We sketch a constructive formal theory TCF+ of computable functionals, based on the partial continuous functionals as their intended domain. Such a task had long ago been started by Dana Scott [12, 15], under the well-known abbreviation LCF (logic of computable functionals). The present approach differs from Scott’s in two aspects.

(i) The intended semantical domains for the base types are non-flat free algebras, given by their constructors, where the latter are injective and have disjoint ranges; both properties do not hold in the flat case.

(ii) TCF+ has the facility to argue not only about the functionals themselves, but also about their finite approximations.

In this setting we give an informal proof (based on Berger [2]) of Kreisel’s density theorem [7], and an adaption of Plotkin’s definability theorem [10, 11]. We then show that both proofs can be formalized in TCF+.

The naive model of a finitely typed theory like TCF+ is the full set theoretic hierarchy of functionals of finite types. However, this immediately leads to higher cardinalities, and does not lend itself well for a constructive theory of computability. A more appropriate semantics for typed languages has its roots in work of Kreisel [7] (where formal neighborhoods are used) and Kleene [6]. This line of research was developed in a mathematically more satisfactory way by Scott [13] and Ershov [3]. Today this theory is usually presented in the context of abstract domain theory (see [16, 1]); it is based on classical logic. The present work can be seen as an attempt to develop a constructive theory of formal neighborhoods for continuous functionals, in a direct and intuitive style. The task is to replace abstract domain theory by a more concrete, finitary theory of representations. As a framework we use Scott’s information systems (see [14, 8, 16]). In this setup the basic notion is that of a “token”, or unit of information. The elements or points of the domain appear as abstract or “ideal” entities: possibly infinite sets of tokens, which are “consistent” and “deductively closed”.

The paper is organized as follows. Section 1 collects basic facts about information systems, and section 2 contains informal proofs of the density and definability theorems for the case of the non-flat natural numbers, in

Date: February 20, 2010.
enough detail to guide the formalization. Section 3 develops the language and axioms of the theory TCF\(^+\). The formalization of both theorems in TCF\(^+\) is discussed in section 4.

1. Partial Continuous Functionals

1.1. Information systems. The basic idea of information systems is to provide an axiomatic setting to describe approximations of abstract objects (like functions or functionals) by concrete, finite ones. The axioms below are a minor modification of Scott’s [14], due to Larsen and Winskel [8].

An information system is a structure \((A, \text{Con}, \vdash)\) where \(A\) is a countable set (the tokens), \(\text{Con}\) is a nonempty set of finite subsets of \(A\) (the consistent sets) and \(\vdash\) is a subset of \(\text{Con} \times A\) (the entailment relation), which satisfy

\[
\begin{align*}
U \subseteq V \in \text{Con} &\rightarrow U \in \text{Con}, \\
\{a\} &\in \text{Con}, \\
U \vdash a &\rightarrow U \cup \{a\} \in \text{Con}, \\
a \in U \in \text{Con} &\rightarrow U \vdash a, \\
U, V \in \text{Con} &\rightarrow \forall_{a \in V}(U \vdash a) \rightarrow V \vdash b \rightarrow U \vdash b.
\end{align*}
\]

The elements \(U\) of \(\text{Con}\) are called formal neighborhoods. We use \(U, V, W\) to denote finite sets, and write

\[
\begin{align*}
U \vdash V &\text{ for } U \in \text{Con} \land \forall_{a \in V}(U \vdash a), \\
a \uparrow b &\text{ for } \{a, b\} \in \text{Con} \quad (a, b \text{ are consistent}), \\
U \uparrow V &\text{ for } \forall_{a \in U, b \in V}(a \uparrow b).
\end{align*}
\]

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

\[
\begin{align*}
U \subseteq x \rightarrow U \in \text{Con} &\quad (x \text{ is consistent}), \\
x \supseteq U \vdash a &\rightarrow a \in x \quad (x \text{ is deductively closed}).
\end{align*}
\]

For example the deductive closure \(U := \{a \mid U \vdash a\}\) of \(U\) is an ideal. The set of all ideals of \(A\) is denoted by \(|A|\).

Examples. Every countable set \(A\) can be turned into a flat information system by letting the set of tokens be \(A\), \(\text{Con} := \{\emptyset\} \cup \{\{a\} \mid a \in A\}\) and \(U \vdash a\) mean \(a \in U\). In this case the ideals are just the elements of Con.

Consider the algebras \(B\) (booleans), \(N\) (natural numbers), \(P\) (positive numbers written binary), \(D\) (derivations) given by the constructors

\[
\begin{align*}
\text{it}^B, \text{tt}^B &\text{ for } B, \\
0^N \text{ and } S^N \rightarrow ^N &\text{ (successor) for } N,
\end{align*}
\]

\[
\begin{align*}
\text{for } B, \\
0^N \text{ and } S^N \rightarrow ^N &\text{ (successor) for } N,
\end{align*}
\]
For each of them we define an information system $C_\iota = (\text{Tok}_\iota, \text{Con}_\iota, \vdash_\iota)$:

(a) The tokens $a \in \text{Tok}_\iota$ are the constructor expressions $C_{a_1^*} \ldots a_n^*$ where $a_i^*$ is an extended token, i.e., a token or the special symbol $\ast$ which carries no information.

(b) A finite set $U$ of tokens in $\text{Tok}_\iota$ is consistent (i.e., $\in \text{Con}_\iota$) if its elements start with the same $n$-ary constructor $C$, say $U = \{ C_{a_1^*}, \ldots, C_{a_m^*} \}$, and $U_i \in \text{Con}_\iota$ where $U_i$ consists of the (proper) tokens among $a_1^* \ldots a_m^*$, with $U_i$ as in (b) above (and $U \vdash \ast$ defined to be true).

(c) $\{ C_{a_1^*}, \ldots, C_{a_m^*} \} \vdash_\iota C' a^\ast$ is defined to mean $C = C'$, $m \geq 1$ and $U_i \vdash a_i^*$, with $U_i$ as in (b) above.

For example, the tokens for $\mathbb{N}$ are shown in Figure 1. For tokens $a, b$ we have $\{ a \} \vdash_\iota b$ if and only if there is a path from $a$ (up) to $b$ (down). In $\mathbb{D}$, the set $\{ C_0^*, C_0^0 \}$ is consistent, and $\{ C_0^*, C_0^0 \} \vdash_\iota C_0^0$.

A token is called total if it has the form $C_{a_1^*} \ldots a_n^*$ with a total token $a_i^*$ at every argument position. For example, the total tokens for $\mathbb{N}$ are all $S_{S^0}$, and for $\mathbb{D}$ all $\ast$-free constructor trees built from 0 and $C$.

By induction on the formation of tokens, one easily sees the following.

**Lemma** (Comparability). If $\iota$ has at most unary constructors, then any two consistent tokens $a, b$ are comparable, i.e., $\{ a \} \vdash_\iota b$ or $\{ b \} \vdash_\iota a$.

### 1.2. Function spaces.

Let $A = (A, \text{Con}_A, \vdash_A)$ and $B = (B, \text{Con}_B, \vdash_B)$ be information systems. Define the function space $A \rightarrow B = (C, \text{Con}_C, \vdash_C)$ by

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

$$(U_i, b_i) \mid i \in I \} \in \text{Con} := \forall J \subseteq I ( \bigcup_{i \in J} U_j \in \text{Con}_A \rightarrow \{ b_j \mid j \in J \} \in \text{Con}_B), $$

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

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

![Figure 1. Tokens and entailment for $\mathbb{N}$](image-url)
From the definition of Con we know that this set is in ConB. Now define
\( W \vdash (U, b) \) by \( WU \vdash_B b \). Clearly application is monotone in the second
argument, in the sense that \( U \vdash_A U' \) implies \( WU' \subseteq WU \), hence \( WU \vdash_B WU' \). Application is also monotone in the first argument, i.e.,
\[
W \vdash W' \implies WU \vdash_B W'U.
\]
Using this one easily proves that \( A \to B \) is an information system provided
\( A \) and \( B \) are.

For any information system \( A \) the set of all \( C_U := \{ x \in |A| \mid U \subseteq x \} \) with
\( U \in \text{Con} \) forms the basis of a topology on \(|A|\), the Scott topology. The
continuous functions (w.r.t. the Scott topology) from \(|A|\) to \(|B|\) are in a
natural bijective correspondence with the ideals of \( A \to B \):

(a) With any ideal \( r \in |A \to B| \) we can associate a continuous function
\(|r| : |A| \to |B| \) 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 \( f : |A| \to |B| \) we can associate
an ideal \( f : A \to B \) by \( \hat{f} := \{ (U, b) \mid b \in f(U) \} \).

These assignments are inverse to each other, i.e., \( f = |\hat{f}| \) and \( r = \hat{|r|} \). We
usually write \( rz \) for \(|r|z \), and similarly \( (U, b) \in f \) for \( (U, b) \in \hat{f} \).

Lemma (Approximable maps [14]). Let \( A = (A, \text{Con}_A, \vdash_A) \) and \( B =
(B, \text{Con}_B, \vdash_B) \) be information systems. The ideals of \( A \to B \) are exactly
the approximable maps from \( A \) to \( B \), i.e., the relations \( r \subseteq \text{Con}_A \times B \) with

(a) If \( (U, b_1), \ldots, (U, b_n) \in r \), then \( \{b_1, \ldots, b_n\} \in \text{Con}_B \);
(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 \( (U, b) \in r \).

Types are built from base types \( \iota \) (the algebras above) by \( \rho \to \sigma \). For every
type \( \rho \) we define the information system \( C_\rho = (\text{To}, \text{Con}_\rho, \vdash_\rho) \) starting
from the \( C_\iota \) by formation of function spaces \( C_\rho \to \sigma := C_\rho \to C_\sigma \). The set
\( |C_\rho| \) of ideals in \( C_\rho \) is the set of partial continuous functionals of type \( \rho \).
A partial continuous functional \( x \in |C_\rho| \) is computable if it is recursively
enumerable when viewed as a set of tokens. The information systems \( C_\rho \)
enjoy the pleasant property of “coherence”, which amounts to the possibility
of locating inconsistencies in two-element sets of data objects. Generally,
an information system \( A = (A, \text{Con}, \vdash) \) is coherent if it satisfies: \( U \subseteq A \) is
consistent if and only if all of its two-element subsets are.

