Cobham Recursive Set Functions

Moritz Müller

Kurt Gödel Research Center, Vienna, Austria.

joint with A. Beckmann, S. Buss, S.-D. Friedman and N. Thapen

BASICS 2015 Summer School
Logic Summer School in China 2015
Zhejiang Normal University

Computations on arbitrary sets

There are many equivalent definitions of the class of recursive functions on the natural numbers. Different definitions have different uses while the equivalence of all the notions provides evidence for Church's thesis, the thesis that the concept of recursive function is the most reasonable explication of our intuitive notion of effectively calculable function. As the various definitions are lifted to domains other than the integers (e. g., admissible sets) some of the equivalences break down. This break-down provides us with a laboratory for the study of recursion theory. Barwise, 1975

Computations on arbitrary sets

> *There are many equivalent definitions of the class of recursive functions on the natural numbers. Different definitions have different uses while the equivalence of all the notions provides evidence for Church's thesis, the thesis that the concept of recursive function is the most reasonable explication of our intuitive notion of effectively calculable function. As the various definitions are lifted to domains other than the integers (e. g., admissible sets) some of the equivalences break down. This break-down provides us with a laboratory for the study of recursion theory.* Barwise, 1975

A computable function over $\mathbb{N}$ is one

Recursion theoretic view obtainable from certain initial functions by means of composition, primitive recursion and the μ-operator.

Definability theoretic view Σ_1-definable in the language of arithmetic.

Primitive recursive set functions (Jensen, Karp 1971)

are obtained from initial functions

- constant $0 = \emptyset$
- projections
- pair $x, y \mapsto \{x, y\}$
- union $x \mapsto \bigcup x$
- conditional $\mathrm{cond}_{\in}(x, y, u, v) := \texttt{if } u \in v \texttt{ then } x \texttt{ else } y$

by composition and ϵ-recursion

- if $g(z, y, \vec{x})$ is PRSF, then so is $f(y, \vec{x}) = g(\{f(u, \vec{x}) : u \in y\}, y, \vec{x})$

Primitive recursive set functions (Jensen, Karp 1971)

are obtained from initial functions

- constant $0 = \emptyset$
- projections
- pair $x, y \mapsto \{x, y\}$
- union $x \mapsto \bigcup x$
- conditional $\text{cond}_{\in}(x, y, u, v) := \texttt{if } u \in v \texttt{ then } x \texttt{ else } y$

by composition and ϵ-recursion

- if $g(z, y, \vec{x})$ is PRSF, then so is $f(y, \vec{x}) = g(\{f(u, \vec{x}) : u \in y\}, y, \vec{x})$

Generalizes primitive recursive computations to arbitrary sets

Goal Similarly generalize polynomial time computations to arbitrary sets

Need Recursion theoretic definition of PTIME

Cobham's definition of polynomial time 1965

The polynomial time function over $\mathbb{N}$ are those obtained from initial functions

constant 0, projections, successors $s_0(x) = 2x$ and $s_1(x) = 2x + 1$

smash $x\#y = 2^{|x| \cdot |y|}$ where $|x| = \lceil \log(x+1) \rceil$

by composition and limited recursion on notation

if h, g_0, g_1, t are polynomial time, then so is

$f(0, \vec{x}) = h(\vec{x})$

$f(s_b(y), \vec{x}) = g_b(f(y, \vec{x}), y, \vec{x})$ where $b \in \{0, 1\}$

Cobham's definition of polynomial time 1965

The polynomial time function over $\mathbb{N}$ are those obtained from initial functions

constant 0, projections, successors $s_0(x) = 2x$ and $s_1(x) = 2x + 1$

smash $x\#y = 2^{|x| \cdot |y|}$ where $|x| = \lceil \log(x+1) \rceil$

by composition and limited recursion on notation

if h, g_0, g_1, t are polynomial time, then so is

$f(0, \vec{x}) = h(\vec{x})$

$f(s_b(y), \vec{x}) = g_b(f(y, \vec{x}), y, \vec{x})$ where $b \in \{0, 1\}$

provided $f(y, \vec{x}) \leq t(y, \vec{x})$ for all $y, \vec{x}$.

