TOWARDS A FORMAL THEORY OF COMPUTABILITY

SIMON HUBER, BASIL A. KARÁDAIS AND HELMUT SCHWICHTENBERG

We sketch a constructive formal theory TCF^+ of computable functionals, based on the partial continuous functionals as their intendend 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 $(A, \mathrm{Con}, \vdash)$ where A is a countable set (the *tokens*), Con is a nonempty set of finite subsets of A (the *consistent* sets) and $\vdash$ is a subset of $\mathrm{Con} \times A$ (the *entailment* relation), which satisfy

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

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

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

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

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

For example the *deductive closure* $\overline{U} := \{\, a \mid U \vdash a \,\}$ of U is an ideal. The set of all ideals of $\boldsymbol{A}$ is denoted by $|\boldsymbol{A}|$.

Examples. 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.

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

$$\begin{aligned}
&\mathtt{tt}^{\mathbf{B}}, \mathtt{ff}^{\mathbf{B}} \quad \text{for } \mathbf{B},\\
&0^{\mathbf{N}} \text{ and } \mathrm{S}^{\mathbf{N} \to \mathbf{N}} \text{ (successor) for } \mathbf{N},
\end{aligned}$$

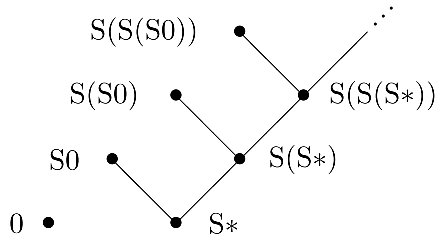

FIGURE 1. Tokens and entailment for **N**

$$1^{\mathbf{P}},\ S_0^{\mathbf{P}\to\mathbf{P}} \text{ (append 0) and } S_1^{\mathbf{P}\to\mathbf{P}} \text{ (append 1) for } \mathbf{P},$$
$$0^{\mathbf{D}} \text{ (axiom) and } \mathrm{C}^{\mathbf{D}\to\mathbf{D}\to\mathbf{D}} \text{ (rule) for } \mathbf{D}.$$

For each of them we define an information system $\boldsymbol{C}_\iota = (\mathrm{Tok}_\iota, \mathrm{Con}_\iota, \vdash_\iota)$:

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

(b) A finite set U of tokens in Tok_ι is *consistent* (i.e., $\in \mathrm{Con}_\iota$) if its elements start with the same n-ary constructor C, say $U = \{\mathrm{C}\vec{a_1^*}, \dots, \mathrm{C}\vec{a_m^*}\}$, and $U_i \in \mathrm{Con}_\iota$ where U_i consists of the (proper) tokens among $a_{1i}^*, \dots, a_{mi}^*$.

(c) $\{\mathrm{C}\vec{a_1^*}, \dots, \mathrm{C}\vec{a_m^*}\} \vdash_\iota \mathrm{C}'\vec{a^*}$ is defined to mean $\mathrm{C} = \mathrm{C}'$, $m \geq 1$ and $U_i \vdash a_i^*$, with U_i as in (b) above (and $U \vdash *$ defined to be true).

For example, the tokens for **N** are shown in Figure 1. For tokens a, b we have $\{a\} \vdash b$ if and only if there is a path from a (up) to b (down). In **D**, the set $\{\mathrm{C}0*, \mathrm{C}*0\}$ is consistent, and $\{\mathrm{C}0*, \mathrm{C}*0\} \vdash \mathrm{C}00$.

A token is called *total* if it has the form $\mathrm{C}\vec{a}$ with a total token a_i at every argument position. For example, the total tokens for **N** are all S^n0, and for **D** all $*$-free constructor trees built from 0 and C.

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

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

1.2. **Function spaces.** Let $\boldsymbol{A} = (A, \mathrm{Con}_A, \vdash_A)$ and $\boldsymbol{B} = (B, \mathrm{Con}_B, \vdash_B)$ be information systems. Define the *function space* $\boldsymbol{A} \to \boldsymbol{B} = (C, \mathrm{Con}, \vdash)$ 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 $WU \vdash_B b$. 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.,

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

Using this one easily proves that $\boldsymbol{A} \to \boldsymbol{B}$ is an information system provided $\boldsymbol{A}$ and $\boldsymbol{B}$ are.

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

(a) With any ideal $r \in |\boldsymbol{A} \to \boldsymbol{B}|$ 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 ideal $\hat{f}\colon \boldsymbol{A} \to \boldsymbol{B}$ by $\hat{f} := \{\, (U,b) \mid b \in f(\overline{U}) \,\}$.

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

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

(a) *If $(U,b_1),\dots,(U,b_n) \in r$, then $\{b_1,\dots,b_n\} \in \mathrm{Con}_B$;*
(b) *If $(U,b_1),\dots,(U,b_n) \in r$ and $\{b_1,\dots,b_n\} \vdash_B b$, 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 ι (the algebras above) by $\rho \to \sigma$. For every type ρ we define the information system $\boldsymbol{C}_\rho = (\mathrm{Tok}_\rho, \mathrm{Con}_\rho, \vdash_\rho)$ starting from the $\boldsymbol{C}_\iota$ by formation of function spaces $\boldsymbol{C}_{\rho\to\sigma} := \boldsymbol{C}_\rho \to \boldsymbol{C}_\sigma$. The set $|\boldsymbol{C}_\rho|$ of ideals in $\boldsymbol{C}_\rho$ is the set of *partial continuous functionals* of type ρ. A partial continuous functional $x \in |\boldsymbol{C}_\rho|$ is *computable* if it is recursively enumerable when viewed as a set of tokens. The information systems $\boldsymbol{C}_\rho$ 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 $\boldsymbol{A} = (A, \mathrm{Con}, \vdash)$ is *coherent* if it satisfies: $U \subseteq A$ is consistent if and only if all of its two-element subsets are.

It is easy to see that every constructor C generates a continuous function $r_{\mathrm{C}} := \{\, (\vec{U}, \mathrm{C}\vec{a^*}\,) \mid \vec{U} \vdash \vec{a^*} \,\}$ in the function space (where $(\vec{U}, b)$ means $(U_1, \dots (U_n, b) \dots))$, and that

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

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. 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

$$|r_{\mathrm{C}}|\emptyset y = \emptyset = |r_{\mathrm{C}}|x\emptyset, \qquad |r_{\mathrm{C}_1}|\emptyset = \emptyset = |r_{\mathrm{C}_2}|\emptyset,$$

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:

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

Every defined constant D comes with a system of *computation rules*, consisting of finitely many equations $D\vec{P}_i(\vec{y}_i) = M_i$ $(i = 1, \dots, n)$ with free variables of $\vec{P}_i(\vec{y}_i)$ and M_i among $\vec{y}_i$, where the $\vec{P}_i(\vec{y}_i)$ must be "constructor patterns", i.e., lists of applicative terms built from constructors and distinct variables, with each constructor C occurring in a context $\mathrm{C}\vec{P}$ (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 $\mathrm{P}\colon \mathbf{N} \to \mathbf{N}$ defined by the computation rules $\mathrm{P}0 = 0$, $\mathrm{P}(\mathrm{S}n) = n$,
(ii) Gödel's primitive recursion operators $\mathcal{R}^\tau_{\mathbf{N}}\colon \mathbf{N} \to \tau \to (\mathbf{N} \to \tau \to \tau) \to \tau$ with computation rules $\mathcal{R}0fg = f$, $\mathcal{R}(\mathrm{S}n)fg = gn(\mathcal{R}nfg)$, and
(iii) the least-fixed-point operators Y_ρ of type $(\rho \to \rho) \to \rho$ defined by the computation rule $Y_\rho f = f(Y_\rho f)$.

For every closed term $\lambda_{\vec{x}} M$ of type $\vec{\rho} \to \sigma$ we inductively define a set $[\![\lambda_{\vec{x}} M]\!]$ of tokens of type $\vec{\rho} \to \sigma$.

