objects by construction (synthesized), but rather defined by the property that they can always be destructed (analysed). Therefore we require the "closure property"

$$(3) \quad {}^{\mathrm{co}}T^{-}_{\mathbb{Y}} : \forall_x(x \in {}^{\mathrm{co}}T_{\mathbb{Y}} \to (x \equiv L) \vee \exists_{x_1,x_2}(x_1, x_2 \in {}^{\mathrm{co}}T_{\mathbb{Y}} \wedge x \equiv Bx_1x_2)).$$

A set built by construction steps (synthesis) is meant to be the least set closed under these steps. Similary, a set described by destruction (analysis) is meant to be the largest set closed under destruction. Hence we require the "greatest fixed point" property

$$(4) \quad {}^{\mathrm{co}}T^{+}_{\mathbb{Y}} : \forall_x(x \in X \to (x \equiv L) \vee \exists_{x_1,x_2}(x_1, x_2 \in {}^{\mathrm{co}}T_{\mathbb{Y}} \cup X \wedge x \equiv Bx_1x_2)) \to X \subseteq {}^{\mathrm{co}}T_{\mathbb{Y}}.$$

${}^{\mathrm{co}}T^{+}_{\mathbb{Y}}$ expresses that every competitor X satisfying the closure property is below ${}^{\mathrm{co}}T_{\mathbb{Y}}$.

We also consider binary versions of $T_{\mathbb{Y}}$ and ${}^{\mathrm{co}}T_{\mathbb{Y}}$, called *similarity* $\sim_{\mathbb{Y}}$ and *bisimilarity* $\approx_{\mathbb{Y}}$. The clauses for $\sim_{\mathbb{Y}}$ are

$$(5) \quad \begin{array}{l} (\sim_{\mathbb{Y}})^{+}_0 : L \sim_{\mathbb{Y}} L, \\ (\sim_{\mathbb{Y}})^{+}_1 : \forall_{x_1,x_1'}(x_1 \sim_{\mathbb{Y}} x_1' \to \forall_{x_2,x_2'}(x_2 \sim_{\mathbb{Y}} x_2' \to Bx_1x_2 \sim_{\mathbb{Y}} Bx_1'x_2')) \end{array}$$

and the elimination or least-fixed-point property is

$$(6) \quad \begin{array}{l} \sim^{-}_{\mathbb{Y}} : X(L,L) \to \forall_{x_1,x_2}((x_1 \sim_{\mathbb{Y}} x_2 \wedge Xx_1x_2) \to \\ \qquad\qquad \forall_{x_1',x_2'}((x_1' \sim_{\mathbb{Y}} x_2' \wedge Xx_1'x_2') \to \\ \qquad\qquad X(Bx_1x_2, Bx_1'x_2'))) \to \\ \qquad \sim_{\mathbb{Y}} \subseteq X. \end{array}$$

The elimination (or closure) property for $\approx_{\mathbb{Y}}$ is

$$\begin{aligned}\approx^{-}_{\mathbb{Y}}\colon \forall_{x,x'}(x \approx_{\mathbb{Y}} x' \to ((x \equiv L) \land (x' \equiv L)) \lor \\ \exists_{x_1,x_2,x_1',x_2'}(x_1 \approx_{\mathbb{Y}} x_1' \land x_2 \approx_{\mathbb{Y}} x_2' \land \\ x \equiv Bx_1x_2 \land x' \equiv Bx_1'x_2'))\end{aligned} \tag{7}$$

and the introduction (or greatest-fixed-point or coinduction) property is

(8)

$$\begin{aligned}\approx^{+}_{\mathbb{Y}}\colon \forall_{x,x'}(Xxx' \to ((x \equiv L) \land (x' \equiv L)) \lor \\ \exists_{x_1,x_2,x_1',x_2'}((x_1 \approx_{\mathbb{Y}} x_1' \lor Xx_1x_1') \land (x_2 \approx_{\mathbb{Y}} x_2' \lor Xx_2x_2') \land \\ x \equiv Bx_1x_2 \land x' \equiv Bx_1'x_2')) \to \\ X \subseteq \approx_{\mathbb{Y}}.\end{aligned}$$

We show that $x \approx_{\mathbb{Y}} x'$ implies $x \equiv x'$. Generally we have

Lemma 2.2.1 (Bisimilarity). *For every closed base type bisimilarity implies equality.*

Proof. As an example we prove this for $\mathbb{Y}$. Let a range over tokens for $\mathbb{Y}$. By induction on the height $|a^*|$ of extended tokens a^* we prove that for all ideals x, x' and extended tokens a^* from $a^* \in x$ we can infer $a^* \in x'$. It suffices to consider the case $Ba_1^*a_2^*$. From $x \approx_{\mathbb{Y}} x'$ we obtain by the closure property x_1, x_2, x_1', x_2' with

$$x_1 \approx x_1' \land x_2 \approx x_2' \land x \equiv Bx_1x_2 \land x' \equiv Bx_1'x_2'.$$

Then $a_i^* \in x_i$ (for $i = 1, 2$), and by IH $a_i^* \in x_i'$. Thus $Ba_1^*a_2^* \in x'$. □

From the Bisimilarity Lemma we obtain

Proposition 2.2.2 (Characterization of $\approx_{\mathbb{Y}}$).

$$x \approx_{\mathbb{Y}} x' \leftrightarrow x, x' \in {}^{\mathrm{co}}T_{\mathbb{Y}} \land x \equiv x'.$$

Proof. "$\to$". By Lemma 2.2.1 it remains to prove $x \approx_{\mathbb{Y}} x' \to x \in {}^{\mathrm{co}}T_{\mathbb{Y}}$. To this end we apply ${}^{\mathrm{co}}T^{+}_{\mathbb{Y}}$ with competitor $X := \{\, x \mid \exists_{x'}(x \approx_{\mathbb{Y}} x') \,\}$. It suffices to prove the premise. Fix x, x' with $x \approx_{\mathbb{Y}} x'$. The goal is

$$\begin{aligned}(x \equiv L) \lor \exists_{x_1,x_2}((x_1 \in {}^{\mathrm{co}}T_{\mathbb{Y}} \lor \exists_{x_1'}(x_1 \approx x_1')) \land \\ (x_2 \in {}^{\mathrm{co}}T_{\mathbb{Y}} \lor \exists_{x_2'}(x_2 \approx x_2')) \land x \equiv Bx_1x_2).\end{aligned}$$

By the closure property $\approx^{-}_{\mathbb{Y}}$ we have

$$(x \equiv L \land x' \equiv L) \lor \exists_{x_1,x_2,x_1',x_2'}(x_1 \approx x_1' \land x_2 \approx x_2' \land x \equiv Bx_1x_2 \land x' \equiv Bx_1'x_2')).$$

In the first case we have $x \equiv L$ and are done. In the second case we have x_1, x_2, x_1', x_2' with $x_1 \approx x_1'$, $x_2 \approx x_2'$ and $x \equiv Bx_1x_2$, and are done as well.

"←". We prove $x \in {}^{\mathrm{co}}T_{\mathbb{Y}} \to x \equiv x' \to x \approx_{\mathbb{Y}} x'$ by the greatest-fixed-point property $\approx^{+}_{\mathbb{Y}}$ with competitor $X := \{\, x, x' \mid x \in {}^{\mathrm{co}}T_{\mathbb{Y}} \land x \equiv x' \,\}$. It suffices to prove the premise. Fix x, x' with $x \in {}^{\mathrm{co}}T_{\mathbb{Y}} \land x \equiv x'$. The goal is

$$\begin{aligned}(x \equiv L \land x' \equiv L) \lor \exists_{x_1,x_2,x_1',x_2'}((x_1 \approx x_1' \lor (x_1 \in {}^{\mathrm{co}}T_{\mathbb{Y}} \land x_1 \equiv x_1')) \land \\ (x_2 \approx x_2' \lor (x_2 \in {}^{\mathrm{co}}T_{\mathbb{Y}} \land x_2 \equiv x_2')) \land \\ x \equiv Bx_1x_2 \land x' \equiv Bx_1'x_2')).\end{aligned}$$

By the closure property ${}^{\mathrm{co}}T^{-}_{\mathbb{Y}}$ applied to $x \in {}^{\mathrm{co}}T_{\mathbb{Y}}$ we have

$$(x \equiv L) \lor \exists_{x_1,x_2}(x_1 \in {}^{\mathrm{co}}T_{\mathbb{Y}} \land x_2 \in {}^{\mathrm{co}}T_{\mathbb{Y}} \land x \equiv Bx_1x_2).$$

In the first case we have $x \equiv L$ and are done, since $x \equiv x'$. In the second case we have $x_1, x_2 \in {}^{\mathrm{co}}T_{\mathbb{Y}}$ with $x \equiv Bx_1x_2$. Then we are done as well with $x_1' := x_1$ and $x_2' := x_2$, since again $x \equiv x'$. □

2.2.4. Constructors as continuous functions. Let ι be a closed base type. Every constructor C generates the following ideal in the function space determined by the type of the constructor:

$$r_{\mathrm{C}} := \{\, (\vec{U}, \mathrm{C}\vec{a^*}) \mid \vec{U} \vdash \vec{a^*} \,\},$$

where $(\vec{U}, a)$ abbreviates $(U_1, (U_2, \dots (U_n, a) \dots))$.

According to the general definition of a continuous function associated to an ideal in a function space the continuous map $|r_{\mathrm{C}}|$ satisfies

$$|r_{\mathrm{C}}|(\vec{x}) = \{\, \mathrm{C}\vec{a^*} \mid \exists_{\vec{U} \subseteq \vec{x}}(\vec{U} \vdash \vec{a^*}) \,\}.$$

(For $\mathbb{N}$ we have $|r_S|(\{0\}) = \{S0, S*\}$ and $|r_S|(\{S0, S*\}) = \{SS0, SS*, S*\}$.) An immediate consequence is that the (continuous maps corresponding to) constructors are injective and their ranges are disjoint.

Lemma 2.2.3 (Constructors are injective and have disjoint ranges). *Let ι be a closed base type and* C *be a constructor of ι. Then*

$$|r_{\mathrm{C}}|(\vec{x}) \subseteq |r_{\mathrm{C}}|(\vec{y}) \leftrightarrow \vec{x} \subseteq \vec{y}.$$

If $\mathrm{C}_1, \mathrm{C}_2$ are distinct constructors of ι, then $|r_{\mathrm{C}_1}|(\vec{x}) \neq |r_{\mathrm{C}_2}|(\vec{y})$, since the two ideals are non-empty and disjoint.

Proof. Immediate from the definitions. □

Lemma (Ideals of a closed base type). *Every non-empty ideal in the information system associated to a closed base type has the form $|r_{\mathrm{C}}|(\vec{x})$ with a constructor* C *and ideals $\vec{x}$.*

Proof. Let z be a non-empty ideal and $\mathrm{C}a_0^*b_0^* \in z$, where for simplicity we assume that C is a binary constructor. Let $x := \{\, a \mid \mathrm{C}a* \in z \,\}$ and $y := \{\, b \mid \mathrm{C}*b \in z \,\}$; clearly x, y are ideals. We claim that $z = |r_{\mathrm{C}}|(x, y)$.

For $\supseteq$ consider $\mathrm{C}a^*b^*$ with $a^* \in x \cup \{*\}$ and $b^* \in y \cup \{*\}$. Then by definition $\{\mathrm{C}a^**, \mathrm{C}*b^*\} \subseteq z$, hence $\mathrm{C}a^*b^* \in z$ by deductive closure. Conversely, notice that an arbitrary element of z must have the form $\mathrm{C}a^*b^*$, because of consistency. Then $\{\mathrm{C}a^**, \mathrm{C}*b^*\} \subseteq z$ again by deductive closure. Hence $a^* \in x \cup \{*\}$ and $b^* \in y \cup \{*\}$, and therefore $\mathrm{C}a^*b^* \in |r_{\mathrm{C}}|(x, y)$. □

It is in this proof that we need entailment to be a relation between finite sets of tokens and single tokens, not just a binary relation between tokens. Information systems with the latter property are called *atomic*.

The information systems $\boldsymbol{C}_\tau$ enjoy the pleasant property of *coherence*, which amounts to the possibility to locate inconsistencies in two-element sets of data objects. Generally, an information system $\boldsymbol{A} = (A, \mathrm{Con}, \vdash)$ is *coherent* if it satisfies: $U \subseteq A$ is consistent if and only if all of its two-element subsets are.

LEMMA 2.2.4. *Let $\boldsymbol{A}$ and $\boldsymbol{B}$ be information systems. If $\boldsymbol{B}$ is coherent, then so is $\boldsymbol{A} \to \boldsymbol{B}$.*

PROOF. Let $\boldsymbol{A} = (A, \mathrm{Con}_A, \vdash_A)$ and $\boldsymbol{B} = (B, \mathrm{Con}_B, \vdash_B)$ be information systems, and consider $\{(U_1, b_1), \dots, (U_n, b_n)\} \subseteq \mathrm{Con}_A \times B$. Assume

$$\forall_{1 \le i < j \le n}(\{(U_i, b_i), (U_j, b_j)\} \in \mathrm{Con}).$$

We have to show $\{(U_1, b_1), \dots, (U_n, b_n)\} \in \mathrm{Con}$. Let $I \subseteq \{1, \dots, n\}$ and $\bigcup_{i \in I} U_i \in \mathrm{Con}_A$. We must show $\{\, b_i \mid i \in I \,\} \in \mathrm{Con}_B$. Now since $\boldsymbol{B}$ is coherent by assumption, it suffices to show that $\{b_i, b_j\} \in \mathrm{Con}_B$ for all $i, j \in I$. So let $i, j \in I$. By assumption we have $U_i \cup U_j \in \mathrm{Con}_A$, and hence $\{b_i, b_j\} \in \mathrm{Con}_B$. □

2.3. Terms

We now set up a term language to denote partial continuous functionals. It can be seen as a as a common extension of Gödel's T (1958) and Plotkin's PCF (1977); we call it T^+.

2.3.1. A common extension T^+ of Gödel's T and Plotkin's PCF.

DEFINITION (Terms). *Terms* of T^+ are built from (typed) variables and (typed) constants (constructors C or defined constants D; see the definition below) by application and abstraction:

$$M, N ::= x^\tau \mid \mathrm{C}^\tau \mid D^\tau \mid (\lambda_{x^\tau} M^\sigma)^{\tau \to \sigma} \mid (M^{\tau \to \sigma} N^\tau)^\sigma.$$

The set $\mathrm{FV}(M)$ of free variables of a term M is defined by

$$\mathrm{FV}(x) := \{x\}, \qquad \mathrm{FV}(C), \mathrm{FV}(D) := \emptyset,$$
$$\mathrm{FV}(\lambda_x M) := \mathrm{FV}(M) \setminus \{x\}, \qquad \mathrm{FV}(MN) := \mathrm{FV}(M) \cup \mathrm{FV}(N).$$

DEFINITION (Conversion). We define a conversion relation $\mapsto_\beta$ for terms similarly to what we did for derivation terms:

$$(\lambda_x M(x))^{\tau\to\sigma} N^\tau \mapsto_\beta M(N)^\sigma.$$

In addition we will employ another conversion relation $\mapsto_\eta$ defined by

$$\lambda_x(Mx) \mapsto_\eta M \quad \text{if } x \notin \mathrm{FV}(M), \text{ and } M \text{ is not an abstraction.}$$

DEFINITION (Computation rule). Every defined constant D comes with a system of *computation rules*, consisting of finitely many equations

$$D\vec{P}_i(\vec{y}_i) := M_i \qquad (i = 1, \dots, n \text{ where } n \geq 0) \tag{9}$$

with free variables of $\vec{P}_i(\vec{y}_i)$ and M_i among $\vec{y}_i$, where the arguments on the left hand side must be "constructor patterns", i.e., lists of applicative terms built from constructors and distinct variables. To ensure consistency of the defining equations, we require that for $i \neq j$ $\vec{P}_i$ and $\vec{P}_j$ have disjoint free variables, and either $\vec{P}_i$ and $\vec{P}_j$ are non-unifiable[2] (i.e., there is no substitution which identifies them), or else for the "most general unifier" ϑ of $\vec{P}_i$ and $\vec{P}_j$ we have $M_i\vartheta = M_j\vartheta$. Notice that the substitution ϑ assigns to the variables $\vec{y}_i$ in M_i constructor patterns $\vec{R}_k(\vec{z})$ $(k = i, j)$. A further requirement on a system of computation rules $D\vec{P}_i(\vec{y}_i) := M_i$ is that the lengths of all $\vec{P}_i(\vec{y}_i)$ are the same; this number is called the *arity* of D, denoted by $\mathrm{ar}(D)$. A substitution instance of a left hand side of (9) is called a *D-redex*.

More formally, constructor patterns are defined inductively by (we write $\vec{P}(\vec{x}\,)$ to indicate all variables in $\vec{P}$):

(a) x is a constructor pattern.
(b) The empty list is a constructor pattern.
(c) If $\vec{P}(\vec{x}\,)$ and $Q(\vec{y}\,)$ are constructor patterns whose variables $\vec{x}$ and $\vec{y}$ are disjoint, then $(\vec{P}, Q)(\vec{x}, \vec{y}\,)$ is a constructor pattern.
(d) If C is a constructor and $\vec{P}$ a constructor pattern, then so is $\mathrm{C}\vec{P}$.

REMARK. The requirement of disjoint variables in constructor patterns $\vec{P}_i$ and $\vec{P}_j$ used in computation rules of a defined constant D is needed to ensure that applying the most general unifier produces constructor patterns again. However, for readability we take this as an implicit convention, and write computation rules with possibly non-disjoint variables.

Examples of constants D defined by computation rules are abundant. In particular, the (structural) recursion and corecursion operators will be defined by computation rules.

[2]A detailed treatment of unification is in Appendix A

The simplest example is the constant $\bot_\iota$ of type ι with no computation rules. The boolean connectives andb, impb and orb are defined by

$$\begin{array}{lll} \mathtt{tt}\ \mathrm{andb}\ y := y, & \mathtt{ff}\ \mathrm{impb}\ y := \mathtt{tt}, & \mathtt{tt}\ \mathrm{orb}\ y := \mathtt{tt}, \\ x\ \mathrm{andb}\ \mathtt{tt} := x, & \mathtt{tt}\ \mathrm{impb}\ y := y, & x\ \mathrm{orb}\ \mathtt{tt} := \mathtt{tt}, \\ \mathtt{ff}\ \mathrm{andb}\ y := \mathtt{ff}, & x\ \mathrm{impb}\ \mathtt{tt} := \mathtt{tt}, & \mathtt{ff}\ \mathrm{orb}\ y := y, \\ x\ \mathrm{andb}\ \mathtt{ff} := \mathtt{ff}, & & x\ \mathrm{orb}\ \mathtt{ff} := x. \end{array}$$

Notice that when two such rules overlap, their right hand sides are equal under any unifier of the left hand sides.

Decidable *equality* $=_\iota \colon \iota \to \iota \to \mathbb{B}$ for a closed base type ι can be defined easily by computation rules. For example,

$$\begin{array}{ll} (0 =_{\mathbb{N}} 0) := \mathtt{tt}, & (Sn =_{\mathbb{N}} 0) := \mathtt{ff}, \\ (0 =_{\mathbb{N}} Sm) := \mathtt{ff}, & (Sn =_{\mathbb{N}} Sm) :=\ (n =_{\mathbb{N}} m) \end{array}$$

and similarly for $\mathbb{Y}$ with constructors L (leaf) and B (branch)

$$\begin{array}{ll} (L =_{\mathbb{Y}} L) := \mathtt{tt}, & (Bxy =_{\mathbb{Y}} L) := \mathtt{ff}, \\ (L =_{\mathbb{Y}} Bxy) := \mathtt{ff}, & (Bxy =_{\mathbb{Y}} Bx'y') :=\ (x =_{\mathbb{Y}} x'\ \mathrm{andb}\ y =_{\mathbb{Y}} y'). \end{array}$$

For $\mathbb{L}(\iota)$ with ι a closed base type we define $=_{\mathbb{L}(\iota)} \colon \mathbb{L}(\iota) \to \mathbb{L}(\iota) \to \mathbb{B}$ by

$$\begin{array}{ll} ([] =_{\mathbb{L}(\iota)} []) := \mathtt{tt}, & (a{::}\ell =_{\mathbb{L}(\iota)} []) := \mathtt{ff}, \\ ([] =_{\mathbb{L}(\iota)} a{::}\ell) := \mathtt{ff}, & (a{::}\ell =_{\mathbb{L}(\iota)} a'{::}\ell') :=\ (a =_\iota a'\ \mathrm{andb}\ \ell =_{\mathbb{L}(\iota)} \ell'). \end{array}$$

For the base type $\mathbb{N}$ of natural numbers we have the doubling function

$$\begin{array}{ll} \mathrm{Double}(0) & := 0, \\ \mathrm{Double}(S(n)) & := S(S(\mathrm{Double}(n))). \end{array}$$

Addition (written infix) is defined similarly, this time with a parameter m:

$$\begin{array}{ll} m + 0 & := m, \\ m + S(n) & := S(m + n). \end{array}$$

Multiplication (again written infix) is defined using addition by

$$\begin{array}{ll} m \cdot 0 & := 0, \\ m \cdot S(n) & := (m \cdot n) + m. \end{array}$$

Similarly we can define all primitive recursive functions.

Up to now we have mainly considered examples of total functions, in the sense that total arguments are mapped to total values. But recall that in our setting functions need not be total. To give an example consider the closed base type $\mathbb{S}(\mathbb{D})$ of streams defined by the single constructor type

$$\mathrm{SCons} \colon \mathbb{D} \to \mathbb{S}(\mathbb{D}) \to \mathbb{S}(\mathbb{D}).$$

We write $\mathrm{C}_d u$ or $d :: u$ for $\mathrm{SCons}(d, u)$. This base type differs from the ones previously considered by not having a nullary constructor. As a consequence it does not have non-empty total ideals, but clearly cototal ones. An example is $\{\, \mathrm{C}_d^n(*) \mid n \geq 1 \,\}$.

We define Map of type $(\tau \to \sigma) \to \mathbb{S}(\tau) \to \mathbb{S}(\sigma)$ mapping its function argument $h\colon \tau \to \sigma$ over a stream u of type $\mathbb{S}(\tau)$ by the computation rule

$$\mathrm{Map}_h(a :: u) := (ha) :: \mathrm{Map}_h(u).$$

As an example of how to define standard arithmetical functions in T^+ we consider the quotient-and-remainder function qr of type $\mathbb{N} \to \mathbb{N} \to \mathbb{N} \times \mathbb{N}$. Its defining equarions are

$$\begin{aligned} \mathrm{qr}(0, m) &:= (0, 0), \\ \mathrm{qr}(n+1, m) &:= \begin{cases} (q, r+1) & \text{if } r+1 < m, \\ (q+1, 0) & \text{else} \end{cases} \qquad \text{with } (q, r) := \mathrm{qr}(n, m). \end{aligned}$$

Lemma 2.3.1 (NatQRProp). *Given n, m with $0 < m$. Let $(q, r) := \mathrm{qr}(n, m)$. Then $n = q * m + r$ and $r < m$.*

Proof. By induction on n. The base case is clear. In the step we distinguish cases. *Case* $r + 1 < m$. Then $\mathrm{qr}(n+1, m) = (q, r+1)$. The claim follows by induction hypothesis. *Case* $m \leq r + 1$. Then $m = r + 1$ since $r < m$ (by induction hypothesis) and $m \leq r + 1$ (by case assumption). By definition we have $\mathrm{qr}(n+1, m) = (q+1, 0)$. We obtain

$$n + 1 = q * m + r + 1 = q * m + m = (q + 1) * m + 0. \qquad \square$$

2.3.2. Recursion operators. Important examples of such constants D are the (structural) higher type *recursion operators* $\mathcal{R}_\iota^\tau$ introduced by Hilbert (1925) and Gödel (1958). They are used to construct maps from the base type ι to the value type τ, by recursion on the structure of ι.

For instance, $\mathcal{R}_{\mathbb{N}}^\tau$ has type $\mathbb{N} \to \tau \to (\mathbb{N} \to \tau \to \tau) \to \tau$. It is defined by the computation rules

$$\begin{aligned} \mathcal{R}_{\mathbb{N}}^\tau 0 a f &:= a, \\ \mathcal{R}_{\mathbb{N}}^\tau (Sn) a f &:= f n (\mathcal{R}_{\mathbb{N}}^\tau n a f). \end{aligned}$$

The first argument is the recursion argument, the second one gives the base value, and the third one gives the step function, mapping the recursion argument and the previous value to the next value. For example, $\mathcal{R}_{\mathbb{N}}^{\mathbb{N}} n m \lambda_{n,l}(Sl)$ defines addition $m + n$ by recursion on n. For $\lambda_{n,l}(Sl)$ we often write $\lambda_{\_,l}(Sl)$ since the bound variable n is not used.

It will be convenient to write a list of constructor types

$$(\vec{\alpha}_i \to (\xi)_{\nu < n_i} \to \xi)_{i<k} \quad \text{as} \quad ((\rho_{i\nu}(\xi))_{\nu < n_i} \to \xi)_{i<k}.$$

DEFINITION (Type of $\mathcal{R}^\tau_\iota$). Let a base type ι be given by a list of constructor types $((\rho_{i\nu}(\xi))_{\nu<n_i} \to \xi)_{i<k}$. Let τ be a type. We define the type of the recursion operator $\mathcal{R}^\tau_\iota$ to be

$$\iota \to ((\rho_{i\nu}(\iota \times \tau))_{\nu<n_i} \to \tau)_{i<k} \to \tau.$$

Here ι is the type of the recursion argument, and each $(\rho_{i\nu}(\iota\times\tau))_{\nu<n_i} \to \tau$ is called a *step type*. Usage of $\iota \times \tau$ rather than τ in the step types can be seen as a "strengthening", since then one has more data available to construct the value of type τ. Moreover, for recursive argument types we avoid the product type in $\iota \times \tau$ and take the two argument types ι and τ instead ("duplication").

DEFINITION (Computation rules for $\mathcal{R}^\tau_\iota$). Let

$$\alpha_0 \to \ldots \to \alpha_{m-1} \to (\xi)_{i<n} \to \xi$$

be the type of the i-th constructor C_i of ι and consider a term $\mathrm{C}_i\vec{x}$ of type ι. We write $\vec{x}^P = x^P_0, \ldots, x^P_{m-1}$ for the *parameter arguments* $x^{\alpha_0}_0, \ldots, x^{\alpha_{m-1}}_{m-1}$ and $\vec{x}^R = x^R_0, \ldots, x^R_{n-1}$ for the *recursive arguments* $x^\iota_m, \ldots, x^\iota_{m+n-1}$. Writing $\mathcal{R}$ for $\mathcal{R}^\tau_\iota$ we take as its computation rules

$$\mathcal{R}(\mathrm{C}_i\vec{x})\vec{f} := f_i\vec{x}(\mathcal{R}x^R_0\vec{f}\,)\ldots(\mathcal{R}x^R_{n-1}\vec{f}\,).$$

EXAMPLES.

$$\begin{aligned}
&\mathcal{R}^\tau_{\mathbb{B}}\colon \mathbb{B} \to \tau \to \tau \to \tau,\\
&\mathcal{R}^\tau_{\mathbb{N}}\colon \mathbb{N} \to \tau \to (\mathbb{N} \to \tau \to \tau) \to \tau,\\
&\mathcal{R}^\tau_{\mathbb{P}}\colon \mathbb{P} \to \tau \to (\mathbb{P} \to \tau \to \tau) \to (\mathbb{P} \to \tau \to \tau) \to \tau,\\
&\mathcal{R}^\tau_{\mathbb{Y}}\colon \mathbb{Y} \to \tau \to (\mathbb{Y} \to \tau \to \mathbb{Y} \to \tau \to \tau) \to \tau,\\
&\mathcal{R}^\tau_{\mathbb{L}(\rho)}\colon \mathbb{L}(\rho) \to \tau \to (\rho \to \mathbb{L}(\rho) \to \tau \to \tau) \to \tau,\\
&\mathcal{R}^\tau_{\rho+\sigma}\colon \rho + \sigma \to (\rho \to \tau) \to (\sigma \to \tau) \to \tau,\\
&\mathcal{R}^\tau_{\rho\times\sigma}\colon \rho \times \sigma \to (\rho \to \sigma \to \tau) \to \tau.
\end{aligned}$$

It is a helpful exercise to write out the computation rules for these particular recursion operators.

There is an important variant of recursion, where no recursive calls occur. This variant is called the *cases operator*; it distinguishes cases according to the outer constructor form. For a base type ι given by a list of constructor types $((\rho_{i\nu}(\xi))_{\nu<n_i} \to \xi)_{i<k}$ and a result type τ the type of the cases operator $\mathcal{C}^\tau_\iota$ is

$$\iota \to ((\rho_{i\nu}(\iota))_{\nu<n_i} \to \tau)_{i<k} \to \tau.$$

The simplest example (for type $\mathbb{B}$) is *if-then-else*. Another example is

$$\mathcal{C}^\tau_{\mathbb{N}}\colon \mathbb{N} \to \tau \to (\mathbb{N} \to \tau) \to \tau.$$

It could be used to define the *predecessor* function on $\mathbb{N}$ by the term

$$\mathrm{P}m := \mathcal{C}^{\mathbb{N}}_{\mathbb{N}} m 0(\lambda_n n).$$

However, it is easier to define the predecessor function by the computation rules $\mathrm{P}0 := 0$ and $\mathrm{P}(Sn) := n$.

REMARK. When computing the value of a cases term, we do not want to (eagerly) evaluate all arguments, but rather compute the test argument first and depending on the result (lazily) evaluate at most one of the other arguments. This phenomenon is well known in functional languages; for instance, in SCHEME the `if`-construct is called a *special form* (as opposed to an operator). Therefore instead of taking the cases operator applied to a full list of arguments, one rather uses a `case`-construct to build this term; it differs from the former only in that it employs lazy evaluation. Hence the predecessor function is $[\texttt{if}\ m\ 0\ \lambda_n n]$ (which is often written in the form $[\mathbf{case}\ m\ \mathbf{of}\ 0 \mid \lambda_n n]$).

General recursion with respect to a measure. In practice it often happens that one needs to recur to an argument which is not an immediate component of the present constructor object; this is not allowed in structural recursion. Of course, in order to ensure that the recursion terminates we have to assume that the recurrence is w.r.t. a given well-founded set; for simplicity we restrict ourselves to the algebra $\mathbb{N}$. However, we do allow that the recurrence is with respect to a measure function μ, with values in $\mathbb{N}$. The operator $\mathcal{F}$ of *general recursion* then is defined by

$$\mathcal{F}\mu x G = Gx(\lambda_y[\mathbf{if}\ \mu y < \mu x\ \mathbf{then}\ \mathcal{F}\mu y G\ \mathbf{else}\ \varepsilon]), \tag{10}$$

where ε denotes a canonical inhabitant of the range. We leave it as an exercise to prove that $\mathcal{F}$ is definable from an appropriate structural recursion operator.

As an example for the use of $\mathcal{F}$ we define a function NatToPos converting a natural number ≥ 1 written in unary (i.e., built from the constructors 0 and S) into the same natural number written in binary (i.e., built from the constructors 1, S_0 and S_1). This uses the auxiliary functions $\mathrm{Even}\colon \mathbb{N} \to \mathbb{B}$ defined by

$$\begin{aligned} \mathrm{Even}(0) &:= \mathtt{tt}, \\ \mathrm{Even}(S(0)) &:= \mathtt{ff}, \\ \mathrm{Even}(S(S(n))) &:= \mathrm{Even}(n) \end{aligned}$$

and $\mathrm{Half}\colon \mathbb{N} \to \mathbb{N}$ defined by

$$\begin{aligned} \mathrm{Half}(0) &:= 0, \\ \mathrm{Half}(S(0)) &:= 0, \\ \mathrm{Half}(S(S(n))) &:= S(\mathrm{Half}(n)). \end{aligned}$$

Then $\mathrm{NatToPos}\colon \mathbb{N} \to \mathbb{P}$ is defined by

$$\mathrm{NatToPos}(n) = \mathcal{F}(\mathrm{id}, n, G)$$

with $\mathrm{id}(n) = n$ and $G\colon \mathbb{N} \to (\mathbb{N} \to \mathbb{P}) \to \mathbb{P}$ defined by

$$G(n, f) = \begin{cases} S_0(f(\mathrm{Half}(n))) & \text{if } \mathrm{Even}(n), \\ 1 & \text{if } n = S(0), \\ S_1(f(\mathrm{Half}(n))) & \text{otherwise.} \end{cases}$$

2.3.3. Corecursion. The computation rules for $\mathcal{R}$ work from the leaves towards the root, and terminate because total ideals are finite. If, however, we deal with cototal ideals, then a similar operator is available to define functions with cototal ideals as values, namely "corecursion".