Cobham's definition of polynomial time 1965

The polynomial time function over $\mathbb{N}$ are those obtained from initial functions

constant 0, projections, successors $s_0(x) = 2x$ and $s_1(x) = 2x + 1$

smash $\quad x\#y = 2^{|x| \cdot |y|} \quad$ where $|x| = \lceil \log(x+1) \rceil$

by composition and limited recursion on notation

if h, g_0, g_1, t are polynomial time, then so is

$$f(0, \vec{x}) = h(\vec{x})$$
$$f(s_b(y), \vec{x}) = g_b(f(y, \vec{x}), y, \vec{x}) \quad \text{where } b \in \{0, 1\}$$

provided $\quad f(y, \vec{x}) \leq t(y, \vec{x})$ for all $y, \vec{x}$.

Equivalent proviso:

- t a #-term: built from variables, $1 = s_1(0)$ and $\#$.
- $|f(y, x_1, x_2 \ldots)| \leq p(|y|, |x_1|, |x_2| \ldots)$ for some polynomial p

Set composition and set smash

Set composition $$x \odot y := \begin{cases} y & \text{if } x = 0 \\ \{u \odot y : u \in x\} & \text{if } x \neq 0 \end{cases}$$

Set composition and set smash

Set composition $$x \odot y := \begin{cases} y & \text{if } x = 0 \\ \{u \odot y : u \in x\} & \text{if } x \neq 0 \end{cases}$$

Set smash $$x \# y := y \odot \{u \# y : u \in x\}$$

Set composition and set smash

Set composition $\quad x \odot y := \begin{cases} y & \text{if } x = 0 \\ \{u \odot y : u \in x\} & \text{if } x \neq 0 \end{cases}$

Set smash $\quad x\#y := y \odot \{u\#y : u \in x\}$

#-term $t(\vec{x})$ built from variables $\vec{x}$, 1=$\{0\}$, $\odot$ and #

- There are polynomials p, q such that for all $\vec{x}$

  $\mathsf{rk}(t(x_1, x_2 \ldots)) \leq p(\mathsf{rk}(x_1), \mathsf{rk}(x_2) \ldots)$

  $|\mathsf{tc}(t(x_1, x_2 \ldots))| \leq q(|\mathsf{tc}(x_1)|, |\mathsf{tc}(x_2)| \ldots)$

Set composition and set smash

Set composition $x \odot y := \begin{cases} y & \text{if } x = 0 \\ \{u \odot y : u \in x\} & \text{if } x \neq 0 \end{cases}$

Set smash $x \# y := y \odot \{u \# y : u \in x\}$

#-term $t(\vec{x})$ built from variables $\vec{x}$, $1{=}\{0\}$, $\odot$ and $\#$

- There are polynomials p, q such that for all $\vec{x}$

  $\mathsf{rk}(t(x_1, x_2 \ldots)) \leq p(\mathsf{rk}(x_1), \mathsf{rk}(x_2) \ldots)$

  $|\mathsf{tc}(t(x_1, x_2 \ldots))| \leq q(|\mathsf{tc}(x_1)|, |\mathsf{tc}(x_2)| \ldots)$

Intuition #-terms play the role of polynomial length bounds.

Intuition consider only hereditarily finite x: finite Mostowski graph, one can compute $f(x) = g(\{f(u) : u \in x\}, x)$ with oracle g in parallel time $\approx \mathsf{rk}(x)$ and total work $\approx |\mathsf{tc}(x)|$.

Bounding relation $\preccurlyeq$

A single-valued embedding of x into y is $\tau : \mathsf{tc}(x) \to \mathsf{tc}(y)$ st

if $u \neq v$, then $\tau(u) \neq \tau(v)$

if $u \in v$, then $\tau(u) \in \mathsf{tc}(\tau(v))$

Example if $x \subseteq y$, then the identity is such an embedding.

