A theory of computable functionals

Helmut Schwichtenberg

Mathematisches Institut, LMU, München

University of Canterbury, Christchurch, 12 Feb 2016

- Formulas and predicates
- A theory of computable functionals
- Brouwer - Heyting - Kolmogorov and decorations
- The type of a formula or predicate
- Realizability
- Extracted terms

Simultaneously define formulas and predicates

$$
\begin{aligned}
A, B &::= P\vec{r} \mid A \to B \mid \forall_x A, \\
P, Q &::= X \mid \{\, \vec{x} \mid A \,\} \mid \mu_X(\forall_{\vec{x}_i}((A_{i\nu})_{\nu<n_i} \to X\vec{r}_i))_{i<k}
\end{aligned}
$$

Need restriction: X at most strictly positive in $A_{i\nu}$.

$$
\begin{aligned}
T_{\mathbf{N}} &:= \mu_X(X0, \forall_n(Xn \to X(\mathrm{S}n))), \\
\mathrm{Even} &:= \mu_X(X0, \forall_n(Xn \to X(\mathrm{S}(\mathrm{S}n)))), \\
\mathrm{Eq} &:= \mu_X(\forall_x Xxx), \\
\mathrm{Ex}_Y &:= \mu_X(\forall_x(Yx \to X)), \\
\mathrm{Cap}_{Y,Z} &:= \mu_X(\forall_{\vec{x}}(Y\vec{x} \to Z\vec{x} \to X\vec{x})), \\
\mathrm{Cup}_{Y,Z} &:= \mu_X(\forall_{\vec{x}}(Y\vec{x} \to X\vec{x}),\ \forall_{\vec{x}}(Z\vec{x} \to X\vec{x})).
\end{aligned}
$$

Abbreviations

$$
\begin{aligned}
\exists_x A &:= \mathrm{Ex}_{\{x|A\}}, \\
P \cap Q &:= \mathrm{Cap}_{P,Q}, \\
P \cup Q &:= \mathrm{Cup}_{P,Q}.
\end{aligned}
$$

- Formulas and predicates
- A theory of computable functionals
- Brouwer - Heyting - Kolmogorov and decorations
- The type of a formula or predicate
- Realizability
- Extracted terms

Relation to type theory

- Main difference: partial functionals are first class citizens.
- "Logic enriched": Formulas and types kept separate.
- Minimal logic: $\to, \forall$ only. $\mathrm{Eq}(x, y)$ (Leibniz equality), $\exists$, $\vee$, $\wedge$ inductively defined (Russell, Martin-Löf).
- $\mathbf{F} := \mathrm{Eq}(\mathrm{ff}, \mathrm{tt})$. Ex-falso-quodlibet: $\mathbf{F} \to A$ provable.
- "Decorations" $\to^{\mathrm{nc}}$, $\forall^{\mathrm{nc}}$ (i) allow abstract theory (ii) remove unused data.

Theory of computable functionals TCF

Typed variables, ranging over the partial continuous functionals.
Minimal logic, with intro and elim for $\to$ and $\forall$. Axioms:

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

Induction = elimination for totality over $\mathbf{N}$.

$$T_{\mathbf{N}}^- \colon \forall_n(T_{\mathbf{N}}n \to X0 \to \forall_n(T_{\mathbf{N}}n \to Xn \to X(\mathrm{S}n)) \to Xn).$$

Remarks

- Every "competitor" X satisfying the clauses contains $T_{\mathbf{N}}$.
- Induction for $\mathbf{N}$, which only holds for total numbers.
- Fits the logical elimination rules (main part comes first).
- "Strengthened" step formula $\forall_n(T_{\mathbf{N}}n \to Xn \to X(\mathrm{S}n))$.

For nullary predicates $P = \{ \mid A \}$ and $Q = \{ \mid B \}$ we write $A \wedge B$ for $P \cap Q$ and $A \vee B$ for $P \cup Q$. Introduction axioms:

$$
\begin{aligned}
&\forall_x (A \to \exists_x A), \\
&A \to B \to A \wedge B, \\
&A \to A \vee B, \qquad B \to A \vee B.
\end{aligned}
$$

Elimination axioms:

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

Equalities

(i) Defined function constants D introduced by computation rules, written $\ell = r$, but intended as left-to-right rewrites.

(ii) Leibniz equality Eq (inductively defined).