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

For every constructor C and defined constant D we have

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

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

Here $(\vec{U}, V) \subseteq [\![\lambda_{\vec{x}} M]\!]$ means $(\vec{U}, b) \in [\![\lambda_{\vec{x}} M]\!]$ for all (finitely many) $b \in V$, and $(\vec{U}, b)$ denotes $(U_1, \dots (U_n, b) \dots)$. 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

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

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

Theorem. (a) *For every term M, $[\![\lambda_{\vec{x}} M]\!]$ 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 $[\![M]\!]_{\vec{x}}^{\vec{u}} := \bigcup_{\vec{U} \subseteq \vec{u}} [\![M]\!]_{\vec{x}}^{\vec{U}}$ with $[\![M]\!]_{\vec{x}}^{\vec{U}} := \{\, b \mid (\vec{U}, b) \in [\![\lambda_{\vec{x}} M]\!] \,\}$. Notice that a consequence of (A) is

(1) $c \in [\![MN]\!]_{\vec{x}}^{\vec{u}} \leftrightarrow \exists_{V \subseteq [\![N]\!]_{\vec{x}}^{\vec{u}}} ((V, c) \in [\![M]\!]_{\vec{x}}^{\vec{u}})$ (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

$$\dfrac{\dfrac{\dfrac{\hat{W} \vdash (V, b)}{(\hat{W}, V, b) \in [\![\lambda_f f]\!]} \qquad \dfrac{(\hat{W}, W_i, b_i) \in [\![\lambda_f Y]\!] \qquad \dfrac{\hat{W} \vdash (V_{ij}, b_{ij})}{(\hat{W}, V_{ij}, b_{ij}) \in [\![\lambda_f f]\!]}}{(\hat{W}, b_i) \in [\![\lambda_f (Yf)]\!]}}{(\hat{W}, b) \in [\![\lambda_f (f(Yf))]\!]}}{(W, b) \in [\![Y]\!]}\ (D), \text{ assuming } W \vdash \hat{W}$$

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

"$\rightarrow$". By induction on the D-height. We have $(\hat{W}, W_i, b_i) \in [\![\lambda_f Y]\!]$, $\hat{W} \vdash W_i$ and $\hat{W} \vdash (V, b)$. By induction hypothesis $W_i^{n_i} \emptyset \vdash b_i$, and $\hat{W}^{n_i} \emptyset \vdash W_i^{n_i} \emptyset$ by monotonicity of application. Because of $\hat{W}^{n+1} \emptyset \vdash \hat{W}^n \emptyset$ (proved by induction on n, using monotonicity) we obtain $\hat{W}^n \emptyset \vdash b_i$ with $n := \max n_i$, i.e., $\hat{W}^n \emptyset \vdash V$. Recall that $\hat{W} \vdash (V, b)$ was defined to mean $\hat{W} V \vdash b$. Hence $\hat{W}(\hat{W}^n \emptyset) \vdash b$ 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 $V := W^n \emptyset =: \{\, b_i \mid i \in I \,\}$. Then $W^n \emptyset \vdash b_i$, hence by induction hypothesis $(W, b_i) \in [\![Y]\!]$. Substituting W for $\hat{W}$ and all W_i in the derivation above gives the claim $(W, b) \in [\![Y]\!]$. □

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

$$b \in [\![Y]\!] w \leftrightarrow \exists_k (b \in w^{k+1} \emptyset). \tag{2}$$

Proof. Since $w^{k+1}\emptyset$ for fixed k is continuous in w, from $b \in w^{k+1}\emptyset$ we can infer $W^{k+1}\emptyset \vdash b$ 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 ρ (notation $x \in G_\rho$) and the equivalence relation $x_1 \approx x_2$ between them are defined inductively.

(a) For an algebra ι, the total ideals x are those of the form $\mathrm{C}\vec{z}$ with C a constructor of ι and $\vec{z}$ total (C denotes the continuous function $|r_{\mathrm{C}}|$). Two total ideals x_1, x_2 are equivalent (written $x_1 \approx_\iota x_2$) if both are of the form $\mathrm{C}\vec{z}_i$ with the same constructor C of ι, and $z_{1j} \approx_\iota z_{2j}$ for all j.
(b) An ideal r of type $\rho \to \sigma$ is total if and only if for all total z of type ρ, the result $|r|z$ of applying r to z is total. For $f, g \in G_{\rho\to\sigma}$ define $f \approx_{\rho\to\sigma} g$ 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 $f \in G_\rho$, $g \in |\boldsymbol{C}_\rho|$ and $f \subseteq g$, then $g \in G_\rho$.*

Proof. By induction on ρ. For base types ι use induction on the definition of $f \in G_\iota$. *Case* $\rho \to \sigma$. Assume $f \in G_{\rho\to\sigma}$ and $f \subseteq g$. We show $g \in G_{\rho\to\sigma}$. 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 $f_1, f_2 \in |\boldsymbol{C}_{\rho\to\sigma}|$ and $x \in |\boldsymbol{C}_\rho|$.*

Proof. By the definition of $|r|$,

$$
\begin{aligned}
&|f_1 \cap f_2|x \\
&= \{\, b \in \mathrm{Tok}_\sigma \mid \exists_{U\subseteq x}((U, b) \in f_1 \cap f_2)\,\} \\
&= \{\, b \in \mathrm{Tok}_\sigma \mid \exists_{U_1\subseteq x}((U_1, b) \in f_1)\,\} \cap \{\, b \in \mathrm{Tok}_\sigma \mid \exists_{U_2\subseteq x}((U_2, b) \in f_2)\,\} \\
&= |f_1|x \cap |f_2|x.
\end{aligned}
$$

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 ρ. For ι use induction on the definitions of $f \approx_\iota g$ and G_ι. *Case* $\rho \to \sigma$.

$$
\begin{aligned}
f \approx_{\rho\to\sigma} g &\leftrightarrow \forall_{x\in G_\rho}(fx \approx_\sigma gx) \\
&\leftrightarrow \forall_{x\in G_\rho}(fx \cap gx \in G_\sigma) \quad \text{by induction hypothesis}
\end{aligned}
$$

$$\begin{aligned} &\leftrightarrow \forall_{x\in G_\rho}((f\cap g)x \in G_\sigma) \quad \text{by the last lemma} \\ &\leftrightarrow f\cap g \in G_{\rho\to\sigma}. \end{aligned}$$ □

Theorem. *$x \approx_\rho y$ implies $fx \approx_\sigma fy$, for $x, y \in G_\rho$ and $f \in G_{\rho\to\sigma}$.*

Proof. Since $x \approx_\rho y$ we have $x\cap y \in G_\rho$ by the previous lemma. Now $fx, fy \supseteq f(x\cap y)$ and hence $fx\cap fy \in G_\sigma$. But this implies $fx \approx_\sigma fy$ again by the previous lemma. □

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

$$\forall_{U\in\mathrm{Con}_\rho}\exists_{x\in G_\rho}(U\subseteq x)$$

(i.e., $G_\rho \subseteq |\boldsymbol{C}_\rho|$ is dense w.r.t. the Scott topology), and *separating* if

$$\forall_{U,V\in\mathrm{Con}_\rho}(U \not\asymp_\rho V \to \vec{z}\in G \wedge U\vec{z} \not\asymp_\iota V\vec{z}).$$

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

$$\begin{aligned} &|\mathrm{C}a_1^*\dots a_n^*| := \max\{\,|a_i^*| \mid i=1,\dots,n\,\}+1, \qquad |\ast| := 0, \\ &|(U,b)| := \max\{|U|,|b|\}+1, \\ &|\{\,a_i \mid i\in I\,\}| := \max\{\,|a_i|+1 \mid i \in I\,\}. \end{aligned}$$

Remark. Let $U \in \mathrm{Con}_\iota$ 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 $\mathrm{C}\vec{U} \vdash U$ (and also $U \vdash \mathrm{C}\vec{U}$), and $|U_i| < |U|$ (where $\mathrm{C}\vec{U} := \{\,\mathrm{C}\vec{a^*} \mid a_i^* \in U_i$ if $U_i \neq \emptyset$, and $a_i^* = \ast$ otherwise$\,\}$).