Bounding relation $\preccurlyeq$

A single-valued embedding of x into y is $\tau : \mathrm{tc}(x) \to \mathrm{tc}(y)$ st

if $u \neq v$, then $\tau(u) \neq \tau(v)$

if $u \in v$, then $\tau(u) \in \mathrm{tc}(\tau(v))$

Example if $x \subseteq y$, then the identity is such an embedding.

A (multi-valued) embedding of x into y is $\tau : \mathrm{tc}(x) \to P(\mathrm{tc}(y)) \setminus \{0\}$ st

if $u \neq v$, then $\tau(u) \cap \tau(v) = 0$

if $u \in v$ and $v' \in \tau(v)$, then there exists $u' \in \tau(u)$ such that $u' \in \mathrm{tc}(v')$

Bounding relation $\preccurlyeq$

A single-valued embedding of x into y is $\tau : \mathrm{tc}(x) \to \mathrm{tc}(y)$ st

if $u \neq v$, then $\tau(u) \neq \tau(v)$

if $u \in v$, then $\tau(u) \in \mathrm{tc}(\tau(v))$

Example if $x \subseteq y$, then the identity is such an embedding.

A (multi-valued) embedding of x into y is $\tau : \mathrm{tc}(x) \to P(\mathrm{tc}(y)) \setminus \{0\}$ st

if $u \neq v$, then $\tau(u) \cap \tau(v) = 0$

if $u \in v$ and $v' \in \tau(v)$, then there exists $u' \in \tau(u)$ such that $u' \in \mathrm{tc}(v')$

- Then $\mathrm{rk}(x) \leq \mathrm{rk}(y)$ and $|\mathrm{tc}(x)| \leq |\mathrm{tc}(y)|$.

Intuition Then x is structurally no more complex than y.

Cobham recursive set functions

are obtained from initial functions

constant 0, projections, pair $\{x, y\}$, union $\bigcup x$, $\mathrm{cond}_{\in}(x, y, u, v)$, smash $x\#y$

by composition and Cobham recursion

if g, τ, t are CRSF, then so is

$$f(y, \vec{x}) = g(\{f(u, \vec{x}) : u \in y\}, y, \vec{x})$$

Cobham recursive set functions

are obtained from initial functions

constant 0, projections, pair $\{x, y\}$, union $\bigcup x$, $\mathrm{cond}_{\in}(x, y, u, v)$, smash $x\#y$

by composition and Cobham recursion

if g, τ, t are CRSF, then so is

$$f(y, \vec{x}) = g(\{f(u, \vec{x}) : u \in y\}, y, \vec{x})$$

provided $\tau(\cdot, y, \vec{x}) : f(y, \vec{x}) \preccurlyeq t(y, \vec{x})$ for all $y, \vec{x}$.

(i.e. $u \mapsto \tau(u, y, \vec{x})$ is an embedding of $f(y, \vec{x})$ into $t(y, \vec{x})$)

Cobham recursive set functions

are obtained from initial functions

constant 0, projections, pair $\{x, y\}$, union $\bigcup x$, $\mathrm{cond}_{\in}(x, y, u, v)$, smash $x \# y$

by composition and Cobham recursion

if g, τ, t are CRSF, then so is

$$f(y, \vec{x}) = g(\{f(u, \vec{x}) : u \in y\}, y, \vec{x})$$

provided $\tau(\cdot, y, \vec{x}) : f(y, \vec{x}) \preccurlyeq t(y, \vec{x})$ for all $y, \vec{x}$.
(i.e. $u \mapsto \tau(u, y, \vec{x})$ is an embedding of $f(y, \vec{x})$ into $t(y, \vec{x})$)

- equivalent: demand t to be a #-term.
- equivalent: allow "impredicative" $\tau(\cdot, y, \vec{x}, f(y, \vec{x}))$.