It is easy to see that every constructor \( C \) generates a continuous function
\( r_C := \{ (U, Ca^x) \mid \hat{U} \vdash a^x \} \) in the function space (where \( (U, b) \) means
\( (U_1, \ldots, (U_n, b) \ldots) \), and that
\[
|r_C| \sqsubseteq |r_C| \forall x \rightarrow \sqsubseteq \forall \sqsubseteq \forall y.
\]
If $C_1, C_2$ are distinct constructors of $\iota$, then $|r_{C_1}|\vec{x} \neq |r_{C_2}|\vec{y}$, since the two ideals are non-empty and disjoint. Hence constructors are injective and have disjoint ranges. Notice that neither property holds for flat information systems, since for them, by monotonicity, constructors need to be strict (i.e., if one argument is the empty ideal, then the value is as well). But then

$$|r_C|\emptyset \vec{y} = \emptyset = |r_C|\emptyset \vec{x}, \quad |r_{C_1}|\emptyset = \emptyset = |r_{C_2}|\emptyset,$$

where $C$ is a binary and $C_1, C_2$ are unary constructors.

2. Computable functionals

2.1. Terms and their denotational semantics. Terms are built from (typed) variables and (typed) constants (constructors $C$ or defined constants $D$, see below) by application and abstraction:

$$M, N ::= x^\rho | C^\rho | D^\rho | (\lambda x^\rho M^\sigma)^{\rho \rightarrow \sigma} | (M^{\rho \rightarrow \sigma} N^\rho)^\sigma.$$

Every defined constant $D$ comes with a system of computation rules, consisting of finitely many equations $D\vec{P}(\vec{y}_i) = M_i (i = 1, \ldots, n)$ with free variables of $\vec{P}(\vec{y}_i)$ and $M_i$ among $\vec{y}_i$, where the $\vec{P}(\vec{y}_i)$ must be “constructor patterns”, i.e., lists of applicative terms built from constructors and distinct variables, with each constructor $C$ occurring in a context $C\vec{P}(\vec{y}_i)$ of base type.

We assume that $\vec{P}_i$ and $\vec{P}_j$ for $i \neq j$ are non-unifiable. Examples are

(i) the predecessor function $P: \mathbb{N} \rightarrow \mathbb{N}$ defined by the computation rules

$$P0 = 0, \quad P(Sn) = n,$$

(ii) Gödel’s primitive recursion operators $R^\tau_N: \mathbb{N} \rightarrow \tau \rightarrow (\mathbb{N} \rightarrow \tau \rightarrow \tau) \rightarrow \tau$ with computation rules $R0fg = f, \quad R(Sn)fg = gn(Rnf)$, and

(iii) the least-fixed-point operators $Y^\rho$ of type $(\rho \rightarrow \rho) \rightarrow \rho$ defined by the computation rule $Y^\rho f = f(Y^\rho f)$.

For every closed term $\lambda x^\rho M$ of type $\vec{\rho} \rightarrow \sigma$ we inductively define a set $\llbracket \lambda x^\rho M \rrbracket$ of tokens of type $\vec{\rho} \rightarrow \sigma$.

$$U_i \vdash b \quad (\vec{U}, b) \in \llbracket \lambda x^\rho M \rrbracket (V), \quad \frac{(U, V, c) \in \llbracket \lambda x^\rho M \rrbracket \quad (U, c) \in \llbracket \lambda x^\rho N \rrbracket}{\frac{(U, c) \in \llbracket \lambda x^\rho (MN) \rrbracket}{(U, V, c) \in \llbracket \lambda x^\rho (MN) \rrbracket}} (A).$$

For every constructor $C$ and defined constant $D$ we have

$$\frac{\vec{V} \vdash b^\vec{\rho}}{(U, \vec{V}, Cb) \in \llbracket \lambda x^\rho C \rrbracket (C)}, \quad \frac{(U, \vec{V}, b) \in \llbracket \lambda x^\rho M \rrbracket \quad \vec{W} \vdash \vec{P}(\vec{V})}{(U, \vec{W}, b) \in \llbracket \lambda x^\rho D \rrbracket (D)},$$

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

Here $(\vec{U}, V) \subseteq \llbracket \lambda x^\rho M \rrbracket$ means $(\vec{U}, b) \in \llbracket \lambda x^\rho M \rrbracket$ for all (finitely many) $b \in V$, and $(\vec{U}, b)$ denotes $(U_1, \ldots, U_n, b, \ldots)$. For a constructor pattern $\vec{P}(\vec{x})$.
and a list $\vec{V}$ of the same length and types as $\vec{x}$, $\vec{P}(\vec{V})$ is a list of formal neighborhoods of the same length and types as $\vec{P}(\vec{x})$: $x(V)$ is $V$, and

$$(C\vec{P})(\vec{V}) := \{ C\vec{b}^* | b_i^* \in P_i(\vec{V}) \text{ if } P_i(\vec{V}) \neq \emptyset, \text{ and } b_i^* = * \text{ otherwise} \}.$$ 

The height of a derivation of $(\vec{U}, b) : \in [\lambda xM]$ is defined as usual, by adding 1 at each rule. We define its $D$-height similarly, where only rules (D) count.

**Theorem.** (a) For every term $M$, $[\lambda xM]$ is an ideal.
(b) If a term $M$ converts to $M'$ by $\beta\eta$-conversion or application of a computation rule, then its value is preserved, i.e., $[M] = [M']$.

For a term $M$ with free variables among $\vec{x}$ and an assignment $\vec{x} \mapsto \vec{u}$ of ideals $\vec{u}$ to $\vec{x}$ let $\llbracket M \rrbracket_{\vec{x}}^\vec{u} := \bigcup_{\vec{u} \subseteq \vec{a}} \llbracket M \rrbracket_{\vec{x}}^{\vec{a}}$ with $\llbracket M \rrbracket_{\vec{x}}^{\vec{a}} := \{ b | (\vec{u}, b) \in [\lambda xM] \}$. Notice that a consequence of (A) is

(1) $c \in \llbracket MN \rrbracket_{\vec{x}}^\vec{u} \iff \exists_{\vec{y} \subseteq [N]_{\vec{x}}}((V, c) \in \llbracket M \rrbracket_{\vec{x}}^\vec{y})$ (continuity of application).

**Proposition.** For every $n > 0$, there is a derivation of $(W, b) \in [Y]$ with $D$-height $n$ if and only if $W^n(\emptyset) \vdash b$.

**Proof.** Every derivation of $(W, b) \in [Y]$ must have the form

$$
\begin{array}{c}
\begin{array}{c}
\vdash (V, b) \quad \vdash (V, b) \\
(\vec{W}, V, b) \in [\lambda f] \\
\end{array}
\end{array}
\begin{array}{c}
\begin{array}{c}
\vdash (V, b) \quad \vdash (V, b) \\
(\vec{W}, V, b) \in [\lambda f] \\
\end{array}
\end{array}$$

$$
\begin{array}{c}
\begin{array}{c}
\vdash (V, b) \\
(\vec{W}, b) \in [\lambda f(Yf)] \\
\end{array}
\end{array}
\begin{array}{c}
\begin{array}{c}
\vdash (V, b) \\
(\vec{W}, b) \in [\lambda f(Yf)] \\
\end{array}
\end{array}$$

$$
\begin{array}{c}
\begin{array}{c}
\vdash (V, b) \\
(\vec{W}, b) \in [Y] \\
\end{array}
\end{array}
\begin{array}{c}
\begin{array}{c}
\vdash (V, b) \\
(\vec{W}, b) \in [Y] \\
\end{array}
\end{array}
$$

(D), assuming $W \vdash \vec{W}$

with $V := \{ b_i | i \in I \}$, $W_i := \{ (V_{ij}, b_{ij}) | j \in I_i \}$.

“$\rightarrow$”. By induction on the $D$-height. We have $(\vec{W}, W_i, b_i) \in [\lambda f Y]$, $\vec{W} \vdash W_i$ and $\vec{W} \vdash (V, b)$. By induction hypothesis $W_i^n(\emptyset) \vdash b_i$, and $W^n(\emptyset) \vdash W_i^n(\emptyset)$ by monotonicity of application. Because of $W^n(\emptyset) \vdash W^n(\emptyset)$ (proved by induction on $n$, using monotonicity) we obtain $\vec{W}^n(\emptyset) \vdash b_i$ with $n := \max n_i$, i.e., $\vec{W}^n(\emptyset) \vdash V$. Recall that $\vec{W} \vdash (V, b)$ was defined to mean $\vec{W} V \vdash b$. Hence $\vec{W}(W^n(\emptyset)) \vdash b$ and therefore $W^{n+1}(\emptyset) \vdash b$.

“$\leftarrow$”. By induction on $n$. Let $W(W^n(\emptyset)) \vdash b$, i.e., $W \vdash (V, b)$ with $V := W^n(\emptyset) =: \{ b_i | i \in I \}$. Then $W^n(\emptyset) \vdash b_i$, hence by induction hypothesis $(W, b_i) \in [Y]$. Substituting $W$ for $W$ and all $W_i$ in the derivation above gives the claim $(W, b) \in [Y]$.  

**Corollary.** The fixed point operator $Y$ has the property

$$
b \in [Y] w \leftrightarrow \exists_k (b \in w^{k+1}(\emptyset)).$$
Proof. Since \( w^{k+1} \) for fixed \( k \) is continuous in \( w \), from \( b \in w^{k+1} \) we can infer \( W^{k+1} \vdash b \) for some \( W \subseteq w \), and conversely. Moreover \( b \in [Y]w \) is equivalent to \( (W,b) \in [Y] \) for some \( W \subseteq w \), by \((A)\). Now apply the proposition. \( \square \)

2.2. Total functionals. We now single out the total continuous functionals from the partial ones. Our main goal will be the density theorem, which says that every finite functional can be extended to a total one.

The total ideals \( x \) of type \( \rho \) (notation \( x \in G_\rho \)) and the equivalence relation \( x_1 \approx x_2 \) between them are defined inductively.

(a) For an algebra \( \iota \), the total ideals \( x \) are those of the form \( C\bar{z} \) with \( C \) a constructor of \( \iota \) and \( \bar{z} \) total (\( C \) denotes the continuous function \( |r_C| \)).

Two total ideals \( x_1, x_2 \) are equivalent (written \( x_1 \approx x_2 \)) if both are of the form \( C\bar{z} \) with the same constructor \( C \) of \( \iota \), and \( z_{1j} \approx z_{2j} \) for all \( j \).