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

Theorem (Density). *For every type $\rho = \rho_1 \to \dots \to \rho_p \to \iota$ we have decidable formulas TExt_ρ and Sep_ρ^i $(i=1,\dots,p)$ such that*

(a) *$\forall_{U\in\mathrm{Con}_\rho}(U \subseteq \{\,a \mid \mathrm{TExt}_\rho(U,a)\,\} \in G_\rho)$ and*
(b) *$\forall_{U,V\in\mathrm{Con}_\rho}(U \not\asymp_\rho V \to \vec{z}_{U,V} \in G \wedge U\vec{z}_{U,V} \not\asymp_\iota V\vec{z}_{U,V})$, where $\vec{z}_{U,V} = z_{U,V,1},\dots,z_{U,V,p}$ and $z_{U,V,i} = \{\,a \mid \mathrm{Sep}_\rho^i(U,V,a)\,\}$.*

Proof. By induction on ρ.

Case ι, (a). Given $U \in \mathrm{Con}_\iota$ we define a token a_U by induction on the height $|U|$ such that $\{a_U\} \vdash U$ and $G_\iota a_U$. For $U=\emptyset$ let a_U be the nullary constructor of ι. If $U \neq \emptyset$, define U_i from U as in the remark above; then $\mathrm{C}\vec{U} \vdash U$ and $|U_i| < |U|$. Hence for $a_U := Ca_{U_1}\dots a_{U_n}$ we have $G_\iota a_U$ by induction hypothesis, and $\{a_U\} \vdash C\vec{U} \vdash U$ by the definition of entailment. So we can put $\mathrm{TExt}_\iota(U, a) := (\{a_U\} \vdash a)$.

Case ι, (b). There is nothing to show.

Case $\rho \to \sigma$, (a). Fix $W = \{ (U_i, a_i) \mid i < n \} \in \mathrm{Con}_{\rho\to\sigma}$. Consider $i < j < n$ with $a_i \not\uparrow a_j$, thus $U_i \not\uparrow U_j$. By induction hypothesis (b) for ρ we have $\vec{z}_{ij} \in G$ such that $U_i\vec{z}_{ij} \not\uparrow_\iota U_j\vec{z}_{ij}$. Define for every $U \in \mathrm{Con}_\rho$ a set I_U of indices $k < n$ such that "U behaves as U_k with respect to the $\vec{z}_{ij}$":

$$\begin{aligned} I_U := \{ k < n \mid {} & \forall_{i<k}(a_i \not\uparrow a_k \to U\vec{z}_{ik} \vdash_\iota U_k\vec{z}_{ik}) \wedge \\ & \forall_{j>k}(a_k \not\uparrow a_j \to U\vec{z}_{kj} \vdash_\iota U_k\vec{z}_{kj}) \}. \end{aligned}$$

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

$$V_U := \{ a_k \mid k \in I_U \} \in \mathrm{Con}_\sigma.$$

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 $a_i \not\uparrow a_j$. Then $U\vec{z}_{ij} \vdash_\iota U_j\vec{z}_{ij}$ and $U\vec{z}_{ij} \vdash_\iota U_i\vec{z}_{ij}$, thus $U_i\vec{z}_{ij} \uparrow_\iota U_j\vec{z}_{ij}$. But $U_i\vec{z}_{ij} \not\uparrow_\iota U_j\vec{z}_{ij}$ by the choice of the $\vec{z}_{ij}$ for $U_i \not\uparrow U_j$.

By induction hypothesis (a) $V_U \subseteq y_{V_U} := \{ a \mid \mathrm{TExt}_\sigma(V_U, a) \} \in G_\sigma$. Let

$$r := \{ (U, a) \mid (a \in y_{V_U} \wedge \forall_{i,j<n}(a_i \not\uparrow a_j \to G_\iota(U\vec{z}_{ij}))) \vee V_U \vdash a \}, \tag{3}$$

We claim $W \subseteq r \in G_{\rho\to\sigma}$; then we can define $\mathrm{TExt}_{\rho\to\sigma}(W, (U, a))$ 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 $r \in |\boldsymbol{C}_{\rho\to\sigma}|$ 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 $(U, b_1), \dots, (U, b_n) \in r$ and $\{b_1, \dots, b_n\} \vdash b$ 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 $\{b_1, \dots, b_n\} \subseteq y_{V_U}$, hence $b \in y_{V_U}$ and thus $(U, b) \in r$. Otherwise $V_U \vdash \{b_1, \dots, b_n\} \vdash b$ 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 $\forall_{i,j<n}(a_i \not\uparrow a_j \to G_\iota(U'\vec{z}_{ij}))$. We show $a \in y_{V_{U'}}$. But $U\vec{z}_{ij}$ and $U'\vec{z}_{ij}$ both contain a total token, for every i, j with $a_i \not\uparrow a_j$, 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 $r \in G_{\rho\to\sigma}$. Let $x \in G_\rho$. We show that $rx \in G_\sigma$, i.e.,

$$\{ a \in \mathrm{Tok}_\sigma \mid \exists_{U\subseteq x}((U, a) \in r) \} \in G_\sigma.$$

Recall $\vec{z}_{ij} \in G$ for all $i < j < n$ with $a_i \not\uparrow a_j$. Hence $x\vec{z}_{ij} \in G_\iota$ for all such i, j. Since every total ideal of base type contains a total token we have $U_{ij} \subseteq x$ with $G_\iota(U_{ij}\vec{z}_{ij})$. Let U be the union of all U_{ij}'s. Then $G_\iota(U\vec{z}_{ij})$.

Hence $(U, a) \in r$ for all $a \in y_{V_U}$, i.e., $y_{V_U} \subseteq rx$ and therefore $rx \in G_\sigma$, by the Extension Lemma.

Case $\rho \to \sigma$, (b). Let $W_1, W_2 \in \mathrm{Con}_{\rho\to\sigma}$ with $W_1 \not\uparrow W_2$. Pick $(U_i, a_i) \in W_i$ such that $U_1 \uparrow U_2$ and $a_1 \not\uparrow a_2$. By induction hypothesis (a) for ρ

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

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

$$\{a_1\}\vec{z}_{a_1,a_2} \not\uparrow_\iota \{a_2\}\vec{z}_{a_1,a_2},$$

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

$$\begin{aligned}\mathrm{Sep}^1_{\rho\to\sigma}(W_1, W_2, a) &:= \mathrm{TExt}_\rho(U_1 \cup U_2, a),\\ \mathrm{Sep}^{i+1}_{\rho\to\sigma}(W_1, W_2, a) &:= \mathrm{Sep}^i_\sigma(\{a_1\}, \{a_2\}, a).\end{aligned} \qquad \square$$