To understand the type of a corecursion operator let a base type ι be given by a list of constructor types

$$(\rho_{i\nu}(\iota))_{\nu<n_i} \to \iota \quad (i < k).$$

The product of these k constructor types is isomorphic to

$$\sum_{i<k} \prod_{\nu<n_i} \rho_{i\nu}(\iota) \to \iota$$

and the type of the recursion operator $\mathcal{R}^\tau_\iota$ is isomorphic to

$$\iota \to \Big(\sum_{i<k} \prod_{\nu<n_i} \rho_{i\nu}(\iota \times \tau) \to \tau\Big) \to \tau.$$

This way of subsuming the types of all constructors of a base type into a single type suggests the following definition of the *destructor* $\mathcal{D}_\iota$ of a base type ι.

Definition (Destructor). For a base type ι given by a list of constructor types $((\rho_{i\nu}(\xi))_{\nu<n_i} \to \xi)_{i<k}$ we define the type of the destructor $\mathcal{D}_\iota$ to be

$$\iota \to \sum_{i<k} \prod_{\nu<n_i} \rho_{i\nu}(\iota).$$

The computation rules for $\mathcal{D}_\iota$ disassemble a constructor-built pattern into its parts. For instance, the computation rules for $\mathcal{D}_\mathbb{N}$ of type $\mathbb{N} \to \mathrm{uysum}(\mathbb{N})$

(or $\mathbb{U}+\mathbb{N}$) are

$$\begin{aligned}\mathcal{D}_{\mathbb{N}}(0) &:= \mathrm{DummyL},\\ \mathcal{D}_{\mathbb{N}}(S(n)) &:= \mathrm{Inr}(n).\end{aligned}$$

The computation rules for $\mathcal{D}_{\mathbb{P}}$ of type $\mathbb{P}\to$ uysum$(\mathbb{P}+\mathbb{P})$ (or $\mathbb{U}+(\mathbb{P}+\mathbb{P})$) are

$$\begin{aligned}\mathcal{D}_{\mathbb{P}}(1) &:= \mathrm{DummyL},\\ \mathcal{D}_{\mathbb{P}}(S_0(p)) &:= \mathrm{Inr}(\mathrm{InL}^{\mathbb{P}\to\mathbb{P}+\mathbb{P}}(p)),\\ \mathcal{D}_{\mathbb{P}}(S_1(p)) &:= \mathrm{Inr}(\mathrm{InR}^{\mathbb{P}\to\mathbb{P}+\mathbb{P}}(p)).\end{aligned}$$

The corecursion operator ${}^{\mathrm{co}}\mathcal{R}^{\tau}_{\iota}$ is used to construct a map from τ to ι by "corecursion" on the structure of ι. Its type is

$$\tau\to\Big(\tau\to\sum_{i<k}\prod_{\nu<n_i}\rho_{i\nu}(\iota+\tau)\Big)\to\iota. \tag{11}$$

Examples (Types and computation rules of corecursion operators).

$$\begin{aligned}&{}^{\mathrm{co}}\mathcal{R}^{\tau}_{\mathbb{B}}\colon \tau\to(\tau\to\mathbb{U}+\mathbb{U})\to\mathbb{B},\\ &{}^{\mathrm{co}}\mathcal{R}^{\tau}_{\mathbb{N}}\colon \tau\to(\tau\to\mathbb{U}+(\mathbb{N}+\tau))\to\mathbb{N},\\ &{}^{\mathrm{co}}\mathcal{R}^{\tau}_{\mathbb{P}}\colon \tau\to(\tau\to\mathbb{U}+((\mathbb{P}+\tau)+(\mathbb{P}+\tau)))\to\mathbb{P},\\ &{}^{\mathrm{co}}\mathcal{R}^{\tau}_{\mathbb{Y}}\colon \tau\to(\tau\to\mathbb{U}+(\mathbb{Y}+\tau)\times(\mathbb{Y}+\tau))\to\mathbb{Y},\\ &{}^{\mathrm{co}}\mathcal{R}^{\tau}_{\mathbb{L}(\rho)}\colon \tau\to(\tau\to\mathbb{U}+\rho\times(\mathbb{L}(\rho)+\tau))\to\mathbb{L}(\rho),\\ &{}^{\mathrm{co}}\mathcal{R}^{\tau}_{\mathbb{S}(\rho)}\colon \tau\to(\tau\to\rho\times(\mathbb{S}(\rho)+\tau))\to\mathbb{S}(\rho).\end{aligned}$$

The computation rule for each of these is defined below. For $f\colon\rho\to\tau$ and $g\colon\sigma\to\tau$ we denote $\lambda_x(\mathcal{R}^{\tau}_{\rho+\sigma}xfg)$ of type $\rho+\sigma\to\tau$ by $[f,g]$, and similary for ternary sumtypes etcetera. The identity functions id below are of type $\iota\to\iota$ with ι the respective base type.

$$\begin{aligned}{}^{\mathrm{co}}\mathcal{R}^{\tau}_{\mathbb{B}}xf &:= [\lambda\_\,\mathtt{tt},\lambda\_\,\mathtt{ff}](fx),\\ {}^{\mathrm{co}}\mathcal{R}^{\tau}_{\mathbb{N}}xf &:= [\lambda\_\,0,\lambda_y(S([\mathrm{id}^{\mathbb{N}\to\mathbb{N}},P_{\mathbb{N}}]y))](fx),\\ {}^{\mathrm{co}}\mathcal{R}^{\tau}_{\mathbb{P}}xf &:= [\lambda\_\,1,\lambda_y(S_0([\mathrm{id},P_{\mathbb{P}}]y)),\lambda_y(S_1([\mathrm{id},P_{\mathbb{P}}]y))](fx),\\ {}^{\mathrm{co}}\mathcal{R}^{\tau}_{\mathbb{Y}}xf &:= [\lambda\_L,\lambda_{y_0,y_1}(B([\mathrm{id},P_{\mathbb{Y}}]y_0)([\mathrm{id},P_{\mathbb{Y}}]y_1))](fx),\\ {}^{\mathrm{co}}\mathcal{R}^{\tau}_{\mathbb{L}(\rho)}xf &:= [\lambda\_[],\lambda_{y_0,y_1}(y_0::[\mathrm{id},P_{\mathbb{L}(\rho)}]y_1)](fx),\\ {}^{\mathrm{co}}\mathcal{R}^{\tau}_{\mathbb{S}(\rho)}xf &:= (fx)_0::[\mathrm{id},P_{\mathbb{S}(\rho)}](fx)_1\end{aligned}$$

with $(fx)_i$ the i-th component of the pair fx, and $P_\alpha := \lambda_x({}^{\mathrm{co}}\mathcal{R}^{\tau}_{\alpha}xf)$ for $\alpha\in\{\mathbb{N},\mathbb{P},\mathbb{Y},\mathbb{L}(\rho),\mathbb{S}(\rho)\}$.

DEFINITION. The (single) computation rule for ${}^{\mathrm{co}}\mathcal{R}^\tau_\iota$ of type (11) is

$$ {}^{\mathrm{co}}\mathcal{R}^\tau_\iota x f := [g_0, \dots, g_{k-1}](fx) $$

where g_i of type $\prod_{\nu<n_i} \rho_{i\nu}(\iota+\tau) \to \iota$ is defined as

$$ g_i := \lambda_{\vec{x}}(\mathrm{C}_i(N_\nu)_{\nu<n_i}) \qquad \text{with } x_\nu \colon \rho_{i\nu}(\iota+\tau), $$

$$ N_\nu := \begin{cases} x_\nu & \text{if } \rho_{i\nu}(\xi) \text{ is a parameter arg. type,} \\ [\mathrm{id}^{\iota\to\iota}, P^{\tau\to\iota}]x_\nu^{\iota+\tau} & \text{otherwise,} \end{cases} $$

and $P := \lambda_x({}^{\mathrm{co}}\mathcal{R}^\tau_\iota x f)$ contains the corecursive call.

REMARK. It can be difficult to read the computation rules for corecursion operators. However, it helps if we know some properties of the "step" function f. For instance we have

$$ {}^{\mathrm{co}}\mathcal{R}^\tau_{\mathbb{N}} x f = \begin{cases} 0 & \text{if } fx = \mathrm{DummyL}^{\mathbb{U}+(\mathbb{N}+\tau)} \\ Sn & \text{if } fx = \mathrm{Inr}(\mathrm{InL}^{\mathbb{N}\to\mathbb{N}+\tau} n) \\ S({}^{\mathrm{co}}\mathcal{R}^\tau_{\mathbb{N}} x' f) & \text{if } fx = \mathrm{Inr}(\mathrm{InR}^{\tau\to\mathbb{N}+\tau} x') \end{cases} $$

$$ {}^{\mathrm{co}}\mathcal{R}^\tau_{\mathbb{Y}} x f = \begin{cases} L & \text{if } fx = \mathrm{DummyL}^{\mathbb{U}+(\mathbb{Y}+\tau)\times(\mathbb{Y}+\tau)} \\ Buv & \text{if } fx = \mathrm{Inr}\langle \mathrm{InL}^{\mathbb{Y}\to\mathbb{Y}+\tau} u, \mathrm{InL}^{\mathbb{Y}\to\mathbb{Y}+\tau} v\rangle \\ Bu({}^{\mathrm{co}}\mathcal{R}^\tau_{\mathbb{Y}} y f) & \text{if } fx = \mathrm{Inr}\langle \mathrm{InL}^{\mathbb{Y}\to\mathbb{Y}+\tau} u, \mathrm{InR}^{\tau\to\mathbb{Y}+\tau} y\rangle \\ B({}^{\mathrm{co}}\mathcal{R}^\tau_{\mathbb{Y}} y f)v & \text{if } fx = \mathrm{Inr}\langle \mathrm{InR}^{\tau\to\mathbb{Y}+\tau} y, \mathrm{InL}^{\mathbb{Y}\to\mathbb{Y}+\tau} v\rangle \\ B({}^{\mathrm{co}}\mathcal{R}^\tau_{\mathbb{Y}} y f)({}^{\mathrm{co}}\mathcal{R}^\tau_{\mathbb{Y}} z f) & \text{if } fx = \mathrm{Inr}\langle \mathrm{InR}^{\tau\to\mathbb{Y}+\tau} y, \mathrm{InR}^{\tau\to\mathbb{Y}+\tau} z\rangle \end{cases} $$

$$ {}^{\mathrm{co}}\mathcal{R}^\tau_{\mathbb{S}(\rho)} x f = \begin{cases} a :: u & \text{if } fx = \langle a, \mathrm{InL}^{\mathbb{S}(\rho)\to\mathbb{S}(\rho)+\tau} u\rangle \\ a :: {}^{\mathrm{co}}\mathcal{R}^\tau_{\mathbb{S}(\rho)} x' f & \text{if } fx = \langle a, \mathrm{InR}^{\tau\to\mathbb{S}(\rho)+\tau} x'\rangle. \end{cases} $$

Recall that Map of type $(\tau \to \sigma) \to \mathbb{S}(\tau) \to \mathbb{S}(\sigma)$ maps its function argument $h \colon \tau \to \sigma$ over a stream u of type $\mathbb{S}(\tau)$ (see Section 2.3.1, page 24). It is an easy exercise to give an alternative definition of the function Map by means of the corecursion operator.

REMARK. It is possible to define interesting cototal objects by means of corecursion operators. For instance the rightmost infinite path in the type $\mathbb{Y}$ of binary trees is $t_R := {}^{\mathrm{co}}\mathcal{R}^\tau_{\mathbb{Y}} x_0 f_0$ with τ, x_0 arbitrary (for instance $\tau := \mathbb{U}$, $x_0 := \mathrm{Dummy}^{\mathbb{U}}$) and

$$ f_0 x^\tau := \mathrm{Inr}\langle \mathrm{InL}^{\mathbb{Y}\to\mathbb{Y}+\tau} 0, \mathrm{InR}^{\tau\to\mathbb{Y}+\tau} x\rangle. $$

For the leftmost path we similary have t_L.

Another example is a function converting a real number given as a Cauchy sequence of rationals together a Cauchy modulus into an infinite

stream of signed digits $\{-1, 0, 1\}$. Such a function can be defined by a simple corecursion. One can extract it from a proof (using "coinduction") of the fact that such a conversion exists.

2.4. Denotational semantics

We now set up a connection between the model $(|\mathbf{A}_\rho|)_\rho$ of partial continuous functionals described in Section 2.1 and the term system T^+ from Section 2.3. The main point is to clarify how we can use computation rules to define an ideal z in a function space. The general idea is to inductively define the set of tokens (U, a) that make up z. It is convenient to define the value $[\![\lambda_{\vec{x}} M]\!]$, where M is a term with free variables among $\vec{x}$. Since this value is a token set, we can define inductively the relation $(\vec{U}, a) \in [\![\lambda_{\vec{x}} M]\!]$.

For a constructor pattern $\vec{P}(\vec{x})$ and a list $\vec{V}$ of the same length and types as $\vec{x}$ we define a list $\vec{P}(\vec{V})$ of formal neighborhoods of the same length and types as $\vec{P}(\vec{x})$, by induction on $\vec{P}(\vec{x})$. $x(V)$ is the singleton list V, and for $\langle\rangle$ we take the empty list. $(\vec{P}, Q)(\vec{V}, \vec{w})$ is covered by the induction hypothesis. Finally

$$(\mathrm{C}\vec{P})(\vec{V}) := \{\, \mathrm{C}\vec{a^*} \mid a_i^* \in P_i(\vec{V_i}) \text{ if } P_i(\vec{V_i}) \neq \emptyset, \text{ and } a_i^* = * \text{ otherwise} \,\}.$$

We use the following notation. $(\vec{U}, a)$ means $(U_1, (U_2, \dots (U_n, a)) \dots)$, and $(\vec{U}, V) \subseteq [\![\lambda_{\vec{x}} M]\!]$ means $(\vec{U}, a) \in [\![\lambda_{\vec{x}} M]\!]$ for all (finitely many) $a \in V$.

Definition (Inductive, of $(\vec{U}, a) \in [\![\lambda_{\vec{x}} M]\!]$).

$$\frac{U_i \vdash a}{(\vec{U}, a) \in [\![\lambda_{\vec{x}} x_i]\!]}(V), \qquad \frac{(\vec{U}, V, a) \in [\![\lambda_{\vec{x}} M]\!] \qquad (\vec{U}, V) \subseteq [\![\lambda_{\vec{x}} N]\!]}{(\vec{U}, a) \in [\![\lambda_{\vec{x}} (MN)]\!]}(A).$$

For every constructor C and defined constant D we have

$$\frac{\vec{V} \vdash \vec{a^*}}{(\vec{U}, \vec{V}, \mathrm{C}\vec{a^*}) \in [\![\lambda_{\vec{x}} \mathrm{C}]\!]}(\mathrm{C}), \qquad \frac{(\vec{U}, \vec{V}, a) \in [\![\lambda_{\vec{x},\vec{y}} M]\!] \qquad \vec{W} \vdash \vec{P}(\vec{V})}{(\vec{U}, \vec{W}, a) \in [\![\lambda_{\vec{x}} D]\!]}(D)$$

with one such rule (D) for every computation rule $D\vec{P}(\vec{y}) = M$.

This "denotational semantics" has good properties; however, we do not carry out the proofs here but rather refer to the literature. Some of these proofs are given in Appendix B. First of all, one can prove that $[\![\lambda_{\vec{x}} M]\!]$ is an ideal. Moreover, our definition above of the denotation of a term is reasonable in the sense that it is not changed by an application of the standard (β- and η-) conversions or a computation rule.

CHAPTER 3

A theory TCF of partial continuous functionals

After getting clear about the domains we intend to reason about, the partial continuous functionals, we now set up a theory to prove their properties. The main concepts are those of inductively and coinductively defined predicates.

3.1. Formulas and their computational content

Formulas will built up from prime formulas $P\vec{t}$ by implication $\to$ and universal quantification $\forall_x$; here the t_i are terms, x is a variable and P is a predicate of a certain arity (a list of types). Types and terms are defined as in Chapter 2. We often write $\vec{t} \in P$ for $P\vec{t}$.

Predicates can be inductively or coinductively defined. An example for the former is $T_{\mathbb{L}(\mathbb{N})}$, which is defined by the clauses (i) $[] \in T_{\mathbb{L}(\mathbb{N})}$ and (ii) $\forall_{n\in\mathbb{N}}(\ell \in T_{\mathbb{L}(\mathbb{N})} \to n :: \ell \in T_{\mathbb{L}(\mathbb{N})})$. An example for the latter is ${}^{\mathrm{co}}T_{\mathbb{L}(\mathbb{N})}$ defined by a closure axiom saying that every $\ell \in {}^{\mathrm{co}}T_{\mathbb{L}(\mathbb{N})}$ is of the form $n :: \ell'$ with $n \in \mathbb{N}$ and $\ell' \in {}^{\mathrm{co}}T_{\mathbb{L}(\mathbb{N})}$ again. According to Kolmogorov (1932) a formula can be seen as a problem, asking for a solution. In the inductive example a solution for $4 :: 2 :: 0 :: [] \in T_{\mathbb{L}(\mathbb{N})}$ would be the generating (finite) sequence $[]$, $0 :: []$, $2 :: 0 :: []$, $4 :: 2 :: 0 :: []$, and in the coinductive example a solution for a prime formula $t \in {}^{\mathrm{co}}T_{\mathbb{L}(\mathbb{N})}$ would be an (infinite) stream of natural numbers. Generally, a solution for an inductive predicate is a finite construction tree, and for a coinductive predicate a finitely branching possibly infinite destruction tree. Such trees can be seen as ideals of the closed base types considered in Section 2.2.2. A solution for a problem posed by the formula $A \to B$ is a computable functional mapping solutions of A into solutions of B.

Sometimes the solution of a problem does not need all available input. We therefore mark the sources of such computationally superfluous input – that is, some (co)inductive predicates – as "non-computational" (n.c.).

Assume an infinite supply of predicate variables, each of its own *arity* (a list of types). We distinguish two sorts of predicate variables, "computationally relevant" ones $X^{\mathrm{c}}, Y^{\mathrm{c}}, Z^{\mathrm{c}}, W^{\mathrm{c}} \dots$ and "non-computational" ones X^{nc}, Y^{nc}, Z^{nc}, W^{nc} ..., and use $X, Y, Z, W \dots$ for both.

Let Z be a predicate variable. By $\bar{Z}$ we denote the result of applying the predicate variable Z to a list of terms of fitting types, and by $\tilde{Z}$ lists of those.

DEFINITION (Clauses and predicate forms). Let X be a predicate variable. An *X-clause* is a formula

$$K := \forall_{\vec{x}}(\tilde{Y}^{\mathrm{c}} \to \tilde{Z}^{\mathrm{nc}} \to (\forall_{\vec{y}_i}(\tilde{W}_i^{\mathrm{nc}} \to \bar{X}_i))_{i<n} \to \bar{X})$$

with all predicate variables Y_i^{c}, Z_i^{nc}, W_i^{nc} occurring exactly once and distinct from each other and from X, and all $\bar{X}_i$ coming from the fixed X. A premise of a clause is called a *parameter* premise if X does not occur in it, and a *recursive* premise otherwise. A clause K is *non-recursive* if it has no recursive premises.

Let $\vec{K}$ be a list of X-clauses. We call $I^{\mathrm{c}} := \mu_{X^{\mathrm{c}}}\vec{K}$ and $I^{\mathrm{nc}} := \mu_{X^{\mathrm{nc}}}\vec{K}$ (with $\vec{K}$ not empty) *predicate forms* (and use I for both), and similarly with ${}^{\mathrm{co}}I$ for I and ν for μ.

EXAMPLES. Recall that $\mathrm{Nil}^{\mathbb{L}(\alpha)}$ and $\mathrm{Cons}^{\alpha\to\mathbb{L}(\alpha)\to\mathbb{L}(\alpha)}$ are the two constructors of the base type $\mathbb{L}(\alpha)$ of lists, written $[]$ and $::$ (infix).

1. Let Y of arity (α) and X of arity $(\mathbb{L}(\alpha))$ be predicate variables. Then

$$K_0 := ([] \in X),$$
$$K_1 := \forall_x(x \in Y \to \forall_\ell(\ell \in X \to x :: \ell \in X))$$

are clauses and both *relative totality* $T_{\mathbb{L}}(Y)$ (or $T_{\mathbb{L},Y}$) defined by $\mu_X(K_0, K_1)$ and also *relative cototality* ${}^{\mathrm{co}}T_{\mathbb{L}}(Y)$ (or ${}^{\mathrm{co}}T_{\mathbb{L},Y}$) defined by $\nu_X(K_0, K_1)$ are predicate forms with Y a parameter predicate variable. Note that we can omit the type parameter α, since it can be read off from the arity of Y.

2. Alternatively let Y of arity (α, α) and X of arity $(\mathbb{L}(\alpha), \mathbb{L}(\alpha))$ be predicate variables. Then

$$K_0 := X([], []),$$
$$K_1 := \forall_{x,x'}(Y(x, x') \to \forall_{\ell,\ell'}(X(\ell, \ell') \to X(x :: \ell, x' :: \ell')))$$

are clauses and both *similarity* $\sim_{\mathbb{L}}(Y)$ defined by $\mu_X(K_0, K_1)$ and *bisimilarity* $\approx_{\mathbb{L}}(Y)$ defined by $\nu_X(K_0, K_1)$ are predicate forms with Y a parameter predicate variable.

Note that a predicate form I may contain type variables $\vec{\alpha}$ and predicate variables $\vec{Y}$. We write $I(\vec{\rho}, \vec{P})$ for the result of substituting in I the types $\vec{\rho}$ for $\vec{\alpha}$ and the predicates $\vec{P}$ for $\vec{Y}$.

DEFINITION (Constructor types of a predicate form). From every clause K we obtain a constructor type by

- omitting quantifiers,

- dropping all n.c. predicates and from the c.r. predicates their arguments, and
- replacing the remaining predicate variables by type variables.

That is, from the clause

$$\forall_{\vec{x}}(\tilde{Y}^{\mathrm{c}} \to \tilde{Z}^{\mathrm{nc}} \to (\forall_{\vec{y}_i}(\tilde{W}_i^{\mathrm{nc}} \to \bar{X}_i))_{i<n} \to \bar{X})$$

we obtain the constructor type $\vec{\alpha} \to (\xi)_{i<n} \to \xi$. With every predicate form $I^{\mathrm{c}} := (\mu/\nu)_{X^{\mathrm{c}}}\vec{K}$ we associate the list $\vec{\kappa}$ of constructor types.

Definition (Predicates and formulas).

$$\begin{aligned} P, Q &::= X \mid \{\, \vec{x} \mid A \,\} \mid I(\vec{\rho}, \vec{P}\,) \mid {}^{\mathrm{co}}I(\vec{\rho}, \vec{P}\,) && \text{(predicates)},\\ A, B &::= P\vec{t} \mid A \to B \mid \forall_x A && \text{(formulas)} \end{aligned}$$

with $I/{}^{\mathrm{co}}I$ a predicate form. To take care of the difference between X^{c} and X^{nc} we define the *final predicate* of a predicate or formula by

$$\begin{aligned} \mathrm{fp}(X) &:= X, & \mathrm{fp}(P\vec{t}\,) &:= \mathrm{fp}(P),\\ \mathrm{fp}(\{\, \vec{x} \mid A \,\}) &:= \mathrm{fp}(A), & \mathrm{fp}(A \to B) &:= \mathrm{fp}(B),\\ \mathrm{fp}((I/{}^{\mathrm{co}}I)(\vec{\rho}, \vec{P}\,)) &:= I/{}^{\mathrm{co}}I, & \mathrm{fp}(\forall_x A) &:= \mathrm{fp}(A). \end{aligned}$$

We call a predicate or formula C *non-computational* (n.c., or *Harrop*) if its final predicate $\mathrm{fp}(C)$ is of the form X^{nc} or I^{nc}, else *computationally relevant* (c.r.). We require that all predicate substitutions involved in $(I/{}^{\mathrm{co}}I)(\vec{\rho}, \vec{P}\,)$ substitute c.r. predicates for c.r. predicate variables and n.c. predicates for n.c. predicate variables. Such predicate substitutions are called *sharp*.

Predicates of the form $I(\vec{\rho}, \vec{P}\,)$ are called *inductive*, and predicates of the form ${}^{\mathrm{co}}I(\vec{\rho}, \vec{P}\,)$ *coinductive*.

The terms $\vec{t}$ are those introduced in Section 2.3.1, i.e., typed terms built from typed variables and constants by abstraction and application, and (importantly) those with a common reduct are identified.

A predicate of the form $\{\, \vec{x} \mid C \,\}$ is called a *comprehension term*. We identify $\{\, \vec{x} \mid C(\vec{x}\,) \,\}\vec{t}$ with $C(\vec{t}\,)$. For a predicate C of arity $(\rho, \vec{\sigma}\,)$ we write Ct for $\{\, \vec{y} \mid Ct\vec{y} \,\}$.

It is a natural question to ask what the type of a "realizer" or "witness" of a c.r. predicate or formula C should be.

DEFINITION (Type $\tau(C)$ of a c.r. predicate or formula C). Assume a global injective assignment of type variables ζ to c.r. predicate variables X^{c}.

$$\begin{aligned} \tau(X^{\mathrm{c}}) &:= \zeta, & \tau(P\vec{t}\,) &:= \tau(P), \\ \tau(\{\,\vec{x} \mid A\,\}) &:= \tau(A), & \tau(A \to B) &:= \begin{cases} \tau(A) \to \tau(B) & (A \text{ c.r.}) \\ \tau(B) & (A \text{ n.c.}), \end{cases} \\ \tau(I(\vec{\rho}, \vec{P}\,)) &:= \iota_{\vec{\kappa}(\vec{\rho},\tau(\vec{P}^{\mathrm{c}}))}, & \tau(\forall_x A) &:= \tau(A). \end{aligned}$$

In the I-case we have assumed $I = (\mu/\nu)_X \vec{K}$ with X-clauses $\vec{K}$. Every K_i has an assigned constructor type κ_i. Free in $\vec{\kappa}$ are the type variables $\vec{\alpha}$ from $\vec{K}$ and the type variables $\vec{\zeta}$ globally assigned to the c.r. predicate variables $\vec{Y}^{\mathrm{c}}$ in $\vec{K}$. Now $\vec{\kappa}(\vec{\rho}, \tau(\vec{P}^{\mathrm{c}}))$ is the result of substituting $\vec{\rho}$ for $\vec{\alpha}$ and of the (already generated) types $\tau(P_i^{\mathrm{c}})$ for ζ_i in $\vec{\kappa}$.

3.2. Examples of inductive predicates

A simple example of an inductive predicate is totality $T_{\mathbb{N}}$ of the natural numbers. It is defined as

$$T_{\mathbb{N}} := \mu_X(K_0, K_1)$$

with

$$\begin{aligned} K_0 &:= (0 \in X), \\ K_1 &:= \forall_n(n \in X \to Sn \in X). \end{aligned}$$

Depending on whether the predicate variable X is n.c. or c.r. we have an n.c. or a c.r. totality predicate.

Recall that a variable of type τ ranges over arbitrary objects of type τ, which may be partial. However, in practice we ofter want to argue on total objects only. To make such a restriction easy to read we introduce two sorts of variable names: a general one written $\hat{x}$ ranging over arbitrary (possibly partial) objects, and a special one written x ranging over total objects only. Then we use the abbreviation

$$\forall_x A(x) := \forall_{\hat{x}}(\hat{x} \in T_\tau \to A(\hat{x})).$$

We will follow this convention from now on. Hence the clause K_1 above should now be written

$$K_1 := \forall_{\hat{n}}(\hat{n} \in X \to S\hat{n} \in X).$$

Another particularly important example of an inductive predicate is *Leibniz equality*, defined simply by

$$\mathrm{EqD} := \mu_{X^{\mathrm{nc}}}(\forall_{\hat{x}} X^{\mathrm{nc}} \hat{x}\hat{x}) \qquad (\mathrm{D} \text{ for "inductively } \mathbf{d}\text{efined"}).$$

We will use the abbreviation

$$(t \equiv s) := \mathrm{EqD}(t, s).$$

The missing logical connectives existence, disjunction and conjunction can also be defined inductively. Existence is defined inductively by

$$\begin{aligned}\mathrm{Ex}_{Y^{\mathrm{c}}} &:= \mu_{X^{\mathrm{c}}}(\forall_{\hat{x}}(\hat{x} \in Y^{\mathrm{c}} \to X^{\mathrm{c}})),\\ \mathrm{ExNc}_{Y} &:= \mu_{X^{\mathrm{nc}}}(\forall_{\hat{x}}(\hat{x} \in Y \to X^{\mathrm{nc}})).\end{aligned}$$

Then by definition

$$\tau(\mathrm{Ex}) = \mu_{\xi}(\beta \to \xi) = \mathbb{I}(\beta).$$

We use the abbreviation

$$\begin{aligned}\exists_{\hat{x}} A &:= \mathrm{Ex}_{\{\hat{x}|A\}},\\ \exists^{\mathrm{nc}}_{\hat{x}} A &:= \mathrm{ExNc}_{\{\hat{x}|A\}},\end{aligned}$$

and again since the decoration is determined by the c.r./n.c. status of the parameter predicate we usually leave out the decoration and just write $\exists$.