(iii) Pointwise equality between partial continuous functionals, defined inductively as well.

(iv) If ℓ and r have a finitary algebra as their type, $\ell = r$ by (i) is a boolean term. Take $\mathrm{Eq}((\ell = r)^{\mathbf{B}}, \mathtt{tt})$.

In TCF formulas $A(r)$ and $A(s)$ are identified if $r, s \in \mathrm{T}^+$ have a common reduct.

$$\mathrm{Eq}^+ \colon \forall_x \mathrm{Eq}(x^\rho, x^\rho)$$
$$\mathrm{Eq}^- \colon \forall_{x,y}(\mathrm{Eq}(x,y) \to \forall_x Xxx \to Xxy).$$

Compatibility of Eq: $\forall_{x,y}(\mathrm{Eq}(x,y) \to A(x) \to A(y))$.

(Use Eq^- with $\{\, x, y \mid A(x) \to A(y) \,\}$ for X.)

Define falsity by $\mathbf{F} := \mathrm{Eq}(\mathrm{ff}, \mathrm{tt})$.

Ex-falso-quodlibet: $\mathrm{TCF} \vdash \mathbf{F} \to A$ where A has no strictly positive occurrences of (i) predicate variables (ii) inductive predicates without nullary clauses.

Proof.

1. Show $\mathbf{F} \to \mathrm{Eq}(x^\rho, y^\rho)$.

$$\begin{array}{ll} \mathrm{Eq}(\mathcal{R}^\rho_{\mathbf{B}}\mathrm{ff}xy, \mathcal{R}^\rho_{\mathbf{B}}\mathrm{ff}xy) & \text{by } \mathrm{Eq}^+ \\ \mathrm{Eq}(\mathcal{R}^\rho_{\mathbf{B}}\mathrm{tt}xy, \mathcal{R}^\rho_{\mathbf{B}}\mathrm{ff}xy) & \text{by compatibility from } \mathrm{Eq}(\mathrm{ff}, \mathrm{tt}) \\ \mathrm{Eq}(x^\rho, y^\rho) & \text{by conversion.} \end{array}$$

2. Show $\mathbf{F} \to A$, by induction on A. Case $I\vec{s}$.
Let K_0 be the nullary clause, with final conclusion $I\vec{t}$.
By IH from $\mathbf{F}$ we can derive all parameter premises, hence $I\vec{t}$.
From $\mathbf{F}$ we also have $\mathrm{Eq}(s_i, t_i)$ by 1.
Hence $I\vec{s}$ by compatibility.
The cases $A \to B$ and $\forall_x A$ are obvious. $\square$

- Formulas and predicates
- A theory of computable functionals
- Brouwer - Heyting - Kolmogorov and decorations
- The type of a formula or predicate
- Realizability
- Extracted terms

Brouwer - Heyting - Kolmogorov

Have $\to^{\pm}$, $\forall^{\pm}$, $I^{\pm}$. BHK-interpretation:

- p proves $A \to B$ if and only if p is a construction transforming any proof q of A into a proof $p(q)$ of B.
- p proves $\forall_{x^\rho} A(x)$ if and only if p is a construction such that for all a^ρ, $p(a)$ proves $A(a)$.

Leaves open:

- What is a "construction"?
- What is a proof of a prime formula?

Proposal:

- Construction: computable functional.
- Proof of a prime formula $I\vec{r}$: generation tree.

Example: generation tree for $\mathrm{Even}(6)$ should consist of a single branch with nodes $\mathrm{Even}(0)$, $\mathrm{Even}(2)$, $\mathrm{Even}(4)$ and $\mathrm{Even}(6)$.

Decoration

Which of the variables $\vec{x}$ and assumptions $\vec{A}$ are actually used in the "solution" provided by a proof of

$$\forall_{\vec{x}}(\vec{A} \to I\vec{r})?$$

To express this we split each of $\to, \forall$ into two variants:

- a "computational" one $\to^{\mathrm{c}}, \forall^{\mathrm{c}}$ and
- a "non-computational" one $\to^{\mathrm{nc}}, \forall^{\mathrm{nc}}$ (with restricted rules)

and consider

$$\forall^{\mathrm{nc}}_{\vec{x}}\forall^{\mathrm{c}}_{\vec{y}}(\vec{A} \to^{\mathrm{nc}} \vec{B} \to^{\mathrm{c}} X\vec{r}).$$

This will lead to a different (simplified) algebra ι_I associated with the inductive predicate I.