Bootstrapping CRSF

- **Bounded replacement** if g, τ, t are CRSF, then so is

  $f(x) = \{g(u, x) : u \in x\}$ provided $\tau(\cdot, x) : f(x) \preccurlyeq t(x)$

Bootstrapping CRSF

- **Bounded replacement** if g, τ, t are CRSF, then so is

  $$f(x) = \{g(u, x) : u \in x\} \quad \text{provided } \tau(\cdot, x) : f(x) \preccurlyeq t(x)$$

- **Separation** if g is CRSF, then so is $f(x) = \{u \in x : g(u) \neq 0\}$;

Bootstrapping CRSF

- **Bounded replacement** if g, τ, t are CRSF, then so is

  $f(x) = \{g(u, x) : u \in x\}$ provided $\tau(\cdot, x) : f(x) \preccurlyeq t(x)$

- **Separation** if g is CRSF, then so is $f(x) = \{u \in x : g(u) \neq 0\}$;

  $h(u) := \texttt{if } g(u) \in \{0\} \texttt{ then } 0 \texttt{ else } \{u\}$
  Bounded replacement gives $f(x) = \{\bigcup h(u) : u \in x\}$
  Proviso satisfied since $f(x) \subseteq x$. □

Bootstrapping CRSF

- **Bounded replacement** if g, τ, t are CRSF, then so is

  $f(x) = \{g(u, x) : u \in x\}$ provided $\tau(\cdot, x) : f(x) \preccurlyeq t(x)$

- **Separation** if g is CRSF, then so is $f(x) = \{u \in x : g(u) \neq 0\}$;

  $h(u) := \texttt{if } g(u) \in \{0\} \texttt{ then } 0 \texttt{ else } \{u\}$
  Bounded replacement gives $f(x) = \{\bigcup h(u) : u \in x\}$
  Proviso satisfied since $f(x) \subseteq x$. □

- $\langle x, y\rangle, x \cup y, x \setminus y, x \cap y, x \odot y$ and $\mathrm{tc}(x) = x \cup \bigcup\{\mathrm{tc}(u) : u \in x\}$ are CRSF.

Bootstrapping CRSF

- **Bounded replacement** if g, τ, t are CRSF, then so is

  $f(x) = \{g(u, x) : u \in x\}$ provided $\tau(\cdot, x) : f(x) \preccurlyeq t(x)$

- **Separation** if g is CRSF, then so is $f(x) = \{u \in x : g(u) \neq 0\}$;

  $h(u) := \texttt{if } g(u) \in \{0\} \texttt{ then } 0 \texttt{ else } \{u\}$
  Bounded replacement gives $f(x) = \{\bigcup h(u) : u \in x\}$
  Proviso satisfied since $f(x) \subseteq x$. □

- $\langle x, y\rangle, x \cup y, x \setminus y, x \cap y, x \odot y$ and $\mathrm{tc}(x) = x \cup \bigcup\{\mathrm{tc}(u) : u \in x\}$ are CRSF.

- CRSF relations (i.e. char function is CRSF) are

  closed under Boolean combinations and bounded quantification.

Bootstrapping CRSF

- **Bounded replacement** if g, τ, t are CRSF, then so is

  $f(x) = \{g(u, x) : u \in x\}$ provided $\tau(\cdot, x) : f(x) \preccurlyeq t(x)$

- **Separation** if g is CRSF, then so is $f(x) = \{u \in x : g(u) \neq 0\}$;

  $h(u) := \texttt{if } g(u) \in \{0\} \texttt{ then } 0 \texttt{ else } \{u\}$
  Bounded replacement gives $f(x) = \{\bigcup h(u) : u \in x\}$
  Proviso satisfied since $f(x) \subseteq x$. □

- $\langle x, y\rangle, x \cup y, x \setminus y, x \cap y, x \odot y$ and $\mathrm{tc}(x) = x \cup \bigcup\{\mathrm{tc}(u) : u \in x\}$ are CRSF.