For a context where only total objects are of interest we have

$$\begin{aligned}\mathrm{ExDT}_{Y^{\mathrm{c}}} &:= \mu_{X^{\mathrm{c}}}(\forall_{\hat{x}}(\hat{x} \in T^{\mathrm{c}} \to \hat{x} \in Y^{\mathrm{c}} \to X^{\mathrm{c}})),\\ \mathrm{ExLT}_{Y^{\mathrm{c}}} &:= \mu_{X^{\mathrm{c}}}(\forall_{\hat{x}}(\hat{x} \in T^{\mathrm{c}} \to \hat{x} \in Y^{\mathrm{nc}} \to X^{\mathrm{c}})),\\ \mathrm{ExRT}_{Y^{\mathrm{c}}} &:= \mu_{X^{\mathrm{c}}}(\forall_{\hat{x}}(\hat{x} \in T^{\mathrm{nc}} \to \hat{x} \in Y^{\mathrm{c}} \to X^{\mathrm{c}})),\\ \mathrm{ExNcT}_{Y} &:= \mu_{X^{\mathrm{nc}}}(\forall_{\hat{x}}(\hat{x} \in T \to \hat{x} \in Y \to X^{\mathrm{nc}})).\end{aligned}$$

Here D is for "double", L for "left" and R for "right". Then by definition

$$\begin{aligned}\tau(\mathrm{ExDT}) &= \mu_{\xi}(\tau \to \beta \to \xi) = \tau \times \beta\\ \tau(\mathrm{ExLT}) &= \mu_{\xi}(\tau \to \xi) \qquad = \mathbb{I}(\tau),\\ \tau(\mathrm{ExRT}) &= \mu_{\xi}(\beta \to \xi) \qquad = \mathbb{I}(\beta).\end{aligned}$$

To make these formulas more readable we can again use our convention concerning the two sorts $\hat{x}$ and x of variable names. Then the inductive predicates above are written as

$$\begin{aligned}\mathrm{ExDT}_{Y^{\mathrm{c}}} &:= \mu_{X^{\mathrm{c}}}(\forall_{x}(x \in Y^{\mathrm{c}} \to X^{\mathrm{c}})),\\ \mathrm{ExLT}_{Y^{\mathrm{c}}} &:= \mu_{X^{\mathrm{c}}}(\forall_{x}(x \in Y^{\mathrm{nc}} \to X^{\mathrm{c}})),\\ \mathrm{ExRT}_{Y^{\mathrm{c}}} &:= \mu_{X^{\mathrm{c}}}(\forall^{\mathrm{nc}}_{x}(x \in Y^{\mathrm{c}} \to X^{\mathrm{c}})),\\ \mathrm{ExNcT}_{Y} &:= \mu_{X^{\mathrm{nc}}}(\forall_{x}(x \in Y \to X^{\mathrm{nc}})).\end{aligned}$$

We use the abbreviations

$$\begin{array}{ll}
\exists^{\mathrm{d}}_x A := \mathrm{ExDT}_{\{x|A\}} & \text{if } A \text{ is c.r.,}\\
\exists^{\mathrm{l}}_x A := \mathrm{ExLT}_{\{x|A\}} & \text{if } A \text{ is n.c.,}\\
\exists^{\mathrm{r}}_x A := \mathrm{ExRT}_{\{x|A\}} & \text{if } A \text{ is c.r.,}\\
\exists^{\mathrm{nc}}_x A := \mathrm{ExNcT}_{\{x|A\}} & \text{for arbitrary } A.
\end{array}$$

Disjunction is a special case of union

$$\mathrm{Cup}_{Y,Z} := \mu_{X^{\mathrm{c}}}(\forall_{\vec{x}}(Y\vec{x} \to X^{\mathrm{c}}\vec{x}),\ \forall_{\vec{x}}(Z\vec{x} \to X^{\mathrm{c}}\vec{x})).$$

Since the parameter predicates Y, Z can be chosen as either c.r. or n.c. we obtain the variants

$$\begin{array}{ll}
\mathrm{CupD}_{Y^{\mathrm{c}},Z^{\mathrm{c}}} & := \mu_{X^{\mathrm{c}}}(\forall_{\vec{x}}(Y^{\mathrm{c}}\vec{x} \to X^{\mathrm{c}}\vec{x}),\ \forall_{\vec{x}}(Z^{\mathrm{c}}\vec{x} \to X^{\mathrm{c}}\vec{x})),\\
\mathrm{CupL}_{Y^{\mathrm{c}},Z^{\mathrm{nc}}} & := \mu_{X^{\mathrm{c}}}(\forall_{\vec{x}}(Y^{\mathrm{c}}\vec{x} \to X^{\mathrm{c}}\vec{x}),\ \forall_{\vec{x}}(Z^{\mathrm{nc}}\vec{x} \to X^{\mathrm{c}}\vec{x})),\\
\mathrm{CupR}_{Y^{\mathrm{nc}},Z^{\mathrm{c}}} & := \mu_{X^{\mathrm{c}}}(\forall_{\vec{x}}(Y^{\mathrm{nc}}\vec{x} \to X^{\mathrm{c}}\vec{x}),\ \forall_{\vec{x}}(Z^{\mathrm{c}}\vec{x} \to X^{\mathrm{c}}\vec{x})),\\
\mathrm{CupU}_{Y^{\mathrm{nc}},Z^{\mathrm{nc}}} & := \mu_{X^{\mathrm{c}}}(\forall_{\vec{x}}(Y^{\mathrm{nc}}\vec{x} \to X^{\mathrm{c}}\vec{x}),\ \forall_{\vec{x}}(Z^{\mathrm{nc}}\vec{x} \to X^{\mathrm{c}}\vec{x})),\\
\mathrm{CupNc}_{Y,Z} & := \mu_{X^{\mathrm{nc}}}(\forall_{\vec{x}}(Y\vec{x} \to X^{\mathrm{nc}}\vec{x}),\ \forall_{\vec{x}}(Z\vec{x} \to X^{\mathrm{nc}}\vec{x})).
\end{array}$$

Here D is for "double", L for "left", R for "right" and U for "uniform". Then by definition

$$\begin{array}{lll}
\tau(\mathrm{CupD}) = \mu_\xi(\beta_0 \to \xi, \beta_1 \to \xi) & = \beta_0 + \beta_1,\\
\tau(\mathrm{CupL}) = \mu_\xi(\beta \to \xi, \xi) & = \beta + \mathbb{U} = \mathtt{ysumu}(\beta),\\
\tau(\mathrm{CupR}) = \mu_\xi(\xi, \beta \to \xi) & = \mathbb{U} + \beta = \mathtt{uysum}(\beta),\\
\tau(\mathrm{CupU}) = \mu_\xi(\xi, \xi) & = \mathbb{B}.
\end{array}$$

We use the abbreviations

$$\begin{array}{ll}
P \cup^{\mathrm{d}} Q & := \mathrm{CupD}_{P,Q},\\
P \cup^{\mathrm{l}} Q & := \mathrm{CupL}_{P,Q},\\
P \cup^{\mathrm{r}} Q & := \mathrm{CupR}_{P,Q},\\
P \cup^{\mathrm{u}} Q & := \mathrm{CupU}_{P,Q},\\
P \cup^{\mathrm{nc}} Q & := \mathrm{CupNc}_{P,Q}.
\end{array}$$

In case of nullary predicates we use

$$\begin{aligned} A \vee^{\mathrm{d}} B &:= \mathrm{CupD}_{\{|A\},\{|B\}}, \\ A \vee^{\mathrm{l}} B &:= \mathrm{CupL}_{\{|A\},\{|B\}}, \\ A \vee^{\mathrm{r}} B &:= \mathrm{CupR}_{\{|A\},\{|B\}}, \\ A \vee^{\mathrm{u}} B &:= \mathrm{CupU}_{\{|A\},\{|B\}}, \\ A \vee^{\mathrm{nc}} B &:= \mathrm{CupNc}_{\{|A\},\{|B\}}. \end{aligned}$$

Since the "decoration" is determined by the c.r./n.c. status of the two parameter predicates we usually leave it out in $\vee^{\mathrm{d}}, \vee^{\mathrm{l}}, \vee^{\mathrm{r}}, \vee^{\mathrm{u}}$ and just write $\vee$. However in the final nc-variant we suppress even the information which clause has been used, and hence must keep the notation $\vee^{\mathrm{nc}}$.

Similarly conjunction is a special case of intersection

$$\mathrm{Cap}_{Y,Z} := \mu_{X^{\mathrm{c}}}(\forall_{\vec{x}}(Y\vec{x} \to Z\vec{x} \to X^{\mathrm{c}}\vec{x})),$$

and we obtain the variants

$$\begin{aligned} \mathrm{CapD}_{Y^{\mathrm{c}},Z^{\mathrm{c}}} &:= \mu_{X^{\mathrm{c}}}(\forall_{\vec{x}}(Y^{\mathrm{c}}\vec{x} \to Z^{\mathrm{c}}\vec{x} \to X^{\mathrm{c}}\vec{x})), \\ \mathrm{CapL}_{Y^{\mathrm{c}},Z^{\mathrm{nc}}} &:= \mu_{X^{\mathrm{c}}}(\forall_{\vec{x}}(Y^{\mathrm{c}}\vec{x} \to Z^{\mathrm{nc}}\vec{x} \to X^{\mathrm{c}}\vec{x})), \\ \mathrm{CapR}_{Y^{\mathrm{nc}},Z^{\mathrm{c}}} &:= \mu_{X^{\mathrm{c}}}(\forall_{\vec{x}}(Y^{\mathrm{nc}}\vec{x} \to Z^{\mathrm{c}}\vec{x} \to X^{\mathrm{c}}\vec{x})), \\ \mathrm{CapNc}_{Y,Z} &:= \mu_{X^{\mathrm{nc}}}(\forall_{\vec{x}}(Y\vec{x} \to Z\vec{x} \to X^{\mathrm{nc}}\vec{x})). \end{aligned}$$

Then by definition

$$\begin{aligned} \tau(\mathrm{CapD}) = \mu_\xi(\beta_0 \to \beta_1 \to \xi) &= \beta_0 \times \beta_1 \\ \tau(\mathrm{CapL}) = \tau(\mathrm{CapR}) = \mu_\xi(\beta \to \xi) &= \mathbb{I}(\beta). \end{aligned}$$

We use the abbreviations

$$\begin{aligned} P \cap^{\mathrm{d}} Q &:= \mathrm{CapD}_{P,Q}, \\ P \cap^{\mathrm{l}} Q &:= \mathrm{CapL}_{P,Q}, \\ P \cap^{\mathrm{r}} Q &:= \mathrm{CapR}_{P,Q}, \\ P \cap^{\mathrm{nc}} Q &:= \mathrm{CapNc}_{P,Q}. \end{aligned}$$

In case of nullary predicates we use

$$\begin{aligned} A \wedge^{\mathrm{d}} B &:= \mathrm{CapD}_{\{|A\},\{|B\}}, \\ A \wedge^{\mathrm{l}} B &:= \mathrm{CapL}_{\{|A\},\{|B\}}, \\ A \wedge^{\mathrm{r}} B &:= \mathrm{CapR}_{\{|A\},\{|B\}}, \\ A \wedge^{\mathrm{nc}} B &:= \mathrm{CapNc}_{\{|A\},\{|B\}}. \end{aligned}$$

Again since the decoration is determined by the c.r./n.c. status of the two parameter predicates we usually leave out the decoration and just write $\wedge$.

3.3. Axioms of TCF

We define a theory of continuous functionals, called TCF. Formulas are the ones defined above, involving typed variables. Derivations use the rules of minimal logic for $\to$ and $\forall$, and the axioms introduced below. However, because of the distinction between n.c. and c.r. predicates and formulas we have an extra degree of freedom. By an *n.c. part* of a derivation we mean a subderivation with an n.c. end formula. Such n.c. parts will not contribute to the computational content of the whole derivation, and hence we can ignore all decorations in those parts (i.e., use a modified notion of equality of formulas there).

3.3.1. Axioms for inductive predicates. For each inductive predicate there are "clauses" or introduction axioms, together with a "least-fixed-point" or elimination axiom. To grasp the general form of these axioms it is convenient to write a clause

$$\forall_{\vec{x}}(\tilde{Y}^{\mathrm{c}} \to \tilde{Z}^{\mathrm{nc}} \to (\forall_{\vec{y}_i}(\tilde{W}_i^{\mathrm{nc}} \to \bar{X}_i))_{i<n} \to \bar{X}) \quad \text{as} \quad \forall_{\vec{x}}((A_\nu(X))_{\nu<n} \to X\vec{t}\,).$$

Definition (Introduction and elimination axioms for inductive predicates). For an inductive predicate $\mu_X(\forall_{\vec{x}_i}((A_{i\nu}(X))_{\nu<n_i} \to X\vec{t}_i))_{i<k} =: I$ we have k introduction axioms I_i^+ $(i<k)$ and one elimination axiom I^-:

$$I_i^+ \colon \forall_{\vec{x}_i}((A_{i\nu}(I))_{\nu<n_i} \to I\vec{t}_i), \tag{12}$$

$$I^- \colon (\forall_{\vec{x}_i}((A_{i\nu}(I\cap X))_{\nu<n_i} \to X\vec{t}_i))_{i<k} \to I \subseteq X \tag{13}$$

($I\cap X$ was inductively defined above). (13) expresses that every competitor X satisfying the same clauses contains I. We take all substitution instances of I_i^+, I^- (w.r.t. substitutions for type and predicate variables) as axioms.

Remarks. (i) We use a "strengthened" form of the "step formula", namely $\forall_{\vec{x}_i}(A_{i\nu}(I\cap X))_{\nu<n_i} \to X\vec{t}_i$ rather than $\forall_{\vec{x}_i}(A_{i\nu}(X))_{\nu<n_i} \to X\vec{t}_i$. In applications of the least-fixed-point axiom this simplifies the proof of the "step", since we have an additional I-hypothesis available.

(ii) Notice that there is no circularity here for the inductive predicate $Y\cap Z := \mathrm{Cap}_{Y,Z}$, since there are no recursive calls in this particular inductive definition and hence $\cap$ does not occur in

$$\mathrm{Cap}^-_{Y,Z} \colon \forall_{\vec{x}}(Y\vec{x} \to Z\vec{x} \to X\vec{x}) \to \mathrm{Cap}_{Y,Z} \subseteq X.$$

(iii) The elimination axiom (13) could equivalently be written as

$$I^- \colon \forall_{\vec{x}}(I\vec{x} \to (\forall_{\vec{x}_i}((A_{i\nu}(I\cap X))_{\nu<n_i} \to X\vec{t}_i))_{i<k} \to X\vec{x}\,)$$

In this form it fits better with our (i.e., Gentzen's) way to write the logical elimination rules, where the main premise comes first. More importantly, its type (cf. Section 3.1) will then be the type of the recursion operator $\mathcal{R}^\tau_\iota$

taken as the "computational content" (cf. Section 4.1) of the elimination axiom for I. Therefore in the implementation of TCF this form is used. However, for readability we often prefer the form (13) in the present notes.

In Section 3.2 we considered several basic examples of inductive predicates. Their introduction and elimination axioms have important consequences, which we shall study now.

Totality $T_{\mathbb{N}}$ for the natural numbers has the introduction axioms

$$(T_{\mathbb{N}})_0^+ : 0 \in T_{\mathbb{N}},$$
$$(T_{\mathbb{N}})_1^+ : \forall_{\hat{n}}(\hat{n} \in T_{\mathbb{N}} \to S\hat{n} \in T_{\mathbb{N}}).$$

Its elimination axiom is

$$(T_{\mathbb{N}})^- : 0 \in X \to \forall_{\hat{n}}(\hat{n} \in T_{\mathbb{N}} \to \hat{n} \in X \to S\hat{n} \in X) \to \forall_{\hat{n}}(\hat{n} \in T_{\mathbb{N}} \to \hat{n} \in X)$$

or in abbreviated form (recall our convention on using both $\hat{n}$ and n as variable names)

$$(T_{\mathbb{N}})^- : 0 \in X \to \forall_n(n \in X \to Sn \in X) \to \forall_n(n \in X).$$

This is the usual induction axiom for (total) natural numbers.

The n.c. Leibniz equality has the introduction axiom

$$(\mathrm{EqD})_0^+ : \forall_{\hat{x}}(\hat{x} \equiv \hat{x}).$$

Its elimination axiom is

$$(\mathrm{EqD})^- : \forall_{\hat{x}} X\hat{x}\hat{x} \to \forall_{\hat{x},\hat{y}}(\hat{x} \equiv \hat{y} \to X\hat{x}\hat{y}).$$

From this definition we can deduce the property Leibniz used as a definition.

LEMMA 3.3.1 (Compatibility of EqD). $\forall_{\hat{x},\hat{y}}(\hat{x} \equiv \hat{y} \to A(\hat{x}) \to A(\hat{y}))$.

PROOF. By the elimination axiom with $X := \{\, \hat{x}, \hat{y} \mid A(\hat{x}) \to A(\hat{y}) \,\}$. □

Using compatibility of EqD one easily proves symmetry and transitivity.

An important usage of EqD in TCF is that it allows to introduce *falsity* and hence *negation*. Recall that the language of TCF contains constructors for base types. For the base type $\mathbb{B}$ of booleans we have as constructors Frege's "Wahrheitswerte" $\mathtt{tt}$ and $\mathtt{ff}$. Using these we can define

DEFINITION (Falsity, Negation). (a) Falsity $\mathbf{F}$ is defined by

$$\mathbf{F} := (\mathtt{ff} \equiv \mathtt{tt}).$$

(b) The negation $\neg A$ of a formula A is defined by

$$\neg A := (A \to \mathbf{F}).$$

Now using the fact that we identify terms with a common reduct and that we have recursion operators in our language we can prove "ex-falso-quodlibet" for formulas $I\vec{t}$ with I a predicate form. Easy examples are

LEMMA 3.3.2 (Ex-falso for EqD and $T_{\mathbb{N}}$). TCF *proves*

(a) $\mathbf{F} \to \forall_{\hat{x},\hat{y}}(\hat{x} \equiv \hat{y})$,
(b) $\mathbf{F} \to \forall_{\hat{n}}(\hat{n} \in T_{\mathbb{N}})$.

PROOF. (a) We show $\mathrm{Ef}_{\mathrm{EqD}} \colon \mathbf{F} \to \hat{x}^\tau \equiv \hat{y}^\tau$. To see this, we first obtain $\mathcal{R}^\tau_{\mathbb{B}}\mathtt{ff}\hat{x}\hat{y} \equiv \mathcal{R}^\tau_{\mathbb{B}}\mathtt{ff}\hat{x}\hat{y}$ from the introduction axiom. Then from $\mathtt{ff} \equiv \mathtt{tt}$ we get $\mathcal{R}^\tau_{\mathbb{B}}\mathtt{tt}\hat{x}\hat{y} \equiv \mathcal{R}^\tau_{\mathbb{B}}\mathtt{ff}\hat{x}\hat{y}$ by compatibility. Now $\mathcal{R}^\tau_{\mathbb{B}}\mathtt{tt}\hat{x}\hat{y}$ converts to $\hat{x}$ and $\mathcal{R}^\tau_{\mathbb{B}}\mathtt{ff}\hat{x}\hat{y}$ converts to $\hat{y}$. Hence $\hat{x} \equiv \hat{y}$, since we identify terms with a common reduct.

(b) We show $\mathrm{Ef}_{T_{\mathbb{N}}} \colon \mathbf{F} \to \hat{n} \in T_{\mathbb{N}}$. Assume $\mathbf{F}$. Then $\hat{n} \equiv 0$ by (a), hence $\hat{n} \in T_{\mathbb{N}}$ by $0 \in T_{\mathbb{N}}$ and compatibility. □

A similar result holds for arbitrary predicates and formulas. We postpone it until the axioms for coinductive predicates have been introduced.

An important use of Leibniz equality EqD is that it allows to turn a term t of type $\mathbb{B}$ into a formula $\mathrm{atom}(t)$, defined by

DEFINITION (Boolean terms as formulas).

$$\mathrm{atom}(t) := (t \equiv \mathtt{tt}).$$

This opens up a convenient way to deal with equality on closed base types. The computation rules ensure that, for instance, the boolean term $St =_{\mathbb{N}} Ss$, or more precisely $=_{\mathbb{N}}(St, Ss)$, is identified with $t =_{\mathbb{N}} s$. We can now turn this boolean term into the formula $(St =_{\mathbb{N}} Ss) \equiv \mathtt{tt}$, which again is abbreviated by $St =_{\mathbb{N}} Ss$, but this time with the understanding that it is a formula. Then (importantly) the two formulas $St =_{\mathbb{N}} Ss$ and $t =_{\mathbb{N}} s$ are identified because the latter is a reduct of the first. Consequently there is no need to prove the implication $St =_{\mathbb{N}} Ss \to t =_{\mathbb{N}} s$ explicitly.

Recall the inductive definitions of the logical connectives existence, disjunction and conjunction given above. For nullary predicates $P = \{ \mid A \}$ and $Q = \{ \mid B \}$ we write $A \vee B$ for $P \cup Q$ and $A \wedge B$ for $P \cap Q$. For simplicity we only consider the "double" versions. Then the introduction axioms are

$$\begin{aligned}
&\forall_x(A \to \exists^{\mathrm{d}}_x A),\\
&A \to A \vee^{\mathrm{d}} B, \qquad B \to A \vee^{\mathrm{d}} B,\\
&A \to B \to A \wedge^{\mathrm{d}} B,
\end{aligned}$$

and the elimination axioms are (now written in the equivalent form mentioned above, where the main premise comes first)

$$\begin{aligned}
&\exists^{\mathrm{d}}_x A \to \forall_x(A \to B) \to B \qquad (x \notin \mathrm{FV}(B)),\\
&A \vee^{\mathrm{d}} B \to (A \to C) \to (B \to C) \to C,\\
&A \wedge^{\mathrm{d}} B \to (A \to B \to C) \to C.
\end{aligned}$$

3.3.2. Axioms for coinductive predicates. For each coinductive predicate there is a closure axiom, together with a "greatest-fixed-point" axiom. For example, for the base type $\mathbb{Y}$ of of binary trees

- the cototality predicate ${}^{\mathrm{co}}T_{\mathbb{Y}}$ is defined by the closure axiom (3) (page 18) and the greatest-fixed-point axiom by (4) (page 18), and
- the bisimilarity predicate $\approx_{\mathbb{Y}}$ by the closure axiom (7) (page 19) and the greatest-fixed-point axiom (8) (page 19).

To understand the general axioms for coinductive predicates note that the conjunction of the k clauses (12) of an inductive predicate I is equivalent to

$$\forall_{\vec{x}}(\bigvee_{i<k}\exists_{\vec{x}_i}(\bigwedge_{\nu<n_i} A_{i\nu}(I)\land\vec{x}\equiv\vec{t}_i)\to I\vec{x}\,).$$

Definition (Closure and greatest-fixed-point axioms). For an inductive predicate $\mu_X(\forall_{\vec{x}_i}((A_{i\nu}(X))_{\nu<n_i}\to X\vec{t}_i))_{i<k} =: I$ we define its dual ${}^{\mathrm{co}}I$ (with ν for μ) by the *closure axiom* ${}^{\mathrm{co}}I^-$ and the *greatest-fixed-point axiom* ${}^{\mathrm{co}}I^+$:

$$(14)\qquad {}^{\mathrm{co}}I^-\colon \forall_{\vec{x}}({}^{\mathrm{co}}I\vec{x}\to\bigvee_{i<k}\exists_{\vec{x}_i}(\bigwedge_{\nu<n_i} A_{i\nu}({}^{\mathrm{co}}I)\land\vec{x}\equiv\vec{t}_i)),$$

$$(15)\qquad {}^{\mathrm{co}}I^+\colon \forall_{\vec{x}}(X\vec{x}\to\bigvee_{i<k}\exists_{\vec{x}_i}(\bigwedge_{\nu<n_i} A_{i\nu}({}^{\mathrm{co}}I\cup X)\land\vec{x}\equiv\vec{t}_i))\to X\subseteq{}^{\mathrm{co}}I.$$

(${}^{\mathrm{co}}I\cup X$ was inductively defined above). The axiom expresses that every "competitor" X satisfying the closure axiom is contained in ${}^{\mathrm{co}}I$. We take all substitution instances of ${}^{\mathrm{co}}I^+$, ${}^{\mathrm{co}}I^-$ (w.r.t. substitutions for type and predicate variables) as axioms.

Again we have used a "strengthened" form of the "step formula", with $A_{i\nu}({}^{\mathrm{co}}I\cup X)$ rather than $A_{i\nu}(X)$. In applications of the greatest-fixed-point axiom this simplifies the proof of the "step", since its conclusion is weaker.

Remark. The greatest-fixed-point axiom (15) could be written as

$$\forall_{\vec{x}}(X\vec{x}\to\forall_{\vec{x}}(X\vec{x}\to\bigvee_{i<k}\exists_{\vec{x}_i}(\bigwedge_{\nu<n_i} A_{i\nu}({}^{\mathrm{co}}I\cup X)\land\vec{x}\equiv\vec{t}_i))\to{}^{\mathrm{co}}I\vec{x}).$$

Then its type will be the type of the corecursion operator ${}^{\mathrm{co}}\mathcal{R}^\tau_\iota$ taken as the "computational content" (cf. Section 4.1) of the greatest-fixed-point axiom for ${}^{\mathrm{co}}I$. Therefore in the implementation of TCF this form is used. However, for readability we prefer the form (15) in the present notes.

Remark. Instead of Leibniz equality $\equiv$ in (14) and (15) we could also use a different equality relation, for instance the n.c. variant $\doteq^{\mathrm{nc}}$ of pointwise equality to be introduced in Section 3.4. This leads to a new variant of ${}^{\mathrm{co}}I$.

EXAMPLES. (1) To show how to construct the dual ${}^{\mathrm{co}}I$ of an inductive predicate I we consider the predicate Even, which is defined by the clauses

$$(\mathrm{Even})^+_0 : 0 \in \mathrm{Even},$$
$$(\mathrm{Even})^+_1 : \forall_n(n \in \mathrm{Even} \to S(Sn) \in \mathrm{Even}).$$

The conjunction of its two clauses is equivalent to

$$\forall_n(n \equiv 0 \vee \exists_{n'}(n' \in \mathrm{Even} \wedge n \equiv S(Sn')) \to n \in \mathrm{Even}).$$

Now the dual ${}^{\mathrm{co}}\mathrm{Even}$ of Even is defined by its closure axiom ${}^{\mathrm{co}}\mathrm{Even}^-$:

$$\forall_n(n \in {}^{\mathrm{co}}\mathrm{Even} \to n \equiv 0 \vee \exists_{n'}(n' \in {}^{\mathrm{co}}\mathrm{Even} \wedge n \equiv S(Sn')))$$

and its greatest-fixed-point axiom ${}^{\mathrm{co}}\mathrm{Even}^+$:

$$\forall_n(Xn \to n \equiv 0 \vee \exists_{n'}(n' \in ({}^{\mathrm{co}}\mathrm{Even} \cup X) \wedge n \equiv S(Sn'))) \to X \subseteq {}^{\mathrm{co}}\mathrm{Even}.$$

(2) Consider the inductive predicate I of arity $(\mathbb{R})$ defined by the clause

$$\forall_{d,x',x}(d \in \mathbb{D} \wedge x' \in \mathbb{R} \wedge |x'| \le_{\mathbb{R}} 1 \wedge x' \in I \wedge x =_{\mathbb{R}} \frac{x'+d}{2} \to x \in I).$$

Here it is assumed that the real numbers $\mathbb{R}$ together with the relations $=_{\mathbb{R}}$ and $\le_{\mathbb{R}}$ are available. $\mathbb{D}$ is the base type of signed digits $\{-1, 0, 1\}$. The dual ${}^{\mathrm{co}}I$ is defined by its closure axiom ${}^{\mathrm{co}}I^-$:

$$\forall_x(x \in {}^{\mathrm{co}}I \to$$
$$\exists^{\mathrm{r}}_{d,x',y}(d \in \mathbb{D} \wedge x' \in \mathbb{R} \wedge |x'| \le_{\mathbb{R}} 1 \wedge x' \in {}^{\mathrm{co}}I \wedge y =_{\mathbb{R}} \frac{x'+d}{2} \wedge x =_{\mathbb{R}} y)$$

and its greatest-fixed-point (or coinduction) axiom ${}^{\mathrm{co}}I^+$:

$$\forall_x(x \in X \to \exists^{\mathrm{r}}_{d,x',y}($$
$$d \in \mathbb{D} \wedge x' \in \mathbb{R} \wedge |x'| \le_{\mathbb{R}} 1 \wedge (x' \in {}^{\mathrm{co}}I \cup X) \wedge y =_{\mathbb{R}} \frac{x'+d}{2} \wedge x =_{\mathbb{R}} y)) \to$$
$$X \subseteq {}^{\mathrm{co}}I).$$

REMARK. For n.c. inductive or coinductive predicates the axioms are formed as in the c.r. case, using $\vee^{\mathrm{nc}}$ for the closure axiom of ${}^{\mathrm{co}}I^{\mathrm{nc}}$. But there is an important restriction: for I^{nc} with more than one clause the elimination axiom $(I^{\mathrm{nc}})^-$ can only be used with a *non-computational* competitor predicate. This is needed in the proof of the soundness theorem. However, this restriction does not apply to I^{nc} defined by one clause only. Important examples of such *one-clause-nc* inductive predicates are Leibniz equality and the non-computational variants of the existential quantifier and of conjunction.

Generally, an inductive predicate is always contained in its dual.

LEMMA 3.3.3. $I \subseteq {}^{\mathrm{co}}I$, $I^{\mathrm{nc}} \subseteq {}^{\mathrm{co}}I^{\mathrm{nc}}$.

PROOF. The least-fixed-point axiom (13) for I (i.e., I^-) is equivalent to

$$\forall_{\vec{x}}(\bigvee_{i<k}\exists_{\vec{x}_i}(\bigwedge_{\nu<n_i} A_{i\nu}(I\cap X)\land \vec{x}\equiv\vec{t}_i)\to X\vec{x}\,)\to I\subseteq X.$$

It suffices that its premise holds with ${}^{\mathrm{co}}I$ for X. This follows from the greatest-fixed-point axiom (15) (i.e., ${}^{\mathrm{co}}I^+$), with the competitor predicate

$$X:=\{\,\vec{x}\mid \bigvee_{i<k}\exists_{\vec{x}_i}(\bigwedge_{\nu<n_i} A_{i\nu}(I\cap {}^{\mathrm{co}}I)\land \vec{x}\equiv\vec{t}_i)\,\}.$$

This means that we have to show the premise of (15) with this X, i.e.,

$$\forall_{\vec{x}}(X\vec{x}\to \bigvee_{i<k}\exists_{\vec{x}_i}(\bigwedge_{\nu<n_i} A_{i\nu}({}^{\mathrm{co}}I\cup X)\land \vec{x}\equiv\vec{t}_i)).$$

But if we unfold the premise $X\vec{x}$, this follows from $I\cap {}^{\mathrm{co}}I\subseteq {}^{\mathrm{co}}I\cup X$. For I^{nc} the proof is similar. □

REMARK. In case of an inductive predicate with non-recursive clauses only also the reverse inclusions ${}^{\mathrm{co}}I\subseteq I$, ${}^{\mathrm{co}}I^{\mathrm{nc}}\subseteq I^{\mathrm{nc}}$. Hence it is not necessary to consider ${}^{\mathrm{co}}I$. Examples are the inductively defined logical connectives $\exists$, $\vee$, $\wedge$ and Leibniz equality.

LEMMA 3.3.4. $I\subseteq I^{\mathrm{nc}}$, ${}^{\mathrm{co}}I\subseteq {}^{\mathrm{co}}I^{\mathrm{nc}}$.

PROOF. Let $I:=\mu_X(\forall_{\vec{x}_i}((A_{i\nu}(X))_{\nu<n_i}\to X\vec{t}_i))_{i<k}$.

For $I\subseteq I^{\mathrm{nc}}$ we use the elimination axiom (13) with I^{nc} as competitor predicate:

$$(\forall_{\vec{x}_i}((A_{i\nu}(I\cap I^{\mathrm{nc}}))_{\nu<n_i}\to I^{\mathrm{nc}}\vec{t}_i))_{i<k}\to I\subseteq I^{\mathrm{nc}}.$$

It suffices to prove the premises. Let $i<k$, fix $\vec{x}_i$ and assume $A_{i\nu}(I\cap I^{\mathrm{nc}})$ for all $\nu<n_i$. Since $A_{i\nu}(X)$ is strictly positive in X we obtain $A_{i\nu}(I^{\mathrm{nc}})$ for all $\nu<n_i$ and hence $I^{\mathrm{nc}}\vec{x}_i$ by $(I^{\mathrm{nc}})^+_i$.

For ${}^{\mathrm{co}}I\subseteq {}^{\mathrm{co}}I^{\mathrm{nc}}$ we use the greatest-fixed-point axiom for ${}^{\mathrm{co}}I^{\mathrm{nc}}$ with ${}^{\mathrm{co}}I$ as competitor predicate:

$$\forall_{\vec{x}}({}^{\mathrm{co}}I\vec{x}\to \bigvee_{i<k}\exists_{\vec{x}_i}(\bigwedge_{\nu<n_i} A_{i\nu}({}^{\mathrm{co}}I^{\mathrm{nc}}\cup {}^{\mathrm{co}}I)\land \vec{x}\equiv\vec{t}_i))\to {}^{\mathrm{co}}I\subseteq {}^{\mathrm{co}}I^{\mathrm{nc}}.$$

It suffices to prove the premise, which again follows from the fact that $A_{i\nu}(X)$ is strictly positive in X. □

Now we are ready to generalize Lemma 3.3.2 on Ex-falso for EqD and $T_{\mathbb{N}}$ to arbitrary predicates and formulas. However, we have to take care of the following issues:

- Predicate variables might occur as non "strictly positive parts".
- (Co)inductive predicates as strictly positive parts might not have a nullary clause.

DEFINITION. The notion of a strictly positive part (s.p.p) of a predicate or formula C is defined inductively.

(a) C is a strictly positive part of C.
(b) If I is a strictly positive part of C and A a premise of a non-recursive clause of I then A is a strictly positive part of C.
(c) If $\{\,\vec{x} \mid A(\vec{x})\,\}$ is a strictly positive part of C then so is $A(\vec{t}\,)$.
(d) If $A \to B$ is a strictly positive part of C then so is B.
(e) If $\forall_{\hat{x}} A(\hat{x})$ is a strictly positive part of C then so is $A(t)$.

THEOREM 3.3.5 (Ex-falso-quodlibet). *Let C be a predicate or formula.* TCF *proves*

$$\begin{cases} \forall_{\vec{\hat{x}}}(\mathbf{F} \to P\vec{\hat{x}}\,) & \text{if } C \text{ is a predicate } P \\ \mathbf{F} \to A & \text{if } C \text{ is a formula } A \end{cases}$$

from assumptions

$$\begin{cases} \forall_{\vec{\hat{x}}}(\mathbf{F} \to Y\vec{\hat{x}}\,) & \text{if } Y \text{ is a predicate variable strictly positive in } C \\ \forall_{\vec{\hat{x}}}(\mathbf{F} \to I\vec{\hat{x}}\,) & \text{if } I \text{ has no non-recursive clause and is a s.p.p. of } C \end{cases}$$

PROOF. By Lemma 3.3.2 we have $\mathrm{Ef}_{\mathrm{EqD}} \colon \mathbf{F} \to \hat{x}^\tau \equiv \hat{y}^\tau$. The claim can now be proved by induction on C.

Case $I\vec{s}$. If I has no non-recursive clause we can use the assumption $\forall_{\vec{\hat{x}}}(\mathbf{F} \to I\vec{\hat{x}}\,)$. Otherwise let K_i be a non-recursive clause, with final conclusion $I\vec{t}$. By induction hypothesis from $\mathbf{F}$ we can derive all parameter premises. Hence $I\vec{t}$. From $\mathbf{F}$ we also obtain $s_i \equiv t_i$, by the remark above. Hence $I\vec{s}$ by compatibility.

Case ${}^{\mathrm{co}}I\vec{s}$. Use Lemma 3.3.3. The cases $Y\vec{s}$, $A \to B$ and $\forall_{\hat{x}} A$ are obvious. □

3.4. Equality and extensionality

Equality at closed base types of level 0 is easy to handle. For simplicity we only consider the type $\mathbb{Y}$ of binary trees. Recall that our theory TCF has an intended model, determined by the ideals of the information systems $\boldsymbol{A}_\tau$. We have seen in the Bisimilarity Lemma 2.2.1 (on page 19) that for closed base types bisimilarity implies equality, which in TCF is formalized by the inductively defined Leibniz equality EqD. Therefore we take as an axiom:

AXIOM (Bisimilarity). *For every closed base type bisimilarity implies Leibniz equality.*

This axiom is justified by the fact that it holds in our intended model. As a consequence we can prove in TCF

PROPOSITION 3.4.1 (Characterization of equality at $T_{\mathbb{Y}}$ and ${}^{\mathrm{co}}T_{\mathbb{Y}}$).