(b) An ideal \( r \) of type \( \rho \rightarrow \sigma \) is total if and only if for all total \( z \) of type \( \rho \), the result \( |r|z \) of applying \( r \) to \( z \) is total. For \( f,g \in G_{\rho \rightarrow \sigma} \) define \( f \approx_{\rho \rightarrow \sigma} g \) by \( \forall x \in G_\rho (fx \approx_\sigma gx) \).

We show that \( x \approx_\rho y \) implies \( fx \approx_\sigma fy \), following Longo and Moggi [9].

**Lemma (Extension).** If \( f \in G_\rho \), \( g \in |C_\rho| \) and \( f \subseteq g \), then \( g \in G_\rho \).

**Proof.** By induction on \( \rho \). For base types \( \iota \) use induction on the definition of \( f \in G_\iota \). Case \( \rho \rightarrow \sigma \). Assume \( f \in G_{\rho \rightarrow \sigma} \) and \( f \subseteq g \). We show \( g \in G_{\rho \rightarrow \sigma} \).

So let \( x \in G_\rho \). We show \( gx \in G_\sigma \). But \( gx \supseteq fx \in G_\sigma \), so the claim follows by the induction hypothesis. \( \square \)

**Lemma.** \( (f_1 \cap f_2)x = f_1x \cap f_2x \), for \( f_1, f_2 \in |C_{\rho \rightarrow \sigma}| \) and \( x \in |C_\rho| \).

**Proof.** By the definition of \( |r| \),

\[
|f_1 \cap f_2| x
= \{ b \in \text{Tok}_\sigma \mid \exists U \subseteq x ((U,b) \in f_1 \cap f_2) \}
= \{ b \in \text{Tok}_\sigma \mid \exists U \subseteq x ((U_1,b) \in f_1) \} \cap \{ b \in \text{Tok}_\sigma \mid \exists U \subseteq x ((U_2,b) \in f_2) \}
= |f_1| x \cap |f_2| x.
\]

The part “\( \subseteq \)” of the middle equality is obvious. For “\( \supseteq \)”, let \( U_i \subseteq x \) with \( (U_i,b) \in f_i \) be given. Choose \( U = U_1 \cup U_2 \). Then clearly \( (U,b) \in f_i \) (as \( \{(U_i,b)\} \vdash (U,b) \) and \( f_i \) is deductively closed). \( \square \)

**Lemma.** \( f \approx_\rho g \) if and only if \( f \cap g \in G_\rho \), for \( f, g \in G_\rho \).

**Proof.** By induction on \( \rho \). For \( \iota \) use induction on the definitions of \( f \approx_\iota g \) and \( G_\iota \). Case \( \rho \rightarrow \sigma \).

\[
f \approx_{\rho \rightarrow \sigma} g \iff \forall x \in G_\rho (fx \approx_\sigma gx)
\iff \forall x \in G_\rho (fx \cap gx \in G_\sigma) \quad \text{by induction hypothesis}
\]
∀x∈Gρ((f\cap g)x∈Gσ) \text{ by the last lemma}

f\cap g \in G_{ρ→σ}.

\square

Theorem. \(x\approx_ρ y\) implies \(fx\approx_σ fy\), for \(x, y \in G_ρ\) and \(f \in G_ρ→G_σ\).

Proof. Since \(x\approx_ρ y\) we have \(x \cap y \in G_ρ\) by the previous lemma. Now \(fx, fy \supseteq f(x \cap y)\) and hence \(fx \cap fy \in G_σ\). But this implies \(fx\approx_σ fy\) again by the previous lemma. \(\square\)

We prove the density theorem, which says that every finitely generated functional (i.e., every \(U\) with \(U \in \text{Con}_ρ\)) can be extended to a total one. A type \(ρ\) is called dense if

\[∀_{U∈\text{Con}_ρ}∃_{x∈G_ρ}(U \subseteq x)\]

(i.e., \(G_ρ \subseteq |C_ρ|\) is dense w.r.t. the Scott topology), and separating if

\[∀_{U,V∈\text{Con}_ρ}(U \not\subseteq V \rightarrow \exists z \in G \land U z \not\subseteq V z)\]

We prove that every type \(ρ\) is both dense and separating. Define the height \(|a^∗|\) of an extended token \(a^∗\), and \(|U|\) of a formal neighborhood \(U\), by

\[|Ca^1 \ldots a^n| := \max\{ |a^i| | i = 1, \ldots, n \} + 1, \quad |∗| := 0,\]

\[|(U, b)| := \max\{ |U_i| | b \} + 1,\]

\[|\{ a_i \mid i ∈ I \}| := \max\{ |a_i| + 1 | i ∈ I \}.\]

Remark. Let \(U ∈ \text{Con}\) be non-empty. Then every token in \(U\) starts with the same constructor \(C\). Let \(U_i\) consist of all tokens at the \(i\)-th argument position of some token in \(U\). Then \(CU^U \vdash U\) (and also \(U \vdash CU^U\)), and \(|U_i| < |U|\) (where \(CU^U := \{ Ca^i \mid a^i ∈ U_i \text{ if } U_i \neq \emptyset, \text{ and } a^i = * \text{ otherwise} \}\)).

We write \(G_ρ a\) to mean that \(a\) is a total token (i.e., a constructor tree without \(*\)), and \(G_ρ U\) to mean that \(U\) contains a total token. For \(W = \{ (U_i, a_i) | i < n \}\) we have \(Wx := \{ a_i | U_i \subseteq x \}\). Hence if \(x\) is decidable, then so is \(Wx\).

Theorem (Density). For every type \(ρ = ρ_1 → \ldots → ρ_p → \iota\) we have decidable formulas \(\text{TExt}_ρ\) and \(\text{Sep}_ρ^i (i = 1, \ldots, p)\) such that

(a) \(∀_{U∈\text{Con}_ρ}(U \subseteq \{ a \mid \text{TExt}_ρ(U, a) \} \in G_ρ)\) and

(b) \(∀_{U,V∈\text{Con}_ρ}(U \not\subseteq V \rightarrow \exists_{U,V} \in G \land U z_{U,V} \not\subseteq V z_{U,V}), \text{ where } z_{U,V} = z_{U,V,1}, \ldots, z_{U,V,p} \text{ and } z_{U,V,i} = \{ a \mid \text{Sep}_ρ^i(U, V, a) \}\).

Proof. By induction on \(ρ\).

Case \(\iota\). (a). Given \(U \in \text{Con}_ρ\) we define a token \(a_U\) by induction on the height \(|U|\) such that \(\{ a_U \} \vdash U\) and \(G_ρ a_U\). For \(U = \emptyset\) let \(a_U^U\) be the nullary constructor of \(\iota\). If \(U \neq \emptyset\), define \(U_i\) from \(U\) as in the remark above; then \(CU^U \vdash U\) and \(|U_i| < |U|\). Hence for \(a_U := Ca_{U_1} \ldots a_{U_n}\) we have \(G_ρ a_U\) by
induction hypothesis, and \( \{a_U\} \vdash C \overline{U} \vdash U \) by the definition of entailment. So we can put \( \mathsf{TExt}_i(U, a) := \{a_U\} \vdash a \).

Case \( i \), (b). There is nothing to show.

Case \( \rho \rightarrow \sigma \), (a). Fix \( W = \{ (U_i, a_i) \mid i \in \mathbb{N}^i \} \subseteq \mathsf{Con}_{\mathbb{N}^i} \). Consider \( i < j < n \) with \( a_i \not\vdash a_j \), thus \( U_i \not\vdash U_j \). By induction hypothesis (b) for \( \rho \) we have \( \widetilde{z}_{ij} \in G \) such that \( U_i \overline{z}_{ij} \not\vdash, U_j \overline{z}_{ij} \). Define for every \( U \in \mathsf{Con}_\rho \) a set \( I_U \) of indices \( k < n \) such that “\( U \) behaves as \( U_k \) with respect to the \( \widetilde{z}_{ij} \)’s. Then

\[
I_U := \{ k < n \mid \forall_i < k (a_i \not\vdash a_k \rightarrow U \overline{z}_{ik} \vdash U_k \overline{z}_{ik}) \land \\
\forall_j > k (a_k \not\vdash a_j \rightarrow U \overline{z}_{kj} \vdash U_k \overline{z}_{kj}) \}.
\]

Notice that \( k \in I_{U_k} \). We first show

\[ V_U := \{ a_k \mid k \in I_U \} \subseteq \mathsf{Con}_\sigma. \]

It suffices to prove \( a_i \uparrow a_j \) for \( i, j \in I_U \) with \( i < j \). Since \( a_i \uparrow a_j \) is decidable we can argue indirectly. Assume \( a_i \not\vdash a_j \). Then \( U \overline{z}_{ij} \vdash U_i \overline{z}_{ij} \) and \( U \overline{z}_{ij} \vdash U_j \overline{z}_{ij} \), thus \( U_i \overline{z}_{ij} \vdash U_j \overline{z}_{ij} \). But \( U_i \overline{z}_{ij} \vdash U_j \overline{z}_{ij} \) by the choice of the \( \widetilde{z}_{ij} \) for \( U_i \not\vdash U_j \).

By induction hypothesis (a) \( V_U \subseteq y_{V_U} \vdash \{ a \mid \mathsf{TExt}_\sigma(V_U, a) \} \in \mathsf{G}_\sigma \). Let

\[
(3) \quad r := \{ (U, a) \mid (a \in y_{V_U} \land \forall_i < n (a_i \not\vdash a_j \rightarrow G_i(U \overline{z}_{ij}))) \lor V_U \vdash a \},
\]

We claim \( W \subseteq r \in \mathsf{G}_{\mathbb{N}^i} \); then we can define \( \mathsf{TExt}_{\mathbb{N}^i}(W, (U, a)) \) to be the defining formula of \( r \). Since \( k \in I_{U_k} \) we have \( a_k \in V_{U_k} \), thus \( (U_k, a_k) \in r \).

For \( r \in \mathsf{C}_{\mathbb{N}^i} \) we verify the properties of approximable maps.

First we show that \( (U, a) \in r \) and \( (U, b) \in r \) imply \( a \uparrow b \). But from the premises we obtain \( a, b \in y_{V_U} \) and hence \( a \uparrow b \).