- CRSF relations (i.e. char function is CRSF) are
  closed under Boolean combinations and bounded quantification.

  $\exists u \in y R(u, \vec{x})$ has char function $y, \vec{x} \mapsto \bigcup\{\chi_R(u, \vec{x}) : u \in y\}$.
  Use Bounded replacement. Proviso: values are $\subseteq 1$ □

Example: the rank function

Suffices to show $\mathsf{rk}^+(x) := \mathsf{rk}(x) + 1$ is CRSF. Define

$$\mathsf{rk}^+(x) := \mathsf{Succ}\Big(\bigcup\{\mathsf{rk}^+(u) : u \in x\} \Big) \qquad \text{where } \mathsf{Succ}(x) := x \cup \{x\}$$

by Cobham Recursion.

Example: the rank function

Suffices to show $\mathsf{rk}^+(x) := \mathsf{rk}(x) + 1$ is CRSF. Define

$$\mathsf{rk}^+(x) := \mathsf{Succ}\Big(\bigcup\{\mathsf{rk}^+(u) : u \in x\} \Big) \qquad \text{where } \mathsf{Succ}(x) := x \cup \{x\}$$

by Cobham Recursion. Proviso: multi-valued embedding (into $\{x\}$)

$$\tau(\alpha, x) := \{u \in \mathsf{tc}(\{x\}) : \mathsf{rk}(u) = \alpha\}.$$

Example: the rank function

Suffices to show $\mathsf{rk}^+(x) := \mathsf{rk}(x) + 1$ is CRSF. Define

$$\mathsf{rk}^+(x) := \mathsf{Succ}\Big(\bigcup\{\mathsf{rk}^+(u) : u \in x\} \Big) \qquad \text{where } \mathsf{Succ}(x) := x \cup \{x\}$$

by Cobham Recursion. Proviso: multi-valued embedding (into $\{x\}$)

$$\tau(\alpha, x) := \{u \in \mathsf{tc}(\{x\}) : \mathsf{rk}(u) = \alpha\}.$$

Need: τ is CRSF. By Separation, it suffices to show: $\mathsf{rk}(x) \leq \mathsf{rk}(y)$ is CRSF.

Example: the rank function

Suffices to show $\mathrm{rk}^+(x) := \mathrm{rk}(x) + 1$ is CRSF. Define

$$\mathrm{rk}^+(x) := \mathrm{Succ}\Big(\bigcup\{\mathrm{rk}^+(u) : u \in x\} \Big) \qquad \text{where } \mathrm{Succ}(x) := x \cup \{x\}$$

by Cobham Recursion. Proviso: multi-valued embedding (into $\{x\}$)

$$\tau(\alpha, x) := \{u \in \mathrm{tc}(\{x\}) : \mathrm{rk}(u) = \alpha\}.$$

Need: τ is CRSF. By Separation, it suffices to show: $\mathrm{rk}(x) \leq \mathrm{rk}(y)$ is CRSF.

$$f(x, y) := \Big\{ \; u \in \mathrm{tc}(\{x\}) : u \subseteq \bigcup\{f(x, v) : v \in y\} \; \Big\}$$

is CRSF by Cobham recursion. Proviso: identity embeds into $\{x\}$.

Example: the rank function

Suffices to show $\mathsf{rk}^+(x) := \mathsf{rk}(x) + 1$ is CRSF. Define

$$\mathsf{rk}^+(x) := \mathsf{Succ}\Big(\bigcup\{\mathsf{rk}^+(u) : u \in x\} \Big) \qquad \text{where } \mathsf{Succ}(x) := x \cup \{x\}$$

by Cobham Recursion. Proviso: multi-valued embedding (into $\{x\}$)

$$\tau(\alpha, x) := \{u \in \mathsf{tc}(\{x\}) : \mathsf{rk}(u) = \alpha\}.$$

Need: τ is CRSF. By Separation, it suffices to show: $\mathsf{rk}(x) \leq \mathsf{rk}(y)$ is CRSF.

$$f(x, y) := \Big\{ \; u \in \mathsf{tc}(\{x\}) : u \subseteq \bigcup\{f(x, v) : v \in y\} \; \Big\}$$

is CRSF by Cobham recursion. Proviso: identity embeds into $\{x\}$.