(a) $\forall_{x,x'}(x \sim_{\mathbb{Y}} x' \leftrightarrow x, x' \in T_{\mathbb{Y}} \wedge x \equiv x')$.
(b) $\forall_{x,x'}(x \approx_{\mathbb{Y}} x' \leftrightarrow x, x' \in {}^{\mathrm{co}}T_{\mathbb{Y}} \wedge x \equiv x')$.

PROOF. (b). The proof of Proposition 2.2.2 relies on Lemma 2.2.1, which we just added as an axiom. The rest of the proof uses poperties of ${}^{\mathrm{co}}T_{\mathbb{Y}}$ and $\approx_{\mathbb{Y}}$ available in TCF.

(a). Similar to (b), using $T^{\pm}_{\mathbb{Y}}$, $\sim^{\pm}_{\mathbb{Y}}$ instead. For the proof of $x \sim_{\mathbb{Y}} x' \to x \equiv x'$ use (b) and $\sim_{\mathbb{Y}} \subseteq \approx_{\mathbb{Y}}$ (which follows from Lemma 3.3.3). □

This characterization of equality at $T_{\mathbb{Y}}$ and ${}^{\mathrm{co}}T_{\mathbb{Y}}$ is useful because it gives us a tool (induction, coinduction) to prove equalities $t \equiv t'$, which otherwise would be difficult.

COROLLARY 3.4.2.
(a) $\forall_x(x \sim_{\mathbb{Y}} x \leftrightarrow x \in T_{\mathbb{Y}})$.
(b) $\forall_x(x \approx_{\mathbb{Y}} x \leftrightarrow x \in {}^{\mathrm{co}}T_{\mathbb{Y}})$.

PROOF. Immediate from Proposition 3.4.1. □

COROLLARY 3.4.3.
(a) $\sim_{\mathbb{Y}}$ *is an equivalence relation on* $T_{\mathbb{Y}}$.
(b) $\approx_{\mathbb{Y}}$ *is an equivalence relation on* ${}^{\mathrm{co}}T_{\mathbb{Y}}$.

PROOF. Immediate from Proposition 3.4.1. □

REMARK. For closed base types like $\mathbb{Y}$ we can also relate $\sim_{\mathbb{Y}}$ to the binary boolean-valued function $=_{\mathbb{Y}}\colon \mathbb{Y} \to \mathbb{Y} \to \mathbb{B}$ defined in Section 2.3.1. One easily proves that

$$\forall_x(x \in T_{\mathbb{Y}} \to x = x),$$
$$\forall_x(x \in T_{\mathbb{Y}} \to \forall_y(y \in T_{\mathbb{Y}} \to x = y \to x \sim_{\mathbb{Y}} y)).$$

Usage of $=_{\mathbb{Y}}$ has the advantage that proofs may become shorter, since we identify terms with a common reduct. Pointwise equality $\doteq_{\mathbb{Y}}$ is defined to be $\sim_{\mathbb{Y}}$.

Up to now we have mainly dealt with base types. However, our theory TCF allows function types as well. We extend the notions of totality and pointwise equality from base types to function types. For simplicity we only consider parameter-free types.

DEFINITION (Totality and pointwise equality for function types).

$$(f \in T_{\tau\to\sigma}) := \forall_x(x \in T_\tau \to fx \in T_\sigma),$$
$$(f \doteq_{\tau\to\sigma} g) := f, g \in T_{\tau\to\sigma} \wedge \forall_{x,y}(x \doteq_\tau y \to fx \doteq_\sigma gy).$$

Extensionality is defined as diagonalization of pointwise equality:

DEFINITION (Extensionality).

$$(x \in \mathrm{Ext}_\tau) := (x \doteq_\tau x).$$

EXAMPLE (A non-extensional functional). Define f, g of type $\mathbb{N} \to \mathbb{N}$ by the computation rules $fn = 0$ and $g0 = 0$, $g(Sn) = gn$. Then $f\bot_{\mathbb{N}} = 0$ because of the computation rules for f. For $g\bot_{\mathbb{N}}$ no computation rule fits, but because of the inductive definition of $(\vec{U}, a) \in [\![\lambda_{\vec{x}} M]\!]$ in Section 2.4 (page 77) $[\![g\bot_{\mathbb{N}}]\!]$ is the empty ideal $[\![\bot_{\mathbb{N}}]\!]$. Hence $f \doteq g$, i.e., $f, g \in T_{\mathbb{N}\to\mathbb{N}}$ and $\forall_{n,m}(n \doteq_{\mathbb{N}} m \to fn \doteq_{\mathbb{N}} gm)$. The latter holds since $n \doteq_{\mathbb{N}} m$ implies $n \in T_{\mathbb{N}}$ and $n \equiv m$. Therefore the functional F defined by $Fh = h\bot_{\mathbb{N}}$ maps the pointwise equal f, g to different values.

By Corollary 3.4.2 (page 45) we know the equivalence of $\mathrm{Ext}_{\mathbb{Y}}$ and $T_{\mathbb{Y}}$; this also holds for arbitrary closed base types. This equivalence can be extended to closed types of level 1:

LEMMA 3.4.4. *The predicates* Ext_τ *and* T_τ *are equivalent for closed types of level* ≤ 1.

PROOF. For closed base types this has been proved in Corollary 3.4.2 (for the special case of the base type $\mathbb{Y}$). In case of level 1 we use induction on the height of the type, defined by

$$|\tau \to \sigma| := 1 + \max\{|\tau|, |\sigma|\}$$

Let $\tau \to \sigma$ be a closed type of level 1. The following are equivalent.

$f \in \mathrm{Ext}_{\tau\to\sigma}$
$f \doteq_{\tau\to\sigma} f$
$\forall_{x,y}(x \doteq_\tau y \to fx \doteq_\sigma fy)$
$\forall_{x\in T_\tau}(fx \doteq_\sigma fx)$ by Corollary 3.4.2, since $\mathrm{lev}(\tau) = 0$
$\forall_{x\in T_\tau}(fx \in \mathrm{Ext}_\sigma)$.

By induction hypothesis the final formula is equivalent to $f \in T_{\tau\to\sigma}$. □

For arbitrary closed types τ the relation $\doteq_\tau$ is a "partial equivalence relation", which means the following.

LEMMA 3.4.5. *For every closed type* τ *the relation* $\doteq_\tau$ *is an equivalence relation on* Ext_τ.

PROOF. By induction on the height $|\tau|$ of τ. *Case* ι. Use Corollary 3.4.3.

Case $\tau \to \sigma$. We first prove symmetry of $\doteq_{\tau\to\sigma}$. Let $f \doteq_{\tau\to\sigma} g$. The goal is $g \doteq_{\tau\to\sigma} f$. Assume $x \doteq_\tau y$. The goal now is $gx \doteq_\sigma fy$. From $x \doteq_\tau y$ we obtain $y \doteq_\tau x$ by symmetry of $\doteq_\tau$, hence $fy \doteq_\sigma gx$ from $f \doteq_{\tau\to\sigma} g$, hence $gx \doteq_\sigma fy$ by symmetry of $\doteq_\sigma$.

We finally prove transitivity of $\doteq_{\tau\to\sigma}$. Let $f \doteq_{\tau\to\sigma} g$ and $g \doteq_{\tau\to\sigma} h$. The goal is $f \doteq_{\tau\to\sigma} h$. Assume $x \doteq_\tau y$. The goal now is $fx \doteq_\sigma hy$. From $x \doteq_\tau y$ we obtain $y \doteq_\tau x$ by symmetry of $\doteq_\tau$, hence $x \doteq_\tau x$ by transitivity of $\doteq_\tau$. Then $fx \doteq_\sigma gx$ follows from $f \doteq_{\tau\to\sigma} g$. We also have $gx \doteq_\sigma hy$ from $g \doteq_{\tau\to\sigma} h$. Using transitivity of $\doteq_\sigma$ we obtain $fx \doteq_\sigma hy$. □

LEMMA 3.4.6 (Compatibility of terms). *For every term $t(\vec{x}\,)$ with extensional constants and free variables among $\vec{x}$ we have*

$$\forall_{\vec{x},\vec{y}}(\vec{x} \doteq_{\vec{\rho}} \vec{y} \to t(\vec{x}\,) \doteq_\tau t(\vec{y}\,)).$$

PROOF. This is proved by induction on t. *Case* x. Immediate. *Case* c. By assumption $c \doteq_\tau c$. *Case* $\lambda_x t(x,\vec{x}\,)$. Let $\vec{x} \doteq_{\vec{\rho}} \vec{y}$. The goal is $\lambda_x t(x,\vec{x}\,) \doteq_{\tau\to\sigma} \lambda_x t(x,\vec{y}\,)$, which by definition means

$$\forall_{x,y}(x \doteq_\tau y \to t(x,\vec{x}\,) \doteq_\sigma t(y,\vec{y}\,)).$$

Assume $x \doteq_\tau y$. With $\vec{x} \doteq_{\vec{\rho}} \vec{y}$ the claim $t(x,\vec{x}\,) \doteq_\sigma t(y,\vec{y}\,)$ holds by the IH. *Case* $t(\vec{x}\,)s(\vec{x}\,)$. Let $\vec{x} \doteq_{\vec{\rho}} \vec{y}$. By IH we have $t(\vec{x}\,) \doteq_{\tau\to\sigma} t(\vec{y}\,)$, i.e.,

$$\forall_{x,y}(x \doteq_\tau y \to t(\vec{x}\,)x \doteq_\sigma t(\vec{y}\,)y).$$

Again by IH we have $s(\vec{x}\,) \doteq_\tau s(\vec{y}\,)$. Hence $t(\vec{x}\,)s(\vec{x}\,) \doteq_\sigma t(\vec{y}\,)s(\vec{y}\,)$. □

LEMMA 3.4.7 (Extensionality of terms). *For every term $t(\vec{x}\,)$ with extensional constants and free variables among $\vec{x}$ we have*

$$\forall_{\vec{x}}(\vec{x} \in \mathrm{Ext}_{\vec{\rho}} \to t(\vec{x}\,) \in \mathrm{Ext}_\tau).$$

PROOF. Let $t(\vec{x}\,)$ with free variables among $\vec{x}$ be given, and assume $\vec{x} \in \mathrm{Ext}_{\vec{\rho}}$. By Lemma 3.4.6 applied to $\vec{x},\vec{x}$ we obtain $t(\vec{x}\,) \doteq_\tau t(\vec{x}\,)$, hence $t(\vec{x}\,) \in \mathrm{Ext}_\tau$. □

CHAPTER 4

Computational content of proofs

We have already mentioned that (co)inductive predicates can be declared as either computationally relevant (c.r.) or non-computational (n.c.). But what is the computational content in the c.r. case? We first address this question for (co)inductive predicates, and then extend it to arbitrary formulas. Next we study in what sense a proof of a c.r. formula A provides us with concrete computational content. This can be seen as a "witness" for the validity of A, or – in the sense of Kolmogorov (1932) – a "solution" to problem A.

Finally we take a step back and reflect on what we have done. We formally define what it means for a term to "realize" the c.r. formula A. We extract from a proof M of A a term $\mathrm{et}(M)$ and (again formally) prove that it is a realizer of A. In this proof we need "invariance axioms" stating that every c.r. formula not involving realizability is invariant under realizability, formally $A \leftrightarrow \exists_z(z\ \mathbf{r}\ A)$, where $z\ \mathbf{r}\ A$ means "z realizes A".

4.1. Realizers

Assume that we have a global assignment giving for every c.r. predicate variable X of arity $\vec{\rho}$ an n.c. predicate variable $X^{\mathbf{r}}$ of arity $(\vec{\rho}, \xi)$ where ξ is the type variable associated with X. We will also introduce $I^{\mathbf{r}}/{}^{\mathrm{co}}I^{\mathbf{r}}$ for (co)inductive predicates $I/{}^{\mathrm{co}}I$. A formula or predicate C is called $\mathbf{r}$-free if it does not contain any of these $X^{\mathbf{r}}$, $I^{\mathbf{r}}$ or ${}^{\mathrm{co}}I^{\mathbf{r}}$. A derivation M is called $\mathbf{r}$-free if it contains $\mathbf{r}$-free formulas only.

Definition ($C^{\mathbf{r}}$ for $\mathbf{r}$-free predicates and formulas C). For every $\mathbf{r}$-free predicate or formula C we define a predicate or formula $C^{\mathbf{r}}$. For n.c. C let $C^{\mathbf{r}} := C$. In case C is c.r. $C^{\mathbf{r}}$ is an n.c. predicate of arity $(\vec{\sigma}, \tau(C))$ with $\vec{\sigma}$ the arity of C. We often write $z\ \mathbf{r}\ C$ for $C^{\mathbf{r}}z$ in case C is a c.r. formula. For c.r. *predicates* X let $X^{\mathbf{r}}$ be the n.c. predicate variable provided, and

$$\{\, \vec{x} \mid A \,\}^{\mathbf{r}} := \{\, \vec{x}, z \mid z\ \mathbf{r}\ A \,\}.$$

Now consider a c.r. (co)inductive predicate

$$I/{}^{\mathrm{co}}I := (\mu/\nu)_X((K_i(X))_{i<k}$$

with associated base type ι_I given by the constructor types $(\kappa_i(\xi))_{i<k}$ where $\kappa_i(\xi) := \tau(K_i(X))$. The i-th constructor of ι_I is $\mathrm{C}_i\colon \kappa_i(\iota_I)$. Let s be a variable of type $\tau(I)$ and ϑ the substitution $\xi \mapsto \tau(I)$, $X^{\mathbf{r}} \mapsto \{\, \vec{x}, s \mid Y\vec{x}s \,\}$. We define n.c. predicates $I^{\mathbf{r}}$ and ${}^{\mathrm{co}}I^{\mathbf{r}}$ by

$$I^{\mathbf{r}}/{}^{\mathrm{co}}I^{\mathbf{r}} := (\mu/\nu)_Y((C_i\ \mathbf{r}\ K_i(X))\vartheta)_{i<k}.$$

The substitution ϑ is necessary since the arity of Y (and hence of $I^{\mathbf{r}}/{}^{\mathrm{co}}I^{\mathbf{r}}$) must be $(\vec{\rho}, \tau(I))$ and not $(\vec{\rho}, \xi)$. For c.r. *formulas* let

$$\begin{aligned} z\ \mathbf{r}\ P\vec{t} &:= P^{\mathbf{r}}\vec{t}z, \\ z\ \mathbf{r}\ (A \to B) &:= \begin{cases} \forall_w(w\ \mathbf{r}\ A \to zw\ \mathbf{r}\ B) & \text{if } A \text{ is c.r.} \\ A \to z\ \mathbf{r}\ B & \text{if } A \text{ is n.c.} \end{cases} \\ z\ \mathbf{r}\ \forall_x A &:= \forall_x(z\ \mathbf{r}\ A). \end{aligned}$$

Example. As an easy example for the construction of $I^{\mathbf{r}}$ consider the predicate Even, defined by $\mu_X(K_0(X), K_1(X))$ with $K_0(X) := (0 \in X)$ and $K_1(X) := \forall_n(n \in X \to S(Sn) \in X)$. The associated base type ι_{Even} is given by the constructor types $\kappa_0(\xi) := \xi$ and $\kappa_1(\xi) := \xi \to \xi$, i.e., $\iota_{\mathrm{Even}} = \mathbb{N}$ with constructors $\mathrm{C}_0 := 0$ and $\mathrm{C}_1 := S$. Let ϑ be the substitution $\xi \mapsto \mathbb{N}$, $X^{\mathbf{r}} \mapsto \{\, n, m \mid Ynm \,\}$. Since $S\ \mathbf{r}\ K_1(X)$ is $\forall_{n,m}(X^{\mathbf{r}}nm \to X^{\mathbf{r}}(S(Sn), Sm))$ we obtain

$$I^{\mathbf{r}} := \mu_Y(Y00, \forall_{n,m}(Ynm \to Y(S(Sn), Sm))).$$

We express Kolmogorov's view of formulas as problems by means of *invariance axioms*:

Axiom (Invariance under realizability). *For* $\mathbf{r}$*-free c.r. formulas* A *we require as axioms*

$$\mathrm{InvAll}_A\colon \forall_z(z\ \mathbf{r}\ A \to A). \tag{16}$$

$$\mathrm{InvEx}_A\colon A \to \exists_z(z\ \mathbf{r}\ A). \tag{17}$$

Realizers of totality and cototality predicates will be of special interest for us. Notice that the types $\tau(T_\iota)$ and $\tau({}^{\mathrm{co}}T_\iota)$ are both ι. Moreover we have

Lemma 4.1.1 (Realizers of totality). *For closed base types* ι *the following are equivalent.*

(a) $T_\iota^{\mathbf{r}}xy$,
(b) $x \sim_\iota^{\mathrm{nc}} y$,
(c) $x \in T_\iota^{\mathrm{nc}} \wedge x \equiv y$.

Proof. (a) $\leftrightarrow$ (b). Both $T_\iota^{\mathbf{r}}xy$ and $x \sim_\iota^{\mathrm{nc}} y$ satify the same clauses. Use the respective elimination axiom in each of the two directions.

(b) $\leftrightarrow$ (c). Use Corollary 3.4.2 (page 45). □

LEMMA 4.1.2 (Realizers of cototality). *For closed base types ι the following are equivalent.*

(a) ${}^{\mathrm{co}}T_\iota^{\mathbf{r}} xy$,
(b) $x \approx_\iota^{\mathrm{nc}} y$,
(c) $x \in {}^{\mathrm{co}}T_\iota^{\mathrm{nc}} \land x \equiv y$.

PROOF. As an example we give the proof for $\mathbb{N}$. Since we have n.c. goals only, decorations are omitted.

(a) $\to$ (b). We use the greatest-fixed-point axiom for $\approx_{\mathbb{N}}$:

$$\forall_{n,m}(Xnm \to (n \equiv 0 \land m \equiv 0) \lor \exists_{n',m'}((n' \approx_{\mathbb{N}} m' \lor Xn'm') \land n \equiv Sn' \land m \equiv Sm')) \to X \subseteq \approx_{\mathbb{N}}$$

and apply it with ${}^{\mathrm{co}}T_{\mathbb{N}}^{\mathbf{r}}$ for X. It suffices to prove the premise. Assume ${}^{\mathrm{co}}T_{\mathbb{N}}^{\mathbf{r}} nm$; the goal is

$$C := (n \equiv 0 \land m \equiv 0) \lor \exists_{n',m'}((n' \approx_{\mathbb{N}} m' \lor {}^{\mathrm{co}}T_{\mathbb{N}}^{\mathbf{r}} n'm') \land n \equiv Sn' \land m \equiv Sm').$$

By the closure axiom $({}^{\mathrm{co}}T_{\mathbb{N}}^{\mathbf{r}})^-$ we have

$$(n \equiv 0 \land m \equiv 0) \lor \exists_{n',m'}({}^{\mathrm{co}}T_{\mathbb{N}}^{\mathbf{r}} n'm' \land n \equiv Sn' \land m \equiv Sm').$$

We argue by cases (i.e., use $\lor^-$).

Case 1. $n \equiv 0 \land m \equiv 0$. Go for the l.h.s. of the disjunction C.

Case 2. $\exists_{n',m'}({}^{\mathrm{co}}T_{\mathbb{N}}^{\mathbf{r}} n'm' \land n \equiv Sn' \land m \equiv Sm')$. Go for the r.h.s. of C.

(b) $\to$ (a). Recall ${}^{\mathrm{co}}T_{\mathbb{N}} := \nu_X(0 \in X,\ \forall_{n \in X}(Sn \in X))$, hence by definition

$${}^{\mathrm{co}}T_{\mathbb{N}}^{\mathbf{r}} := \nu_{X^{\mathbf{r}}}(X^{\mathbf{r}}00,\ \forall_{n,m}(X^{\mathbf{r}}nm \to X^{\mathbf{r}}(Sn)(Sm))).$$

We need to show $n \approx_{\mathbb{N}} m \to {}^{\mathrm{co}}T_{\mathbb{N}}^{\mathbf{r}} nm$. To this end we use the greatest-fixed-point axiom for ${}^{\mathrm{co}}T_{\mathbb{N}}^{\mathbf{r}}$:

$$\forall_{n,m}(Xnm \to (n \equiv 0 \land m \equiv 0) \lor \exists_{n',m'}(n', m' \in ({}^{\mathrm{co}}T_{\mathbb{N}}^{\mathbf{r}} \cup X) \land n \equiv Sn' \land m \equiv Sm')) \to X \subseteq {}^{\mathrm{co}}T_{\mathbb{N}}^{\mathbf{r}}$$

and apply it with $\approx_{\mathbb{N}}$ for X. It suffices to prove the premise. Assume $n \approx_{\mathbb{N}} m$; the goal is

$$C := (n \equiv 0 \land m \equiv 0) \lor \exists_{n',m'}((n', m' \in ({}^{\mathrm{co}}T_{\mathbb{N}}^{\mathbf{r}} \cup \approx_{\mathbb{N}}) \land n \equiv Sn' \land m \equiv Sm')).$$

By the closure axiom $(\approx_{\mathbb{N}})^-$ we have

$$n \approx_{\mathbb{N}} m \to (n \equiv 0 \land m \equiv 0) \lor \exists_{n',m'}(n' \approx_{\mathbb{N}} m' \land n \equiv Sn' \land m \equiv Sm').$$

We argue by cases (i.e., use $\lor^-$).

Case 1. $n \equiv 0 \land m \equiv 0$. Go for the l.h.s. of the disjunction C.

Case 2. $\exists_{n',m'}(n' \approx_{\mathbb{N}} m' \land n \equiv Sn' \land m \equiv Sm')$. Go for the r.h.s. of C.

(b) $\leftrightarrow$ (c). Use the Bisimilarity axiom and Proposition 3.4.1 (page 44). □

Next we study what our general definition says about realizers for the c.r. inductively defined decorated connectives.

Recall that for the sum type $\rho+\sigma$ we had the constructors $(\mathrm{InL}_{\rho\sigma})^{\rho\to\rho+\sigma}$ and $(\mathrm{InR}_{\rho\sigma})^{\sigma\to\rho+\sigma}$. In the special situation that one of the two parameter types is the unit type $\mathbb{U}$ it is common to view the sum type $\mathbb{U}+\sigma$ as a unary algebra form, with constructors DummyL of type $\mathbb{U}+\sigma$ and Inr of type $\sigma\to\mathbb{U}+\sigma$. Similarly $\rho+\mathbb{U}$ is viewed as a unary algebra form, with constructors Inl of type $\rho\to\rho+\mathbb{U}$ and DummyR of type $\rho+\mathbb{U}$.

LEMMA 4.1.3 (Realizers for $\vee$). *$z\ \mathbf{r}\ (A\vee B)$ is equivalent to*

$\exists_x(x\ \mathbf{r}\ A\wedge z\equiv \mathrm{InL}(x))\vee^{\mathrm{nc}}\exists_y(y\ \mathbf{r}\ B\wedge z\equiv\mathrm{InR}(y))$ *for A,B c.r.*

$\exists_x(x\ \mathbf{r}\ A\wedge z\equiv \mathrm{Inl}(x))\vee^{\mathrm{nc}}(B\wedge z\equiv\mathrm{DummyR})$ *for A c.r. and B n.c.*

$(A\wedge z\equiv\mathrm{DummyL})\vee^{\mathrm{nc}}\exists_y(y\ \mathbf{r}\ B\wedge z\equiv\mathrm{Inr}(y))$ *for A n.c. and B c.r.*

$(A\wedge z\equiv \mathtt{tt})\vee^{\mathrm{nc}}(B\wedge z\equiv\mathtt{ff})$ *for A,B n.c.*

PROOF. As an example we consider the case A n.c. and B c.r. Recall $\mathrm{OrR}_{X^{\mathrm{nc}},Y^{\mathrm{c}}}:=\mu_Z(X^{\mathrm{nc}}\to Z,Y^{\mathrm{c}}\to Z)$. Then

$$\mathrm{OrR}^{\mathbf{r}}_{X^{\mathrm{nc}},Y^{\mathbf{r}}}:=\mu_{Z^{\mathbf{r}}}(X^{\mathrm{nc}}\to\mathrm{DummyL}\in Z^{\mathbf{r}},\forall_y(y\ \mathbf{r}\ Y\to\mathrm{Inr}(y)\in Z^{\mathbf{r}})).$$

Now substituting X^{nc} by A and $Y^{\mathbf{r}}$ by $\{\,y\mid y\ \mathbf{r}\ B\,\}$ in the introduction axioms gives

$$(\mathrm{OrR}^{\mathbf{r}}_{A,\{y\mid y\mathbf{r}B\}})^+_0\colon A\to\mathrm{DummyL}\ \mathbf{r}\ (A\vee B),$$
$$(\mathrm{OrR}^{\mathbf{r}}_{A,\{y\mid y\mathbf{r}B\}})^+_1\colon\forall_y(y\ \mathbf{r}\ B\to\mathrm{Inr}(y)\ \mathbf{r}\ (A\vee B)).$$

This suffices for "$\leftarrow$": if $A\wedge z\equiv\mathrm{DummyL}$, then from $(\mathrm{OrR}^{\mathbf{r}}_{A,\{y\mid y\mathbf{r}B\}})^+_0$ we obtain $z\ \mathbf{r}\ (A\vee B)$, and if we have y with $y\ \mathbf{r}\ B$ and $z\equiv\mathrm{Inr}(y)$, then from $(\mathrm{OrR}^{\mathbf{r}}_{A,\{y\mid y\mathbf{r}B\}})^+_1$ we again obtain $z\ \mathbf{r}\ (A\vee B)$.

Conversely, the elimination axiom $(\mathrm{OrR}^{\mathbf{r}}_{X^{\mathrm{nc}},Y^{\mathbf{r}}})^-$ is

$$(X^{\mathrm{nc}}\to\mathrm{DummyL}\in Z)\to\forall_y(y\ \mathbf{r}\ Y\to\mathrm{Inr}(y)\in Z)\to\mathrm{OrR}^{\mathbf{r}}_{X^{\mathrm{nc}},Y^{\mathbf{r}}}\subseteq Z.$$

Substitute Z by $\{\,z\mid(A\wedge z\equiv\mathrm{DummyL})\vee^{\mathrm{nc}}\exists_y(y\ \mathbf{r}\ B\wedge z\equiv\mathrm{Inr}(y))\,\}$. Then with A for X^{nc} and $\{\,y\mid y\ \mathbf{r}\ B\,\}$ for $Y^{\mathbf{r}}$ the two premises become provable and we obtain

$$\forall_z(z\ \mathbf{r}\ (A\vee B)\to(A\wedge z\equiv\mathrm{DummyL})\vee^{\mathrm{nc}}\exists_y(y\ \mathbf{r}\ B\wedge z\equiv\mathrm{Inr}(y))).\qquad\square$$

Similarly we have

LEMMA 4.1.4 (Realizers for $\wedge$). *$z\ \mathbf{r}\ (A\wedge B)$ is equivalent to*

$z\equiv\langle\mathrm{lft}(z),\mathrm{rht}(z)\rangle\wedge(\mathrm{lft}(z)\ \mathbf{r}\ A)\wedge(\mathrm{rht}(z)\ \mathbf{r}\ B)$ *for A c.r. and B c.r.*

$(z\ \mathbf{r}\ A)\wedge B$ *for A c.r. and B n.c.*

$A\wedge(z\ \mathbf{r}\ B)$ *for A n.c. and B c.r.*

PROOF. *Case* A, B c.r. Recall $\mathrm{AndD}_{X^{\mathrm{c}},Y^{\mathrm{c}}} := \mu_{Z^{\mathrm{c}}}(X^{\mathrm{c}} \to Y^{\mathrm{c}} \to Z^{\mathrm{c}})$. Then

$$\mathrm{AndD}^{\mathbf{r}}_{X^{\mathbf{r}},Y^{\mathbf{r}}} := \mu_{Z^{\mathbf{r}}}(\forall_x(x \in X^{\mathbf{r}} \to \forall_y(y \in Y^{\mathbf{r}} \to \langle x, y\rangle \in Z^{\mathbf{r}}))).$$

Now substituting $X^{\mathbf{r}}$ by $\{\, x \mid x \;\mathbf{r}\; A \,\}$ and $Y^{\mathbf{r}}$ by $\{\, y \mid y \;\mathbf{r}\; B \,\}$ in the introduction axiom gives

$$(\mathrm{AndD}^{\mathbf{r}}_{\{x|x\mathbf{r}A\},\{y|y\mathbf{r}B\}})^+_0 \colon \forall_x((x \;\mathbf{r}\; A) \to \forall_y(y \;\mathbf{r}\; B \to \langle x, y\rangle \;\mathbf{r}\; (A \wedge B))).$$

This suffices for "$\leftarrow$". Conversely, the elimination axiom $(\mathrm{AndD}^{\mathbf{r}}_{X^{\mathbf{r}},Y^{\mathbf{r}}})^-$ is

$$\forall_x(x \in X^{\mathbf{r}} \to \forall_y(y \in Y^{\mathbf{r}} \to \langle x, y\rangle \in Z)) \to \mathrm{AndD}^{\mathbf{r}}_{X^{\mathbf{r}},Y^{\mathbf{r}}} \subseteq Z.$$

Substitute Z by $\{\, z \mid z \equiv \langle \mathrm{lft}(z), \mathrm{rht}(z)\rangle \wedge (\mathrm{lft}(z) \;\mathbf{r}\; A) \wedge (\mathrm{rht}(z) \;\mathbf{r}\; B) \,\}$. Then with $\{\, x \mid x \;\mathbf{r}\; A \,\}$ for $X^{\mathbf{r}}$ and $\{\, y \mid y \;\mathbf{r}\; B \,\}$ for $Y^{\mathbf{r}}$ the premise is provable and we obtain

$$\forall_z(z \;\mathbf{r}\; (A \wedge B) \to z \equiv \langle \mathrm{lft}(z), \mathrm{rht}(z)\rangle \wedge (\mathrm{lft}(z) \;\mathbf{r}\; A) \wedge (\mathrm{rht}(z) \;\mathbf{r}\; B)).$$

Case A c.r. and B n.c. Recall $\mathrm{AndL}_{X^{\mathrm{c}},Y^{\mathrm{nc}}} := \mu_{Z^{\mathrm{c}}}(X^{\mathrm{c}} \to Y^{\mathrm{nc}} \to Z^{\mathrm{c}})$. Then

$$\mathrm{AndL}^{\mathbf{r}}_{X^{\mathbf{r}},Y^{\mathrm{nc}}} := \mu_{Z^{\mathbf{r}}}(\forall_z(z \;\mathbf{r}\; X \to Y^{\mathrm{nc}} \to z \in Z^{\mathbf{r}})).$$

Now substituting $X^{\mathbf{r}}$ by $\{\, z \mid z \;\mathbf{r}\; A \,\}$ and Y^{nc} by B in the introduction axiom gives

$$(\mathrm{AndL}^{\mathbf{r}}_{\{z|z\mathbf{r}A\},B})^+_0 \colon \forall_z((z \;\mathbf{r}\; A) \to B \to z \;\mathbf{r}\; (A \wedge B)).$$

This suffices for "$\leftarrow$". Conversely, the elimination axiom $(\mathrm{AndL}^{\mathbf{r}}_{X,Y^{\mathrm{nc}}})^-$ is

$$\forall_z(z \;\mathbf{r}\; X \to Y^{\mathrm{nc}} \to z \in Z) \to \mathrm{AndL}^{\mathbf{r}}_{X,Y^{\mathrm{nc}}} \subseteq Z.$$

Substitute Z by $\{\, z \mid (z \;\mathbf{r}\; A) \wedge B \,\}$. Then with $\{\, z \mid z \;\mathbf{r}\; A \,\}$ for X and B for Y^{nc} the premise is provable and we obtain

$$\forall_z(z \;\mathbf{r}\; (A \wedge B) \to (z \;\mathbf{r}\; A) \wedge B). \qquad \square$$

LEMMA 4.1.5 (Realizers for $\exists$). *$z \;\mathbf{r}\; \exists_x A \leftrightarrow \exists_x(z \;\mathbf{r}\; A)$ for A c.r.*

PROOF. Recall $\mathrm{Ex}_Y := \mu_X(\forall_x(x \in Y \to X))$. Then

$$\mathrm{Ex}^{\mathbf{r}}_{Y^{\mathbf{r}}} := \mu_{X^{\mathbf{r}}}(\forall_{x,z}(Y^{\mathbf{r}}xz \to X^{\mathbf{r}}z)).$$

Now substituting $Y^{\mathbf{r}}$ by $\{\, x, z \mid z \;\mathbf{r}\; A \,\}$ in the introduction axiom gives

$$(\mathrm{Ex}^{\mathbf{r}}_{\{x,z|z\mathbf{r}A\}})^+_0 \colon \forall_{x,z}(z \;\mathbf{r}\; A \to z \;\mathbf{r}\; \exists_x A)$$

Conversely, the elimination axiom $(\mathrm{Ex}^{\mathbf{r}}_{Y^{\mathbf{r}}})^-$ is

$$\forall_z(z \in \mathrm{Ex}^{\mathbf{r}}_{Y^{\mathbf{r}}} \to \forall_{x,z}(Y^{\mathbf{r}}xz \to z \in X) \to z \in X).$$

which is equivalent to

$$\forall_z(z \in \mathrm{Ex}^{\mathbf{r}}_{Y^{\mathbf{r}}} \to \forall_z(\exists_x Y^{\mathbf{r}}xz \to z \in X) \to z \in X).$$

Substituting X by $\{\, z \mid \exists_x(Y^{\mathbf{r}}xz)\,\}$ makes the middle part provable. Thus with $\{\, x, z \mid z\ \mathbf{r}\ A\,\}$ for $Y^{\mathbf{r}}$ we obtain $\forall_z(z\ \mathbf{r}\ \exists_x A \to \exists_x(z\ \mathbf{r}\ A))$ from $(\mathrm{Ex}^{\mathbf{r}}_{\{x,z|z\mathbf{r}A\}})^-$. □

4.2. Extracted terms, soundness

Let M be a proof in TCF of a c.r. formula A. Assume M is an $\mathbf{r}$-free proof, i.e., M contains no realizability predicates $I^{\mathbf{r}}$ or ${}^{\mathrm{co}}I^{\mathbf{r}}$. We define its *extracted term* $\mathrm{et}(M)$, of type $\tau(A)$, with the aim to express M's computational content. It will be a term built up from variables, constructors, recursion operators, destructors and corecursion operators by λ-abstraction and application.