Next we show that \( (U, b_1), \ldots, (U, b_n) \in r \) and \( \{ b_1, \ldots, b_n \} \vdash \) imply \( (U, b) \in r \). We argue by cases. If the left hand side of the disjunction in (3) holds for one \( b_k \), then \( \{ b_1, \ldots, b_n \} \subseteq y_{V_U} \), hence \( b \in y_{V_U} \), and thus \( (U, b) \in r \).

Otherwise \( V_U \vdash \{ b_1, \ldots, b_n \} \vdash b \) and therefore \( (U, b) \in r \) as well.

Finally we show that \( (U, a) \in r \) and \( U' \vdash U \) imply \( U', a) \in r \). We again argue by cases. If the left hand side of the disjunction in (3) holds, we have \( a \in y_{V_U} \), and from \( U' \vdash U \) we obtain \( \forall_i < n (a_i \not\vdash a_j \rightarrow G_i(U \overline{z}_{ij})) \). We show \( a \in y_{V_{U'}} \). But \( U_i \overline{z}_{ij} \) and \( U' \overline{z}_{ij} \) both contain a total token, for every \( i, j \) with \( a_i \not\vdash a_j \), which must be the same since \( U' \vdash U \). Thus \( I_U = I_{U'} \), hence \( V_U = V_{U'} \). Now assume \( V_U \vdash a \). But \( U' \vdash U \) implies \( I_U \subseteq I_{U'} \), hence \( V_U \subseteq V_{U'} \), hence \( V_{U'} \vdash a \) and therefore \( (U', a) \in r \).

It remains to prove \( r \in \mathsf{G}_{\mathbb{N}^i} \). Let \( x \in \mathsf{G}_\rho \). We show that \( rx \in \mathsf{G}_\sigma \), i.e.,

\[
\{ a \in \mathsf{Tok}_\sigma \mid \exists U \subseteq x ((U, a) \in r) \} \in \mathsf{G}_\sigma.
\]

Recall \( \overline{z}_{ij} \in G \) for all \( i < j < n \) with \( a_i \not\vdash a_j \). Hence \( x \overline{z}_{ij} \in G_i \) for all such \( i, j \). Since every total ideal of base type contains a total token we have \( U_{ij} \subseteq x \) with \( G_i(U_{ij} \overline{z}_{ij}) \). Let \( U \) be the union of all \( U_{ij} \)’s. Then \( G_i(U \overline{z}_{ij}) \).
Hence \((U,a) \in r\) for all \(a \in y \nu_U\), i.e., \(y \nu_U \subseteq rx\) and therefore \(rx \in G_\sigma\), by the Extension Lemma.

Case \(\rho \rightarrow \sigma\), (b). Let \(W_1,W_2 \in \text{Con}_{\rho \rightarrow \sigma}\) with \(W_1 \not\subseteq W_2\). Pick \((U_1,a_1) \in W_i\) such that \(U_1 \upharpoonright U_2\) and \(a_1 \not\subseteq a_2\). By induction hypothesis (a) for \(\rho\)

\[ U_1 \cup U_2 \subseteq z_{U_1,U_2} := \{ a \mid \text{TExt}_\rho(U_1 \cup U_2, a) \} \in G_\rho. \]

Then \(a_1 \in W_i z_{U_1,U_2}\). From the induction hypothesis (b) for \(\sigma\) we obtain \(z_{a_1,a_2} \in G\) such that

\[ \{a_1\} z_{a_1,a_2} \not\subseteq \{a_2\} z_{a_1,a_2}, \]

where \(\sigma = \sigma_1 \rightarrow \ldots \rightarrow \sigma_p \rightarrow i\) and \(z_{a_1,a_2,i} := \{ a \mid \text{Sep}_i^\rho(\{a_1\}, \{a_2\}, a) \} \) for \(i = 1, \ldots, p\). Hence \(W_i z_{U_1,U_2} z_{a_1,a_2,i} \not\subseteq W_2 z_{U_1,U_2} z_{a_1,a_2,i}\). Therefore

\[ \text{Sep}_{p \rightarrow i}^\rho(W_1,W_2,a) := \text{TExt}_\rho(U_1 \cup U_2, a), \]

\[ \text{Sep}_{p+1 \rightarrow i}^\rho(W_1,W_2,a) := \text{Sep}_i^\rho(\{a_1\}, \{a_2\}, a). \]

\[ \square \]

2.3. Definability. There will be two kinds of (natural) numbers: (i) total tokens in the algebra \(N\), and (ii) total ideals of type \(N\). Recall that the total tokens in \(N\) are iterated applications of the successor constructor \(S\) to the zero constructor \(0\). We call them index numbers and write \(n \in N\) for the \(n\)-th such token. Then \(\overline{n}\) is a total ideal of type \(N\).

In the statement of the definability theorem below we will need fixed enumerations \((e_n)_{n \in N}\) of all tokens and \((E_n)_{n \in N}\) of all formal neighborhoods, one for each type. We will also need some special computable functionals:

The parallel conditional \(\text{pcond}: B \rightarrow \rho \rightarrow \rho \rightarrow \rho\). It is defined by the clauses

\begin{align*}
(4) & \quad U \vdash \mathfrak{t} \rightarrow V \vdash a \rightarrow (U,V,W,a) \in \text{pcond}, \\
(5) & \quad U \vdash \mathfrak{f} \rightarrow W \vdash a \rightarrow (U,V,W,a) \in \text{pcond}, \\
(6) & \quad V \vdash a \rightarrow W \vdash a \rightarrow (U,V,W,a) \in \text{pcond}.
\end{align*}

We also need the least-fixed-point axiom, which says that any set of tokens \((U,V,W,a)\) satisfying (4)–(6) is a superset of pcond. It is easy to see that pcond is an ideal.

**Lemma** (Properties of pcond).

\begin{align*}
(7) & \quad \mathfrak{t} \in z \rightarrow \text{pcond}(z,x,y) = x, \\
(8) & \quad \mathfrak{f} \in z \rightarrow \text{pcond}(z,x,y) = y, \\
(9) & \quad a \in x \rightarrow a \in y \rightarrow a \in \text{pcond}(z,x,y).
\end{align*}

**Proof.** (7). Assume \(\mathfrak{t} \in z\). “\(\supseteq\)” Let \(a \in x\). We show \(a \in \text{pcond}(z,x,y)\). It suffices to find \(U \subseteq z\), \(V \subseteq x\) and \(W \subseteq y\) such that \((U,V,W,a) \in \text{pcond}\). Since \((\{\mathfrak{t}\},\{a\},\emptyset,a) \in \text{pcond}\) by (4) we can take \(\{\mathfrak{t}\}\) for \(U\), \(\{a\}\) for \(V\) and \(\emptyset\) for \(W\). “\(\subseteq\)” Let \(a \in \text{pcond}(z,x,y)\). We show \(a \in x\). By continuity of
application we have $U \subseteq z$, $V \subseteq x$ and $W \subseteq y$ such that $(U, V, W, a) \in \text{pcond}$. It suffices to show $V \vdash a$. This will follow from the rules for pcond, since (because of $t \in z$ the token $(U, V, W, a)$ must have entered pcond by clause (4) or (6). Formally we make use of the least-fixed-point axiom for pcond, and apply it to $C := \{ (U, V, W, a) \mid \{ u \} \vdash U \rightarrow V \vdash a \}$. We show that $C$ satisfies (4)–(6). For (5) we must show

$$U \vdash \text{ff} \rightarrow W \vdash a \rightarrow \{ u \} \vdash U \rightarrow V \vdash a.$$ 

This follows from ex-falso-quodlibet, since $\{ u \} \vdash U$ and $U \vdash \text{ff}$ implies $\{ u \} \vdash \text{ff}$, a contradiction. (4) and (6) have the desired conclusion $V \vdash a$ among their premises. But now the least-fixed-point axiom for pcond implies $(U, V, W, a) \in C$ (since $t \in z$ and $U \subseteq z$ imply $\{ u \} \vdash U$) and hence $V \vdash a$.

(8) is proved similarly. (9). It suffices to have $V \subseteq x$ and $W \subseteq y$ such that $(\emptyset, V, W, a) \in \text{pcond}$. Use (6) with $\{ a \}$ for $V$ and $W$. \qed

A continuous variant of the union. The continuous variant $\cup_\#$ of the union has type $\rho \rightarrow \mathbb{N} \rightarrow \rho$; its defining clauses are

$$(10) \quad U \uparrow e_n \rightarrow V \vdash n \rightarrow U \vdash a \rightarrow (U, V, a) \in \cup_\#,$$

$$(11) \quad \{ e_n \} \vdash a \rightarrow V \vdash n \rightarrow (U, V, a) \in \cup_\#,$$

and again we require the least-fixed-point axiom for $\cup_\#$. $U \uparrow e_n$ means $\forall a \in U (a \uparrow e_n)$. It is easy to see that $\cup_\#$ is an ideal.

**Lemma (Properties of $\cup_\#$).**

$$(12) \quad \forall a \in x (a \uparrow e_n) \rightarrow x \cup_\# \overline{n} = x \cup \{ e_n \},$$

$$(13) \quad e_n \in x \cup_\# \overline{n}.$$ 

**Proof.** (12). Assume $a \uparrow e_n$ for all $a \in x$.

“$\supseteq$”. Let $a \in x \cup \{ e_n \}$. We show $a \in x \cup_\# \overline{n}$. It suffices to find $U \subseteq x$, $V \subseteq \overline{n}$ such that $(U, V, a) \in \cup_\#$. Let $U := \{ a \}$ in case $a \in x$, and $U := \emptyset$ in case $\{ e_n \} \vdash a$. Then $(U, \{ a \}, a) \in \cup_\#$ by (10) or (11), respectively.

“$\subseteq$”. Let $a \in x \cup_\# \overline{n}$. We show $a \in x \cup \{ e_n \}$. By continuity of application we have $U \subseteq x$ and $V \subseteq \overline{n}$ such that $(U, V, a) \in \cup_\#$. Let

$$C := \{ (U, V, a) \mid U \vdash a \lor \exists k \in \mathbb{N} (\{ e_k \} \vdash a \land V \vdash k) \}.$$

$C$ satisfies (10) and (11). Hence by the least-fixed-point axiom for $\cup_\#$ we have $(U, V, a) \in C$. If $U \vdash a$ the claim is immediate, since $U \subseteq x$. Otherwise we have $k \in \mathbb{N}$ such that $\{ e_k \} \vdash a$ and $V \vdash k$. But $V \subseteq \overline{n}$ implies $k = n$. Hence $\{ e_n \} \vdash a$ and therefore $a \in \{ e_n \}$.