Each inductive predicate is marked as computationally relevant (c.r.) or non-computational (n.c., or Harrop): $\mu_X^{\mathrm{nc}}(K_0, \dots, K_{k-1})$. Then the elimination scheme must be restricted to n.c. formulas.

We usually write $\to$, $\forall$, μ for $\to^{\mathrm{c}}$, $\forall^{\mathrm{c}}$, μ^{c}. Notice that in the clauses of an n.c. inductive predicate $\mu_X^{\mathrm{nc}}\vec{K}$ decorations play no role.

For the even numbers we now have two variants:

$$\begin{aligned}\mathrm{Even} &:= \mu_X(X0, \forall_n^{\mathrm{nc}}(Xn \to X(\mathrm{S}(\mathrm{S}n)))),\\ \mathrm{Even}^{\mathrm{nc}} &:= \mu_X^{\mathrm{nc}}(X0, \forall_n(Xn \to X(\mathrm{S}(\mathrm{S}n)))).\end{aligned}$$

Generally for every c.r. inductive predicate I (i.e., defined as $\mu_X\vec{K}$) we have an n.c. variant I^{nc} defined as $\mu_X^{\mathrm{nc}}\vec{K}$.

$$
\begin{aligned}
\mathrm{ExD}_Y &:= \mu_X(\forall_x(Yx \to X)),\\
\mathrm{ExL}_Y &:= \mu_X(\forall_x(Yx \to^{\mathrm{nc}} X)).\\
\mathrm{ExR}_Y &:= \mu_X(\forall_x^{\mathrm{nc}}(Yx \to X)),\\
\mathrm{ExU}_Y &:= \mu_X^{\mathrm{nc}}(\forall_x(Yx \to X)).
\end{aligned}
$$

D for "double", L for "left", R for "right", U for "uniform".
Abbreviations

$$
\begin{aligned}
\exists_x^{\mathrm{d}} A &:= \mathrm{ExD}_{\{x|A\}},\\
\exists_x^{\mathrm{l}} A &:= \mathrm{ExL}_{\{x|A\}},\\
\exists_x^{\mathrm{r}} A &:= \mathrm{ExR}_{\{x|A\}},\\
\exists_x^{\mathrm{u}} A &:= \mathrm{ExU}_{\{x|A\}}.
\end{aligned}
$$

$$
\begin{aligned}
\mathrm{CupD}_{Y,Z} &:= \mu_X(Y \to X,\ Z \to X),\\
\mathrm{CupL}_{Y,Z} &:= \mu_X(Y \to X,\ Z \to^{\mathrm{nc}} X),\\
\mathrm{CupR}_{Y,Z} &:= \mu_X(Y \to^{\mathrm{nc}} X,\ Z \to X),\\
\mathrm{CupU}_{Y,Z} &:= \mu_X(Y \to^{\mathrm{nc}} X,\ Z \to^{\mathrm{nc}} X),\\
\mathrm{CupNC}_{Y,Z} &:= \mu_X^{\mathrm{nc}}(Y \to X,\ Z \to X).
\end{aligned}
$$

The final nc-variant suppresses even the information which clause has been used. Abbreviations

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

- Formulas and predicates
- A theory of computable functionals
- Brouwer - Heyting - Kolmogorov and decorations
- The type of a formula or predicate
- Realizability
- Extracted terms

Examples

Let $a, b \in \mathbf{Q}$, $x \in \mathbf{R}$, $k \in \mathbf{Z}$, $f \in \mathbf{R} \to \mathbf{R}$.

- $\forall_{a,b,x}(a < b \to x \leq b \vee^{\mathrm{u}} a \leq x)$ has type $\mathbf{Q} \to \mathbf{Q} \to \mathbf{R} \to \mathbf{B}$.
- $\forall_{a,b,x}(a < b \to x < b \vee^{\mathrm{d}} a < x)$ has type $\mathbf{Q} \to \mathbf{Q} \to \mathbf{R} \to \mathbf{Z} + \mathbf{Z}$.
- The formula

$$\begin{aligned}\forall_{f,k}(f(0) \leq 0 \leq f(1) \to \\ &\forall_{a,b}\left(\frac{1}{2^k}|b-a| \leq |f(b)-f(a)|\right) \to \\ &\exists^{\mathrm{l}}_x f(x)=0)\end{aligned}$$

has type $(\mathbf{R} \to \mathbf{R}) \to \mathbf{Z} \to \mathbf{R}$.