Definition (Extracted term). For an $\mathbf{r}$-free proof M of a c.r. formula A we define its extracted term $\mathrm{et}(M)$ by

$$
\begin{aligned}
\mathrm{et}(u^A) &:= z_u^{\tau(A)} \quad (z_u^{\tau(A)} \text{ uniquely associated to } u^A),\\
\mathrm{et}((\lambda_{u^A} M^B)^{A\to B}) &:= \begin{cases} \lambda_{z_u}\mathrm{et}(M) & \text{if } A \text{ is c.r.}\\ \mathrm{et}(M) & \text{if } A \text{ is n.c.}\end{cases}\\
\mathrm{et}((M^{A\to B}N^A)^B) &:= \begin{cases} \mathrm{et}(M)\mathrm{et}(N) & \text{if } A \text{ is c.r.}\\ \mathrm{et}(M) & \text{if } A \text{ is n.c.}\end{cases}\\
\mathrm{et}((\lambda_x M^A)^{\forall_x A}) &:= \mathrm{et}(M),\\
\mathrm{et}((M^{\forall_x A(x)}t)^{A(t)}) &:= \mathrm{et}(M).
\end{aligned}
$$

It remains to define extracted terms for the axioms. Consider a (c.r.) inductively defined predicate I. For its introduction and elimination axioms define

$$
\begin{aligned}
\mathrm{et}(I_i^+) &:= \mathrm{C}_i,\\
\mathrm{et}(I^-) &:= \mathcal{R},
\end{aligned}
$$

where both the constructor C_i and the recursion operator $\mathcal{R}$ refer to the base type ι_I associated with I. For the closure and greatest-fixed-point axioms of ${}^{\mathrm{co}}I$ define

$$
\begin{aligned}
\mathrm{et}({}^{\mathrm{co}}I^-) &:= \mathcal{D},\\
\mathrm{et}({}^{\mathrm{co}}I_i^+) &:= {}^{\mathrm{co}}\mathcal{R},
\end{aligned}
$$

where again both the destructor $\mathcal{D}$ and the corecursion operator ${}^{\mathrm{co}}\mathcal{R}$ refer to the base type ι_I associated with I. For the elimination axiom $(I^{\mathrm{nc}})^-$ of a one-clause-nc inductive predicate with a c.r. competitor predicate the extracted term is the identity.

From the Soundness Theorem 4.2.1 below it will follow that the term extracted from a closed **r**-free proof of a c.r. formula A realizes A. As a preparation we first attend the axioms. Let I be an inductive predicate and ι_I its associated base type. One can show that the extracted term of $I^{\pm}$, ${}^{\mathrm{co}}I^{\pm}$ realizes the respective axiom[1]. Proofs of these facts are automatically generated in Minlog.

THEOREM 4.2.1 (Soundness). *Let M be an* **r***-free derivation of a formula A from assumptions $u_i\colon C_i$ $(i<n)$. Then we can derive*

$$\begin{cases} \mathrm{et}(M)\ \mathbf{r}\ A & \text{if } A \text{ is c.r.} \\ A & \text{if } A \text{ is n.c.} \end{cases}$$

from assumptions

$$\begin{cases} z_{u_i}\ \mathbf{r}\ C_i & \text{if } C_i \text{ is c.r.} \\ C_i & \text{if } C_i \text{ is n.c.} \end{cases}$$

PROOF. *Case* $u\colon A$. *Subcase* A c.r. Then $\mathrm{et}(u)=z_u$. *Subcase* A n.c. Immediate.

Case $c\colon A$. *Subcase* A c.r. The axioms have been treated above. *Subcase* A n.c. Immediate.

Case $(\lambda_{u^A} M^B)^{A\to B}$ with B c.r. We must derive $\mathrm{et}(\lambda_u M)\ \mathbf{r}\ (A\to B)$. To this end we distinguish subcases. *Subcase* A c.r. Then the goal

$$\forall_z(z\ \mathbf{r}\ A\to \mathrm{et}(M)(z)\ \mathbf{r}\ B)$$

follows from the induction hypothesis by $\to^+$ and $\forall^+$.

Subcase A n.c. Then the goal is

$$A\to \mathrm{et}(\lambda_u M)\ \mathbf{r}\ B.$$

Recall that $\mathrm{et}(\lambda_u M)=\mathrm{et}(M)$. By induction hypothesis we have a derivation of $\mathrm{et}(M)\ \mathbf{r}\ B$ from A, which is what we want.

Case $(\lambda_{u^A} M^B)^{A\to B}$ with B n.c. We need a derivation of $A\to B$.

Subcase A c.r. By induction hypothesis we have a derivation of B from $z\ \mathbf{r}\ A$. Using the invariance axiom $A\to\exists_z(z\ \mathbf{r}\ A)$ we obtain the required derivation of B from A as follows.

$$\dfrac{\dfrac{A\to\exists_z(z\ \mathbf{r}\ A)\qquad A}{\exists_z(z\ \mathbf{r}\ A)}\qquad \begin{array}{c}[z\ \mathbf{r}\ A]\\ \mid \mathrm{IH}\\ B\end{array}}{B}\ \exists^-$$

Subcase A n.c. By induction hypothesis we have a derivation of B from A, which is what we want.

[1] In Appendix C such proofs for some (co)inductive predicates are written out.

Case $(M^{A\to B}N^A)^B$ with B c.r. We need a derivation of $\mathrm{et}(MN)\ \mathbf{r}\ B$. To this end we distinguish subcases. *Subcase* A c.r. Then $\mathrm{et}(MN) = \mathrm{et}(M)\mathrm{et}(N)$. By induction hypothesis we have derivations of $\mathrm{et}(M)\ \mathbf{r}\ (A \to B)$ and hence of

$$\forall_z(z\ \mathbf{r}\ A\ \to\ \mathrm{et}(M)z\ \mathbf{r}\ B)$$

and of $\mathrm{et}(N)\ \mathbf{r}\ A$. This gives the claim. *Subcase* A n.c. Then $\mathrm{et}(MN) = \mathrm{et}(M)$. By induction hypothesis we have derivations of $\mathrm{et}(M)\ \mathbf{r}\ (A \to B)$ and hence of

$$A\ \to\ \mathrm{et}(M)\ \mathbf{r}\ B$$

and of A. Applying the former to the latter gives $\mathrm{et}(M)\ \mathbf{r}\ B$.

Case $(M^{A\to B}N^A)^B$ with B n.c. The goal is to find a derivation of B. *Subcase* A c.r. By induction hypothesis we have derivations of $A \to B$ and of $\mathrm{et}(N)\ \mathbf{r}\ A$. Now using the invariance axiom $\forall_z(z\ \mathbf{r}\ A \to A)$ we obtain the required derivation of B by $\to^-$ from the derivation of $A \to B$ and

$$\dfrac{\dfrac{\forall_z(z\ \mathbf{r}\ A \to A) \qquad \mathrm{et}(N)}{\mathrm{et}(N)\ \mathbf{r}\ A \to A} \qquad \begin{array}{l}\mid \mathrm{IH}\\ \mathrm{et}(N)\ \mathbf{r}\ A\end{array}}{A}$$

Subcase A n.c. By induction hypothesis we have derivations of $A \to B$ and of A, hence also a derivation of B.

Case $(\lambda_x M^A)^{\forall_x A}$ with $\forall_x A$ c.r. We need a derivation of $\mathrm{et}(\lambda_x M)\ \mathbf{r}\ \forall_x A$. By definition $\mathrm{et}(\lambda_x M) = \mathrm{et}(M)$. Hence we must derive

$$\mathrm{et}(M)\ \mathbf{r}\ \forall_x A, \quad \text{which is} \quad \forall_x(\mathrm{et}(M)\ \mathbf{r}\ A).$$

This follows from the induction hypothesis.

Case $(\lambda_x M^A)^{\forall_x A}$ with $\forall_x A$ n.c. By induction hypothesis we have a derivation of A. Apply $\forall^+$.

Case $(M^{\forall_x A(x)}t)^{A(t)}$ with $A(t)$ c.r. We must derive $\mathrm{et}(Mt)\ \mathbf{r}\ A(t)$. By definition $\mathrm{et}(Mt) = \mathrm{et}(M)$, and by induction hypothesis we can derive

$$\mathrm{et}(M)\ \mathbf{r}\ \forall_x A(x), \quad \text{which is} \quad \forall_x(\mathrm{et}(M)\ \mathbf{r}\ A(x)).$$

Case $(M^{\forall_x A(x)}t)^{A(t)}$ with $A(t)$ n.c. By induction hypothesis we have a derivation of $\forall_x A(x)$. Apply $\forall^-$. □

CHAPTER 5

Real analysis

We are interested in *exact real numbers*, as opposed to floating point numbers. The final goal is to develop the basics of real analysis in such a way that from a proof of an existence formula one can extract a program. For instance, from a proof of the intermediate value theorem we want to extract a program that, given an arbitrary error bound $\frac{1}{2^p}$, computes a rational x where the given function is zero up to the error bound.

5.1. Exact real arithmetic

5.1.1. Cauchy sequences, equality. We shall view a real as a Cauchy sequence of rationals with a separately given modulus.

DEFINITION 5.1.1. A real number x is a pair $((a_n)_{n\in\mathbb{N}}, M)$ with $a_n \in \mathbb{Q}$ and $M\colon \mathbb{P} \to \mathbb{N}$ such that $(a_n)_n$ is a *Cauchy sequence* with modulus M, that is

$$|a_n - a_m| \le \frac{1}{2^p} \quad \text{for } n, m \ge M(p)$$

and M is weakly increasing (that is $M(p) \le M(q)$ for $p \le q$). M is called *Cauchy modulus* of x.

We shall loosely speak of a real $(a_n)_n$ if the Cauchy modulus M is clear from the context or inessential. Every rational a is tacitly understood as the real represented by the constant sequence $a_n = a$ with the constant modulus $M(p) = 0$.

DEFINITION 5.1.2. Two reals $x := ((a_n)_n, M)$, $y := ((b_n)_n, N)$ are called *equivalent* (or *equal* and written $x = y$, if the context makes clear what is meant), if

$$|a_{M(p+1)} - b_{N(p+1)}| \le \frac{1}{2^p} \quad \text{for all } p \in \mathbb{P}.$$

We want to show that this is an equivalence relation. Reflexivity and symmetry are clear. For transitivity we use the following lemma:

LEMMA 5.1.3 (RealEqChar). *For reals $x := ((a_n)_n, M)$, $y := ((b_n)_n, N)$ the following are equivalent:*
(a) *$x = y$;*

(b) $\forall_p \exists_{n_0} \forall_{n \geq n_0} (|a_n - b_n| \leq \frac{1}{2^p})$.

PROOF. (a) implies (b). For $n \geq M(p+2), N(p+2)$ we have

$$\begin{aligned} |a_n - b_n| &\leq |a_n - a_{M(p+2)}| + |a_{M(p+2)} - b_{N(p+2)}| + |b_{N(p+2)} - b_n| \\ &\leq \frac{1}{2^{p+2}} + \frac{1}{2^{p+1}} + \frac{1}{2^{p+2}}. \end{aligned}$$

(b) implies (a). Let $q \in \mathbb{P}$, and $n \geq n_0, M(p+1), N(p+1)$ with n_0 provided for q by (b). Then

$$\begin{aligned} |a_{M(p+1)} - b_{N(p+1)}| &\leq |a_{M(p+1)} - a_n| + |a_n - b_n| + |b_n - b_{N(p+1)}| \\ &\leq \frac{1}{2^{p+1}} + \frac{1}{2^q} + \frac{1}{2^{p+1}}. \end{aligned}$$

The claim follows, because this holds for every $q \in \mathbb{P}$. □

REMARK 5.1.4 (RealSeqEqToEq). An immediate consequence is that any two reals with the same Cauchy sequence (but possibly different moduli) are equal.

LEMMA 5.1.5 (RealEqTrans). *Equality between reals is transitive.*

PROOF. Let $(a_n)_n$, $(b_n)_n$, $(c_n)_n$ be the Cauchy sequences for x, y, z. Assume $x = y$, $y = z$ and pick n_1, n_2 for $p+1$ according to the lemma above. Then $|a_n - c_n| \leq |a_n - b_n| + |b_n - c_n| \leq \frac{1}{2^{p+1}} + \frac{1}{2^{p+1}}$ for $n \geq n_1, n_2$. □

5.1.2. The Archimedian property. For every function on the reals we certainly want compatibility with equality. This however is not always the case; here is an important example.

LEMMA 5.1.6 (RealBound). *For every real $x := ((a_n)_n, M)$ we can find p_x such that $|a_n| \leq 2^{p_x}$ for all n.*

PROOF. Let $n_0 := M(1)$ and p_x be such that $\max\{\, |a_n| \mid n \leq n_0 \,\} + \frac{1}{2} \leq 2^{p_x}$. Then $|a_n| \leq 2^{p_x}$ for all n. □

Clearly this assignment of p_x to x is not compatible with equality.

5.1.3. Nonnegative and positive reals. A real $x := ((a_n)_n, M)$ is called *nonnegative* (written $x \in \mathbb{R}^{0+}$) if

$$-\frac{1}{2^p} \leq a_{M(p)} \quad \text{for all } p \in \mathbb{P}.$$

It is *p-positive* (written $x \in_p \mathbb{R}^+$, or $x \in \mathbb{R}^+$ if p is not needed) if

$$\frac{1}{2^p} \leq a_{M(p+1)}.$$

We want to show that both properties are compatible with equality. First we prove a useful characterization of nonnegative reals.

LEMMA 5.1.7 (RealNNegChar). *For a real $x := ((a_n)_n, M)$ the following are equivalent:*

(a) $x \in \mathbb{R}^{0+}$;
(b) $\forall_p \exists_{n_0} \forall_{n \geq n_0} (-\frac{1}{2^p} \leq a_n)$.

PROOF. (a) implies (b). For $n \geq M(p+1)$ we have

$$\begin{aligned} -\frac{1}{2^p} &\leq -\frac{1}{2^{p+1}} + a_{M(p+1)} \\ &= -\frac{1}{2^{p+1}} + (a_{M(p+1)} - a_n) + a_n \\ &\leq -\frac{1}{2^{p+1}} + \frac{1}{2^{p+1}} + a_n. \end{aligned}$$

(b) implies (a). Let $q \in \mathbb{P}$ and $n \geq n_0, M(p)$ with n_0 provided by (b) (for q). Then

$$\begin{aligned} -\frac{1}{2^p} - \frac{1}{2^q} &\leq -\frac{1}{2^p} + a_n \\ &= -\frac{1}{2^p} + (a_n - a_{M(p)}) + a_{M(p)} \\ &\leq -\frac{1}{2^p} + \frac{1}{2^p} + a_{M(p)}. \end{aligned}$$

The claim follows, because this holds for every q. □

LEMMA 5.1.8 (RealNNegCompat). *If $x = y$ and $x \in \mathbb{R}^{0+}$, then $y \in \mathbb{R}^{0+}$.*

PROOF. Let $x := ((a_n)_n, M)$ and $y := ((b_n)_n, N)$. Assume $x \in \mathbb{R}^{0+}$ and $x = y$, and let p be given. Pick n_0 according to the lemma above and n_1 according to the characterization of equality of reals in Lemma 5.1.3 (RealEqChar) (both for $p+1$). Then for $n \geq n_0, n_1$

$$-\frac{1}{2^p} \leq -\frac{1}{2^{p+1}} + a_n \leq (b_n - a_n) + a_n.$$

Hence $y \in \mathbb{R}^{0+}$ by definition. □

LEMMA 5.1.9 (RealPosChar). *For a real $x := ((a_n)_n, M)$ the following are equivalent:*

(a) $x \in_p \mathbb{R}^+ \to \forall_n (M(p+1) \leq n \to \frac{1}{2^{p+1}} \leq a_n)$.
(b) $\forall_{n \geq n_0} (\frac{1}{2^q} \leq a_n) \to x \in_{q+1} \mathbb{R}^+$.

PROOF. (a) implies (b). Assume $x \in_p \mathbb{R}^+$, that is $\frac{1}{2^p} \leq a_{M(p+1)}$. Then for $M(p+1) \leq n$ we have

$$\frac{1}{2^{p+1}} \leq -\frac{1}{2^{p+1}} + a_{M(p+1)} = -\frac{1}{2^{p+1}} + (a_{M(p+1)} - a_n) + a_n \leq a_n.$$

(b) implies (a). Assume $\forall_{n\geq n_0}(\frac{1}{2^q} \leq a_n)$.

$$\begin{aligned}\frac{1}{2^{q+1}} &< -\frac{1}{2^{q+2}} + \frac{1}{2^q} \\ &\leq -\frac{1}{2^{q+2}} + a_n && \text{for } n_0 \leq n \\ &\leq (a_{M(q+2)} - a_n) + a_n && \text{for } M(q+2) \leq n.\end{aligned}$$

Hence $x \in_{q+1} \mathbb{R}^+$. □

5.1.4. Arithmetical functions. Given real numbers $x := ((a_n)_n, M)$ and $y := ((b_n)_n, N)$, we define $x + y$, $-x$, $|x|$, $x \cdot y$, and $\frac{1}{x}$ (the latter only provided that $|x| \in_q \mathbb{R}^+$) as represented by the respective sequence (c_n) of rationals with modulus L:

	c_n	$L(p)$
$x + y$	$a_n + b_n$	$\max\big(M(p+1), N(p+1)\big)$
$-x$	$-a_n$	$M(p)$
$\lvert x\rvert$	$\lvert a_n\rvert$	$M(p)$
$x \cdot y$	$a_n \cdot b_n$	$\max\big(M(p+1+p_y), N(p+1+p_x)\big)$
$\frac{1}{x}$ for $\lvert x\rvert \in_q \mathbb{R}^+$	$\begin{cases}\frac{1}{a_n} & \text{if } a_n \neq 0 \\ 0 & \text{if } a_n = 0\end{cases}$	$M(2(q+1)+p)$

where 2^{p_x} is the upper bound provided by Lemma 5.1.6 (RealBound).

LEMMA 5.1.10. *For reals x, y also $x+y$, $-x$, $|x|$, $x\cdot y$ and (provided that $|x| \in_q \mathbb{R}^+$) also $1/x$ are reals.*

PROOF. We restrict ourselves to the cases $x \cdot y$ and $1/x$.

$$\begin{aligned}|a_n b_n - a_m b_m| &= |a_n(b_n - b_m) + (a_n - a_m)b_m| \\ &\leq |b_n - b_m| \cdot |a_n| + |a_n - a_m| \cdot |b_m| \\ &\leq |b_n - b_m| \cdot 2^{p_x} + |a_n - a_m| \cdot 2^{p_y} \leq \frac{1}{2^p}\end{aligned}$$

for $n, m \geq \max\big(M(p+1+p_y), N(p+1+p_x)\big)$.

For $1/x$ assume $|x| \in_q \mathbb{R}^+$. Then by the (proof of our) characterization of positivity in Lemma 5.1.9 (RealPosChar), $\frac{1}{2^{q+1}} \leq |a_n|$ for $n \geq M(q+1)$. Hence

$$\begin{aligned}\Big|\frac{1}{a_n} - \frac{1}{a_m}\Big| &= \frac{|a_m - a_n|}{|a_n a_m|} \\ &\leq 2^{2(q+1)}|a_m - a_n| && \text{for } n, m \geq M(q+1) \\ &\leq \frac{1}{2^p} && \text{for } n, m \geq M(2(q+1)+p).\end{aligned}$$

The claim now follows from the assumption that M is weakly increasing. □

LEMMA 5.1.11. *For reals* x, y, z

$$x + (y + z) = (x + y) + z \qquad x \cdot (y \cdot z) = (x \cdot y) \cdot z$$
$$x + 0 = x \qquad x \cdot 1 = x$$
$$x + (-x) = 0 \qquad 0 < |x| \to x \cdot \frac{1}{x} = 1$$
$$x + y = y + x \qquad x \cdot y = y \cdot x$$
$$x \cdot (y + z) = x \cdot y + x \cdot z$$

PROOF. For $0 < |x| \to x \cdot \frac{1}{x} = 1$ the Cauchy sequences are finally the same, which suffices. In all other cases both the Cauchy sequences and the moduli are the same, hence both sides are actually identical. □

LEMMA 5.1.12. *The functions* $x + y$, $-x$, $|x|$, $x \cdot y$ *and (provided that* $|x| \in_q \mathbb{R}^+$*) also* $1/x$ *are compatible with equality.*

PROOF. Routine. For instance in case $x + y$ because of the commutativity of $+$ it suffices to prove $x = y \to x + z = y + z$. But this follows immediately from Lemma 5.1.3 (RealEqChar): the n_0 for the conclusion can be the same as for the premise. □

LEMMA 5.1.13. *For reals* x, y *from* $x \cdot y = 1$ *we can infer* $0 < |x|$.

PROOF. Pick p such that $|b_n| \leq 2^p$ for all n. Pick n_0 such that $n_0 \leq n$ implies $\frac{1}{2} \leq a_n \cdot b_n$. Then $\frac{1}{2} \leq |a_n| \cdot 2^p$ for $n_0 \leq n$, and hence $\frac{1}{2^{p+1}} \leq |a_n|$. □

LEMMA 5.1.14. *For reals* x, y,

(a) $x, y \in \mathbb{R}^{0+} \to x + y, x \cdot y \in \mathbb{R}^{0+}$,
(b) $x, y \in \mathbb{R}^{+} \to x + y, x \cdot y \in \mathbb{R}^{+}$,
(c) $x \in \mathbb{R}^{0+} \to -x \in \mathbb{R}^{0+} \to x = 0$.

PROOF. (a), (b). Routine. (c). Let p be given. Pick n_0 such that $-\frac{1}{2^p} \leq a_n$ and $-\frac{1}{2^p} \leq -a_n$ for $n \geq n_0$. Then $|a_n| \leq \frac{1}{2^p}$. □

5.1.5. Comparison of reals. We write $x \leq y$ for $y - x \in \mathbb{R}^{0+}$ and $x < y$ for $y - x \in \mathbb{R}^{+}$. Unwinding the definitions yields that $x \leq y$ is to say that for every p, $a_{L(p)} \leq b_{L(p)} + \frac{1}{2^p}$ with $L(p) := \max(M(p), N(p))$, or equivalently (using Lemma 5.1.7 (RealNNegChar)) that for every p there exists n_0 such that $a_n \leq b_n + \frac{1}{2^p}$ for all $n \geq n_0$. Furthermore, $x < y$ is a shorthand for the presence of p with $a_{L(p+1)} + \frac{1}{2^p} \leq b_{L(p+1)}$ with L the maximum of M and N, or equivalently (using Lemma 5.1.9 (RealPosChar)) for the presence of p, q with $a_n + \frac{1}{2^p} \leq b_n$ for all $n \geq q$; we then write $x <_p y$ (or $x <_{p,q} y$) whenever we want to call these witnesses.

LEMMA 5.1.15 (RealPosLe). *If* $x \leq y$ *and* $x \in_p \mathbb{R}^+$, *then* $y \in_{p+2} \mathbb{R}^+$.

PROOF. Easy. Demo in Minlog. □

LEMMA 5.1.16 (RealPosSemiCompat). *If $x = y$ and $x \in_p \mathbb{R}^+$, then we have $y \in_{p+2} \mathbb{R}^+$.*

PROOF. Easy consequence of Lemma 5.1.15. Demo in Minlog. □

LEMMA 5.1.17 (RealApprox). $\forall_{x,p}\exists_a(|a - x| \le \frac{1}{2^p})$.

PROOF. Let $x = ((a_n), M)$. Given p, pick $a_{M(p)}$. We show $|a_{M(p)} - x| \le \frac{1}{2^p}$, that is $|a_{M(p)} - a_{M(q)}| \le \frac{1}{2^p} + \frac{1}{2^q}$ for every q. But this follows from

$$|a_{M(p)} - a_{M(q)}| \le |a_{M(p)} - a_{M(p+q)}| + |a_{M(p+q)} - a_{M(q)}| \le \frac{1}{2^p} + \frac{1}{2^q}. \quad \square$$

LEMMA 5.1.18. *For reals x, y, z,*

$$\begin{array}{ll} x \le x & x \not< x \\ x \le y \to y \le x \to x = y & x < y \to y < z \to x < z \\ x \le y \to y \le z \to x \le z & x < y \to x + z < y + z \\ x \le y \to x + z \le y + z & x < y \to 0 < z \to x \cdot z < y \cdot z \\ x \le y \to 0 \le z \to x \cdot z \le y \cdot z & \end{array}$$

PROOF. From Section 5.1.4. □

Here we have left out information on witnesses p for statements proving a $<$-formula. Such estimates can easily be given explicitly. Examples:

LEMMA 5.1.19 (RealPosPlus). $0 \le x \to 0 <_p y \to 0 <_{p+3} x + y$.

PROOF. From $0 \le x$ we have $\forall_q\exists_{n_0}\forall_{n\ge n_0}(-\frac{1}{2^q} \le a_n)$. From $0 <_p y$ we have some n_1 such that $\forall_{n\ge n_1}(\frac{1}{2^{p+1}} \le b_n)$. Pick n_0 for $p+2$. Then $n_0, n_1 \le n$ implies $0 \le a_n + \frac{1}{2^{p+2}}$ and $\frac{1}{2^{p+2}} \le b_n - \frac{1}{2^{p+2}}$, hence $\frac{1}{2^{p+2}} \le a_n + b_n$. Now Lemma 5.1.9 (RealPosChar) gives $0 <_{p+3} x + y$. □

LEMMA 5.1.20. $x \le y \to y <_p z \to x <_{p+5} z$.

PROOF. This follows from Lemma 5.1.19 (RealPosPlus). □

As is to be expected in view of the existential and universal character of the predicates $<$ and $\le$ on the reals, we have:

LEMMA 5.1.21 (LeIsNotGt). $x \le y \leftrightarrow y \not< x$.

PROOF. $\to$. Assume $x \le y$ and $y < x$. By Lemma 5.1.20 we obtain $x < x$, a contradiction.

$\leftarrow$. It clearly suffices to show $0 \not< z \to z \le 0$, for a real z given by $(c_n)_n$. Assume $0 \not< z$. We must show $\forall_p\exists_{n_0}\forall_{n\ge n_0}(c_n \le \frac{1}{2^p})$. Let p be given. By assumption $0 \not< z$, hence $\neg\exists_q(\frac{1}{2^q} \le c_{M(q+1)})$. For $q := p + 1$ this implies $c_{M(p+2)} < \frac{1}{2^{p+1}}$, hence $c_n \le c_{M(p+2)} + \frac{1}{2^{p+2}} < \frac{1}{2^p}$ for $M(p + 2) \le n$. □

Constructively, we cannot compare two reals, but we can compare every real with a nontrivial interval.

LEMMA 5.1.22 (ApproxSplit). *Let x, y, z be given and assume $x < y$. Then either $z \leq y$ or $x \leq z$.*

PROOF. Let $x := ((a_n)_n, M)$, $y := ((b_n)_n, N)$, $z := ((c_n)_n, L)$. Assume $x <_p y$, that is (by definition) $\frac{1}{2^p} \leq b_n - a_n$ for $n := \max(M(p+2), N(p+2))$. Let $m := \max(n, L(p+2))$.

Case $c_m \leq \frac{a_n+b_n}{2}$. We show $z \leq y$. It suffices to prove $c_l \leq b_l$ for $l \geq m$. This follows from

$$c_l \leq c_m + \frac{1}{2^{p+2}} \leq \frac{a_n + b_n}{2} + \frac{b_n - a_n}{4} = b_n - \frac{b_n - a_n}{4} \leq b_n - \frac{1}{2^{p+2}} \leq b_l.$$

Case $c_m \not\leq \frac{a_n+b_n}{2}$. We show $x \leq z$. This follows from $a_l \leq c_l$ for $l \geq m$:

$$a_l \leq a_n + \frac{1}{2^{p+2}} \leq a_n + \frac{b_n - a_n}{4} = \frac{a_n + b_n}{2} - \frac{b_n - a_n}{4} \leq c_m - \frac{1}{2^{p+2}} \leq c_l. \quad \square$$

Notice that the boolean object determining whether $z \leq y$ or $x \leq z$ depends on the representation of x, y and z. In particular this assignment is *not* compatible with our equality relation.

One might think that the non-available comparison of two reals could be circumvented by using a maximum function. Indeed, such a function can easily be defined (component-wise), and it has the expected properties $x, y \leq \max(x, y)$ and $x, y \leq z \to \max(x, y) \leq z$. But what is missing is the knowledge that $\max(x, y)$ equals one of its arguments, i.e., we do not have $\max(x, y) = x \vee \max(x, y) = y$.

However, in many cases it is sufficient to pick the up to ε largest real out of finitely many given ones. This is indeed possible. We give the proof for two reals; it can be easily generalized.

LEMMA 5.1.23 (Maximum of two reals). *Let $x := ((a_n)_n, M)$ and $y := ((b_n)_n, N)$ be reals, and $p \in \mathbb{P}$. Then either $x \leq y + \frac{1}{2^p}$ or else $y \leq x + \frac{1}{2^p}$.*

PROOF. Let $m := \max\big(M(p+1), N(p+1)\big)$.

Case $a_m \leq b_m$. Then for $m \leq n$

$$a_n \leq a_m + \frac{1}{2^{p+1}} \leq b_m + \frac{1}{2^{p+1}} \leq b_n + \frac{1}{2^p}.$$

This holds for all $n \geq m$, therefore $x \leq y + \frac{1}{2^p}$.

Case $b_m < a_m$. Then for $m \leq n$

$$b_n \leq b_m + \frac{1}{2^{p+1}} < a_m + \frac{1}{2^{p+1}} \leq a_n + \frac{1}{2^p}.$$

This holds for all $n \geq m$, therefore $y \leq x + \frac{1}{2^p}$. $\square$

5.2. Continuous functions

We consider real-valued functions defined on open, closed or half-open intervals $I \subseteq \mathbb{R}$. Let c^∞, d^∞ range over $\mathbb{R} \cup \{\pm\infty\}$.

DEFINITION 5.2.1. A *uniformly continuous function* $f\colon I \to \mathbb{R}$ is given by

$$h\colon (I \cap \mathbb{Q}) \to \mathbb{N} \to \mathbb{Q} \qquad \text{(called approximating map)},$$

together with further data:

(a) A map $\alpha\colon \mathbb{P} \to \mathbb{N}$ such that for $a \in I$ each $(h(a,n))_n$ is a Cauchy sequence with (uniform) modulus α.
(b) A modulus $\omega\colon \mathbb{P} \to \mathbb{P}$ of (uniform) continuity, such that $\omega(p)$ satisfies for all $a, b \in I$

$$|a-b| \le \frac{1}{2^{\omega(p)-1}} \to |h(a,n) - h(b,n)| \le \frac{1}{2^p} \quad \text{for } n \ge \alpha(p).$$

(c) Lower and upper bounds $\mu, \nu \in \mathbb{Q}$ for all $h(a,n)$ with $a \in I$.

We require $\alpha(p) \le \alpha(q)$ and $\omega(p) \le \omega(q)$ for all $p \le q$.