(13). Assume $n \in \mathbb{N}$. It suffices to have $U \subseteq x$ and $V \subseteq \overline{n}$ such that $(U, V, e_n) \in \cup_\#$. Use (11) with $e_n$ for $a$, $\emptyset$ for $U$ and $\{ n \}$ for $V$. \qed
A continuous variant of consistency. We define $\uparrow_\#$ of type $\rho \to N \to B$ by the clauses
\begin{align}
(14) & \quad U \vdash E_n \to V \vdash n \to (U, V, \mathfrak{t}) \in \uparrow_\#, \\
(15) & \quad a \in U \to b \in E_n \to V \vdash n \to a \notin b \to (U, V, \mathfrak{f}) \in \uparrow_\#.
\end{align}
Again we require the least-fixed-point axiom; it is easy to see that $\uparrow_\#$ is an ideal.

Lemma (Properties of $\uparrow_\#$).
\begin{align}
(16) & \quad \mathfrak{t} \in x \uparrow_\# \overline{\pi} \leftrightarrow x \supseteq E_n, \\
(17) & \quad \mathfrak{f} \in x \uparrow_\# \overline{\pi} \leftrightarrow \exists a \in x, b \in E_n (a \notin b).
\end{align}

Proof. (16). Let $n \in N$. “$\rightarrow$”. Assume $\mathfrak{t} \in x \uparrow_\# \overline{\pi}$. We show $x \supseteq E_n$. By continuity of application we have $U \subseteq x$ and $V \subseteq \overline{\pi}$ such that $(U, V, \mathfrak{t}) \in \uparrow_\#$. Let $C$ be the predicate consisting of all $(U, V, c)$ such that
\begin{align*}
(c = \mathfrak{t} & \rightarrow \exists k \in N (U \vdash E_k \land V \vdash k)) \\
(c = \mathfrak{f} & \rightarrow \exists a \in U, k \in N, b \in E_n (V \vdash k \land a \notin b)).
\end{align*}
$C$ satisfies (14) and (15). Hence by the least-fixed-point axiom for $\uparrow_\#$ we have $(U, V, \mathfrak{t}) \in C$, i.e., $k \in N$ such that $U \vdash E_k$ and $V \vdash k$. Using $V \subseteq \overline{\pi}$ we obtain $k = n$. Now $U \subseteq x$ implies $x \supseteq E_n$.

“$\leftarrow$”. Assume $x \supseteq E_n$. We show $\mathfrak{t} \in x \uparrow_\# \overline{\pi}$. It suffices to find $U \subseteq x$ and $V \subseteq \overline{\pi}$ such that $(U, V, \mathfrak{t}) \in \uparrow_\#$. Take $E_n$ for $U$ and $\{n\}$ for $V$. Then $(U, V, \mathfrak{t}) \in \uparrow_\#$ by (14).

(17) is proved similarly. For “$\rightarrow$” we can use the same $C$, and for “$\leftarrow$” use (15) instead of (14). \hfill $\square$

Let $\iota$ have at most unary constructors, i.e., be one of $N$, $B$ or $P$. A partial continuous functional $\Phi$ of type $\rho_1 \to \cdots \to \rho_p \to \iota$ is recursive in $p\text{cond}$, $\cup_\#$ and $\uparrow_\#$ if it can be defined explicitly by a term involving the constructors for $\iota$ and $N$, the constants predecessor, the fixed point operators $Y_\rho$, the parallel conditional $p\text{cond}$ and the continuous variants of union and of consistency.

Theorem (Definability). A partial continuous functional is computable if and only if it is recursive in $p\text{cond}$, $\cup_\#$ and $\uparrow_\#$.

Proof. The fact that the constants are defined by the rules above implies that the ideals they denote are recursively enumerable. Hence every functional recursive in $p\text{cond}$, $\cup_\#$ and $\uparrow_\#$ is computable. For the converse let $\Phi$ be computable of type $\rho_1 \to \cdots \to \rho_p \to \iota$. Then $\Phi$ is a primitive recursively enumerated set of tokens $(E_{f_n}, \ldots, E_{f_p}, e_{gn})$ where $f_1, \ldots, f_p$ and $g$ are fixed primitive recursive functions on index numbers. Let $\overline{f}$ denote a continuous extension of $f$ to ideals, such that $\overline{f_n} = \overline{f\pi}$. Such an $\overline{f}$
obtained by reading \( f \)'s primitive recursion equations as computation rules in the sense of 2.1.

Let \( \vec{\varphi} = \varphi_1, \ldots, \varphi_p \) be arbitrary continuous functionals of types \( \rho_1, \ldots, \rho_p \), respectively. We show that \( \Phi \) is definable by the equation \( \Phi \vec{\varphi} = Yw_{\vec{\varphi}}0 \) with \( w_{\vec{\varphi}} \) of type \( (N \to \iota) \to N \to \iota \) given by

\[
w_{\vec{\varphi}}x := \text{pcond}(\varphi_1 \uparrow \# f_1 x \land \cdots \land \varphi_p \uparrow \# f_p x, \psi(x + 1) \cup \# g x, \psi(x + 1)).
\]

Here \( \land \) is the parallel and of type \( \mathcal{B} \to \mathcal{B} \to \mathcal{B} \), defined by \( \land(p, q) := \text{pcond}(p, q, \{ \text{ff} \}) \). To simplify notation we assume \( p = 1 \) in the argument to follow, and write \( w \) for \( w_{\vec{\varphi}} \). For later reference we split the rest of the argument into steps.

**Step 1.** We first prove that

\[
\forall n(\exists k \in w_{\omega_{k+1}}0 n \to \exists \leq l \leq n+k(\varphi \supseteq E_{fl} \land \{ e_{gl} \} \vdash a)).
\]

The proof is by induction on \( k \). For the base case assume \( a \in w_{\omega_{n+1}}0 \), i.e.,

\[
a \in \text{pcond}(\varphi \uparrow \# f n, \emptyset \cup \# g n, \emptyset).
\]

Then clearly \( \varphi \supseteq E_{fn} \) and \( \{ e_{gn} \} \vdash a \).

**Step 2.** For the step \( k \to k + 1 \) we have

\[
a \in w_{\omega_{k+1}}0 n = w(w_{\omega_{k+1}}0 n) = \text{pcond}(\varphi \uparrow \# f n, v \cup \# g n, v),
\]

with \( v := w_{\omega_{k+1}}0 (n+1) \). Then either \( a \in v \) (and we are done by the induction hypothesis) or else \( \varphi \supseteq E_{fn} \) and \( \{ e_{gn} \} \vdash a \).

**Step 3.** Now \( \Phi \varphi \supseteq Yw_{\vec{\varphi}}0 \) follows easily. Assume \( a \in Yw_{\vec{\varphi}}0 \). Then \( a \in w_{\omega_{k+1}}0 n \) for some \( k \), by (2). Therefore there is an \( l \) with \( 0 \leq l \leq k \) such that \( \varphi \supseteq E_{fl} \) and \( \{ e_{gl} \} \vdash a \). But this implies \( a \in \Phi \varphi \).

**Step 4.** For the converse assume \( a \in \Phi \varphi \). Then for some \( U \subseteq \varphi \) we have \( (U, a) \in \Phi \). By our assumption on \( \Phi \) this means that we have an \( n \) such that \( U = Ef_n \) and \( a = e_{gn} \). We show

\[
a \in w_{\omega_{k+1}}0 (n-k) \text{ for } k \leq n.
\]

The proof is by induction on \( k \). For \( k = 0 \) because of \( \varphi \supseteq E_{fn} \) we have \( \text{tt} \in \varphi \uparrow \# f n \) and hence \( w_{\psi n} = \psi(n + 1) \cup \# g n \supseteq e_{gn} = a \), for any \( \psi \).
Step 5. For the step $k \mapsto k + 1$ by definition of $w (:= w^{\varphi})$
\[ v' := w^{k + 2}(\emptyset(n - k - 1)) \]
\[ = w(w^{k + 1}(\emptyset)) (n - k - 1) \]
\[ = \text{pcond}(\varphi \uparrow \# f(n - k - 1), v \cup \# g(n - k - 1), v) \]
with $v := w^{k + 1}(\emptyset(n - k))$. By induction hypothesis $a \in v$; we show $a \in v'$.
If $a$ and $e_{g(n - k - 1)}$ are inconsistent, $a \in \Phi \varphi$ and $(E_{f(n - k - 1)}, e_{g(n - k - 1)}) \notin \Phi$
imply that $\varphi \cup E_{f(n - k - 1)}$ is inconsistent, hence $\varphi \uparrow f(n - k - 1)$ and therefore $v' = v$.

Step 6. If $a$ and $e_{g(n - k - 1)}$ are consistent, $a$ and $e_{g(n - k - 1)}$ are comparable,
since our underlying algebra $\iota$ has at most unary constructors.

Step 7. In case $\{e_{g(n - k - 1)}\} \vdash a$ we have $v \cup \# g(n - k - 1) \supseteq \{e_{g(n - k - 1)}\} \vdash a$,
and hence $a \in v'$ because of $a \in v$.

Step 8. In case $\{a\} \vdash e_{g(n - k - 1)}$ we have $e_{g(n - k - 1)} \in v$ because of $a \in v$,
hence $v \cup \# g(n - k - 1) = v$ and therefore again $a \in v'$.

Step 9. Now the converse inclusion $\Phi \varphi \subseteq Yw^{\varphi}\overline{0}$ can be seen easily. Since $a \in \Phi \varphi$, the claim just proved for $k := n$ gives $a \in w^{n + 1}(0\overline{0})$, and this implies $a \in Yw^{\varphi}\overline{0}$.
\[ \square \]

3. The Theory $\text{TCF}^+$

We sketch a formal system $\text{TCF}^+$ intended to talk about computable functionals plus their finite approximations, i.e., tokens and formal neighborhoods. Since continuous functionals (i.e., ideals) are possibly infinite sets of tokens, $\text{TCF}^+$ contains for every type $\rho$ set variables $x^{\rho}$. The only existence axiom for sets will be $\Sigma$-comprehension.