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

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

The parallel conditional pcond: $\mathbf{B} \to \rho \to \rho \to \rho$. It is defined by the clauses

$$U \vdash \mathtt{tt} \to V \vdash a \to (U, V, W, a) \in \mathrm{pcond}, \tag{4}$$

$$U \vdash \mathtt{ff} \to W \vdash a \to (U, V, W, a) \in \mathrm{pcond}, \tag{5}$$

$$V \vdash a \to W \vdash a \to (U, V, W, a) \in \mathrm{pcond}. \tag{6}$$

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)**.**

$$\mathtt{tt} \in z \to \mathrm{pcond}(z, x, y) = x, \tag{7}$$

$$\mathtt{ff} \in z \to \mathrm{pcond}(z, x, y) = y, \tag{8}$$

$$a \in x \to a \in y \to a \in \mathrm{pcond}(z, x, y). \tag{9}$$

Proof. (7). Assume $\mathtt{tt} \in z$. "$\supseteq$". Let $a \in x$. We show $a \in \mathrm{pcond}(z, x, y)$. It suffices to find $U \subseteq z$, $V \subseteq x$ and $W \subseteq y$ such that $(U, V, W, a) \in \mathrm{pcond}$. Since $(\{\mathtt{tt}\}, \{a\}, \emptyset, a) \in \mathrm{pcond}$ by (4) we can take $\{\mathtt{tt}\}$ for U, $\{a\}$ for V and $\emptyset$ for W. "$\subseteq$". Let $a \in \mathrm{pcond}(z, x, y)$. We show $a \in x$. By continuity of application we have $U \subseteq z$, $V \subseteq x$ and $W \subseteq y$ such that $(U, V, W, a) \in$ pcond. It suffices to show $V \vdash a$. This will follow from the rules for pcond, since (because of $\mathtt{tt} \in z$) 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 $C := \{\, (U, V, W, a) \mid \{\mathtt{tt}\} \vdash U \to V \vdash a \,\}$. We show that C satisfies (4)–(6). For (5) we must show

$$U \vdash \mathtt{ff} \to W \vdash a \to \{\mathtt{tt}\} \vdash U \to V \vdash a.$$

This follows from ex-falso-quodlibet, since $\{\mathtt{tt}\} \vdash U$ and $U \vdash \mathtt{ff}$ implies $\{\mathtt{tt}\} \vdash \mathtt{ff}$, 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 $\mathtt{tt} \in z$ and $U \subseteq z$ imply $\{\mathtt{tt}\} \vdash U$) and hence $V \vdash a$.

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

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

$$U \uparrow e_n \to V \vdash n \to U \vdash a \to (U, V, a) \in \cup_{\#}, \tag{10}$$

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

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

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

$$\forall_{a \in x}(a \uparrow e_n) \to x \cup_{\#} \overline{n} = x \cup \overline{\{e_n\}}, \tag{12}$$

$$e_n \in x \cup_{\#} \overline{n}. \tag{13}$$

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

"$\supseteq$". Let $a \in x \cup \overline{\{e_n\}}$. 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 $(U, \{n\}, a) \in \cup_{\#}$ by (10) or (11), respectively.

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

$$C := \{\, (U, V, a) \mid U \vdash a \vee \exists_{k \in \mathbb{N}}(\{e_k\} \vdash a \wedge V \vdash k) \,\}.$$

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 $a \in \overline{\{e_n\}}$.

(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. □

A continuous variant of consistency. We define $\uparrow_\#$ of type $\rho \to \mathbf{N} \to \mathbf{B}$ by the clauses

$$U \vdash E_n \to V \vdash n \to (U, V, \mathtt{tt}) \in \uparrow_\#, \tag{14}$$

$$a \in U \to b \in E_n \to V \vdash n \to a \not\uparrow b \to (U, V, \mathtt{ff}) \in \uparrow_\# . \tag{15}$$

Again we require the least-fixed-point axiom; it is easy to see that $\uparrow_\#$ is an ideal.

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

$$\mathtt{tt} \in x \uparrow_\# \overline{n} \leftrightarrow x \supseteq E_n, \tag{16}$$

$$\mathtt{ff} \in x \uparrow_\# \overline{n} \leftrightarrow \exists_{a \in x, b \in E_n}(a \not\uparrow b). \tag{17}$$

Proof. (16). Let $n \in \mathbb{N}$. "$\to$". Assume $\mathtt{tt} \in x \uparrow_\# \overline{n}$. We show $x \supseteq E_n$. By continuity of application we have $U \subseteq x$ and $V \subseteq \overline{n}$ such that $(U, V, \mathtt{tt}) \in \uparrow_\#$. Let C be the predicate consisting of all (U, V, c) such that

$$\begin{aligned}&(c = \mathtt{tt} \to \exists_{k \in \mathbb{N}}(U \vdash E_k \wedge V \vdash k)) \wedge \\ &(c = \mathtt{ff} \to \exists_{a \in U, k \in \mathbb{N}, b \in E_k}(V \vdash k \wedge a \not\uparrow b)).\end{aligned}$$

C satisfies (14) and (15). Hence by the least-fixed-point axiom for $\uparrow_\#$ we have $(U, V, \mathtt{tt}) \in C$, i.e., $k \in \mathbb{N}$ such that $U \vdash E_k$ and $V \vdash k$. Using $V \subseteq \overline{n}$ we obtain $k = n$. Now $U \subseteq x$ implies $x \supseteq E_n$.

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

(17) is proved similarly. For "$\to$" we can use the same C, and for "$\leftarrow$" use (15) instead of (14). □

Let ι have at most unary constructors, i.e., be one of $\mathbf{N}$, $\mathbf{B}$ or $\mathbf{P}$. A partial continuous functional Φ of type $\rho_1 \to \cdots \to \rho_p \to \iota$ is *recursive in* pcond*,* $\cup_\#$ *and* $\uparrow_\#$ if it can be defined explicitly by a term involving the constructors for ι and $\mathbf{N}$, the constants predecessor, the fixed point operators Y_ρ, the parallel conditional pcond 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* pcond*,* $\cup_\#$ *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 pcond, $\cup_\#$ and $\uparrow_\#$ is computable. For the converse let Φ be computable of type $\rho_1 \to \cdots \to \rho_p \to \iota$. Then Φ is a primitive recursively enumerated set of tokens $(E_{f_1 n}, \dots, E_{f_p n}, e_{gn})$ where $f_1, \dots, f_p$ and g are fixed primitive recursive functions on index numbers. Let $\overline{f}$ denote a continuous extension of f to ideals, such that $\overline{fn} = \overline{f}\overline{n}$. Such an $\overline{f}$ is obtained by reading f's primitive recursion equations as computation rules in the sense of 2.1.

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

$$w_{\vec{\varphi}}\psi x := \mathrm{pcond}(\varphi_1 \uparrow_{\#} \overline{f_1}x \wedge \dots \wedge \varphi_p \uparrow_{\#} \overline{f_p}x, \psi(x+1) \cup_{\#} \overline{g}x, \psi(x+1)).$$

Here $\wedge$ is the *parallel and* of type $\mathbf{B} \to \mathbf{B} \to \mathbf{B}$, defined by $\wedge(p, q) := \mathrm{pcond}(p, q, \{\mathsf{ff}\})$. To simplify notation we assume $p = 1$ in the argument to follow, and write w for w_{φ}. For later reference we split the rest of the argument into steps.

Step 1. We first prove that

$$\forall_n (a \in w^{k+1}\emptyset\overline{n} \to \exists_{n \le l \le n+k}(\varphi \supseteq E_{fl} \wedge \{e_{gl}\} \vdash a)). \tag{18}$$

The proof is by induction on k. For the base case assume $a \in w\emptyset\overline{n}$, i.e.,

$$a \in \mathrm{pcond}(\varphi \uparrow_{\#} \overline{fn}, \emptyset \cup_{\#} \overline{gn}, \emptyset).$$

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

Step 2. For the step $k \mapsto k+1$ we have

$$a \in w^{k+2}\emptyset\overline{n} = w(w^{k+1}\emptyset)\overline{n} = \mathrm{pcond}(\varphi \uparrow_{\#} \overline{fn}, v \cup_{\#} \overline{gn}, v),$$

with $v := w^{k+1}\emptyset(\overline{n}+1)$. 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 $\Phi\varphi \supseteq Yw\overline{0}$ follows easily. Assume $a \in Yw\overline{0}$. Then $a \in w^{k+1}\emptyset\overline{0}$ for some k, by (2). Therefore there is an l with $0 \le l \le k$ 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 Φ this means that we have an n such that $U = E_{fn}$ and $a = e_{gn}$. We show

$$a \in w^{k+1}\emptyset(\overline{n-k}) \quad \text{for } k \le n.$$

The proof is by induction on k. For $k = 0$ because of $\varphi \supseteq E_{fn}$ we have $\mathsf{tt} \in \varphi \uparrow_{\#} \overline{fn}$ and hence $w\psi\overline{n} = \psi(\overline{n}+1) \cup_{\#} \overline{gn} \ni e_{gn} = a$, for any ψ.