The type $\tau(C)$ of a formula or predicate C

$\tau(C)$ type or the "nulltype symbol" $\circ$. Extend use of $\rho \to \sigma$ to $\circ$:

$$(\rho \to \circ) := \circ, \quad (\circ \to \sigma) := \sigma, \quad (\circ \to \circ) := \circ.$$

Assume a global injective assignment of a type variable ξ to every c.r. predicate variable X. Let $\tau(C) := \circ$ if C is non-computational. In case C is c.r. let

$$\begin{aligned}
&\tau(P\vec{r}\,) := \tau(P),\\
&\tau(A \to B) := (\tau(A) \to \tau(B)), \quad \tau(A \to^{\mathrm{nc}} B) := \tau(B),\\
&\tau(\forall_{x^\rho} A) := (\rho \to \tau(A)), \quad \tau(\forall^{\mathrm{nc}}_{x^\rho} A) := \tau(A),\\
&\tau(X) := \xi,\\
&\tau(\{\, \vec{x} \mid A \,\}) := \tau(A),\\
&\tau(\underbrace{\mu_X(\forall^{\mathrm{nc}}_{\vec{x}_i}\forall_{\vec{y}_i}(\vec{A}_i \to^{\mathrm{nc}} \vec{B}_i \to X\vec{r}_i))_{i<k}}_{I}) := \underbrace{\mu_\xi(\tau(\vec{y}_i) \to \tau(\vec{B}_i) \to \xi)_{i<k}}_{\iota_I}.
\end{aligned}$$

ι_I is the algebra associated with I.

- Formulas and predicates
- A theory of computable functionals
- Brouwer - Heyting - Kolmogorov and decorations
- The type of a formula or predicate
- Realizability
- Extracted terms

Realizability

For every predicate or formula C we define an n.c. predicate $C^{\mathbf{r}}$.
For n.c. C let

$$C^{\mathbf{r}} := C.$$

In case C is c.r. the arity of $C^{\mathbf{r}}$ is $(\tau(C), \vec{\sigma})$ with $\vec{\sigma}$ the arity of C.
For c.r. formulas define

$$\begin{aligned}
(P\vec{r})^{\mathbf{r}} &:= \{\, u \mid P^{\mathbf{r}} u \vec{r} \,\} \\
(A \to B)^{\mathbf{r}} &:= \begin{cases} \{\, u \mid \forall_v (A^{\mathbf{r}} v \to B^{\mathbf{r}}(uv)) \,\} & \text{if } A \text{ is c.r.} \\ \{\, u \mid A \to B^{\mathbf{r}} u \,\} & \text{if } A \text{ is n.c.} \end{cases} \\
(A \to^{\mathrm{nc}} B)^{\mathbf{r}} &:= \{\, u \mid A \to B^{\mathbf{r}} u \,\} \\
(\forall_x A)^{\mathbf{r}} &:= \{\, u \mid \forall_x A^{\mathbf{r}}(ux) \,\} \\
(\forall^{\mathrm{nc}}_x A)^{\mathbf{r}} &:= \{\, u \mid \forall_x A^{\mathbf{r}} u \,\}.
\end{aligned}$$

For c.r. predicates: given n.c. $X^{\mathbf{r}}$ for all predicate variables X.

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

Consider a c.r. inductive predicate

$$I := \mu_X(\forall^{\mathrm{c/nc}}_{\vec{x}_i}((A_{i\nu})_{\nu<n_i} \to^{\mathrm{c/nc}} X\vec{r}_i))_{i<k}.$$

$\vec{Y}$: all predicate variables strictly positive in some $A_{i\nu}$ except X. Define the witnessing predicate with free predicate variables $\vec{Y}^{\mathbf{r}}$ by

$$I^{\mathbf{r}} := \mu^{\mathrm{nc}}_{X^{\mathbf{r}}}(\forall_{\vec{x}_i,\vec{u}_i}((A^{\mathbf{r}}_{i\nu}u_{i\nu})_{\nu<n_i} \to X^{\mathbf{r}}(\mathrm{C}_i\vec{x}_i\vec{u}_i)\vec{r}_i))_{i<k}$$

with the understanding that

(i) $u_{i\nu}$ occurs only when $A_{i\nu}$ is c.r., and it occurs as an argument in $\mathrm{C}_i\vec{x}_i\vec{u}_i$ only if $A_{i\nu}$ is c.r. and followed by $\to$, and

(ii) only those x_{ij} with $\forall^{\mathrm{c}}_{x_{ij}}$ occur as arguments in $\mathrm{C}_i\vec{x}_i\vec{u}_i$.