DEFINITION 5.2.2. A *continuous function* $g\colon (c^\infty, d^\infty) \to \mathbb{R}$ is given by an approximating map $h\colon ((c^\infty, d^\infty) \cap \mathbb{Q}) \to \mathbb{N} \to \mathbb{Q}$ and a family of uniformly continuous functions $(h\restriction[c,d], \alpha_{c.d}, \omega_{c.d}, \mu_{c.d}, \nu_{c.d})$ for $c^\infty \le c < d \le d^\infty$. We require for $c^\infty < c' \le c < d \le d' < d^\infty$ the monotonicity properties

$$\alpha_{c,d}(p) \le \alpha_{c',d'}(p), \quad \omega_{c,d}(p) \le \omega_{c',d'}(p), \quad \mu_{c',d'} \le \mu_{c,d}, \quad \nu_{c,d} \le \nu_{c',d'}.$$

EXAMPLE 5.2.3 (Squaring). $\mathsf{sq}\colon (-\infty, \infty) \to \mathbb{R}$ is a continuous function given by

(a) the approximating map $h(a,n) := a^2$ and modulus $\alpha_{c,d}(p) := 0$;
(b) the modulus $\omega_{c,d}(p) := p + q + 1$ of uniform continuity, where q is such that $|a+b| \le 2^q$ for $c \le a < b \le d$, because

$$|a-b| \le \frac{1}{2^{p+q}} \to |a^2 - b^2| = |(a-b)(a+b)| \le \frac{1}{2^p};$$

(c) the lower bound $\mu_{c,d} := c^2$ and upper bound $\nu_{c,d} := d^2$ in case $1 \le c$.

Similarly all polynomials with rational coefficients on finite intervals can be viewed as continuous functions in our sense.

EXAMPLE 5.2.4 (Inverse). $\mathsf{inv}\colon (0, \infty) \to \mathbb{R}$ inverting its argument is a continuous function given by the approximating map $h(a,n) := \frac{1}{a}$. It is an easy exercise to define for every compact interval $[\frac{1}{2^q}, d]$ its Cauchy modulus $\alpha_{c,d}(p)$, its modulus $\omega_{c,d}(p)$ of uniform continuity and the lower and upper bounds $\mu_{c,d}$ and $\nu_{c,d}$.

DEFINITION 5.2.5 (Localization $g\restriction p$ of a continuous function). Let a continuous function g be given by $c^\infty, d^\infty, h, \alpha^\infty, \omega^\infty, \mu^\infty, \nu^\infty$. For $p \in \mathbb{Z}^+$ we define $[c,d]$ by

$$[c,d] := \begin{cases} [-2^p, 2^p] & \text{if } c^\infty = -\infty \text{ and } d^\infty = +\infty \\ [c^\infty + \frac{1}{2^p}, 2^p] & \text{if } c^\infty \in \mathbb{R} \text{ and } d^\infty = +\infty \\ [-2^p, d^\infty - \frac{1}{2^p}] & \text{if } c^\infty = -\infty \text{ and } d^\infty \in \mathbb{R} \\ [c^\infty + \frac{1}{2^p}, d^\infty - \frac{1}{2^p}] & \text{if } c^\infty, d^\infty \in \mathbb{R} \end{cases}$$

provided p is large enough to make $[c,d]$ a proper interval. The *localization* $g\restriction p$ is defined to consist of c, d as above, $h\restriction[c,d]$ and

$$\alpha(p) := \alpha^\infty_{c,d}(p), \quad \omega(p) := \omega^\infty_{c,d}(p), \quad \mu(p) := \mu^\infty_{c,d}(p), \quad \nu(p) := \nu^\infty_{c,d}(p).$$

Since the approximating map operates on rationals only, we need to define separately what it means to apply a continuous function in our sense to a real. It suffices to do this for uniformly continuous functions. For continuous ones g we in addition need a witness p for elementhood of the argument x in g's open interval (c^∞, d^∞), i.e., (c,d) (depending on p, as above) such that $x \in (c,d)$. Then we can define the restriction $g\restriction p$ as a uniformly continuous function, and use application $(g\restriction p)(x)$. Notice that $(g\restriction p)(x)$ does not depend on p, since the approximating map h of g is independent of interval bounds c, d, and two real numbers are equal if their Cauchy sequences coincide from one point onwards.

DEFINITION 5.2.6 (Application). Let $f\colon [c,d] \to \mathbb{R}$ be a uniformly continuous function given by $c, d, h, \alpha, \omega, \mu, \nu$. Let further $x = ((a_n)_n, M)$ be an arbitrary real. The *application* $f(x)$ of f to x is defined to be the Cauchy sequence $(h(\pi_{c,d}(a_n), n))_n$ with modulus

$$\lambda_p \max(\alpha(p+1), M(\omega(p+1)-1)).$$

Here the projection $\pi_{c,d}$ is defined by

$$\pi_{c,d}(a) := \begin{cases} c & \text{if } a < c, \\ a & \text{if } c \le a \le d, \\ d & \text{if } d < a. \end{cases}$$

LEMMA 5.2.7 (ContReal). *This is a modulus.*

PROOF. We write a' for $\pi_{c,d}(a)$. Under the assumptions of the definition we have

$$\begin{aligned} |h(a'_n, n) - h(a'_m, m)| &\le |h(a'_n, n) - h(a'_n, m)| + |h(a'_n, m) - h(a'_m, m)| \\ &\le \frac{1}{2^{p+1}} + \frac{1}{2^{p+1}} \end{aligned}$$

if $n, m \geq \alpha(p+1)$ (this gives the first estimate) and $n, m \geq M(\omega(p+1)-1)$ (this gives the second estimate). To see the latter observe that because of $a'_n, a'_m \in [c, d]$ and $m \geq \alpha(p+1)$ it suffices to prove

$$|a'_n - a'_m| \leq \frac{1}{2^{\omega(p+1)-1}}.$$

Because of $n, m \geq M(\omega(p+1)-1)$ we have

$$|a_n - a_m| \leq \frac{1}{2^{\omega(p+1)-1}}.$$

The claim now follows from $|a' - b'| \leq |a - b|$, which is easy to see. □

5.3. Intermediate value theorem

The standard constructive versions of the *intermediate value theorem* read as follows.

THEOREM 5.3.1 (Approximate intermediate value theorem). *Let $f\colon I \to \mathbb{R}$ be continuous and $a < b$ rational numbers in I such that $f(a) \leq 0 \leq f(b)$. Then for every p we can find c with $a < c < b$ such that $|f(c)| \leq \frac{1}{2^p}$.*

A problem with these proofs is that the algorithms they provide are rather bad: in each case one has to partition the interval into as many pieces as the modulus of the continuous function requires for the given error bound, and then for each of these (many) pieces perform certain operations. This problem seems to be unavoidable, since our continuous function may be rather flat. However, we can do somewhat better if we assume a uniform *modulus of increase* (or lower bound on the slope) of f, that is, some $q \in \mathbb{P}$ such that for all $c, d \in \mathbb{Q}$ and all $p \in \mathbb{P}$

$$\frac{1}{2^p} \leq d - c \to \frac{1}{2^{p+q}} \leq f(d) - f(c).$$

Constructively, we cannot compare two reals, but we can compare every real with a nontrivial interval. We already proved this as Lemma 5.1.22 (page 63) called ApproxSplit.

Notice that the boolean object determining whether $z \leq y$ or $x \leq z$ depends on the representation of x, y and z. In particular this assignment is *not* compatible with our equality relation.

We begin with an auxiliary lemma, which from a "correct" interval $c < d$ (that is, $f(c) \leq 0 \leq f(d)$ and $\frac{1}{2^p} \leq d - c$) constructs a new one $c_1 < d_1$ with $d_1 - c_1 = \frac{1}{2}(d - c)$.

LEMMA 5.3.2 (IVTAux). *Let $f\colon I \to \mathbb{R}$ be continuous, with a uniform modulus q of increase. Let $a < b$ be rational numbers in I such that $a \leq c < d \leq b$, say $\frac{1}{2^p} < d - c$, and $f(c) \leq 0 \leq f(d)$. Then we can construct*

c_1, d_1 with $d_1 - c_1 = \frac{1}{2}(d-c)$, such that again $a \le c \le c_1 < d_1 \le d \le b$ and $f(c_1) \le 0 \le f(d_1)$.

PROOF. Let $b_0 = c$ and $b_{n+1} = b_n + \frac{1}{4}(d-c)$ for $n \le 3$, hence $b_4 = d$. From $\frac{1}{2^p} < d - c$ we obtain $\frac{1}{2^{p+2}} \le b_{n+1} - b_n$, so $f(b_n) <_{p+2+q} f(b_{n+1})$.

First compare 0 with the proper interval $f(b_1) < f(b_2)$, using ApproxSplit. In case $0 \le f(b_2)$ let $c_1 = b_0 = c$ and $d_1 = b_2$. In case $f(b_1) \le 0$ compare 0 with the proper interval $f(b_2) < f(b_3)$, using ApproxSplit again. In case $0 \le f(b_3)$ let $c_1 = b_1$ and $d_1 = b_3$. In case $f(b_2) \le 0$ let $c_1 = b_2$ and $d_1 = b_4 = d$. □

THEOREM 5.3.3 (IVT). *Let $f\colon I \to \mathbb{R}$ be continuous, with a uniform modulus of increase. Let $a < b$ be rational numbers in I such that $f(a) \le 0 \le f(b)$. Then we can find $x \in [a,b]$ such that $f(x) = 0$.*

PROOF. Iterating the construction in Lemma 5.3.2 (IVTAux), we construct two sequences $(c_n)_n$ and $(d_n)_n$ of rationals such that for all n

$$\begin{aligned} &a = c_0 \le c_1 \le \cdots \le c_n < d_n \le \cdots \le d_1 \le d_0 = b, \\ &f(c_n) \le 0 \le f(d_n), \\ &d_n - c_n = \frac{1}{2^n}(b-a). \end{aligned}$$

Let x, y be given by the Cauchy sequences $(c_n)_n$ and $(d_n)_n$ with the obvious modulus. As f is continuous, $f(x) = 0 = f(y)$ for the real number $x = y$. □

5.4. Algorithms on stream-represented real numbers

5.4.1. The predicates I and ${}^{\mathrm{co}}I$. We model infinite sequences of signed digits (streams) as objects in the algebra $\mathbb{S}(\mathbb{D}) := \mu_\xi(\mathrm{C}\colon \mathbb{D} \to \xi \to \xi)$, where $\mathbb{D} := \mu_\xi(\mathrm{SdR}\colon \xi, \mathrm{SdM}\colon \xi, \mathrm{SdL}\colon \xi)$ is a 3-element algebra. Such streams will appear as realizers of an inductive predicate I defined by the single clause

$$\forall_{d,x',x}\Big(d \in \mathrm{Sd} \to x' \in I \to x = \frac{x'+d}{2} \to x \in I\Big). \tag{18}$$

Here (and later) x, y, z range over real numbers in $[-1,1]$ and d over integers. Sd is a (formally inductive) predicate expressing that its integer argument d is a signed digit, i.e., $|d| \le 1$. We have chosen (18) rather than the simpler

$$\forall_{d,x}\Big(d \in \mathrm{Sd} \to x \in I \to \frac{x+d}{2} \in I\Big), \tag{19}$$

since we want I to be compatible with the defined equality $=$ on real numbers

$$\forall_{x,y}(x = y \to x \in I \to y \in I), \tag{20}$$

which easily follows from (18) (with reflexivity, symmetry and transitivity of $=$). Using (20) we then obtain (19) from (18) as a lemma.

The dual ${}^{\mathrm{co}}I$ of I is defined by its closure axiom

$$x \in {}^{\mathrm{co}}I \to \exists_{d,x',y}\Big(d \in \mathrm{Sd} \land x' \in {}^{\mathrm{co}}I \land y = \frac{x'+d}{2} \land x = y\Big).$$

Similar to what was done above it can be simplified to

$$x \in {}^{\mathrm{co}}I \to \exists_{d,x'}\Big(d \in \mathrm{Sd} \land x' \in {}^{\mathrm{co}}I \land x = \frac{x'+d}{2}\Big). \tag{21}$$

5.4.2. Average of real numbers. As an example we consider a proof that the average of two real numbers in $[-1,1]$ is in $[-1,1]$ again:

$$x, y \in {}^{\mathrm{co}}I \to \frac{x+y}{2} \in {}^{\mathrm{co}}I.$$

Following Berger and Seisenberger (2010) we begin with an informal proof. The computational content of this proof will be the desired algorithm.

Consider two sets of averages, the second one with a "carry" $i \in \mathbb{Z}$

$$P := \left\{ \frac{x+y}{2} \;\middle|\; x, y \in {}^{\mathrm{co}}I \right\}, \qquad Q := \left\{ \frac{x+y+i}{4} \;\middle|\; x, y \in {}^{\mathrm{co}}I, i \in \mathrm{Sd}_2 \right\},$$

where Sd_2 is a (formally inductive) predicate expressing that the integer i is an extended signed digit, i.e., $|i| \le 2$. It suffices to show that Q satisfies

$$x \in Q \to \exists_{d,x'}\Big(d \in \mathrm{Sd} \land x' \in Q \land x = \frac{x'+d}{2}\Big),$$

for then by the greatest-fixed-point axiom for ${}^{\mathrm{co}}I$ we have $Q \subseteq {}^{\mathrm{co}}I$. Since we also have $P \subseteq Q$ we then obtain $P \subseteq {}^{\mathrm{co}}I$, which is our claim.

Below we will need functions $J, K \colon \mathbb{Z} \to \mathbb{Z}$ such that

$$\forall_i(i = J(i) + 4K(i)) \quad (\text{with } \forall_i(|J(i)| \le 2),\ \forall_i(|i| \le 6 \to |K(i)| \le 1)). \tag{22}$$

We use the following notation for streams u of type $\mathbb{S}(\mathbb{D})$:

$(u)_n :=$ n-th entry in u,

$Tu :=$ rest of u after removal of its first entry.

Clearly $(T^n u)_m = (u)_{n+m}$, where T^n denotes the n-th iteration of T.

Lemma 5.4.1 (CoIAvToAvc).

$$x, y \in {}^{\mathrm{co}}I \to \exists_{i,x',y'}\Big(i \in \mathrm{Sd}_2 \land x', y' \in {}^{\mathrm{co}}I \land \frac{x+y}{2} = \frac{x'+y'+i}{4}\Big).$$

Proof. By the closure axiom (21) for ${}^{\mathrm{co}}I$ we can write $x = \frac{x'+d}{2}$ and $y = \frac{y'+e}{2}$ for some $d, e \in \mathrm{Sd}$ and $x', y' \in {}^{\mathrm{co}}I$. Then

$$\frac{x+y}{2} = \frac{x'+y'+d+e}{4}. \qquad \square$$

The computational content of this proof is a function

$$\mathrm{Av}_{\mathrm{init}}\colon \mathbb{S}(\mathbb{D}) \to \mathbb{S}(\mathbb{D}) \to \mathbb{D}_2 \times \mathbb{S}(\mathbb{D}) \times \mathbb{S}(\mathbb{D})$$
$$\mathrm{Av}_{\mathrm{init}}(u,v) := \langle (u)_0 + (v)_0, Tu, Tv\rangle.$$

LEMMA 5.4.2 (CoIAvcSatCoICl).

$$i \in \mathrm{Sd}_2 \land x, y \in {}^{\mathrm{co}}I \to \exists_{d\in\mathrm{Sd}}\exists_{j\in\mathrm{Sd}_2}\exists_{x',y'\in{}^{\mathrm{co}}I}\Big(\frac{x+y+i}{4} = \frac{\frac{x'+y'+j}{4}+d}{2}\Big).$$

PROOF. By the closure axiom (21) for ${}^{\mathrm{co}}I$ we can write $x = \frac{x'+d}{2}$ and $y = \frac{y'+e}{2}$ for some $d, e \in \mathrm{Sd}$ and $x', y' \in {}^{\mathrm{co}}I$. Then

$$\frac{x+y+i}{4} = \frac{x'+y'+d+e+2i}{8}.$$

Since $|d+e+2i| \le 6$ we can write $d+e+2i = j+4k$ with $|j| \le 2$ and $|k| \le 1$ by the JK-property (22). Therefore

$$\frac{x+y+i}{4} = \frac{x'+y'+j+4k}{8} = \frac{\frac{x'+y'+j}{4}+k}{2}. \qquad \square$$

The computational content of this proof is a function

$$\mathrm{Av}_{\mathrm{step}}\colon \mathbb{D}_2 \times \mathbb{S}(\mathbb{D}) \times \mathbb{S}(\mathbb{D}) \to \mathbb{D} \times \mathbb{D}_2 \times \mathbb{S}(\mathbb{D}) \times \mathbb{S}(\mathbb{D})$$
$$\mathrm{Av}_{\mathrm{step}}\langle i,u,v\rangle := \langle K((u)_0+(v)_0+2i), J((u)_0+(v)_0+2i), Tu, Tv\rangle.$$

By coinduction from Lemma 5.4.2 we obtain

LEMMA 5.4.3 (CoIAvcToCoI). $\forall_z(\exists_{i,x,y}(i \in \mathrm{Sd}_2 \land x, y \in {}^{\mathrm{co}}I \land z = \frac{x+y+i}{4}) \to z \in {}^{\mathrm{co}}I)$.

THEOREM 5.4.4 (CoIAverage). *The average of two real numbers x, y in ${}^{\mathrm{co}}I$ is in ${}^{\mathrm{co}}I$:*

$$x, y \in {}^{\mathrm{co}}I \to \frac{x+y}{2} \in {}^{\mathrm{co}}I.$$

PROOF. Immediate from Lemma 5.4.1 and Lemma 5.4.3. $\square$

The extracted term is short (19 lines in `minlog/examples/analysis/`, file `sdavaux.scm`), and involves the corecursion operator. It represents the function

$$\mathrm{cCoIAverage}\colon \mathbb{S}(\mathbb{D}) \to \mathbb{S}(\mathbb{D}) \to \mathbb{S}(\mathbb{D})$$

defined using corecursion, as follows.

(i) From $u, v \in \mathbb{S}(\mathbb{D})$ form an initial triple $\mathrm{Av}_{\mathrm{init}}(u,v) \in \mathbb{D}_2 \times \mathbb{S}(\mathbb{D}) \times \mathbb{S}(\mathbb{D})$.
(ii) Iterate $\mathrm{Av}_{\mathrm{step}}\colon \mathbb{D}_2 \times \mathbb{S}(\mathbb{D}) \times \mathbb{S}(\mathbb{D}) \to \mathbb{D} \times \mathbb{D}_2 \times \mathbb{S}(\mathbb{D}) \times \mathbb{S}(\mathbb{D})$ starting with $\mathrm{Av}_{\mathrm{init}}(u,v)$.
(iii) Return the stream of the generated $d \in \mathbb{D}$.

5.5. Lookahead

Recall that real numbers in $[-1.1]$ can be represented in the form

$$\sum_{n=0}^{\infty} \frac{d_n}{2^{n+1}} \qquad (d_n \in \{-1, 0, 1\}).$$

We are interested in algorithms for arithmetical functions on real numbers operating on such stream representations $(d_n)_n$. In particular we want to verify lookahead properties of such algorithms.

To this end we inductively define a binary predicate L of arity $(\mathbb{R}, \mathbb{N})$. The intended property of L is

> a realizer of $x \in L_n$ is a list $(d_m)_{m<n}$ of signed digits which approximates x with error bound $\frac{1}{2^n}$, i.e., such that
>
> $$\Big| \sum_{m=0}^{n-1} \frac{d_m}{2^{m+1}} - x \Big| \le \frac{1}{2^n}.$$

Here we have written $x \in L_n$ for $L(x, n)$. Let x, y, z range over real numbers and d over integers. Sd is a (formally inductive) predicate expressing that its integer argument d is a signed digit, i.e., $|d| \le 1$. L is defined by the two clauses

$$L_0^+ \colon \forall_x(|x| \le 1 \to x \in L_0),$$

$$L_1^+ \colon \forall_{x,y,n,d}\Big(d \in \mathrm{Sd} \to x \in L_n \to y = \frac{x+d}{2} \to y \in L_{n+1}\Big).$$

L_1^+ says that if we know the first n digits of a representation of x and that $y = \frac{x+d}{2}$, then we know the first $n+1$ digits of a representation of y.

The induction or least-fixed-point (lfp) axiom for L is

$$\forall_x(|x| \le 1 \to Y(x, 0)) \to$$

$$\forall_{x,y,n,d}\Big(d \in \mathrm{Sd} \to x \in L_n \to Y(x, n) \to y = \frac{x+d}{2} \to Y(y, n+1)\Big) \to L \subseteq Y.$$

The algebra associated with L is $\mathbb{L} := \mu_\xi(U \colon \xi, C \colon \mathbb{D} \to \xi \to \xi)$, i.e., finite lists of signed digits $-1, 0, 1$. We write u, v for objects of type $\mathbb{L}$, and $d :: u$ for Cdu.

We will need some properties of L.

Lemma 5.5.1 (`LCompat`). $x = y \to x \in L_n \to y \in L_n$.

Proof. Use the lfp-axiom for L. □

The term extracted from this proof is the identity on $\mathbb{L}$; it will be omitted in the sequel.

Lemma 5.5.2 (`LToBd`). $x \in L_n \to |x| \le 1$.

PROOF. Use induction on n. □

Parallel to the (simplified) closure axiom (21) for ${}^{\mathrm{co}}I$ (page 68) we have

LEMMA 5.5.3 (`LClosure`). $x{\in}L_{n+1} \to \exists_{d,x'}(d{\in}\mathrm{Sd} \wedge x'{\in}L_n \wedge x = \frac{x'+d}{2})$.

PROOF. Use the lfp-axiom for L. □

The extracted term `cLClosure` has type $\mathbb{L} \to \mathbb{D} \times \mathbb{L}$. It satisfies

$$\mathtt{cLClosure}(d :: u) = \langle d, u\rangle.$$

We now proceed as before, but with L instead of ${}^{\mathrm{co}}I$. As in Section 5.4.2 (page 68) we show

$$\left\{\frac{x+y}{2}\,\middle|\, x, y \in L_{n+1}\right\} \subseteq \left\{\frac{x+y+i}{4}\,\middle|\, x, y \in L_n, i \in \mathrm{Sd}_2\right\} \subseteq L_n,$$

where this time the final conclusion is proved by induction.

LEMMA 5.5.4 (`LAvToAvc`).

$$x, y \in L_{n+1} \to \exists_{i,x',y'}\Big(i \in \mathrm{Sd}_2 \wedge x', y' \in L_n \wedge \frac{x+y}{2} = \frac{x'+y'+i}{4}\Big).$$

PROOF. By Lemma 5.5.3 we can write $x = \frac{x'+d}{2}$ and $y = \frac{y'+e}{2}$ for some $d, e \in \mathrm{Sd}$ and $x', y' \in L_n$. Then

$$\frac{x+y}{2} = \frac{x'+y'+d+e}{4}.$$ □

The extracted term `cLAvToAvc` has type $\mathbb{L} \to \mathbb{L} \to \mathbb{D}_2 \times \mathbb{L} \times \mathbb{L}$ and satisfies

$$\mathtt{cLAvToAvc}(d :: u, e :: v) = \langle d + e, u, v\rangle.$$

LEMMA 5.5.5 (`LAvcSatLCl`).

$$i \in \mathrm{Sd}_2 \wedge x, y \in L_{n+1} \to \exists_{d\in\mathrm{Sd}}\exists_{j\in\mathrm{Sd}_2}\exists_{x',y'\in L_n}\Big(\frac{x+y+i}{4} = \frac{\frac{x'+y'+j}{4} + d}{2}\Big).$$

PROOF. Let $i \in \mathrm{Sd}_2$ and $x, y \in L_{n+1}$. By Lemma 5.5.3 we can write $x = \frac{x'+d}{2}$ and $y = \frac{y'+e}{2}$ for some $d, e \in \mathrm{Sd}$ and $x', y' \in L_n$. Then

$$\frac{x+y+i}{4} = \frac{x'+y'+d+e+2i}{8}.$$

Since $|d+e+2i| \leq 6$ we can write $d+e+2i = j+4k$ with $|j| \leq 2$ and $|k| \leq 1$ by the JK-property (22). Therefore

$$\frac{x+y+i}{4} = \frac{x'+y'+j+4k}{8} = \frac{\frac{x'+y'+j}{4} + k}{2}.$$ □

The extracted term `cLAvcSatLCl` has type $\mathbb{D}_2 \to \mathbb{L} \to \mathbb{L} \to \mathbb{D} \times \mathbb{D}_2 \times \mathbb{L} \times \mathbb{L}$ and satisfies

$$\texttt{cLAvcSatLCl}\langle i, d :: u, e :: v\rangle = \langle K(d+e+2i), J(d+e+2i), u, v\rangle.$$

By induction from Lemma 5.5.5 we obtain

LEMMA 5.5.6 (LAvcToL). $n \in T_{\mathbb{N}} \to i \in \mathrm{Sd}_2 \land x, y \in L_n \to \frac{x+y+i}{4} \in L_n$.

PROOF. By induction on n. *Base.* Assume $i \in \mathrm{Sd}_2$ and $x, y \in L_0$. Then $|x|, |y| \le 1$ by Lemma 5.5.2, hence $\frac{x+y+i}{4} \in L_0$ by the first clause of L. *Step.* Assume $i \in \mathrm{Sd}_2$ and $x, y \in L_{n+1}$. By Lemma 5.5.5 we can write

$$\frac{x+y+i}{4} = \frac{\frac{x'+y'+j}{4} + d}{2}$$

with $x', y' \in L_n$, $j \in \mathrm{Sd}_2$ and $d \in \mathrm{Sd}$. Now we can apply the second clause of L, using the IH to obtain $x', y' \in L_n$ and then compatibility of L. □

The extracted term `cLAvcToL` has type $\mathbb{N} \to \mathbb{D}_2 \to \mathbb{L} \to \mathbb{L} \to \mathbb{L}$ and is defined by recursion over $\mathbb{N}$. It satisfies

$$\texttt{cLAvcToL}(0, \langle i, u, v\rangle) = U,$$
$$\texttt{cLAvcToL}(n+1, \langle i, u, v\rangle) = d :: \texttt{cLAvcToL}(n, w)$$

where

$$\langle d, w\rangle = \texttt{cLAvcSatLCl}\langle i, u, v\rangle.$$

THEOREM 5.5.7 (`LAverage`). $n \in T_{\mathbb{N}} \to x, y \in L_{n+1} \to \frac{x+y}{2} \in L_n$.

PROOF. Immediate by Lemma 5.5.4 and Lemma 5.5.6. □

The extracted term `cLAverage` has type $\mathbb{N} \to \mathbb{L} \to \mathbb{L} \to \mathbb{L}$. It satisfies

$$\texttt{cLAverage}(n, u, v) = \texttt{cLAvcToL}(n, \texttt{cLAvToAvc}(u, v)).$$

The way `cLAverage` operates can be understood as follows.

- From $u, v \in \mathbb{L}$ form the initial triple $\texttt{cLAvToAvc}(u, v) \in \mathbb{D}_2 \times \mathbb{L} \times \mathbb{L}$.
- Recall that `cLAvcSatLCl` has type $\mathbb{D}_2 \to \mathbb{L} \to \mathbb{L} \to \mathbb{D} \times \mathbb{D}_2 \times \mathbb{L} \times \mathbb{L}$. Iterate it n times, starting with $\texttt{cLAvToAvc}(u, v)$.
- Return the list of generated $d \in \mathbb{D}$.

The soundness proof is within TCF and can be automatically generated with `add-sound`.

APPENDIX A

Unification

We show that for any two constructor terms one can decide whether there exists a "unifier", and if so, compute a "most general" one. A solution of this problem has been given by Robinson (1965). In the formulation of the algorithm below we follow Martelli and Montanari (1982).

By a constructor term t, s (term for short) we mean a term built from variables x, y, z and constructors C by application. A *substitution* is a finite set $\vartheta = \{t_1/x_1, \ldots, t_n/x_n\}$ of pairs of variables and terms, such that $x_i \neq x_j$ for $i \neq j$, and $t_i \neq x_i$ for all i. An element t_i/x_i of ϑ is called a *binding* (of x_i to t_i). By $t\vartheta$ we denote the result of simultaneously replacing each variable x_i in t by t_i, and call $t\vartheta$ the *instance* of t induced by ϑ. We shall use ϑ, η, ζ for substitutions. Let ε be the empty substitution. For given substitutions

$$\vartheta = \{t_1/x_1, \ldots, t_n/x_n\}$$
$$\eta = \{s_1/y_1, \ldots, s_m/y_m\},$$

the *composition* $\vartheta\eta$ of ϑ and η is the substitution obtained by deleting in the set

$$\{t_1\eta/x_1, \ldots, t_n\eta/x_n, s_1/y_1, \ldots, s_m/y_m\}$$

all bindings $t_i\eta/x_i$ such that $t_i\eta = x_i$, and also all bindings s_j/y_j such that $y_j \in \{x_1, \ldots, x_n\}$. A substitution ϑ is *idempotent* if $\vartheta\vartheta = \vartheta$. A substitution ϑ is called *more general* than η (written $\eta \leq \vartheta$), if there is a substitution ζ such that $\eta = \vartheta\zeta$. ϑ and η are *equivalent*, if $\vartheta \leq \eta \leq \vartheta$.

It is easy to see that $(t\vartheta)\eta = t(\vartheta\eta)$, and that composition is associative.

We now come to the unification problem. By this we mean the question whether for two given terms t, s there is a substitution ϑ "unifying" the two terms, i.e., with the property $t\vartheta = s\vartheta$.

Let E denote finite equation systems, i.e., multisets

$$\{t_1 = s_1, \ldots, t_n = s_n\}$$

of equations between terms (more precisely pairs of terms). Consider $\{\bot\}$ as a (contradictory) equation system. A substitution ϑ *unifies* E, if for every equation $t = s$ in E we have $t\vartheta = s\vartheta$; no ϑ unifies $\{\bot\}$. ϑ is a *most general unifier (mgu)* of E, if ϑ is a unifier of E and $\eta \leq \vartheta$ for every unifier η of E.

The following characterization of idempotent mgus will be useful in the proof of the Unification Theorem below.

LEMMA A.0.1 (Characterization of idempotent mgu's). *Let ϑ be a unifier of E. Then ϑ is an idempotent mgu of E iff $\eta = \vartheta\eta$ for all unifiers η of E.*

PROOF. Assume that ϑ is a unifier of E.

$\rightarrow$. Let ϑ be an idempotent mgu of E, and assume that η is a unifier of E. Since ϑ is a mgu of E, we have $\eta = \vartheta\zeta$ for some substitution ζ. Hence $\eta = \vartheta\zeta = \vartheta\vartheta\zeta = \vartheta\eta$.

$\leftarrow$. Assume that $\eta = \vartheta\eta$ for all unifiers η of E. Now let η be a unifier of E. Then $\eta \leq \vartheta$; therefore ϑ is a mgu. Since ϑ is a unifier, by assumption we have $\vartheta = \vartheta\vartheta$. □

DEFINITION (Unification algorithm). $E \mapsto_\vartheta E'$ is defined by

(a) $\{t = x\} \cup E \mapsto_\varepsilon \{x = t\} \cup E$, if t is not a variable.
(b) $\{x = x\} \cup E \mapsto_\varepsilon E$.
(c) $\{\mathrm{C}t_1 \dots t_n = \mathrm{C}s_1 \dots s_n\} \cup E \mapsto_\varepsilon \{t_1 = s_1, \dots t_n = s_n\} \cup E$.
(d) $\{\mathrm{C}t_1 \dots t_n = \mathrm{C}'s_1 \dots s_n\} \cup E \mapsto_\varepsilon \{\bot\}$ if $\mathrm{C} \neq \mathrm{C}'$.
(e) $\{x = t, t_1(x) = s_1(x), \dots, t_n(x) = s_n(x)\} \mapsto_{\{t/x\}}$
$\{t_1(t) = s_1(t), \dots t_n(t) = s_n(t)\}$ if $x \notin \mathrm{FV}(t)$.
(f) $\{x = t\} \cup E \mapsto_\varepsilon \{\bot\}$, if $x \in \mathrm{FV}(t)$ and $t \neq x$.

PROPOSITION. *Assume $E \mapsto_\vartheta E'$.*

(a) *If η' is a unifier of E', then $\vartheta\eta'$ is a unifier of E.*
(b) *If η is a unifier of E, then $\eta = \vartheta\eta$ and η is a unifier of E'.*

PROOF. By cases according to the definition of $E \mapsto_\vartheta E'$. Clearly it suffices to treat case (e).

Let η' be a unifier of E'. Then $\{t/x\}\eta'$ is a unifier of E.