Step 5. For the step $k \mapsto k+1$ by definition of w $(:= w_\varphi)$

$$\begin{aligned} v' &:= w^{k+2}\emptyset(\overline{n-k-1}) \\ &= w(w^{k+1}\emptyset)(\overline{n-k-1}) \\ &= \mathrm{pcond}(\varphi \uparrow_\# \overline{f(n-k-1)}, v \cup_\# \overline{g(n-k-1)}, v) \end{aligned}$$

with $v := w^{k+1}\emptyset(\overline{n-k})$. 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 $(E_{f(n-k-1)}, e_{g(n-k-1)}) \in \Phi$ imply that $\varphi \cup E_{f(n-k-1)}$ is inconsistent, hence $\mathrm{ff} \in \varphi \uparrow_\# \overline{f(n-k-1)}$ 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 ι has at most unary constructors.

Step 7. In case $\{e_{g(n-k-1)}\} \vdash a$ we have $v \cup_\# \overline{g(n-k-1)} \supseteq \{e_{g(n-k-1)}\} \vdash a$, 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 $v \cup_\# \overline{g(n-k-1)} = v$ and therefore again $a \in v'$.

Step 9. Now the converse inclusion $\Phi\varphi \subseteq Y w_\varphi \overline{0}$ can be seen easily. Since $a \in \Phi\varphi$, the claim just proved for $k := n$ gives $a \in w_\varphi^{n+1}\emptyset\overline{0}$, and this implies $a \in Y w_\varphi \overline{0}$. □

3. The Theory TCF$^+$

We sketch a formal system 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, TCF$^+$ contains for every type ρ set variables x^ρ. The only existence axiom for sets will be Σ-comprehension.

3.1. **Types and token types.** Recall that (object) types are built from base types ι (the algebras above) by $\rho \to \sigma$. Now in addition for every (object) type ρ we have *token types* Tok^*_ρ (extended tokens of type ρ), Tok_ρ (tokens of type ρ), LTok_ρ (lists of tokens of type ρ), LTok^*_ρ (lists of extended tokens of type ρ); let τ range over token types. The index ρ will be omitted if it is inessential or clear from the context.

We inductively define the extended tokens of an algebra ι. As a generic algebra we take the algebra $\mathbf{D}$ (of derivations), given by the constructors $0^{\mathbf{D}}$ (axiom) and $\mathrm{C}^{\mathbf{D}\to\mathbf{D}\to\mathbf{D}}$ (rule); for other algebras the definitions are similar. The clauses are

$$\mathrm{Tok}^*_{\mathbf{D}}(*), \quad \mathrm{Tok}^*_{\mathbf{D}}(0^{\mathbf{D}}), \quad \mathrm{Tok}^*_{\mathbf{D}}(a^*_1) \to \mathrm{Tok}^*_{\mathbf{D}}(a^*_2) \to \mathrm{Tok}^*_{\mathbf{D}}(\mathrm{C}^{\mathbf{D}\to\mathbf{D}\to\mathbf{D}} a^*_1 a^*_2).$$

(Proper) tokens are defined similarly:

$$\mathrm{Tok}_{\mathbf{D}}(0^{\mathbf{D}}), \quad \mathrm{Tok}^*_{\mathbf{D}}(a^*_1) \to \mathrm{Tok}^*_{\mathbf{D}}(a^*_2) \to \mathrm{Tok}_{\mathbf{D}}(\mathrm{C}^{\mathbf{D}\to\mathbf{D}\to\mathbf{D}} a^*_1 a^*_2).$$

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 $\mathbf{D}$ is defined by

$$\mathrm{LTok}_{\mathbf{D}}(\mathrm{nil}_{\mathbf{D}}), \quad \mathrm{Tok}_{\mathbf{D}}(a) \to \mathrm{LTok}_{\mathbf{D}}(U) \to \mathrm{LTok}_{\mathbf{D}}(a ::_{\mathbf{D}} U).$$

We use $\mathrm{nil}_{\mathbf{D}}$ to denote the empty list, and $a ::_{\mathbf{D}} U$ (or $\mathrm{cons}_{\mathbf{D}}(a, U)$) 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

$$\mathrm{LTok}^*_{\mathbf{D}}(\mathrm{nil}_{\mathbf{D}}), \quad \mathrm{Tok}^*_{\mathbf{D}}(a) \to \mathrm{LTok}^*_{\mathbf{D}}(U) \to \mathrm{LTok}^*_{\mathbf{D}}(a ::_{\mathbf{D}} U).$$

We allow functions of *token-valued types* $\vec{\tau} \to \tau$, defined by primitive recursion. An easy example is $\dot{\in}_{\mathbf{D}} \colon \mathrm{Tok}^*_{\mathbf{D}} \to \mathrm{LTok}^*_{\mathbf{D}} \to \mathrm{Tok}_{\mathbf{B}}$; it is a boolean-valued function, i.e., with values in $\mathrm{Tok}_{\mathbf{B}}$. The recursion equations are

$$\begin{aligned}
&(a^* \mathrel{\dot{\in}_{\mathbf{D}}} \mathrm{nil}) := \mathtt{ff},\\
&(a^* \mathrel{\dot{\in}_{\mathbf{D}}} (b^* ::_{\mathbf{D}} U)) := (a^* =_{\mathbf{D}} b^*) \vee_{\mathbf{B}} a^* \mathrel{\dot{\in}} U,
\end{aligned}$$

where equality $=_{\mathbf{D}} \colon \mathrm{Tok}^*_{\mathbf{D}} \to \mathrm{Tok}^*_{\mathbf{D}} \to \mathrm{Tok}_{\mathbf{B}}$ is defined by

$$\begin{aligned}
&(* =_{\mathbf{D}} *) := (0 =_{\mathbf{D}} 0) := \mathtt{tt},\\
&(* =_{\mathbf{D}} 0) := (* =_{\mathbf{D}} \mathrm{C}a^*_1 a^*_2) := \mathtt{ff},\\
&(0 =_{\mathbf{D}} *) := (0 =_{\mathbf{D}} \mathrm{C}a^*_1 a^*_2) := \mathtt{ff},\\
&(\mathrm{C}a^*_1 a^*_2 =_{\mathbf{D}} *) := (\mathrm{C}a^*_1 a^*_2 =_{\mathbf{D}} 0) := \mathtt{ff},\\
&(\mathrm{C}a^*_1 a^*_2 =_{\mathbf{D}} \mathrm{C}b^*_1 b^*_2) := (a^*_1 =_{\mathbf{D}} b^*_1) \wedge_{\mathbf{B}} (a^*_2 =_{\mathbf{D}} b^*_2),
\end{aligned}$$

and $\vee_{\mathbf{B}} \colon \mathrm{Tok}_{\mathbf{B}} \to \mathrm{Tok}_{\mathbf{B}} \to \mathrm{Tok}_{\mathbf{B}}$ is the disjunction function on $\mathrm{Tok}_{\mathbf{B}}$, defined by $\mathtt{tt} \vee_{\mathbf{B}} b := \mathtt{tt}$ and $\mathtt{ff} \vee_{\mathbf{B}} b := b$.