We write $u\ \mathbf{r}\ A$ for $A^{\mathbf{r}}u$ (u realizes A).

For the even numbers we obtain

$$\mathrm{Even} := \mu_X(X0, \forall^{\mathrm{nc}}_n(Xn \to X(\mathrm{S}(\mathrm{S}n))))$$
$$\mathrm{Even}^{\mathbf{r}} := \mu^{\mathrm{nc}}_{X^{\mathbf{r}}}(X^{\mathbf{r}}00, \forall_{n,m}(X^{\mathbf{r}}mn \to X^{\mathbf{r}}(\mathrm{S}m)(\mathrm{S}(\mathrm{S}n)))).$$

Axiom (Invariance under realizability)

$$\mathrm{Inv}_A \colon A \leftrightarrow \exists^{\mathrm{l}}_u(u \mathbf{\ r\ } A) \qquad \text{for c.r. formulas } A.$$

Lemma

For c.r. formulas A we have

$$(\lambda_u u) \mathbf{\ r\ } (A \to \exists^{\mathrm{l}}_u(u \mathbf{\ r\ } A)),$$
$$(\lambda_u u) \mathbf{\ r\ } (\exists^{\mathrm{l}}_u(u \mathbf{\ r\ } A) \to A).$$

From the invariance axioms we can derive

Theorem (Choice)

$$\forall_x \exists^{\mathrm{l}}_y A(y) \to \exists^{\mathrm{l}}_f \forall_x A(fx) \quad \text{for } A \text{ n.c.}$$
$$\forall_x \exists^{\mathrm{d}}_y A(y) \to \exists^{\mathrm{d}}_f \forall_x A(fx) \quad \text{for } A \text{ c.r.}$$

Theorem (Independence of premise). Assume $x \notin \mathrm{FV}(A)$.

$$(A \to \exists^{\mathrm{l}}_x B) \to \exists^{\mathrm{l}}_x (A \to B) \qquad \text{for } A, B \text{ n.c.}$$
$$(A \to^{\mathrm{nc}} \exists^{\mathrm{l}}_x B) \to \exists^{\mathrm{l}}_x (A \to B) \quad \text{for } B \text{ n.c.}$$
$$(A \to \exists^{\mathrm{d}}_x B) \to \exists^{\mathrm{d}}_x (A \to B) \qquad \text{for } A \text{ n.c., } B \text{ c.r.}$$
$$(A \to^{\mathrm{nc}} \exists^{\mathrm{d}}_x B) \to \exists^{\mathrm{d}}_x (A \to B) \quad \text{for } B \text{ c.r.}$$

- Formulas and predicates
- A theory of computable functionals
- Brouwer - Heyting - Kolmogorov and decorations
- The type of a formula or predicate
- Realizability
- Extracted terms

For derivations M^A with A n.c. let $\mathrm{et}(M^A) := \varepsilon$. Otherwise

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

Extracted terms for the axioms.

- Let I be c.r.

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

  where both C_i and $\mathcal{R}$ refer to the algebra ι_I associated with I.
- For the invariance axioms we take identities.

Theorem (Soundness)

Let M be a derivation of a c.r. formula A from assumptions $u_i \colon C_i$ $(i < n)$. Then we can derive $\mathrm{et}(M)$ $\mathbf{r}$ A from assumptions v_{u_i} $\mathbf{r}$ C_i in case C_i is c.r. and C_i otherwise.

Proof.

By induction on M. $\square$

Conclusion

- Assume M proves A. The derivation in TCF of $\mathrm{et}(M)\ \mathbf{r}\ A$ is automatically generated and can be machine checked.
- Minlog can translate $\mathrm{et}(M)$ into Scheme and Haskell code.
- Coq's extraction returns Ocaml, Scheme or Haskell code, not terms in a "logical" language like T^+.
- Agda views (complete) proofs as programs.