Let η be a unifier of E. Then $x\eta = t\eta$, hence $\eta = \{t/x\}\eta$ (since both substitutions coincide on all variables), and moreover

$$t_i\{t/x\}\eta = t_i\eta = s_i\eta = s_i\{t/x\}\eta.$$

Hence η is a unifier of E'. □

COROLLARY. *Assume*

$$E_1 \mapsto_{\vartheta_1} E_2 \mapsto_{\vartheta_2} \dots E_n \mapsto_{\vartheta_n} E_{n+1}.$$

(a) *If ϑ is a unifier of E_{n+1}, then $\vartheta_1 \dots \vartheta_n\vartheta$ is a unifier of E_1.*
(b) *If η is a unifier of E_1, then $\eta = \vartheta_1 \dots \vartheta_n\eta$ and η is a unifier of E_{n+1}.*

PROOF. The first part clearly follows from the first part of the Proposition. The second part is proved by induction on n. For $n = 0$ there is nothing to show. In the step we split the assumption into

$$E_1 \mapsto_{\vartheta_1} E_2 \quad \text{and} \quad E_2 \mapsto_{\vartheta_2} \dots E_n \mapsto_{\vartheta_n} E_{n+1}.$$

By the second part of the Proposition we have that $\eta = \vartheta_1\eta$ is a unifier of E_2. Hence by IH $\eta = \vartheta_2 \cdots \vartheta_n\eta$ is a unifier of E_{n+1}. Moreover we have $\eta = \vartheta_1\eta = \vartheta_1\vartheta_2 \ldots \vartheta_n\eta$. □

Unification Theorem. *Let E be a finite equation system. Then every sequence*

$$E = E_1 \mapsto_{\vartheta_1} E_2 \mapsto_{\vartheta_2} \ldots$$

terminates with $E_{n+1} = \emptyset$ or $E_{n+1} = \{\bot\}$. In the first case E is unifiable, and $\vartheta_1 \ldots \vartheta_n$ is an idempotent mgu of E. In the second case E is not unifiable.

Proof. Termination is proved using the lexicographic ordering of $\mathbb{N}^3$. To every $E = \{t_1 = s_1, \ldots, t_n = s_n\}$ assign a triple $(n_1, n_2, n_3) \in \mathbb{N}^3$ by

$n_1 :=$ number of variables in E,

$n_2 :=$ number of occurrences of variables and constructors in E,

$n_3 :=$ number of equations $t = x$ in E such that t is not a variable. □

APPENDIX B

Denotational semantics

We set up a connection between the model $(|\mathbf{C}_\rho|)_\rho$ of partial continuous functionals described in Section 2.2.2 and the term system T^+ from Section 2.3.1. The main point is to clarify how we can use computation rules to define an ideal z in a function space.

B.1. Ideals as values of terms

The general idea is to inductively define the set of tokens (U, a) that make up z. It is convenient to define the value $[\![\lambda_{\vec{x}} M]\!]$, where M is a term with free variables among $\vec{x}$. Since this value is a token set, we can define inductively the relation $(\vec{U}, a) \in [\![\lambda_{\vec{x}} M]\!]$.

For a constructor pattern $\vec{P}(\vec{x}\,)$ and a list $\vec{V}$ of the same length and types as $\vec{x}$ we define a list $\vec{P}(\vec{V})$ of formal neighborhoods of the same length and types as $\vec{P}(\vec{x}\,)$, by induction on $\vec{P}(\vec{x}\,)$. $x(V)$ is the singleton list V, and for $\langle\rangle$ we take the empty list. $(\vec{P}, Q)(\vec{V}, \vec{w}\,)$ is covered by the induction hypothesis. Finally

$$(\mathrm{C}\vec{P}\,)(\vec{V}) := \{\, \mathrm{C}\vec{a^*} \mid a_i^* \in P_i(\vec{V_i}) \text{ if } P_i(\vec{V_i}) \neq \emptyset, \text{ and } a_i^* = * \text{ otherwise} \,\}.$$

We use the following notation. $(\vec{U}, a)$ means $(U_1, (U_2, \dots (U_n, a)) \dots)$, and $(\vec{U}, V) \subseteq [\![\lambda_{\vec{x}} M]\!]$ means $(\vec{U}, a) \in [\![\lambda_{\vec{x}} M]\!]$ for all (finitely many) $a \in V$.

Definition B.1.1 (Inductive, of $(\vec{U}, a) \in [\![\lambda_{\vec{x}} M]\!]$).

$$\frac{U_i \vdash a}{(\vec{U}, a) \in [\![\lambda_{\vec{x}} x_i]\!]}(V), \qquad \frac{(\vec{U}, V, a) \in [\![\lambda_{\vec{x}} M]\!] \qquad (\vec{U}, V) \subseteq [\![\lambda_{\vec{x}} N]\!]}{(\vec{U}, a) \in [\![\lambda_{\vec{x}} (MN)]\!]}(A).$$

For every constructor C and defined constant D we have

$$\frac{\vec{V} \vdash \vec{a^*}}{(\vec{U}, \vec{V}, \mathrm{C}\vec{a^*}) \in [\![\lambda_{\vec{x}} \mathrm{C}]\!]}(\mathrm{C}), \qquad \frac{(\vec{U}, \vec{V}, a) \in [\![\lambda_{\vec{x},\vec{y}} M]\!] \qquad \vec{W} \vdash \vec{P}(\vec{V})}{(\vec{U}, \vec{W}, a) \in [\![\lambda_{\vec{x}} D]\!]}(D)$$

with one such rule (D) for every computation rule $D\vec{P}(\vec{y}\,) = M$.

The *height* of a derivation of $(\vec{U}, a) \in [\![\lambda_{\vec{x}} M]\!]$ is defined as usual, by adding 1 at each rule. We define its *D-height* similarly, where only rules (D) count.

We begin with some simple consequences of this definition.

LEMMA B.1.2. *The following transformations preserve D-height:*

$$\vec{V} \vdash \vec{U} \to (\vec{U}, a) \in [\![\lambda_{\vec{x}} M]\!] \to (\vec{V}, a) \in [\![\lambda_{\vec{x}} M]\!], \tag{23}$$

$$(\vec{U}, V, a) \in [\![\lambda_{\vec{x},y} M]\!] \leftrightarrow (\vec{U}, a) \in [\![\lambda_{\vec{x}} M]\!] \quad \textit{if } y \notin \mathrm{FV}(M), \tag{24}$$

$$(\vec{U}, V, a) \in [\![\lambda_{\vec{x},y}(My)]\!] \leftrightarrow (\vec{U}, V, a) \in [\![\lambda_{\vec{x}} M]\!] \quad \textit{if } y \notin \mathrm{FV}(M), \tag{25}$$

$$(\vec{U}, \vec{V}, a) \in [\![\lambda_{\vec{x},\vec{y}}(M(\vec{P}(\vec{y})))]\!] \leftrightarrow (\vec{U}, \vec{P}(\vec{V}), a) \in [\![\lambda_{\vec{x},\vec{z}}(M(\vec{z}))]\!]. \tag{26}$$

PROOF. (23) and (24) are both proved by easy inductions on the respective derivations.

(25). Assume $(\vec{U}, V, a) \in [\![\lambda_{\vec{x},y}(My)]\!]$. By (A) we then have W such that $(\vec{U}, V, W) \subseteq [\![\lambda_{\vec{x},y} y]\!]$ (i.e., $V \vdash W$) and $(\vec{U}, V, W, a) \in [\![\lambda_{\vec{x},y} M]\!]$. By (23) from the latter we obtain $(\vec{U}, V, V, a) \in [\![\lambda_{\vec{x},y} M]\!]$. Now since $y \notin \mathrm{FV}(M)$, (24) yields $(\vec{U}, V, a) \in [\![\lambda_{\vec{x}} M]\!]$, as required. Conversely, assume $(\vec{U}, V, a) \in [\![\lambda_{\vec{x}} M]\!]$. Since $y \notin \mathrm{FV}(M)$, (24) yields $(\vec{U}, V, V, a) \in [\![\lambda_{\vec{x}} M]\!]$. Clearly we have $(\vec{U}, V, V) \subseteq [\![\lambda_{\vec{x},y} y]\!]$. Hence by (A) $(\vec{U}, V, a) \in [\![\lambda_{\vec{x},y}(My)]\!]$, as required. Notice that the D-height did not change in these transformations.

(26). By induction on $\vec{P}$, with a side induction on M. We distinguish cases on M. The cases x_i, C and D follow immediately from (24). In case MN the following are equivalent by the side induction hypothesis:

$$(\vec{U}, \vec{V}, a) \in [\![\lambda_{\vec{x},\vec{y}}((MN)(\vec{P}(\vec{y})))]\!]$$

$$\exists_W((\vec{U}, \vec{V}, W, a) \in [\![\lambda_{\vec{x},\vec{y}}(M(\vec{P}(\vec{y})))]\!] \wedge (\vec{U}, \vec{V}, W) \subseteq [\![\lambda_{\vec{x},\vec{y}}(N(\vec{P}(\vec{y})))]\!])$$

$$\exists_W((\vec{U}, \vec{P}(\vec{V}), W, a) \in [\![\lambda_{\vec{x},\vec{z}}(M(\vec{z}))]\!] \wedge (\vec{U}, \vec{P}(\vec{V}), W) \subseteq [\![\lambda_{\vec{x},\vec{z}}(N(\vec{z}))]\!])$$

$$(\vec{U}, \vec{P}(\vec{V}), a) \in [\![\lambda_{\vec{x},\vec{z}}((MN)(\vec{z}))]\!].$$

The final case is where M is z_i. Then we have to show

$$(\vec{U}, \vec{V}, a) \in [\![\lambda_{\vec{x},\vec{y}}(P(\vec{y}))]\!] \leftrightarrow P(\vec{V}) \vdash a.$$

We distinguish cases on $P(\vec{y})$. If $P(\vec{y})$ is y_j, then both sides are equivalent to $V_j \vdash a$. In case $P(\vec{y})$ is $(\mathrm{C}\vec{Q})(\vec{y})$ the following are equivalent:

$$(\vec{U}, \vec{V}, a) \in [\![\lambda_{\vec{x},\vec{y}}((\mathrm{C}\vec{Q})(\vec{y}))]\!]$$

$$(\vec{U}, \vec{V}, a) \in [\![\lambda_{\vec{x},\vec{y}}(\mathrm{C}\vec{Q}(\vec{y}))]\!]$$

$$(\vec{U}, \vec{Q}(\vec{V}), a) \in [\![\lambda_{\vec{x},\vec{u}}(\mathrm{C}\vec{u})]\!] \quad \text{by IH for } \vec{Q}(\vec{y})$$

$$(\vec{U}, \vec{Q}(\vec{V}), a) \in [\![\lambda_{\vec{x}}\mathrm{C}]\!] \quad \text{by (25)}$$

$$\exists_{\vec{a^*}}(a = \mathrm{C}\vec{a^*} \wedge \vec{Q}(\vec{V}) \vdash \vec{a^*})$$

$$\mathrm{C}\vec{Q}(\vec{V}) \vdash a. \qquad \square$$

Let $\sim$ denote the equivalence relation on formal neighborhoods generated by entailment, i.e., $U \sim V$ means $(U \vdash V) \wedge (V \vdash U)$.

LEMMA B.1.3.

(27) *If $\vec{U} \vdash \vec{P}(\vec{V})$, then there are $\vec{W}$ such that $\vec{U} \sim \vec{P}(\vec{W})$ and $\vec{W} \vdash \vec{V}$.*

PROOF. By induction on $\vec{P}$. The cases x and $\langle\rangle$ are clear, and in case $\vec{P}, Q$ we can apply the induction hypothesis. It remains to treat the case $\mathrm{C}\vec{P}(\vec{x}\,)$. Since $U \vdash \mathrm{C}\vec{P}(\vec{V})$ there is a $\vec{b_0^*}$ such that $\mathrm{C}\vec{b_0^*} \in U$. Let

$$U_i := \{\, a \mid \exists_{\vec{a^*}}(\mathrm{C}\vec{a^*} \in U \wedge a = a_i^*) \,\}.$$

For the constructor pattern $\mathrm{C}\vec{x}$ consider $\mathrm{C}\vec{U}$. By definition

$$\mathrm{C}\vec{U} = \{\, \mathrm{C}\vec{a^*} \mid a_i^* \in U_i \text{ if } U_i \neq \emptyset, \text{ and } a_i^* = * \text{ otherwise} \,\}.$$

We first show $U \sim \mathrm{C}\vec{U}$. Assume $\mathrm{C}\vec{a^*} \in \mathrm{C}\vec{U}$. For each i, if $U_i \neq \emptyset$, then there is an $\vec{a_i^*}$ such that $\mathrm{C}\vec{a_i^*} \in U$ and $a_{ii}^* = a_i^*$, and if $U_i = \emptyset$ then $a_i^* = *$. Hence

$$U \supseteq \{\, \mathrm{C}\vec{a_i^*} \mid U_i \neq \emptyset \,\} \cup \{\mathrm{C}\vec{b_0^*}\} \vdash \mathrm{C}\vec{a^*}.$$

Conversely assume $\mathrm{C}\vec{a^*} \in U$. We define $\mathrm{C}\vec{b^*} \in \mathrm{C}\vec{U}$ by $b_i^* = a_i^*$ if $a_i^* \neq *$, $b_i^* = *$ if $U_i = \emptyset$, and otherwise (i.e., if $a_i^* = *$ and $U_i \neq \emptyset$) take an arbitrary $b_i^* \in U_i$. Clearly $\{\mathrm{C}\vec{b^*}\} \vdash \mathrm{C}\vec{a^*}$.

By definition $\vec{U} \vdash \vec{P}(\vec{V})$. Hence by induction hypothesis there are $\vec{W}$ such that $\vec{U} \sim \vec{P}(\vec{W})$ and $\vec{W} \vdash \vec{V}$. Therefore $U \sim \mathrm{C}\vec{U} \sim \mathrm{C}\vec{P}(\vec{W})$. □

LEMMA B.1.4 (Unification). *If $\vec{P}_1(\vec{V}_1) \sim \cdots \sim \vec{P}_n(\vec{V}_n)$, then $\vec{P}_1, \ldots, \vec{P}_n$ are unifiable with a most general unifier ϑ and there exists $\vec{W}$ such that*

$$(\vec{P}_1\vartheta)(\vec{W}) = \cdots = (\vec{P}_n\vartheta)(\vec{W}) \sim \vec{P}_1(\vec{V}_1) \sim \cdots \sim \vec{P}_n(\vec{V}_n).$$

PROOF. Assume $\vec{P}_1(\vec{V}_1) \sim \cdots \sim \vec{P}_n(\vec{V}_n)$. Then $\vec{P}_1(\vec{V}_1), \ldots, \vec{P}_n(\vec{V}_n)$ are componentwise consistent and hence $\vec{P}_1, \ldots, \vec{P}_n$ are unifiable with a most general unifier ϑ. We now proceed by induction on $\vec{P}_1, \ldots, \vec{P}_n$. If they are either all empty or all variables the claim is trivial. In the case $(\vec{P}_1, P_1), \ldots, (\vec{P}_n, P_n)$ it follows from the linearity condition on variables that a most general unifier of $(\vec{P}_1, P_1), \ldots, (\vec{P}_n, P_n)$ is the union of most general unifiers of $\vec{P}_1, \ldots, \vec{P}_n$ and of $P_1, \ldots, P_n$. Hence the induction hypothesis applies. In the case $\mathrm{C}\vec{P}_1, \ldots, \mathrm{C}\vec{P}_n$ the assumption $\mathrm{C}\vec{P}_1(\vec{V}_1) \sim \cdots \sim \mathrm{C}\vec{P}_n(\vec{V}_n)$ implies $\vec{P}_1(\vec{V}_1) \sim \cdots \sim \vec{P}_n(\vec{V}_n)$ and hence again the induction hypothesis applies. The remaining case is when some are variables and the other ones of the form $\mathrm{C}\vec{P}_i$, say $x, \mathrm{C}\vec{P}_2, \ldots, \mathrm{C}\vec{P}_n$. By assumption

$$V_1 \sim \mathrm{C}\vec{P}_2(\vec{V}_2) \sim \cdots \sim \mathrm{C}\vec{P}_n(\vec{V}_n).$$

By induction hypothesis we obtain the required $\vec{W}$ such that

$$(\vec{P}_2\vartheta)(\vec{W}) = \cdots = (\vec{P}_n\vartheta)(\vec{W}) \sim \vec{P}_2(\vec{V}_2) \sim \cdots \sim \vec{P}_n(\vec{V}_n). \qquad \square$$

LEMMA B.1.5 (Consistency). *$[\![\lambda_{\vec{x}} M]\!]$ is consistent.*

PROOF. Let $(\vec{U}_i, a_i) \in [\![\lambda_{\vec{x}} M]\!]$ for $i = 1, 2$. By coherence it suffices to prove that $(\vec{U}_1, a_1)$ and $(\vec{U}_2, a_2)$ are consistent. We shall prove this by induction on the maximum of the D-heights and a side induction on the maximum of the heights.

Case (V). Let $(\vec{U}_1, a_1), (\vec{U}_2, a_2) \in [\![\lambda_{\vec{x}} x_i]\!]$, and assume that $\vec{U}_1$ and $\vec{U}_2$ are componentwise consistent. Then $U_{1i} \vdash a_1$ and $U_{2i} \vdash a_2$. Since $U_{1i} \cup U_{2i}$ is consistent, a_1 and a_2 must be consistent as well.

Case (C). For $i = 1, 2$ we have

$$\frac{\vec{V}_i \vdash \vec{a_i^*}}{(\vec{U}_i, \vec{V}_i, \mathrm{C}\vec{a_i^*}) \in [\![\lambda_{\vec{x}} \mathrm{C}]\!]}.$$

Assume $\vec{U}_1, \vec{V}_1$ and $\vec{U}_2, \vec{V}_2$ are componentwise consistent. The consistency of $\mathrm{C}\vec{a_1^*}$ and $\mathrm{C}\vec{a_2^*}$ follows from $\vec{V}_i \vdash \vec{a_i^*}$ and the consistency of $\vec{V}_1$ and $\vec{V}_2$.

Case (A). For $i = 1, 2$ we have

$$\frac{(\vec{U}_i, V_i, a_i) \in [\![\lambda_{\vec{x}} M]\!] \qquad (\vec{U}_i, V_i) \subseteq [\![\lambda_{\vec{x}} N]\!]}{(\vec{U}_i, a_i) \in [\![\lambda_{\vec{x}} (MN)]\!]}.$$

Assume $\vec{U}_1$ and $\vec{U}_2$ are componentwise consistent. By the side induction hypothesis for the right premises $V_1 \cup V_2$ is consistent. Hence by the side induction hypothesis for the left hand sides a_1 and a_2 are consistent.

Case (D). For $i = 1, 2$ we have

$$\frac{(\vec{U}_i, \vec{V}_i, a_i) \in [\![\lambda_{\vec{x},\vec{y}_i} M_i(\vec{y}_i)]\!] \qquad \vec{W}_i \vdash \vec{P}_i(\vec{V}_i)}{(\vec{U}_i, \vec{W}_i, a_i) \in [\![\lambda_{\vec{x}} D]\!]}(D)$$

for computation rules $D\vec{P}_i(\vec{y}_i) = M_i(\vec{y}_i)$. Assume $\vec{U}_1, \vec{W}_1$ and $\vec{U}_2, \vec{W}_2$ are componentwise consistent; we must show that a_1 and a_2 are consistent. Since $\vec{W}_1 \cup \vec{W}_2 \vdash \vec{P}_i(\vec{V}_i)$ for $i = 1, 2$, by (27) there are $\vec{V}_1', \vec{V}_2'$ such that $\vec{V}_i' \vdash \vec{V}_i$ and $\vec{W}_1 \cup \vec{W}_2 \sim \vec{P}_i(\vec{V}_i')$. Then by Lemma B.1.4 (on unification) there are $\vec{W}$ such that $(\vec{P}_1\vartheta)(\vec{W}) = (\vec{P}_2\vartheta)(\vec{W}) \sim \vec{P}_i(\vec{V}_i') \vdash \vec{P}_i(\vec{V}_i)$ for $i = 1, 2$, where ϑ is the most general unifier of $\vec{P}_1$ and $\vec{P}_2$. But then also

$$(\vec{y}_i\vartheta)(\vec{W}) \vdash \vec{V}_i,$$

and hence by (23) we have

$$(\vec{U}_i, (\vec{y}_i\vartheta)(\vec{W}), a_i) \in [\![\lambda_{\vec{x},\vec{y}_i} M_i(\vec{y}_i)]\!]$$

with smaller D-height. Now (26) gives

$$(\vec{U}_i, \vec{W}, a_i) \in [\![\lambda_{\vec{x},\vec{z}} M_i(\vec{y}_i)\vartheta]\!]$$

without increasing the D-height. Notice that $M_1(\vec{y}_i)\vartheta = M_2(\vec{y}_i)\vartheta$ by our condition on computation rules. Hence the induction hypothesis applied to $(\vec{U}_1, \vec{W}, a_1), (\vec{U}_2, \vec{W}, a_2) \in [\![\lambda_{\vec{x},\vec{z}} M_1(\vec{y}_1)\vartheta]\!]$ implies the consistency of a_1 and a_2, as required. □

Lemma B.1.6 (Deductive closure). *$[\![\lambda_{\vec{x}} M]\!]$ is deductively closed, i.e., if $W \subseteq [\![\lambda_{\vec{x}} M]\!]$ and $W \vdash (\vec{V}, b)$, then $(\vec{V}, b) \in [\![\lambda_{\vec{x}} M]\!]$.*

Proof. By induction on the maximum of the D-heights and a side induction on the maximum of the heights of $W \subseteq [\![\lambda_{\vec{x}} M]\!]$. We distinguish cases on the last rule of these derivations (which is determined by M).

Case (V). For all $(\vec{U}, a) \in W$ we have

$$\frac{U_i \vdash a}{(\vec{U}, a) \in [\![\lambda_{\vec{x}} x_i]\!]}.$$

We must show $V_i \vdash b$. By assumption $W \vdash (\vec{V}, b)$, hence $W\vec{V} \vdash b$. It suffices to prove $V_i \vdash W\vec{V}$. Let $c \in W\vec{V}$; we show $V_i \vdash c$. There are $\vec{U}$ such that $\vec{V} \vdash \vec{U}$ and $(\vec{U}, c) \in W$. But then by the above $U_i \vdash c$, hence $V_i \vdash U_i \vdash c$.

Case (A). Let $W = \{(\vec{U}_1, a_1), \dots, (\vec{U}_n, a_n)\}$. For each $(\vec{U}_i, a_i) \in W$ there is U_i such that

$$\frac{(\vec{U}_i, U_i, a_i) \in [\![\lambda_{\vec{x}} M]\!] \qquad (\vec{U}_i, U_i) \subseteq [\![\lambda_{\vec{x}} N]\!]}{(\vec{U}_i, a_i) \in [\![\lambda_{\vec{x}} (MN)]\!]}.$$

Define $U := \bigcup\{\, U_i \mid \vec{V} \vdash \vec{U}_i \,\}$. We first show that U is consistent. Let $a, b \in U$. There are i, j such that $a \in U_i$, $b \in U_j$ and $\vec{V} \vdash \vec{U}_i, \vec{U}_j$. Then $\vec{U}_i$ and $\vec{U}_j$ are consistent; hence since $[\![\lambda_{\vec{x}} N]\!]$ is consistent by Lemma B.1.5 a and b are consistent as well.

Next we show $(\vec{V}, U) \subseteq [\![\lambda_{\vec{x}} N]\!]$. Let $a \in U$; we show $(\vec{V}, a) \in [\![\lambda_{\vec{x}} N]\!]$. Fix i such that $a \in U_i$ and $\vec{V} \vdash U_i$, and let $W_i := \{\, (\vec{U}_i, b) \mid b \in U_i \,\} \subseteq [\![\lambda_{\vec{x}} N]\!]$. Since by the side induction hypothesis $[\![\lambda_{\vec{x}} N]\!]$ is deductively closed it suffices to prove $W_i \vdash (\vec{V}, a)$, i.e., $\{\, b \mid b \in U_i \wedge \vec{V} \vdash \vec{U}_i \,\} \vdash a$. But the latter set equals U_i, and $a \in U_i$.

Finally we show $(\vec{V}, U, b) \subseteq [\![\lambda_{\vec{x}} M]\!]$. Let

$$W' := \{(\vec{U}_1, U_1, a_1), \dots, (\vec{U}_n, U_n, a_n)\} \subseteq [\![\lambda_{\vec{x}} M]\!].$$

By side induction hypothesis it suffices to prove that $W' \vdash (\vec{V}, U, b)$, i.e., $\{\, a_i \mid \vec{V} \vdash \vec{U}_i \wedge U \vdash U_i \,\} \vdash b$. But by definition of U the latter set equals $\{\, a_i \mid \vec{V} \vdash \vec{U}_i \,\}$, which in turn entails b because by assumption $W \vdash (\vec{V}, b)$.

Now we can use (A) to infer $(\vec{V}, b) \in [\![\lambda_{\vec{x}} M]\!]$, as required.

Case (C). Assume $W \subseteq [\![\lambda_{\vec{x}} C]\!]$. Then W consists of $(\vec{U}, \vec{U'}, \mathrm{C}\vec{a^*})$ such that $\vec{U'} \vdash \vec{a^*}$. Assume further $W \vdash (\vec{V}, \vec{V'}, b)$. Then

$$\{\, \mathrm{C}\vec{a^*} \mid \exists_{\vec{U},\vec{U'}}((\vec{U}, \vec{U'}, \mathrm{C}\vec{a^*}) \in W \wedge \vec{V} \vdash \vec{U} \wedge \vec{V'} \vdash \vec{U'}) \,\} \vdash b.$$

By definition of entailment b has the form $\mathrm{C}\vec{b^*}$ such that

$$W_i := \{\, a \mid \exists_{\vec{U},\vec{U'},\vec{a^*}}(a = a_i^* \wedge (\vec{U}, \vec{U'}, \mathrm{C}\vec{a^*}) \in W \wedge \vec{V} \vdash \vec{U} \wedge \vec{V'} \vdash \vec{U'}) \,\} \vdash b_i^*.$$

We must show $(\vec{V}, \vec{V'}, \mathrm{C}\vec{b^*}) \in [\![\lambda_{\vec{x}} \mathrm{C}]\!]$, i.e., $\vec{V'} \vdash \vec{b^*}$. It suffices to show $V_i' \vdash W_i$, for every i. Let $a \in W_i$. Then there are $\vec{U}, \vec{U'}, \vec{a^*}$ such that $a = a_i^*$, $(\vec{U}, \vec{U'}, \mathrm{C}\vec{a^*}) \in W$ and $\vec{V'} \vdash \vec{U'}$. Hence $V_i' \vdash U_i' \vdash a_i^* = a$.

Case (D). Let $W = \{(\vec{U}_1, \vec{U}_1'', a_1), \ldots, (\vec{U}_n, \vec{U}_n'', a_n)\}$. For every i there is an $\vec{U}_i'$ such that

$$\frac{(\vec{U}_i, \vec{U}_i', a_i) \in [\![\lambda_{\vec{x},\vec{y_i}} M_i(\vec{y_i})]\!] \qquad \vec{U}_i'' \vdash \vec{P}_i(\vec{U}_i')}{(\vec{U}_i, \vec{U}_i'', a_i) \in [\![\lambda_{\vec{x}} D]\!]}$$

for $D\vec{P}_i(\vec{y_i}) = M_i(\vec{y_i})$ a computation rule. Assume $W \vdash (\vec{V}, \vec{V''}, b)$. We must prove $(\vec{V}, \vec{V''}, b) \in [\![\lambda_{\vec{x}} D]\!]$. Let

$$I := \{\, i \mid 1 \leq i \leq n \wedge \vec{V} \vdash \vec{U}_i \wedge \vec{V''} \vdash \vec{U}_i'' \,\}.$$

Then $\{\, a_i \mid i \in I \,\} \vdash b$, hence $I \neq \emptyset$. For $i \in I$ we have $\vec{V''} \vdash \vec{U}_i'' \vdash \vec{P}_i(\vec{U}_i')$, hence by (27) there are $\vec{V}_i'$ such that $\vec{V''} \sim \vec{P}_i(\vec{V}_i')$ and $\vec{V}_i' \vdash \vec{U}_i'$. In particular for $i, j \in I$

$$\vec{V''} \sim \vec{P}_i(\vec{V}_i') \sim \vec{P}_j(\vec{V}_j').$$

To simplify notation assume $I = \{1, \ldots, m\}$. Hence by the unification lemma $\vec{P}_1, \ldots, \vec{P}_m$ are unifiable with a most general unifier ϑ and there exists $\vec{W}$ such that

$$(\vec{P}_1\vartheta)(\vec{W}) = \cdots = (\vec{P}_m\vartheta)(\vec{W}) \sim \vec{P}_1(\vec{V}_1') \sim \cdots \sim \vec{P}_m(\vec{V}_m').$$

Let $i, j \in I$. Then by the conditions on computation rules $M_i\vartheta = M_j\vartheta$. Also $(\vec{y}_i\vartheta)(\vec{W}) \vdash \vec{V}_i' \vdash \vec{U}_i'$. Therefore by (23)

$$(\vec{V}, (\vec{y}_i\vartheta)(\vec{W}), a_i) \in [\![\lambda_{\vec{x},\vec{y_i}} M_i(\vec{y_i})]\!]$$

and hence by (26)

$$(\vec{V}, \vec{W}, a_i) \in [\![\lambda_{\vec{x},\vec{y_i}} M_i(\vec{y}_i\vartheta)]\!].$$

But $M_i(\vec{y}_i\vartheta) = M_i\vartheta = M_1\vartheta = M_1(\vec{y_1}\vartheta)$ and hence for all $i \in I$

$$(\vec{V}, \vec{W}, a_i) \in [\![\lambda_{\vec{x},\vec{y_i}} M_1(\vec{y_1}\vartheta)]\!].$$

Therefore $X := \{ (\vec{V}, \vec{W}, a_i) \mid i \in I \} \subseteq [\![\lambda_{\vec{x},\vec{y_i}} M_1(\vec{y_1}\vartheta)]\!]$. Since $\{ a_i \mid i \in I \} \vdash b$, we have $X \vdash (\vec{V}, \vec{W}, b)$ and hence the induction hypothesis implies $(\vec{V}, \vec{W}, b) \in [\![\lambda_{\vec{x},\vec{y_i}} M_1(\vec{y_1}\vartheta)]\!]$. Using (26) again we obtain $(\vec{V}, (\vec{y_i}\vartheta)(\vec{W}), b) \in [\![\lambda_{\vec{x},\vec{y_i}} M_1(\vec{y_1})]\!]$. Since $\vec{V}'' \sim \vec{P}_1(\vec{V}_1') \sim \vec{P}_1((\vec{y_1}\vartheta)(\vec{W}))$ we obtain $(\vec{V}, \vec{V}'', b) \in [\![\lambda_{\vec{x}} D]\!]$, by (D). □

Corollary. *$[\![\lambda_{\vec{x}} M]\!]$ is an ideal.*

B.2. Preservation of values

We now prove that Definition B.1.1 of the denotation of a term is reasonable in the sense that this denotation is not changed by an application of the standard (β- and η-) conversions or a computation rule. For the β-conversion part of this proof it is helpful to first introduce a more standard notation, which involves variable environments.

Definition. Assume that all free variables in M are among $\vec{x}$. Let $[\![M]\!]^{\vec{U}}_{\vec{x}} := \{ b \mid (\vec{U}, b) \in [\![\lambda_{\vec{x}} M]\!] \}$ and $[\![M]\!]^{\vec{u},\vec{V}}_{\vec{x},\vec{y}} := \bigcup_{\vec{U} \subseteq \vec{u}} [\![M]\!]^{\vec{U},\vec{V}}_{\vec{x},\vec{y}}$.

From (24) we obtain $[\![M]\!]^{\vec{U},V}_{\vec{x},y} = [\![M]\!]^{\vec{U}}_{\vec{x}}$ if $y \notin \mathrm{FV}(M)$, and similarly for ideals $\vec{u}, v$ instead of $\vec{U}, V$. We have a useful monotonicity property, which follows from the deductive closure of $[\![\lambda_{\vec{x}} M]\!]$.

Lemma B.2.1. (a) *If $\vec{V} \vdash \vec{U}$, $a \vdash b$ and $a \in [\![M]\!]^{\vec{U}}_{\vec{x}}$, then $b \in [\![M]\!]^{\vec{V}}_{\vec{x}}$.*
(b) *If $\vec{v} \supseteq \vec{u}$, $a \vdash b$ and $a \in [\![M]\!]^{\vec{u}}_{\vec{x}}$, then $b \in [\![M]\!]^{\vec{v}}_{\vec{x}}$.*

Proof. (a) $\vec{V} \vdash \vec{U}$, $a \vdash b$ and $(\vec{U}, a) \in [\![\lambda_{\vec{x}} M]\!]$ together imply $(\vec{V}, b) \in [\![\lambda_{\vec{x}} M]\!]$, by the deductive closure of $[\![\lambda_{\vec{x}} M]\!]$. (b) follows from (a). □

Lemma B.2.2. (a) *$[\![x_i]\!]^{\vec{U}}_{\vec{x}} = \overline{U}_i$ and $[\![x_i]\!]^{\vec{u}}_{\vec{x}} = u_i$.*
(b) *$[\![\lambda_y M]\!]^{\vec{U}}_{\vec{x}} = \{ (V, b) \mid b \in [\![M]\!]^{\vec{U},V}_{\vec{x},y} \}$ and $[\![\lambda_y M]\!]^{\vec{u}}_{\vec{x}} = \{ (V, b) \mid b \in [\![M]\!]^{\vec{u},V}_{\vec{x},y} \}$.*
(c) *$[\![MN]\!]^{\vec{U}}_{\vec{x}} = [\![M]\!]^{\vec{U}}_{\vec{x}} [\![N]\!]^{\vec{U}}_{\vec{x}}$ and $[\![MN]\!]^{\vec{u}}_{\vec{x}} = [\![M]\!]^{\vec{u}}_{\vec{x}} [\![N]\!]^{\vec{u}}_{\vec{x}}$.*

Proof. (b) It suffices to prove the first part. But $(V, b) \in [\![\lambda_y M]\!]^{\vec{U}}_{\vec{x}}$ and $b \in [\![M]\!]^{\vec{U},V}_{\vec{x},y}$ are both equivalent to $(\vec{U}, V, b) \in [\![\lambda_{\vec{x},y} M]\!]$.