From a list of extended tokens of $\mathbf{D}$ we obtain a list of (proper) tokens by removing the $*$'s. Define $\mathrm{clean} \colon \mathrm{LTok}^*_{\mathbf{D}} \to \mathrm{LTok}_{\mathbf{D}}$ by

$$\begin{aligned}
&\mathrm{clean}(\mathrm{nil}) := \mathrm{nil}, && \mathrm{clean}(0 :: U) := 0 :: \mathrm{clean}(U),\\
&\mathrm{clean}(* :: U) := \mathrm{clean}(U), && \mathrm{clean}(\mathrm{C}a^*_1 a^*_2 :: U) := \mathrm{C}a^*_1 a^*_2 :: \mathrm{clean}(U).
\end{aligned}$$

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

$$\begin{aligned}
&\mathrm{args}_{\mathrm{C},i}(\mathrm{nil}) := \mathrm{nil},\\
&\mathrm{args}_{\mathrm{C},i}(0 :: U) := \mathrm{args}_{\mathrm{C},i}(U),\\
&\mathrm{args}_{\mathrm{C},i}(\mathrm{C}a^*_1 a^*_2 :: U) := a^*_i :: \mathrm{args}_{\mathrm{C},i}(U).
\end{aligned}$$

Now we can define entailment $\vdash\colon \mathrm{LTok}_{\mathbf{D}} \to \mathrm{Tok}^*_{\mathbf{D}} \to \mathrm{Tok}_{\mathbf{B}}$:

$$\begin{aligned}
&U \vdash * := \mathtt{tt}, && 0 :: U \vdash \mathrm{C}b_1^* b_2^* := U \vdash \mathrm{C}b_1^* b_2^*,\\
&\mathrm{nil} \vdash 0 := \mathtt{ff}, && \mathrm{C}a_1^* a_2^* :: U \vdash 0 := U \vdash 0,\\
&\mathrm{nil} \vdash \mathrm{C}a_1^* a_2^* := \mathtt{ff}, && 0 :: U \vdash 0 := \mathtt{tt},
\end{aligned}$$

and

$$\begin{aligned}
\mathrm{C}a_1^* a_2^* :: U \vdash \mathrm{C}b_1^* b_2^* := {}&\mathrm{clean}(a_1^* :: \mathrm{args}_{\mathrm{C},1}(U)) \vdash b_1^* \wedge_{\mathbf{B}}\\
&\mathrm{clean}(a_2^* :: \mathrm{args}_{\mathrm{C},2}(U)) \vdash b_2^*,
\end{aligned}$$

where $\wedge_{\mathbf{B}}\colon \mathrm{Tok}_{\mathbf{B}} \to \mathrm{Tok}_{\mathbf{B}} \to \mathrm{Tok}_{\mathbf{B}}$ is the conjunction function on $\mathrm{Tok}_{\mathbf{B}}$, defined by $\mathtt{ff} \wedge_{\mathbf{B}} b := \mathtt{ff}$ and $\mathtt{tt} \wedge_{\mathbf{B}} b := b$.

To define consistency for lists of tokens we need an auxiliary function checking the outermost constructor only. Let $\mathrm{PreCon}\colon \mathrm{LTok}_{\mathbf{D}} \to \mathrm{Tok}_{\mathbf{B}}$ be defined by

$$\begin{aligned}
&\mathrm{PreCon}(\mathrm{nil}) := \mathrm{PreCon}(a :: \mathrm{nil}) := \mathtt{tt},\\
&\mathrm{PreCon}(0 :: \mathrm{C}a_1^* a_2^* :: U) := \mathrm{PreCon}(\mathrm{C}a_1^* a_2^* :: 0 :: U) := \mathtt{ff},\\
&\mathrm{PreCon}(0 :: 0 :: U) := \mathrm{PreCon}(0 :: U),\\
&\mathrm{PreCon}(\mathrm{C}a_1^* a_2^* :: \mathrm{C}b_1^* b_2^* :: U) := \mathrm{PreCon}(\mathrm{C}b_1^* b_2^* :: U).
\end{aligned}$$

Using PreCon we can define consistency $\mathrm{Con}\colon \mathrm{LTok}_{\mathbf{D}} \to \mathrm{Tok}_{\mathbf{B}}$ by

$$\begin{aligned}
&\mathrm{Con}(\mathrm{nil}) := \mathrm{Con}(a :: \mathrm{nil}) := \mathtt{tt},\\
&\mathrm{Con}(0 :: \mathrm{C}a_1^* a_2^* :: U) := \mathrm{Con}(\mathrm{C}a_1^* a_2^* :: 0 :: U) := \mathtt{ff},\\
&\mathrm{Con}(0 :: 0 :: U) := \mathrm{Con}(0 :: U),
\end{aligned}$$

and

$$\begin{aligned}
\mathrm{Con}(\mathrm{C}a_1^* a_2^* :: \mathrm{C}b_1^* b_2^* :: U) := {}&\mathrm{PreCon}(\mathrm{C}b_1^* b_2^* :: U) \wedge_{\mathbf{B}}\\
&\mathrm{Con}(\mathrm{clean}(a_1^* :: b_1^* :: \mathrm{args}_{\mathrm{C},1}(U))) \wedge_{\mathbf{B}}\\
&\mathrm{Con}(\mathrm{clean}(a_2^* :: b_2^* :: \mathrm{args}_{\mathrm{C},2}(U))).
\end{aligned}$$

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

We define $G_{\mathbf{D}}\colon \mathrm{Tok}^*_{\mathbf{D}} \to \mathrm{Tok}_{\mathbf{B}}$ expressing totality for extended tokens:

$$G_{\mathbf{D}}(*) := \mathtt{ff}, \qquad G_{\mathbf{D}}(0) := \mathtt{tt}, \qquad G_{\mathbf{D}}(\mathrm{C}a_1^* a_2^*) := G_{\mathbf{D}} a_1^* \wedge_{\mathbf{B}} G_{\mathbf{D}} a_2^*,$$

and also $G_{\mathrm{LTok}_{\mathbf{D}}}\colon \mathrm{LTok}_{\mathbf{D}} \to \mathrm{Tok}_{\mathbf{B}}$ doing the same for lists of tokens

$$G_{\mathrm{LTok}_{\mathbf{D}}}(\mathrm{nil}_{\mathbf{D}}) := \mathtt{ff}, \qquad G_{\mathrm{LTok}_{\mathbf{D}}}(a ::_{\mathbf{D}} U) := G_{\mathbf{D}} a \vee_{\mathbf{B}} G_{\mathrm{LTok}_{\mathbf{D}}} U.$$

Recall that total tokens of $\mathbf{N}$ 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 $n = \langle a_0, a_1, \dots, a_{k-1} \rangle$, and $k = \mathrm{lh}(n)$, $a_i = (n)_i$ for $i < k$.

Tokens of a function type $\rho \to \sigma$ are pairs (U, a) of lists of tokens of type ρ and tokens of type σ. Both projections are given by functions π_1, π_2. Consistency of lists of tokens, application WU and entailment $U \vdash a$ can be defined as described in 1.2.

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 $\ulcorner\cdot\urcorner\colon \mathrm{Tok}^*_{\mathbf{D}} \to \mathbb{N}$ by

$$
\begin{aligned}
&\ulcorner * \urcorner := 0,\\
&\ulcorner 0 \urcorner := \langle \mathrm{SN}(0) \rangle,\\
&\ulcorner \mathrm{C} a_1^* a_2^* \urcorner := \langle \mathrm{SN}(\mathrm{C}), \ulcorner a_1^* \urcorner, \ulcorner a_2^* \urcorner \rangle.
\end{aligned}
$$

Formal neighborhoods are gödelized by $\ulcorner\cdot\urcorner\colon \mathrm{LTok}_\rho \to \mathbb{N}$,

$$\ulcorner a_0 :: a_1 :: \dots a_{k-1} :: \mathrm{nil} \urcorner := \langle \mathrm{SN}(\mathrm{Nhd}), \ulcorner \rho \urcorner, \ulcorner a_0 \urcorner, \ulcorner a_1 \urcorner, \dots, \ulcorner a_{k-1} \urcorner \rangle,$$

where $\ulcorner \iota \urcorner := \langle \mathrm{SN}(\iota) \rangle$, $\ulcorner \rho \to \sigma \urcorner := \langle \mathrm{SN}(\to), \ulcorner \rho \urcorner, \ulcorner \sigma \urcorner \rangle$. It is clear that we can primitive recursively define the converse, mapping the Gödel number $\ulcorner a^* \urcorner$ of an extended token back to a^*, i.e., $e_{\ulcorner a^* \urcorner} = a^*$, and similarly for LTok_ρ.