3.1. Types and token types. Recall that (object) types are built from base types $\iota$ (the algebras above) by $\rho \rightarrow \sigma$. Now in addition for every (object) type $\rho$ we have token types $\text{Tok}_\rho^\tau$ (extended tokens of type $\rho$), $\text{Tok}_\rho$ (tokens of type $\rho$), $\text{LTok}_\rho$ (lists of tokens of type $\rho$), $\text{LTok}_\rho^\tau$ (lists of extended tokens of type $\rho$); let $\tau$ range over token types. The index $\rho$ will be omitted if it is inessential or clear from the context.

We inductively define the extended tokens of an algebra $\iota$. As a generic algebra we take the algebra $\text{D}$ (of derivations), given by the constructors $0^\text{D}$ (axiom) and $C^{\text{D} \rightarrow \text{D} \rightarrow \text{D}}$ (rule); for other algebras the definitions are similar. The clauses are
\[
\text{Tok}_\text{D}(\ast), \quad \text{Tok}_\text{D}(0^\text{D}), \quad \text{Tok}_\text{D}(a_1^\ast) \rightarrow \text{Tok}_\text{D}(a_2^\ast) \rightarrow \text{Tok}_\text{D}(C^{\text{D} \rightarrow \text{D} \rightarrow \text{D} a_1^\ast a_2^\ast}).
\]
(Proper) tokens are defined similarly:

\[ \text{Tok}_D(\emptyset), \quad \text{Tok}_D^*(a_1) \to \text{Tok}_D^*(a_2) \to \text{Tok}_D(C =_D D a_1 a_2). \]

Clearly every token can be viewed as an extended token.

It will be convenient to represent formal neighborhoods as lists of tokens. The algebra of lists of tokens of type \( D \) is defined by

\[ \text{LTok}_D(\text{nil}_D), \quad \text{Tok}_D(a) \to \text{LTok}_D(U) \to \text{LTok}_D(a : D U). \]

We use \( \text{nil}_D \) to denote the empty list, and \( a : D U \) (or \( \text{cons}_D(a, U) \)) to denote the result of constructing a new list from a given one \( U \) by adding \( a \) in front. Similarly the algebra of lists of extended tokens is defined by

\[ \text{LTok}_D^*(\text{nil}_D), \quad \text{Tok}_D^*(a) \to \text{LTok}_D^*(U) \to \text{LTok}_D^*(a : D U). \]

We allow functions of token-valued types \( \vec{\tau} \to \tau \), defined by primitive recursion. An easy example is \( \dot{\epsilon}_D : \text{Tok}_D^* \to \text{LTok}_D^* \to \text{Tok}_B^* \); it is a boolean-valued function, i.e., with values in \( \text{Tok}_B^* \). The recursion equations are

\[
\begin{align*}
(a^* \dot{\epsilon}_D \text{nil}) & := \text{ff}, \\
(a^* \dot{\epsilon}_D (b^* : D U)) & := (a^* =_D b^*) \lor \text{B} a^* \dot{\epsilon} U,
\end{align*}
\]

where equality \( =_D : \text{Tok}_D^* \to \text{Tok}_D^* \to \text{Tok}_D^* \) is defined by

\[
\begin{align*}
(* =_D *) & := (0 =_D 0) := \text{tt}, \\
(* =_D 0) & := (* =_D Ca^*_1 a^*_2) := \text{ff}, \\
(0 =_D *) & := (0 =_D Ca^*_1 a^*_2) := \text{ff}, \\
(C a^*_1 a^*_2 =_D *) & := (Ca^*_1 a^*_2 =_D 0) := \text{ff}, \\
(C a^*_1 a^*_2 =_D C b^*_1 b^*_2) & := (a^*_1 =_D b^*_1) \land \text{B} (a^*_2 =_D b^*_2),
\end{align*}
\]

and \( \lor \text{B} : \text{Tok}_B^* \to \text{Tok}_B^* \to \text{Tok}_B^* \) is the disjunction function on \( \text{Tok}_B^* \), defined by \( \text{tt} \lor \text{B} b := \text{tt} \) and \( \text{ff} \lor \text{B} b := b \).

From a list of extended tokens of \( D \) we obtain a list of (proper) tokens by removing the *'s. Define \( \text{clean} : \text{LTok}_D \to \text{LTok}_D^* \) by

\[
\begin{align*}
\text{clean}(\text{nil}) & := \text{nil}, \\
\text{clean}(0 :: U) & := 0 :: \text{clean}(U), \\
\text{clean}(*) & := \text{clean}(U), \\
\text{clean}(Ca^*_1 a^*_2 :: U) & := Ca^*_1 a^*_2 :: \text{clean}(U).
\end{align*}
\]

We define \( \text{args}_{C,i} : \text{LTok}_D \to \text{LTok}_D^* \) \( (i = 1, 2) \), which from a list of tokens of \( D \) constructs the list of the \( i \)-th arguments of \( C \)-tokens:

\[
\text{args}_{C,i}(\text{nil}) := \text{nil}, \\
\text{args}_{C,i}(0 :: U) := \text{args}_{C,i}(U), \\
\text{args}_{C,i}(Ca^*_1 a^*_2 :: U) := a^*_i :: \text{args}_{C,i}(U).
\]
Now we can define entailment $\vdash: \text{LTok}_D \rightarrow \text{Tok}_D^* \rightarrow \text{Tok}_B$:

\[
\begin{align*}
U \vdash \ast & := \mathfrak{t}, \\
\text{nil} \vdash 0 & := \mathfrak{f}, \\
\text{nil} \vdash \text{Ca}_1^* \text{a}_2^* & := \mathfrak{f}, \\
0 \vdash 0 & := \mathfrak{t}, \\
C_a^* \text{a}_1^* & := \mathfrak{f}, \\
0 \vdash 0 & := \mathfrak{t},
\end{align*}
\]

and

\[
\begin{align*}
\text{Ca}_1^* \text{a}_2^* & \vdash \text{Cb}_1^* \text{b}_2^* := \text{clean}(\text{a}_1^* \vdash \text{args}_{C,1}(U)) \vdash \text{b}_1^* \land_B \\
& \quad \text{clean}(\text{a}_2^* \vdash \text{args}_{C,2}(U)) \vdash \text{b}_2^*,
\end{align*}
\]

where $\land_B: \text{Tok}_B \rightarrow \text{Tok}_B \rightarrow \text{Tok}_B$ is the conjunction function on $\text{Tok}_B$, defined by $\mathfrak{f} \land_B b := \mathfrak{f}$ and $\mathfrak{t} \land_B b := b$.

To define consistency for lists of tokens we need an auxiliary function checking the outermost constructor only. Let $\text{PreCon}: \text{LTok}_D \rightarrow \text{Tok}_B$ be defined by

\[
\begin{align*}
\text{PreCon}(\text{nil}) & := \text{PreCon}(a :: \text{nil}) := \mathfrak{t}, \\
\text{PreCon}(0 :: \text{Ca}_1^* \text{a}_2^* :: U) & := \text{PreCon}(\text{Ca}_1^* \text{a}_2^* :: 0 :: U) := \mathfrak{f}, \\
\text{PreCon}(0 :: 0 :: U) & := \text{PreCon}(0 :: U), \\
\text{PreCon}(\text{Ca}_1^* \text{a}_2^* :: \text{Cb}_1^* \text{b}_2^* :: U) & := \text{PreCon}(\text{Cb}_1^* \text{b}_2^* :: U).
\end{align*}
\]

Using $\text{PreCon}$ we can define consistency $\text{Con}: \text{LTok}_D \rightarrow \text{Tok}_B$ by

\[
\begin{align*}
\text{Con}(\text{nil}) & := \text{Con}(a :: \text{nil}) := \mathfrak{t}, \\
\text{Con}(0 :: \text{Ca}_1^* \text{a}_2^* :: U) & := \text{Con}(\text{Ca}_1^* \text{a}_2^* :: 0 :: U) := \mathfrak{f}, \\
\text{Con}(0 :: 0 :: U) & := \text{Con}(0 :: U),
\end{align*}
\]

and

\[
\begin{align*}
\text{Con}(\text{Ca}_1^* \text{a}_2^* :: \text{Cb}_1^* \text{b}_2^* :: U) & := \text{PreCon}(\text{Cb}_1^* \text{b}_2^* :: U) \land_B \\
& \quad \text{Con}(\text{clean}(\text{a}_1^* :: \text{b}_1^* \vdash \text{args}_{C,1}(U))) \land_B \\
& \quad \text{Con}(\text{clean}(\text{a}_2^* :: \text{b}_2^* \vdash \text{args}_{C,2}(U))).
\end{align*}
\]

We write $a^* \uparrow_{\rho} b^*$ for $\text{Con}(a^* ::_{\rho} b^* ::_{\rho} \text{nil})$.

We define $G_D: \text{Tok}_D^* \rightarrow \text{Tok}_B$ expressing totality for extended tokens:

\[
\begin{align*}
G_D(\ast) & := \mathfrak{f}, \\
G_D(0) & := \mathfrak{t}, \\
G_D(\text{Ca}_1^* \text{a}_2^*) & := G_D \text{a}_1^* \land_B G_D \text{a}_2^*,
\end{align*}
\]

and also $G_{\text{LTok}_D}: \text{LTok}_D \rightarrow \text{Tok}_B$ doing the same for lists of tokens

\[
\begin{align*}
G_{\text{LTok}_D}(\text{nil}) & := \mathfrak{f}, \\
G_{\text{LTok}_D}(a :: U) & := G_D a \lor_B G_{\text{LTok}_D} U.
\end{align*}
\]

Recall that total tokens of $\textbf{N}$ are iterated applications of the successor constructor $S$ to the zero constructor $0$. They are called “index numbers”,

and written \( n \in \mathbb{N} \). Since primitive recursion is available to define token-valued functions, we can construct standard auxiliary functions, like sequence coding. Thus every (index) number \( n \) can be written uniquely as \( n = (a_0, a_1, \ldots, a_{k-1}) \), and \( k = \text{lh}(n) \), \( a_i = (n)_i \) for \( i < k \).

Tokens of a function type \( \rho \to \sigma \) are pairs \((U, a)\) of lists of tokens of type \( \rho \) and tokens of type \( \sigma \). Both projections are given by functions \( \pi_1 \), \( \pi_2 \).