(c) For the first part we argue as follows.

$$\begin{aligned} c \in [\![M]\!]^{\vec{U}}_{\vec{x}} [\![N]\!]^{\vec{U}}_{\vec{x}} &\leftrightarrow \exists_{V \subseteq [\![N]\!]^{\vec{U}}_{\vec{x}}} ((V, c) \in [\![M]\!]^{\vec{U}}_{\vec{x}}) \\ &\leftrightarrow \exists_V ((\vec{U}, V) \subseteq [\![\lambda_{\vec{x}} N]\!] \wedge (\vec{U}, V, c) \in [\![\lambda_{\vec{x}} M]\!]) \\ &\leftrightarrow (\vec{U}, c) \in [\![\lambda_{\vec{x}} (MN)]\!] \quad \text{by } (A) \\ &\leftrightarrow c \in [\![MN]\!]^{\vec{U}}_{\vec{x}}. \end{aligned}$$

The second part is an easy consequence:

$$\begin{aligned}
c \in [\![M]\!]^{\vec{u}}_{\vec{x}}[\![N]\!]^{\vec{u}}_{\vec{x}} &\leftrightarrow \exists_{V \subseteq [\![N]\!]^{\vec{u}}_{\vec{x}}}((V,c) \in [\![M]\!]^{\vec{u}}_{\vec{x}}) \\
&\leftrightarrow \exists_{V \subseteq [\![N]\!]^{\vec{u}}_{\vec{x}}} \exists_{\vec{U} \subseteq \vec{u}}((V,c) \in [\![M]\!]^{\vec{U}}_{\vec{x}}) \\
&\leftrightarrow \exists_{\vec{U}_1 \subseteq \vec{u}} \exists_{V \subseteq [\![N]\!]^{\vec{U}_1}_{\vec{x}}} \exists_{\vec{U} \subseteq \vec{u}}((V,c) \in [\![M]\!]^{\vec{U}}_{\vec{x}}) \\
&\leftrightarrow^{(*)} \exists_{\vec{U} \subseteq \vec{u}} \exists_{V \subseteq [\![N]\!]^{\vec{U}}_{\vec{x}}}((V,c) \in [\![M]\!]^{\vec{U}}_{\vec{x}}) \\
&\leftrightarrow \exists_{\vec{U} \subseteq \vec{u}}(c \in [\![M]\!]^{\vec{U}}_{\vec{x}}[\![N]\!]^{\vec{U}}_{\vec{x}}) \\
&\leftrightarrow \exists_{\vec{U} \subseteq \vec{u}}(c \in [\![MN]\!]^{\vec{U}}_{\vec{x}}) \quad \text{by the first part} \\
&\leftrightarrow c \in [\![MN]\!]^{\vec{u}}_{\vec{x}}.
\end{aligned}$$

Here is the proof of the equivalence marked $(*)$. The upward direction is obvious. For the downward direction we use monotonicity. Assume $\vec{U}_1 \subseteq \vec{u}$, $V \subseteq [\![N]\!]^{\vec{U}_1}_{\vec{x}}$, $\vec{U} \subseteq \vec{u}$ and $(V,c) \in [\![M]\!]^{\vec{U}}_{\vec{x}}$. Let $\vec{U}_2 := \vec{U}_1 \cup \vec{U} \subseteq \vec{u}$. Then by monotonicity $V \subseteq [\![N]\!]^{\vec{U}_2}_{\vec{x}}$ and $(V,c) \in [\![M]\!]^{\vec{U}_2}_{\vec{x}}$. □

COROLLARY. $[\![\lambda_y M]\!]^{\vec{u}}_{\vec{x}} v = [\![M]\!]^{\vec{u},v}_{\vec{x},y}$.

PROOF.

$$\begin{aligned}
b \in [\![\lambda_y M]\!]^{\vec{u}}_{\vec{x}} v &\leftrightarrow \exists_{V \subseteq v}((V,b) \in [\![\lambda_y M]\!]^{\vec{u}}_{\vec{x}}) \\
&\leftrightarrow \exists_{V \subseteq v}(b \in [\![M]\!]^{\vec{u},V}_{\vec{x},y}) \quad \text{by the lemma, part (b)} \\
&\leftrightarrow b \in [\![M]\!]^{\vec{u},v}_{\vec{x},y}.
\end{aligned}$$

□

LEMMA B.2.3 (Substitution). $[\![M(z)]\!]^{\vec{u},[\![N]\!]^{\vec{u}}_{\vec{x}}}_{\vec{x},z} = [\![M(N)]\!]^{\vec{u}}_{\vec{x}}$.

PROOF. By induction on M, and cases on the form of M.
Case $\lambda_y M$. For readability we leave out $\vec{x}$ and $\vec{u}$.

$$\begin{aligned}
[\![\lambda_y M(z)]\!]^{[\![N]\!]}_{z} &= \{\, (V,b) \mid b \in [\![M(z)]\!]^{[\![N]\!],V}_{z,y} \,\} \\
&= \{\, (V,b) \mid b \in [\![M(N)]\!]^{V}_{y} \,\} \quad \text{by induction hypothesis} \\
&= [\![\lambda_y M(N)]\!] \quad \text{by the last lemma, part (b)} \\
&= [\![(\lambda_y M)(N)]\!].
\end{aligned}$$

The other cases are easy. □

LEMMA B.2.4 (Preservation of values, β). $[\![(\lambda_y M(y))N]\!]^{\vec{u}}_{\vec{x}} = [\![M(N)]\!]^{\vec{u}}_{\vec{x}}$.

PROOF. Again we leave out $\vec{x}$, $\vec{u}$. By the last two lemmata and the corollary, $[\![(\lambda_y M(y))N]\!] = [\![\lambda_y M(y)]\!][\![N]\!] = [\![M(y)]\!]^{[\![N]\!]}_{y} = [\![M(N)]\!]$. □

LEMMA B.2.5 (Preservation of values, η). $[\![\lambda_y(My)]\!]^{\vec{u}}_{\vec{x}} = [\![M]\!]^{\vec{u}}_{\vec{x}}$ *if* $y \notin \mathrm{FV}(M)$.

PROOF.

$$\begin{aligned}(V,b) \in [\![\lambda_y(My)]\!]^{\vec{u}}_{\vec{x}} &\leftrightarrow \exists_{\vec{U}\subseteq\vec{u}}((\vec{U},V,b) \in [\![\lambda_{\vec{x},y}(My)]\!]) \\ &\leftrightarrow \exists_{\vec{U}\subseteq\vec{u}}((\vec{U},V,b) \in [\![\lambda_{\vec{x}}M]\!]) \quad \text{by (25)} \\ &\leftrightarrow (V,b) \in [\![M]\!]^{\vec{u}}_{\vec{x}}.\end{aligned}$$

□

LEMMA B.2.6 (Preservation of values, computation rules). *For every computation rule* $D\vec{P}(\vec{y}) = M$ *of a defined constant* D

$$[\![\lambda_{\vec{y}} D\vec{P}(\vec{y})]\!] = [\![\lambda_{\vec{y}} M]\!].$$

PROOF. The following are equivalent:

$$\begin{aligned}&(\vec{V},b) \in [\![\lambda_{\vec{y}} D\vec{P}(\vec{y})]\!] \\ &(\vec{P}(\vec{V}),b) \in [\![D]\!] = [\![\lambda_{\vec{z}} D\vec{z}]\!] \quad \text{by (26)} \\ &(\vec{V},b) \in [\![\lambda_{\vec{y}} M]\!],\end{aligned}$$

where the last step is by definition. □

APPENDIX C

Realizers for axioms

We show that the extracted term of $I^{\pm}$, ${}^{\mathrm{co}}I^{\pm}$ realizes the respective axiom. For the first two claims we only consider the inductive predicate $\sim_{\mathbb{L}}$ with $\mathbb{L}$ the algebra of lists of signed digits.

LEMMA C.0.1. *The constructors of $\mathbb{L}$ realize the clauses of $\sim_{\mathbb{L}}$.*

PROOF. We only consider the second constructor ::. We must show that :: realizes the following formula C equivalent to $(\sim_{\mathbb{L}})^{+}_{1}$:

$$\forall_{s_1,s_2}(s_1 \sim_{\mathbb{D}} s_2 \to \forall_{\ell_1,\ell_2}(\ell_1 \sim_{\mathbb{L}} \ell_2 \to s_1 :: \ell_1 \sim_{\mathbb{L}} s_2 :: \ell_2))$$

i.e., $:: \mathbf{r}\, C$. Pick s_1, s_2. The goal then is

$$:: \mathbf{r}\ (s_1 \sim_{\mathbb{D}} s_2 \to \forall_{\ell_1,\ell_2}(\ell_1 \sim_{\mathbb{L}} \ell_2 \to s_1 :: \ell_1 \sim_{\mathbb{L}} s_2 :: \ell_2)).$$

Pick s with $\sim^{\mathbf{r}}_{\mathbb{D}}(s_1, s_2, s)$. The goal then is

$$:: s\ \mathbf{r}\ \forall_{\ell_1,\ell_2}(\ell_1 \sim_{\mathbb{L}} \ell_2 \to s_1 :: \ell_1 \sim_{\mathbb{L}} s_2 :: \ell_2).$$

Pick ℓ_1, ℓ_2, ℓ with $\sim^{\mathbf{r}}_{\mathbb{L}}(\ell_1, \ell_2, \ell)$. The goal then is

$$(s :: \ell)\ \mathbf{r}\ (s_1 :: \ell_1 \sim_{\mathbb{L}} s_2 :: \ell_2), \quad \text{i.e.,}$$
$$\sim^{\mathbf{r}}_{\mathbb{L}}\ (s_1 :: \ell_1, s_2 :: \ell_2, s :: \ell).$$

But this follows from what we have by the second clause of $\sim^{\mathbf{r}}_{\mathbb{L}}$:

$$\forall_{s_1,s_2,s,\ell_1,\ell_2,\ell}(\sim^{\mathbf{r}}_{\mathbb{D}}(s_1, s_2, s) \to \sim^{\mathbf{r}}_{\mathbb{L}}(\ell_1, \ell_2, \ell) \to \sim^{\mathbf{r}}_{\mathbb{L}}(s_1 :: \ell_1, s_2 :: \ell_2, s :: \ell)). \quad \square$$

LEMMA C.0.2. *The recursion operator $\mathcal{R}^{\alpha}_{\mathbb{L}}$ realizes the least-fixed-point axiom $\sim^{-}_{\mathbb{L}}$.*

PROOF. We equivalently rewrite $\sim^{-}_{\mathbb{L}}$ as $C :=$

$$\begin{aligned}
&\forall_{\ell_1,\ell_2}(\ell_1 \sim_{\mathbb{L}} \ell_2 \to \\
&\quad X[][] \to \\
&\quad \forall_{s_1,s_2,\ell_1,\ell_2}(s_1 \sim_{\mathbb{D}} s_2 \to \ell_1 \sim_{\mathbb{L}} \ell_2 \to X\ell_1\ell_2 \to X(s_1 :: \ell_1, s_2 :: \ell_2)) \to \\
&\quad X\ell_1\ell_2)
\end{aligned}$$

to make its type the same as the one for $\mathcal{R}^{\alpha}_{\mathbb{L}}$:

$$\mathbb{L} \to \alpha \to (\mathbb{D} \to \mathbb{L} \to \alpha \to \alpha) \to \alpha.$$

We must show $\mathcal{R}^\alpha_{\mathbb{L}}\ \mathbf{r}\ C$. Pick ℓ_1, ℓ_2. The goal then is

$$\begin{aligned}&\mathcal{R}^\alpha_{\mathbb{L}}\ \mathbf{r}\ (\ell_1 \sim_{\mathbb{L}} \ell_2 \to\\ &\qquad X[][] \to\\ &\qquad \forall_{s_1,s_2,\ell_1,\ell_2}(s_1 \sim_{\mathbb{D}} s_2 \to \ell_1 \sim_{\mathbb{L}} \ell_2 \to X\ell_1\ell_2 \to X(s_1 :: \ell_1, s_2 :: \ell_2)) \to\\ &\qquad X\ell_1\ell_2).\end{aligned}$$

Pick ℓ with $\sim^{\mathbf{r}}_{\mathbb{L}}(\ell_1, \ell_2, \ell)$. Then the goal is

$$\begin{aligned}&\mathcal{R}^\alpha_{\mathbb{L}}\ell\ \mathbf{r}\ (X[][] \to\\ &\qquad \forall_{s_1,s_2,\ell_1,\ell_2}(s_1 \sim_{\mathbb{D}} s_2 \to \ell_1 \sim_{\mathbb{L}} \ell_2 \to X\ell_1\ell_2 \to X(s_1 :: \ell_1, s_2 :: \ell_2)) \to\\ &\qquad X\ell_1\ell_2).\end{aligned}$$

Pick x with $X^{\mathbf{r}}[][]x$. Then the goal is

$$\begin{aligned}&\mathcal{R}^\alpha_{\mathbb{L}}\ell x\ \mathbf{r}\ (\forall_{s_1,s_2,\ell_1,\ell_2}(s_1 \sim_{\mathbb{D}} s_2 \to \ell_1 \sim_{\mathbb{L}} \ell_2 \to X\ell_1\ell_2 \to X(s_1 :: \ell_1, s_2 :: \ell_2)) \to\\ &\qquad X\ell_1\ell_2).\end{aligned}$$

Pick f with

$$f\ \mathbf{r}\ \forall_{s_1,s_2,\ell_1,\ell_2}(s_1 \sim_{\mathbb{D}} s_2 \to \ell_1 \sim_{\mathbb{L}} \ell_2 \to X\ell_1\ell_2 \to X(s_1 :: \ell_1, s_2 :: \ell_2)),$$

which implies

$$\begin{aligned}\forall_{s_1,s_2,s,\ell_1,\ell_2,\ell,y}(&\sim^{\mathbf{r}}_{\mathbb{D}}(s_1, s_2, s) \to \sim^{\mathbf{r}}_{\mathbb{L}}(\ell_1, \ell_2, \ell) \to X^{\mathbf{r}}\ell_1\ell_2 y \to\\ &X^{\mathbf{r}}(s_1 :: \ell_1, s_2 :: \ell_2, fs\ell y)).\end{aligned}$$

Our goal is

$$X^{\mathbf{r}}(\ell_1, \ell_2, \mathcal{R}^\alpha_{\mathbb{L}}\ell x f) =: Q\ell_1\ell_2\ell.$$

To this end we use the elimination axiom for $\sim^{\mathbf{r}}_{\mathbb{L}}$:

$$\begin{aligned}&\forall_{\ell_1,\ell_2,\ell}(\sim^{\mathbf{r}}_{\mathbb{L}}(\ell_1, \ell_2, \ell) \to\\ &\qquad Q[][][] \to\\ &\qquad \forall_{s_1,s_2,s,\ell_1,\ell_2,\ell}(\sim^{\mathbf{r}}_{\mathbb{D}}(s_1, s_2, s) \to \sim^{\mathbf{r}}_{\mathbb{L}}(\ell_1, \ell_2, \ell) \to Q\ell_1\ell_2\ell \to\\ &\qquad\qquad Q(s_1 :: \ell_1, s_2 :: \ell_2, s :: \ell)) \to\\ &\qquad Q\ell_1\ell_2\ell).\end{aligned}$$

It suffices to prove the premises $Q[][][]$ and $\forall_{s_1,s_2,s,\ell_1,\ell_2,\ell}(\sim^{\mathbf{r}}_{\mathbb{D}}(s_1, s_2, s) \to \sim^{\mathbf{r}}_{\mathbb{L}}(\ell_1, \ell_2, \ell) \to Q\ell_1\ell_2\ell \to Q(s_1 :: \ell_1, s_2 :: \ell_2, s :: \ell))$. By a computation rule for $\mathcal{R}^\alpha_{\mathbb{L}}$ the former is $X^{\mathbf{r}}[][]x$, which we have. For the latter assume s_1, s_2, s, ℓ_1, ℓ_2, ℓ and its premises. We show $Q(s_1 :: \ell_1, s_2 :: \ell_2, s :: \ell)$, i.e.,

$$X^{\mathbf{r}}(s_1 :: \ell_1, s_2 :: \ell_2, \mathcal{R}^\alpha_{\mathbb{L}}(s :: \ell)xf).$$

By the computation rules for $\mathcal{R}^\alpha_{\mathbb{L}}$ this is the same as

$$X^{\mathbf{r}}(s_1 :: \ell_1, s_2 :: \ell_2, fs\ell(\mathcal{R}^\alpha_{\mathbb{L}}\ell xf)).$$

But with $y := \mathcal{R}^{\alpha}_{\mathbb{L}}\ell x f$ this follows from what we have. □

For the final two claims we only consider the coinductive predicate $\approx_{\mathbb{S}}$.

LEMMA C.0.3. *The destructor $\mathcal{D}_{\mathbb{S}}$ realizes the closure axiom $\approx^{-}_{\mathbb{S}}$.*

PROOF. Recall $\approx^{-}_{\mathbb{S}}$:

$$\forall_{u_1,u_2}(u_1 \approx_{\mathbb{S}} u_2 \to \exists_{s_1,s_2,u_1',u_2'}(s_1 \sim_{\mathbb{D}} s_2 \land u_1' \approx_{\mathbb{S}} u_2' \land u_1 \equiv s_1 :: u_1' \land u_2 \equiv s_2 :: u_2'))$$

with cotype ${}^{\mathrm{co}}\mathbb{S} \to \mathbb{D} \times {}^{\mathrm{co}}\mathbb{S}$. The goal is $\mathcal{D}_{\mathbb{S}}\ \mathbf{r} \approx^{-}_{\mathbb{S}}$, which unfolds into

$$\forall_{u_1,u_2,u}(\approx^{\mathbf{r}}_{\mathbb{S}}(u_1,u_2,u) \to \mathcal{D}_{\mathbb{S}}u\ \mathbf{r}\ \exists_{s_1,s_2,u_1',u_2'}(s_1 \sim_{\mathbb{D}} s_2 \land u_1' \approx_{\mathbb{S}} u_2' \land u_1 \equiv s_1 :: u_1' \land u_2 \equiv s_2 :: u_2')).$$

Assume $\approx^{\mathbf{r}}_{\mathbb{S}}(u_1,u_2,u)$. We need to prove

$$\exists_{s_1,s_2,u_1',u_2'}(\mathcal{D}_{\mathbb{S}}u\ \mathbf{r}\ (s_1 \sim_{\mathbb{D}} s_2 \land u_1' \approx_{\mathbb{S}} u_2') \land u_1 \equiv s_1 :: u_1' \land u_2 \equiv s_2 :: u_2').$$

By $(\approx^{\mathbf{r}}_{\mathbb{S}})^{-}$ from $\approx^{\mathbf{r}}_{\mathbb{S}}(u_1,u_2,u)$ we obtain s_1, s_2, s, u_1', u_2', u' such that

$$\sim^{\mathbf{r}}_{\mathbb{D}}(s_1,s_2,s) \land \approx^{\mathbf{r}}_{\mathbb{S}}(u_1',u_2',u') \land u_1 \equiv s_1 :: u_1' \land u_2 \equiv s_2 :: u_2' \land u \equiv s :: u'.$$

Take s_1, s_2, u_1', u_2'. It remains to show $\mathcal{D}_{\mathbb{S}}u\ \mathbf{r}\ (s_1 \sim_{\mathbb{D}} s_2 \land u_1' \approx_{\mathbb{S}} u_2')$. By the computation rule of $\mathcal{D}_{\mathbb{S}}$ we know $\mathcal{D}_{\mathbb{S}}u \equiv \mathcal{D}_{\mathbb{S}}(s :: u') \equiv \langle s, u'\rangle$. Hence we must prove $\sim^{\mathbf{r}}_{\mathbb{D}}(s_1,s_2,s)$ and $\approx^{\mathbf{r}}_{\mathbb{S}}(u_1,u_2,u)$, which we both have. □

LEMMA C.0.4. *The corecursion operator ${}^{\mathrm{co}}\mathcal{R}^{\alpha}_{\mathbb{S}}$ realizes the greatest-fixed-point axiom $\approx^{+}_{\mathbb{S}}$.*

PROOF. We equivalently rewrite $\approx^{+}_{\mathbb{S}}$ as $C :=$

$$\forall_{u_1,u_2}(Xu_1u_2 \to \forall_{u_1,u_2}(Xu_1u_2 \to \exists_{s_1,s_2,u_1',u_2'}(s_1 \sim_{\mathbb{D}} s_2 \land (u_1' \approx_{\mathbb{S}} u_2' \lor Xu_1'u_2') \land u_1 \equiv s_1 :: u_1' \land u_2 \equiv s_2 :: u_2')) \to u_1 \approx_{\mathbb{S}} u_2)$$

to make its cotype the same as the one for ${}^{\mathrm{co}}\mathcal{R}^{\alpha}_{\mathbb{S}}$:

$$\alpha \to (\alpha \to \mathbb{D} \times ({}^{\mathrm{co}}\mathbb{S} + \alpha)) \to {}^{\mathrm{co}}\mathbb{S}.$$

We must show that ${}^{\mathrm{co}}\mathcal{R}^{\alpha}_{\mathbb{S}}$ realizes C, or more formally ${}^{\mathrm{co}}\mathcal{R}^{\alpha}_{\mathbb{S}}\ \mathbf{r}\ C$. The goal then is

$${}^{\mathrm{co}}\mathcal{R}^{\alpha}_{\mathbb{S}}\ \mathbf{r}\ (Xu_1u_2 \to \forall_{u_1,u_2}(Xu_1u_2 \to \exists_{s_1,s_2,u_1',u_2'}(s_1 \sim_{\mathbb{D}} s_2 \land (u_1' \approx_{\mathbb{S}} u_2' \lor Xu_1'u_2') \land u_1 \equiv s_1 :: u_1' \land u_2 \equiv s_2 :: u_2')) \to u_1 \approx_{\mathbb{S}} u_2).$$

Pick u with $X^{\mathbf{r}}u_1u_2u$. Then the goal is

$$^{\mathrm{co}}\mathcal{R}^{\alpha}_{\mathbb{S}}u\ \mathbf{r}\ \forall_{u_1,u_2}(Xu_1u_2 \to \exists_{s_1,s_2,u_1',u_2'}(s_1 \sim_{\mathbb{D}} s_2 \land (u_1' \approx_{\mathbb{S}} u_2' \lor Xu_1'u_2') \land u_1 \equiv s_1 :: u_1' \land u_2 \equiv s_2 :: u_2')) \to u_1 \approx_{\mathbb{S}} u_2).$$

Pick f such that

$$f\ \mathbf{r}\ \forall_{u_1,u_2}(Xu_1u_2 \to \exists_{s_1,s_2,u_1',u_2'}(s_1 \sim_{\mathbb{D}} s_2 \land (u_1' \approx_{\mathbb{S}} u_2' \lor Xu_1'u_2') \land u_1 \equiv s_1 :: u_1' \land u_2 \equiv s_2 :: u_2'))$$

i.e.,

$$\forall_{u_1,u_2,u}(X^{\mathbf{r}}u_1u_2u \to fu\ \mathbf{r}\ \exists_{s_1,s_2,u_1',u_2'}(s_1 \sim_{\mathbb{D}} s_2 \land (u_1' \approx_{\mathbb{S}} u_2' \lor Xu_1'u_2') \land u_1 \equiv s_1 :: u_1' \land u_2 \equiv s_2 :: u_2')).$$

Our goal is

$$\approx^{\mathbf{r}}_{\mathbb{S}}(u_1, u_2, {}^{\mathrm{co}}\mathcal{R}^{\alpha}_{\mathbb{S}}uf).$$

To this end we use the greatest-fixed-point axiom for $\approx^{\mathbf{r}}_{\mathbb{S}}$ in the form

$$\begin{aligned}&\forall_{u_1,u_2,u}(Qu_1u_2u \to\\ &\quad \forall_{u_1,u_2,u}(Qu_1u_2u \to\\ &\qquad \exists_{s_1,s_2,s,u_1',u_2',u'}(\sim^{\mathbf{r}}_{\mathbb{D}}(s_1,s_2,s) \land (\approx^{\mathbf{r}}_{\mathbb{S}}(u_1',u_2',u') \lor Qu_1u_2u') \land\\ &\qquad\quad u_1 \equiv s_1{::}u_1' \land u_2 \equiv s_2{::}u_2' \land u \equiv s{::}u')) \to\\ &\quad \approx^{\mathbf{r}}_{\mathbb{S}}(u_1,u_2,u))\end{aligned}$$

with

$$\exists_{z'}(X^{\mathbf{r}}u_1u_2z' \land u \equiv {}^{\mathrm{co}}\mathcal{R}^{\alpha}_{\mathbb{S}}z'f) =: Qu_1u_2u.$$

It suffices to prove the closure property of Q. Let u_1, u_2, u and also u' be given such that

$$X^{\mathbf{r}}u_1u_2u' \land u \equiv {}^{\mathrm{co}}\mathcal{R}^{\alpha}_{\mathbb{S}}u'f.$$

We need to show

$$\begin{aligned}&\exists_{s_1,s_2,s,u_1',u_2',u'}(\\ &\sim^{\mathbf{r}}_{\mathbb{D}}(s_1,s_2,s) \land (\approx^{\mathbf{r}}_{\mathbb{S}}(u_1',u_2',u') \lor \exists_{u''}(X^{\mathbf{r}}u_1u_2u'' \land u \equiv {}^{\mathrm{co}}\mathcal{R}^{\alpha}_{\mathbb{S}}u''f)) \land\\ &u_1 \equiv s_1 :: u_1' \land u_2 \equiv s_2 :: u_2' \land u \equiv s :: u').\end{aligned} \tag{28}$$

First note that $\sim^{\mathbf{r}}_{\mathbb{D}}(s_1,s_2,s)$ is equivalent to $s_1 \equiv s_2 \equiv s$. Since $X^{\mathbf{r}}u_1u_2u'$ we know

$$fu'\ \mathbf{r}\ \exists_{s_1,s_2,u_1',u_2'}(s_1 \sim_{\mathbb{D}} s_2 \land (u_1' \approx_{\mathbb{S}} u_2' \lor Xu_1'u_2') \land u_1 \equiv s_1 :: u_1' \land u_2 \equiv s_2 :: u_2').$$

Then $fu' \equiv \langle s, w\rangle$ with $\sim^{\mathbf{r}}_{\mathbb{D}}(s_1,s_2,s)$ and $w\ \mathbf{r}\ (u_1' \approx_{\mathbb{S}} u_2' \lor Xu_1'u_2')$, for some s_1, s_2, u_1', u_2' such that $u_1 \equiv s_1 :: u_1'$ and $u_2 \equiv s_2 :: u_2'$. Hence

$$\exists_{u'}(\approx^{\mathbf{r}}_{\mathbb{S}}(u_1',u_2',u') \land w \equiv \mathrm{InL}(u')) \lor \exists_{u''}(X^{\mathbf{r}}u_1'u_2'u'' \land w \equiv \mathrm{InR}(u'')).$$

We distinguish cases on this disjunction. Recall

$$ {}^{\mathrm{co}}\mathcal{R}^{\alpha}_{\mathbb{S}} u f \equiv \begin{cases} s :: u & \text{if } fu \equiv \langle s, \mathrm{InL}(u)\rangle, \\ s :: {}^{\mathrm{co}}\mathcal{R}^{\alpha}_{\mathbb{S}} u' f & \text{if } fu \equiv \langle s, \mathrm{InR}(u')\rangle. \end{cases} $$

Case L. $\approx^{\mathbf{r}}_{\mathbb{S}}(u'_1, u'_2, u') \wedge w \equiv \mathrm{InL}(u')$ for some u'. Then (28) holds, since $u \equiv {}^{\mathrm{co}}\mathcal{R}^{\alpha}_{\mathbb{S}} u' f \equiv s :: u'$.

Case R. $X^{\mathbf{r}} u'_1 u'_2 u'' \wedge w \equiv \mathrm{InR}(u'')$ for some u''. Then again (28) holds with $u' := {}^{\mathrm{co}}\mathcal{R}^{\alpha}_{\mathbb{S}} u'' f$, since $u \equiv {}^{\mathrm{co}}\mathcal{R}^{\alpha}_{\mathbb{S}} u' f \equiv s :: {}^{\mathrm{co}}\mathcal{R}^{\alpha}_{\mathbb{S}} u'' f \equiv s :: u'$. □

Lemma C.0.5 (Identities realize I^- for one-clause-nc I). *For one-clause-nc inductive predicates the elimination axiom with a c.r. competitor predicate is realized by the identity.*

Proof. For $\vec{A}$ n.c. we have

$$ \begin{aligned} &(\lambda_z z)\ \mathbf{r}\ \forall_{\vec{x}}(I\vec{x} \to \forall_{\vec{y}}(\vec{A} \to X\vec{t}\,) \to X\vec{x}\,) \\ &\forall_{\vec{x}}(I\vec{x} \to (\lambda_z z)\ \mathbf{r}\ (\forall_{\vec{y}}(\vec{A} \to X\vec{t}\,) \to X\vec{x}\,)) \\ &\forall_{\vec{x}}(I\vec{x} \to \forall_z(z\ \mathbf{r}\ \forall_{\vec{y}}(\vec{A} \to X\vec{t}\,) \to z\ \mathbf{r}\ X\vec{x}\,)) \\ &\forall_{\vec{x}}(I\vec{x} \to \forall_z(\forall_{\vec{y}}(\vec{A} \to z\ \mathbf{r}\ X\vec{t}\,) \to z\ \mathbf{r}\ X\vec{x}\,)) \end{aligned} $$

which is an instance of the same elimination axiom. □

Bibliography

U. Berger and M. Seisenberger. Proofs, programs, processes. In F. Ferreira et al., editors, *Proceedings CiE 2010*, volume 6158 of *LNCS*, pages 39–48. Springer Verlag, Berlin, Heidelberg, New York, 2010. 5.4.2

Y. L. Ershov. Model C of partial continuous functionals. In R. Gandy and M. Hyland, editors, *Logic Colloquium 1976*, pages 455–467. North-Holland, Amsterdam, 1977. 2

G. Gentzen. Untersuchungen über das logische Schließen I, II. *Mathematische Zeitschrift*, 39:176–210, 405–431, 1935. 1.2

K. Gödel. Über eine bisher noch nicht benützte Erweiterung des finiten Standpunkts. *Dialectica*, 12:280–287, 1958. 2.3, 2.3.2

A. Heyting, editor. *Constructivity in Mathematics*, 1959. North-Holland, Amsterdam.

D. Hilbert. Über das Unendliche. *Mathematische Annalen*, 95:161–190, 1925. 2.3.2

A. N. Kolmogorov. Zur Deutung der intuitionistischen Logik. *Math. Zeitschr.*, 35:58–65, 1932. 3.1, 4

K. G. Larsen and G. Winskel. Using information systems to solve recursive domain equations. *Information and Computation*, 91:232–258, 1991. 2.1

A. Martelli and U. Montanari. An efficient unification algorithm. *ACM Transactions on Programming Languages and Systems*, 4(2):258–282, April 1982. A

G. D. Plotkin. LCF considered as a programming language. *Theoretical Computer Science*, 5:223–255, 1977. 2.3

J. A. Robinson. A machine–oriented logic based on the resolution principle. *Journal of the Association for Computing Machinery*, 12(1):23–41, 1965. A

H. Schwichtenberg and S. S. Wainer. *Proofs and Computations*. Perspectives in Logic. Association for Symbolic Logic and Cambridge University Press, 2012.

D. Scott. Domains for denotational semantics. In E. Nielsen and E. Schmidt, editors, *Automata, Languages and Programming*, volume 140 of *LNCS*, pages 577–613. Springer Verlag, Berlin, Heidelberg, New York, 1982. 2, 2.1

A. S. Troelstra and H. Schwichtenberg. *Basic Proof Theory.* Cambridge University Press, second edition, 2000.

A. S. Troelstra and D. van Dalen. *Constructivism in Mathematics. An Introduction*, volume 121, 123 of *Studies in Logic and the Foundations of Mathematics.* North-Holland, Amsterdam, 1988. 1.3

Index