3.3. **Terms and formulas.** We have variables a^* for Tok^*_ρ (extended tokens of type ρ), a for Tok_ρ (tokens of type ρ) and U for LTok_ρ (lists of tokens of type ρ). 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 Δ-) *prime formulas* are of the form atom(p), with p a term of token type $\mathrm{Tok}_{\mathbf{B}}$. They are decidable in the sense that for each such term p we can prove $p = \mathtt{tt} \lor p = \mathtt{ff}$; in fact, every closed term of type $\mathrm{Tok}_{\mathbf{B}}$ can be evaluated to either $\mathtt{tt}$ and $\mathtt{ff}$. Examples are $a \uparrow_\rho b$, $a \mathrel{\dot\in}_\rho U$, $U \vdash_\rho a$ (which are shorthand for atom$(a \uparrow_\rho b)$, atom$(a \mathrel{\dot\in}_\rho U)$, atom$(U \vdash_\rho a)$). Δ-*formulas* are built from decidable prime formulas by $\to$, $\land$, $\lor$ and *bounded quantifiers*, i.e., $\forall_{a \dot\in U}$, $\exists_{a \dot\in U}$, with a a variable for tokens and U a term for a list of tokens.

In TCF^+ we also allow variables and constants of (object) type ρ, intended to denote sets of tokens. The constants are $[\![\lambda_{\vec{x}} M]\!]$ (with M a term as in

2.1) of type $\vec{\rho} \to \sigma$, and also pcond, $\cup_{\#}$, $\uparrow_{\#}$ of types $\mathbf{B} \to \rho \to \rho \to \rho$, $\rho \to \mathbf{N} \to \rho$ and $\rho \to \mathbf{N} \to \mathbf{B}$, respectively.

Prime Σ-formulas are either decidable prime formulas or else of the form $r \in_\rho x$, with r a term of token type Tok_ρ and x a variable or constant of type ρ. *Σ-formulas* are built as follows.

(a) Every prime Σ-formula is a Σ-formula.
(b) $A_0 \to B$ is a Σ-formula if A_0 is a Δ-formula and B a Σ-formula.
(c) Σ-formulas are closed under $\wedge, \vee$, bounded quantifiers and existential quantifiers over variables of a token type.

Prime formulas are either prime Σ-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 ρ. *Formulas* are built from prime formulas by $\to, \wedge, \vee, \forall, \exists$, 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 $\mathrm{atom}(\mathsf{ff})$. Then $\mathrm{atom}(\mathsf{ff}) \to A$ ("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

$$A(\mathsf{tt}) \to A(\mathsf{ff}) \to A(a),$$
$$A(*) \to A(0) \to \forall_{a^*,b^*}(A(a^*) \to A(b^*) \to A(\mathrm{C}a^*b^*)) \to A(a^*).$$

Moreover $\mathrm{atom}(\mathsf{tt})$ is an axiom. For object types we have *Σ-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 $[\![\lambda_{\vec{x}} M]\!]$ 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 pcond, $\cup_{\#}$ and $\uparrow_{\#}$, as listed in 2.3.
(c) The clauses from 2.2 defining the totality predicates G_ρ and the equivalence relations $x_1 \approx_\rho x_2$, together with their least-fixed-point axioms.

Notice that the latter imply $x_1 \approx_\rho x_2 \to Gx_1 \to Gx_2$.

3.5. **First steps in TCF^+.** We use the abbreviations

$$\begin{aligned} U \subseteq V \quad &\text{for} \quad \forall_{a \dot{\in} U}(a \dot{\in} V), \\ U \vdash V \quad &\text{for} \quad \forall_{a \dot{\in} V}(U \vdash a), \\ U \sim V \quad &\text{for} \quad U \vdash V \wedge V \vdash U, \end{aligned}$$

$$\begin{aligned} a \sim b \quad &\text{for} \quad \{a\} \vdash b \wedge \{b\} \vdash a, \\ x \subseteq y \quad &\text{for} \quad \forall_{a \in x}(a \in y), \\ x = y \quad &\text{for} \quad x \subseteq y \wedge y \subseteq x, \\ U \subseteq x \quad &\text{for} \quad \forall_{a \dot{\in} U}(a \in x). \end{aligned}$$

Terms of (object) type are built from variables and constants by application ts and comprehension $\{ a \mid A \}$. Then $r \in_\rho t$ for t a term of type ρ and r a term of token type Tok_ρ is defined by

$$\begin{aligned} &(r \in_\rho \{ a \mid A(a) \}) := A(r), \\ &(r \in_\rho ts) := \exists_{U \subseteq s}((U, r) \in t) \qquad \text{(continuity of application).} \end{aligned}$$

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

$$a \in_\sigma [\![M]\!] \quad \text{for} \quad \exists_{\vec{U} \subseteq \vec{x}}((\vec{U}, a) \in_{\vec{\rho} \to \sigma} [\![\lambda_{\vec{x}} M]\!]).$$

We can prove Δ-comprehension for lists of tokens

$$\exists_U \forall_a (a \dot{\in} U \leftrightarrow a \dot{\in} V \wedge A), \quad \text{for every } \Delta\text{-formula } A,$$

by induction on V. A convenient notation for U is $[\, a \dot{\in} V \mid A \,]$.

We will need the *extension* $\overline{f}$ of a monotone token-valued function f to ideals. It suffices to do this for $f \colon \mathrm{Tok}^*_{\mathbf{N}} \to \mathrm{Tok}^*_{\mathbf{N}}$. Suppose f is *monotone*, i.e., $\{a^*\} \vdash b^*$ implies $\{fa^*\} \vdash fb^*$. Define $f[\cdot] \colon \mathrm{LTok}^*_{\mathbf{N}} \to \mathrm{LTok}^*_{\mathbf{N}}$ by

$$f[\mathrm{nil}] := \mathrm{nil}, \qquad f[a^* ::_{\mathbf{N}} U] := (fa^*) ::_{\mathbf{N}} f[U].$$

Then $\overline{f} \colon \mathbf{N} \to \mathbf{N}$ is defined by

$$\overline{f} = \{ (U, a) \mid \mathrm{Con}(U) \wedge f[U] \vdash a \}.$$

Clearly $\overline{f}$ is a decidable ideal. If $f \colon \mathrm{Tok}_{\mathbf{N}} \to \mathrm{Tok}_{\mathbf{N}}$ is defined primitive recursively, then by reading f's primitive recursion equations as computation rules we obtain a defined constant $\overline{f}$ (in the sense of 2.1) such that $\overline{fn} = \overline{f}\overline{n}$.

Notice that $\forall_{i<n} A$ with i a variable and n a term of token type $\mathrm{Tok}_{\mathbf{N}}$ can be viewed as bounded quantification. Define $h \colon \mathrm{Tok}^*_{\mathbf{N}} \to \mathrm{LTok}^*_{\mathbf{N}}$ by

$$h(*) := h(0) := \mathrm{nil}, \qquad h(\mathrm{S}a^*) := h(a^*) * (a^* :: \mathrm{nil}),$$

where $*$ appends two lists from $\mathrm{LTok}^*_{\mathbf{N}}$. Then $h(\mathrm{S}^k 0) = [0, \mathrm{S}0, \dots, \mathrm{S}^{k-1}0]$ (i.e., $0 :: \mathrm{S}0 :: \dots \mathrm{S}^{k-1}0 :: \mathrm{nil}$), and we can read $\forall_{i<n} A$ as $\forall_{i \dot{\in} h(n)} A$.