3.2. Enumerations. We assume fixed enumerations \((e_n)_{n \in \mathbb{N}}\) of tokens and \((E_n)_{n \in \mathbb{N}}\) of lists of tokens, for each type. It seems easiest to define them explicitly. Fix for every constructor \( C \) of an algebra a unique “symbol number” \( SN(C) \). We also have a symbol number \( SN(Nhd) \) indicating the code of a formal neighborhood. We define a Gödel numbering \( \llbracket \cdot \rrbracket : \text{Tok}^*_{\mathcal{D}} \to \mathbb{N} \) by

\[
\llbracket \ast \rrbracket := 0, \\
\llbracket 0 \rrbracket := \langle \text{SN}(0) \rangle, \\
\llbracket C a_1^* a_2^* \rrbracket := \langle SN(C), \llbracket a_1^* \rrbracket, \llbracket a_2^* \rrbracket \rangle.
\]

Formal neighborhoods are gődelized by \( \llbracket \cdot \rrbracket : \text{LTok}_\rho \to \mathbb{N} \),

\[
\llbracket a_0 :: a_1 :: \ldots :: a_{k-1} :: \text{nil} \rrbracket := \langle \text{SN}(\text{Nhd}), \llbracket \rho \rrbracket, \llbracket a_0 \rrbracket, \ldots, \llbracket a_{k-1} \rrbracket \rangle, \\
\llbracket \cdot \rrbracket := \langle \text{SN}(\cdot) \rangle, \\
\llbracket \rho \to \sigma \rrbracket := \langle \text{SN}(\to), \llbracket \rho \rrbracket, \llbracket \sigma \rrbracket \rangle.
\]

It is clear that we can primitive recursively define the converse, mapping the Gödel number \( \llbracket a^* \rrbracket \) of an extended token back to \( a^* \), i.e., \( e \llbracket a^* \rrbracket = a^* \), and similarly for \( \text{LTok}_\rho \).

3.3. Terms and formulas. We have variables \( a^* \) for \( \text{Tok}_\rho \) (extended tokens of type \( \rho \)), \( a \) for \( \text{Tok}_\rho \) (tokens of type \( \rho \)) and \( U \) for \( \text{LTok}_\rho \) (lists of tokens of type \( \rho \)). From these, the symbols for token-valued functions and constants for the constructors for tokens, extended tokens and lists of these we can build terms of token types. We identify terms of token type if they have the same normal form w.r.t. the defining primitive recursion equations for the token-valued functions involved.

Decidable (or \( \Delta \)-) prime formulas are of the form \( \text{atom}(p) \), with \( p \) a term of token type \( \text{Tok}_\mathcal{B} \). They are decidable in the sense that for each such term \( p \) we can prove \( p = \mathsf{tt} \lor p = \mathsf{ff} \); in fact, every closed term of type \( \text{Tok}_\mathcal{B} \) can be evaluated to either \( \mathsf{tt} \) and \( \mathsf{ff} \). Examples are \( a \uparrow_\rho b, a \in_\rho U, U \vdash_\rho a \) (which are shorthand for \( \text{atom}(a \uparrow_\rho b), \text{atom}(a \in_\rho U), \text{atom}(U \vdash_\rho a) \)). \( \Delta \)-formulas are built from decidable prime formulas by \( \to, \land, \lor \) and bounded quantifiers, i.e., \( \forall a \in U, \exists a \in U \), with \( a \) a variable for tokens and \( U \) a term for a list of tokens.

In \( \text{TCF}^+ \) we also allow variables and constants of (object) type \( \rho \), intended to denote sets of tokens. The constants are \( [\lambda x M] \) (with \( M \) a term as in
2.1) of type $\vec{\rho} \rightarrow \sigma$, and also $\text{pcond}, \cup \#,$ of types $\text{B} \rightarrow \rho \rightarrow \rho, \rho \rightarrow \text{N} \rightarrow \rho$ and $\rho \rightarrow \text{N} \rightarrow \text{B}$, respectively.

Prime $\Sigma$-formulas are either decidable prime formulas or else of the form $r \in_{\rho} x$, with $r$ a term of token type $\text{Tok}_{\rho}$ and $x$ a variable or constant of type $\rho$. $\Sigma$-formulas are built as follows.

(a) Every prime $\Sigma$-formula is a $\Sigma$-formula.
(b) $A_{0} \rightarrow B$ is a $\Sigma$-formula if $A_{0}$ is a $\Delta$-formula and $B$ a $\Sigma$-formula.
(c) $\Sigma$-formulas are closed under $\land, \lor$, bounded quantifiers and existential quantifiers over variables of a token type.

Prime formulas are either prime $\Sigma$-formulas or else of the form $G_{\rho} x$ (expressing totality of $x$) or $x \approx_{\rho} y$ (expressing equivalence of $x$ and $y$); $x, y$ are variables or constants of type $\rho$. Formulas are built from prime formulas by $\rightarrow, \land, \lor, \forall, \exists$, where the quantifiers are w.r.t all kinds of variables.

3.4. Axioms. TCF$^{+}$ is based on intuitionistic logic. In fact, minimal logic suffices, since falsity can be defined as $\text{atom}(\text{ff})$. Then $\text{atom}(\text{ff}) \rightarrow A$ ("ex-falso-quodlibet") can be proved provided one has it as an axiom for every prime formula (it can be proved for decidable prime formulas).

Therefore the axioms of TCF$^{+}$ are ex-falso-quodlibet for non-decidable prime formulas $A$, plus the usual ones of Heyting arithmetic, adapted to token types. In particular we have the ordinary induction schemes, for arbitrary formulas of the language. Examples are

\[
A(\text{tt}) \rightarrow A(\text{ff}) \rightarrow A(a),
\]
\[
A(*) \rightarrow A(0) \rightarrow \forall a^{*} b^{*} (A(a^{*}) \rightarrow A(b^{*}) \rightarrow A(Ca^{*}b^{*})) \rightarrow A(a^{*}).
\]

Moreover $\text{atom}(\text{tt})$ is an axiom. For object types we have the $\Sigma$-comprehension:

\[
\exists x \forall a (a \in_{\rho} x \leftrightarrow A), \quad \text{for every } \Sigma\text{-formula } A.
\]

A convenient notation for $x$ is $\{ a \mid A \}$. Further axioms are

(a) For every constant $[\lambda x M]$ its defining clauses corresponding to the rules $(V), (A), (C), (D)$ from 2.1, together with their least-fixed-point axioms.
(b) The defining clauses and corresponding least-fixed-point axioms, for $\text{pcond}, \cup \#,$ and $\uparrow \#,$ as listed in 2.3.
(c) The clauses from 2.2 defining the totality predicates $G_{\rho}$ and the equivalence relations $x_{1} \approx_{\rho} x_{2}$, together with their least-fixed-point axioms.

Notice that the latter imply $x_{1} \approx_{\rho} x_{2} \rightarrow Gx_{1} \rightarrow Gx_{2}$.

3.5. First steps in TCF$^{+}$. We use the abbreviations

\[
U \subseteq V \quad \text{for} \quad \forall a \in U (a \in V),
\]
\[
U \vdash V \quad \text{for} \quad \forall a \in V (U \vdash a),
\]
\[
U \sim V \quad \text{for} \quad U \vdash V \land V \vdash U,
\]
Terms of (object) type are built from variables and constants by application $\{ a \mid A \}$.  Then $r \in_\rho t$ for $t$ a term of type $\rho$ and $r$ a term of token type $\text{Tok}_\rho$ is defined by

$$
(r \in_\rho \{ a \mid A(a) \}) := A(r),
$$

$$(r \in_\rho ts) := \exists U \subseteq s ((U, r) \in t) \quad \text{(continuity of application).}
$$

For a term $M$ with free variables among $\bar{x}$ we write

$$
a \in_\sigma [\![ M ]\!] \quad \text{for} \quad \exists U \subseteq x ((U, a) \in \bar{\rho} \rightarrow [\![ \lambda \bar{x} M ]\!]).
$$

We can prove $\Delta$-comprehension for lists of tokens

$$
\exists U \forall a (a \in U \leftrightarrow a \in V \land A), \quad \text{for every} \Delta\text{-formula, A,}
$$

by induction on $V$.  A convenient notation for $U$ is $[a \in V \mid A]$.  We will need the extension $\bar{\mathcal{F}}$ of a monotone token-valued function $f$ to ideals.  It suffices to do this for $f : \text{Tok}_N^* \rightarrow \text{Tok}_N^*$.  Suppose $f$ is monotone, i.e., $\{ a^* \} \vdash b^*$ implies $\{ f a^* \} \vdash f b^*$.  Define $f[\cdot] : \text{LTok}_N^* \rightarrow \text{LTok}_N^*$ by

$$
f[\text{nil}] := \text{nil}, \quad f[a^* :: N U] := (f a^*) :: N f[U].
$$

Then $\bar{\mathcal{F}} : N \rightarrow N$ is defined by

$$
\bar{\mathcal{F}} = \{ (U, a) \mid \text{Con}(U) \land f[U] \vdash a \}.
$$

Clearly $\bar{\mathcal{F}}$ is a decidable ideal.  If $f : \text{Tok}_N \rightarrow \text{Tok}_N$ is defined primitive recursively, then by reading $f$’s primitive recursion equations as computation rules we obtain a defined constant $\bar{\mathcal{F}}$ (in the sense of 2.1) such that $\bar{\mathcal{F}}n = \bar{\mathcal{F}}\bar{\mathcal{F}}$.

Notice that $\forall i \lt_n A$ with $i$ a variable and $n$ a term of token type $\text{Tok}_N$ can be viewed as bounded quantification.  Define $h : \text{Tok}_N^* \rightarrow \text{LTok}_N^*$ by

$$
h(*) := h(0) := \text{nil}, \quad h(S a^*) := h(a^*) \ast (a^* :: \text{nil}),
$$

where $\ast$ appends two lists from $\text{LTok}_N^*$.  Then $h(S^0) = [0, S0, \ldots, S^k \ast -10]$ (i.e., $0 :: S0 :: \ldots S^k \ast -10 :: \text{nil}$), and we can read $\forall i \lt_n A$ as $\forall i \leq h(n) A$.