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

$$(a \dot{\in} Wx) := \exists_{i<n}(U_i \subseteq x \wedge a = a_i).$$

Then $a \dot{\in} Wx$ is a Δ-formula if x is given by $\{ a \mid A \}$ with A a Δ-formula. Therefore by Δ-comprehension for list of tokens we obtain U consisting of all a_i's such that $a_i \mathrel{\dot\in} Wx$. Hence $Wx \vdash a$ can be seen as a Δ-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 ρ, and its proof (by induction on ρ) is to be viewed as employing a "meta"-induction.

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

$$V_U := [\, a_k \mid k \in I_U \,]$$

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

$$r := \{ (U, a) \mid (a \in y_{V_U} \wedge \forall_{i,j<n}(a_i \not\uparrow a_j \to G_\iota(U\vec{z}_{ij}))) \vee V_U \vdash a \},$$

is by Σ-comprehension; in fact, the defining formula is a Δ-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 $W_1, W_2 \in \mathrm{Con}_{\rho\to\sigma}$ with $W_1 \not\uparrow W_2$, and pick $(U_i, a_i) \in W_i$ such that $U_1 \uparrow U_2$ and $a_1 \not\uparrow a_2$. 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 ρ,

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

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

$$\{a_1\}\vec{z}_{a_1,a_2} \not\uparrow_\iota \{a_2\}\vec{z}_{a_1,a_2},$$

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

4.2. **Definability.** We restrict ourselves to the more interesting direction and assume that Φ is given as a primitive recursively enumerated set of tokens (E_{fn}, e_{gn}) where f, g are fixed primitive recursive functions. We need to show that Φ is recursive in pcond, $\cup_\#$ and $\uparrow_\#$, i.e., that it can be defined explicitly by a term involving the constructors for ι and $\mathbf{N}$, the constants predecessor, the fixed point operators Y_ρ, the parallel conditional pcond 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 Φ is definable by the equation $\Phi\varphi = Yw_\varphi\overline{0}$, with w_φ of type $(\mathbf{N} \to \iota) \to \mathbf{N} \to \iota$ given by

$$w_\varphi \psi x := \mathrm{pcond}(\varphi \uparrow_\# \overline{f}x, \psi(x+1) \cup_\# \overline{g}x, \psi(x+1)).$$

In Step 1 by continuity of application we obtain $U \subseteq \varphi \uparrow_\# \overline{fn}$ and $V \subseteq \emptyset \cup_\# \overline{gn}$ such that $(U, V, \emptyset, a) \in \mathrm{pcond}$. For $\varphi \supseteq E_{fn}$ it suffices by (16) to prove $\mathtt{tt} \in \varphi \uparrow_\# \overline{fn}$, which because of $U \subseteq \varphi \uparrow_\# \overline{fn}$ follows from $U \vdash \mathtt{tt}$. This will follow from the rules for pcond, because (since W is $\emptyset$) 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 $C := \{ (U, V, W, a) \mid W \subseteq \emptyset \to U \vdash \mathtt{tt} \}$. We show that C satisfies (4)–(6). For (5) we must show

$$U \vdash \mathtt{ff} \to W \vdash a \to (U, V, W, a) \in C, \quad \text{i.e.,}$$
$$U \vdash \mathtt{ff} \to W \vdash a \to W \subseteq \emptyset \to U \vdash \mathtt{tt}.$$

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 $U \vdash \mathtt{tt}$ among its premises. But now the least-fixed-point axiom for pcond implies $(U, V, \emptyset, a) \in C$ and hence $U \vdash \mathtt{tt}$. For $\{e_{gn}\} \vdash a$ we argue similarly, with $C := \{ (U, V, W, a) \mid W \subseteq \emptyset \to V \vdash a \}$, and obtain $V \vdash a$ and hence $a \in \emptyset \cup_\# \overline{gn}$. 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 $U \subseteq \varphi \uparrow_\# \overline{fn}$, $V \subseteq v \cup_\# \overline{gn}$ and $W \subseteq v$ such that $(U, V, W, a) \in \mathrm{pcond}$. We can prove $W \vdash a \vee (U \vdash \mathtt{tt} \wedge V \vdash a)$ 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 $U \vdash \mathtt{tt}$) and $a \in v \cup_\# \overline{gn}$. From the latter by continuity of application we obtain $V \subseteq v$ and $W \subseteq \overline{gn}$ such that $(V, W, a) \in \cup_\#$. By a least-fixed-point argument (with $C := \{ (V, W, a) \mid \exists_m (m \mathrel{\dot\in} W \wedge \{e_m\} \vdash a) \vee V \vdash a \}$) 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 $m \mathrel{\dot\in} W$, 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 $\mathtt{tt} \in \varphi \uparrow_\# \overline{fn}$ by (16). We show $a \in w\psi\overline{n}$ for an arbitrary ψ, i.e., $a \in \mathrm{pcond}(\varphi \uparrow_\# \overline{fn}, \psi(\overline{n}+1) \cup_\# \overline{gn}, \psi(\overline{n}+1))$. Because of $\mathtt{tt} \in \varphi \uparrow_\# \overline{fn}$ and (7) it is enough to show that $a \in \psi(\overline{n}+1) \cup_\# \overline{gn}$. But $e_{gn} \in \psi(\overline{n}+1) \cup_\# \overline{gn}$ 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 $c \mathrel{\dot\in} E_{f(n-k-1)}$ such that $b \not\asymp c$. Hence $\mathtt{ff} \in \varphi \uparrow_\# \overline{f(n-k-1)}$ by (17), and

$v' = v$ by (8). Step 6 is immediate because of the Comparability Lemma. For Step 7: Here we can infer $a \in v \cup_{\#} \overline{g(n-k-1)}$ from (13). This and the induction hypothesis $a \in v$ yields the claim $a \in v'$ by (9). For Step 8: $v \cup_{\#} \overline{g(n-k-1)} = v$ 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

[1] S. Abramsky and A. Jung. Domain theory. In S. Abramsky, D. Gabbay, and T. Maibaum, editors, *Handbook of Logic in Computer Science*, volume 3, pages 1–168. Clarendon Press, 1994.

[2] U. Berger. Total sets and objects in domain theory. *Annals of Pure and Applied Logic*, 60:91–117, 1993.

[3] Y. L. Ershov. Maximal and everywhere defined functionals. *Algebra i Logika*, 13(4):374–397, 1974.

[4] A. Heyting, editor. *Constructivity in Mathematics*. North–Holland, Amsterdam, 1959.

[5] S. Huber. On the computional content of choice axioms. Master's thesis, Mathematisches Institut der Universität München, 2010.

[6] S. C. Kleene. Countable functionals. In Heyting [4], pages 81–100.

[7] G. Kreisel. Interpretation of analysis by means of constructive functionals of finite types. In Heyting [4], pages 101–128.

[8] K. G. Larsen and G. Winskel. Using information systems to solve recursive domain equations. *Information and Computation*, 91:232–258, 1991.

[9] G. Longo and E. Moggi. The hereditary partial effective functionals and recursion theory in higher types. *The Journal of Symbolic Logic*, 49:1319–1332, 1984.

[10] G. D. Plotkin. LCF considered as a programming language. *Theoretical Computer Science*, 5:223–255, 1977.

[11] G. D. Plotkin. $\mathbf{T}^{\omega}$ as a universal domain. *Journal of Computer and System Sciences*, 17:209–236, 1978.

[12] D. Scott. A type-theoretical alternative to ISWIM, CUCH, OWHY. Manuscript, Oxford University. Published as [15], 1969.

[13] D. Scott. Outline of a mathematical theory of computation. Technical Monograph PRG–2, Oxford University Computing Laboratory, 1970.

[14] 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.

[15] D. Scott. A type-theoretical alternative to ISWIM, CHUCH, OWHY. *Theoretical Computer Science*, 121:411–440, 1993.

[16] V. Stoltenberg-Hansen, E. Griffor, and I. Lindström. *Mathematical Theory of Domains.* Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, 1994.