Every $W$ of token type $\text{LTok}_\rho \rightarrow \sigma$ can be written as $\{ (U_i, a_i) \mid i \lt n \}$.  Here $U_i$, $a_i$ are given as $f(W, i)$, $g(W, i)$ and $n$ as the length $\text{lh}(W)$ of $W$, with $f$, $g$ and $\text{lh}(\cdot)$ defined primitive recursively.  Define

$$
(a \in W x) := \exists i \lt n (U_i \subseteq x \land a = a_i).
$$

Then $a \in W x$ is a $\Delta$-formula if $x$ is given by $\{ a \mid A \}$ with $A$ a $\Delta$-formula.  Therefore by $\Delta$-comprehension for list of tokens we obtain $U$ consisting of
all $a_i$’s such that $a_i \notin W x$. Hence $W x \vdash a$ can be seen as a $\Delta$-formula as well.

4. Formalization

4.1. Density. The informal proof already was written in a form making its formalization in TCF easy. We only discuss the more interesting issues.

The density theorem is parametrized by the type $\rho$, and its proof (by induction on $\rho$) is to be viewed as employing a “meta”-induction.

In the proof that $\rho \to \sigma$ is dense we fixed $W = \{(U_i, a_i) \mid i < n\} \in \text{Con}_\rho\sigma$. Consider $i < j < n$ with $a_i \not\in a_j$, thus $U_i \not\in U_j$. The induction hypothesis (b) for $\rho$ gives $\vec{z}_{ij} \in G$ such that $U_i \vec{z}_{ij} \not\in U_j \vec{z}_{ij}$. The definition of

$$V_U := \{a_k \mid k \in I_U\}$$

can be seen as an application of $\Delta$-comprehension for lists of tokens, since $k \in I_U$ is a $\Delta$-formula. Now the induction hypothesis that $\sigma$ is dense yields $V_U \subseteq y_{V_U} := \{a \mid \text{TExt}_\sigma(V_U, a)\} \in G_\sigma$. The definition (3) of

$$r := \{(U, a) \mid (a \in y_{V_U} \land \forall_{i,j<n}(a_i \not\in a_j \to G_i(U\vec{z}_{ij})) \lor V_U \vdash a\},$$

is by $\Sigma$-comprehension; in fact, the defining formula is a $\Delta$-formula. The rest of the argument can be easily formalized.

The proof that $\rho \to \sigma$ is separating does not present any difficulties. We are given $W_1, W_2 \in \text{Con}_\rho\sigma$ with $W_1 \not\in W_2$, and pick $(U_i, a_i) \in W_i$ such that $U_1 \not\in U_2$ and $a_1 \not\in a_2$. Notice that the $U_i, a_i$ can be defined primitive recursively from $W_1, W_2$, and hence are uniquely determined. By induction hypothesis (a) for $\rho$,

$$U_1 \cup U_2 \subseteq z_{U_1, U_2} := \{a \mid \text{TExt}_\rho(U_1 \cup U_2, a)\} \in G_\rho.$$

Then $a_i \in W_i z_{U_1, U_2}$. From the induction hypothesis (b) for $\sigma$ we obtain $\vec{z}_{a_1, a_2} \in G$ such that (writing $\{a_i\}$ for $\{a_i\}$)

$$\{a_1\} \vec{z}_{a_1, a_2} \not\in \{a_2\} \vec{z}_{a_1, a_2},$$

where $\sigma = \sigma_1 \to \ldots \to \sigma_p \to \iota$ and $\vec{z}_{a_1, a_2, i} := \{a \mid \text{Sep}_i(\{a_1\}, \{a_2\}, a)\}$ for $i = 1, \ldots, p$. Hence $W_1 z_{U_1, U_2} \vec{z}_{a_1, a_2} \not\in W_2 z_{U_1, U_2} \vec{z}_{a_1, a_2}$.  

4.2. Definability. We restrict ourselves to the more interesting direction and assume that $\Phi$ is given as a primitive recursively enumerated set of tokens $(E_{f_n}, e_{gn})$ where $f, g$ are fixed primitive recursive functions. We need to show that $\Phi$ is recursive in peond, $\cup_#$ and $\uparrow_#$, i.e., that it can be defined explicitly by a term involving the constructors for $\iota$ and $N$, the constants predecessor, the fixed point operators $Y_\rho$, the parallel conditional peond and the continuous variants of union and of consistency. In doing so we follow
the steps in the informal proof in 2.3. We show that \( \Phi \) is definable by the equation \( \Phi \varphi = \lambda w, \varphi \bar{\delta} \), with \( w, \varphi \) of type \( (\mathbb{N} \rightarrow \iota) \rightarrow \mathbb{N} \rightarrow \iota \) given by

\[
w, \varphi x := \text{pcond}(\varphi \uparrow \# f x, \varphi(x + 1) \cup \bar{\gamma} x, \psi(x + 1)).
\]

In Step 1 by continuity of application we obtain \( U \subseteq \varphi \uparrow \# \bar{f} n \) and \( V \subseteq \emptyset \cup \bar{\gamma} n \) such that \( (U, V, \emptyset, a) \in \text{pcond} \). For \( \varphi \subseteq E_{fn} \) it suffices by (16) to prove \( \bar{t} \in \varphi \uparrow \# \bar{f} n \), which because of \( U \subseteq \varphi \uparrow \# \bar{f} n \) follows from \( U \vdash \bar{t} \). This will follow from the rules for pcond, because (since \( W = \emptyset \)) the token \( (U, V, \emptyset, a) \) must have entered pcond by rule (4). Formally we make use of the least-fixed-point axiom for pcond, and apply it to \( C := \{(U, V, W, a) \mid W \subseteq \emptyset \rightarrow U \vdash \bar{t}\} \). We show that \( C \) satisfies (4)–(6). For (5) we must show

\[
U \vdash \text{ff} \rightarrow W \vdash a \rightarrow (U, V, W, a) \in C; \quad i.e.,
\]

\[
U \vdash \text{ff} \rightarrow W \vdash a \rightarrow W \subseteq \emptyset \rightarrow U \vdash \bar{t}.
\]

But this follows from ex-falso-quodlibet, since \( W \vdash a \) and \( W \subseteq \emptyset \) are contradictory. (6) is proved similarly, and (4) has the desired conclusion \( U \vdash \bar{t} \) among its premises. But now the least-fixed-point axiom for pcond implies \( (U, V, \emptyset, a) \in C \) and hence \( U \vdash \bar{t} \). For \( \{e_{gn}\} \vdash a \) we argue similarly, with \( C := \{(U, V, W, a) \mid W \subseteq \emptyset \rightarrow V \vdash a\} \), and obtain \( V \vdash a \) and hence \( a \in \emptyset \cup \bar{\gamma} n \). By (12) we conclude that \( \{e_{gn}\} \vdash a \).

The next part of the informal proof was Step 2. Again by continuity of application we obtain \( U \subseteq \varphi \uparrow \# \bar{f} n, V \subseteq v \cup \bar{\gamma} n \) and \( W \subseteq v \) such that \( (U, V, W, a) \in \text{pcond} \). We can prove \( W \vdash a \lor (U \vdash \bar{t} \land V \vdash a) \) as above from the rules for pcond. Hence either \( a \in v \) (and we are done by the induction hypothesis), or else \( \varphi \supseteq E_{fn} \) (which follows as above from \( U \vdash \bar{t} \)) and \( a \in v \cup \bar{\gamma} n \). From the latter by continuity of application we obtain \( V \subseteq v \) and \( W \subseteq \bar{\gamma} n \) such that \( (V, W, a) \in \bar{\gamma} n \). By a least-fixed-point argument (with \( C := \{(V, W, a) \mid \exists m (m \in W \land \{e_{m}\} \vdash a) \lor V \vdash a\} \)) we obtain either \( V \vdash a \) (hence \( a \in v \) and again we are done by the induction hypothesis), or else \( \{e_{m}\} \vdash a \) for an \( m \in G \) such that \( m \in W \), hence \( m = gn \), and therefore \( \{e_{gn}\} \vdash a \). Now the induction used in the informal proof can be applied and we have proved (18) formally.

The informal proof proceeded by Step 3. Since corollary (2) referred to is available in TCF⁺, we have proved the conclusion \( a \in \Phi \varphi \) formally.

Let us now formalize the proof of the reverse direction, i.e., Step 4. In the formalization from \( \varphi \supseteq E_{fn} \) we obtain \( \bar{t} \in \varphi \uparrow \# \bar{f} n \) by (16). We show \( a \in w \psi \bar{n} \) for an arbitrary \( \psi \), i.e., \( a \in \text{pcond}(\varphi \uparrow \# \bar{f} n, \psi(\bar{n} + 1) \cup \bar{\gamma} n, \psi(\bar{n} + 1)) \). Because of \( \bar{t} \in \varphi \uparrow \# \bar{f} n \) and (7) it is enough to show that \( a \in \psi(\bar{n} + 1) \cup \bar{\gamma} n \). But \( e_{gn} \in \psi(\bar{n} + 1) \cup \bar{\gamma} n \) by (13), and we have assumed \( a = e_{gn} \).

Next we consider Step 5. Formally we can infer the existence of \( b \in \varphi \) and \( c \in E_{f(n-k-1)} \) such that \( b \nvdash c \). Hence \( \text{ff} \in \varphi \uparrow \# \bar{f}(n-k-1) \) by (17), and
\( v' = v \) by (8). Step 6 is immediate because of the Comparability Lemma.

For Step 7: Here we can infer \( a \in v \cup_{\mathbb{E}} g(n - k - 1) \) from (13). This and the induction hypothesis \( a \in v \) yields the claim \( a \in v' \) by (9). For Step 8: \( v \cup_{\mathbb{E}} g(n - k - 1) = v \) follows from \( e_{g(n-k-1)} \in v \) by (12). Again this and the induction hypothesis \( a \in v \) yields the claim \( a \in v' \) by (9). For Step 9: The final inference is justified by (2) (applied to \( \{0\}, a \)).

5. Future work

In this paper we attempted to have a first exploratory view on a constructive formal theory of computability TCF\(^+\), where the functionals are studied together with their finite approximations. The attempt was guided by the semantics of non-flat Scott information systems; in particular, it was based on two case studies, namely, the density theorem and the definability theorem. Future work along these lines is to explain TCF\(^+\) in a rigorous and systematic way, as well as test it against further case studies, while an actual implementation on a theorem prover—which should be specially designed to allow for handling functionals and finite approximations alike—remains the ultimate goal of the whole enterprise.

References