Then: $\mathsf{rk}(x) \leq \mathsf{rk}(y) \iff x \in f(x, y)$. $\square$

Normal form for bounds

Transitivity if $\tau_0 : x \preccurlyeq y$ and $\tau_1 : y \preccurlyeq z$, then $\sigma : x \preccurlyeq z$ for some σ.

Normal form for bounds

Transitivity if $\tau_0 : x \preccurlyeq y$ and $\tau_1 : y \preccurlyeq z$, then $\sigma : x \preccurlyeq z$ for some σ.

Monotonicity Let $t(y, ..)$ be a #-term. Then

if $\tau_0 : x \preccurlyeq t(y, ..)$ and $\tau_1 : y \preccurlyeq z$, then $\sigma : x \preccurlyeq t(z, ..)$ for some σ.

Normal form for bounds

Transitivity if $\tau_0 : x \preccurlyeq y$ and $\tau_1 : y \preccurlyeq z$, then $\sigma : x \preccurlyeq z$ for some σ.

Monotonicity Let $t(y, ..)$ be a #-term. Then

if $\tau_0 : x \preccurlyeq t(y, ..)$ and $\tau_1 : y \preccurlyeq z$, then $\sigma : x \preccurlyeq t(z, ..)$ for some σ.

Bounding Theorem

if $f(\vec{x})$ is CRSF, then $\tau(\cdot, \vec{x}) : f(\vec{x}) \preccurlyeq t(\vec{x})$ for some #-term $t(\vec{x})$, τ in CRSF.

Normal form for bounds

Transitivity if $\tau_0 : x \preccurlyeq y$ and $\tau_1 : y \preccurlyeq z$, then $\sigma : x \preccurlyeq z$ for some σ.

Monotonicity Let $t(y, ..)$ be a #-term. Then

if $\tau_0 : x \preccurlyeq t(y, ..)$ and $\tau_1 : y \preccurlyeq z$, then $\sigma : x \preccurlyeq t(z, ..)$ for some σ.

Bounding Theorem

if $f(\vec{x})$ is CRSF, then $\tau(\cdot, \vec{x}) : f(\vec{x}) \preccurlyeq t(\vec{x})$ for some #-term $t(\vec{x})$, τ in CRSF.

Example Neither $x \mapsto P(x)$ nor $x \mapsto |x|$ is CRSF.

Proof: if $f(x)$ is CRSF then
rk$(f(x))$ is polynomially bounded in rk(x), and
$|\text{tc}(f(x))|$ is polynomially bounded in $|\text{tc}(x)|$. $\square$

Closure results

Unbounded replacement

if g is CRSF, then so is

$$f(x) = \{g(u, x) : u \in x\}.$$

Closure results

Unbounded replacement

if g is CRSF, then so is

$$f(x) = \{g(u, x) : u \in x\}.$$

Example $x \times y$ is CRSF.

Proof: two applications of unbounded replacement:

$$x \times y := \bigcup\{\{u\} \times y : u \in x\}$$

$$\{u\} \times y := \{\langle u, v\rangle : v \in y\} \qquad \square$$

Closure results

Unbounded replacement

if g is CRSF, then so is

$$f(x) = \{g(u, x) : u \in x\}.$$

Example $x \times y$ is CRSF.

Proof: two applications of unbounded replacement:

$$x \times y := \bigcup\{\{u\} \times y : u \in x\}$$

$$\{u\} \times y := \{\langle u, v\rangle : v \in y\} \qquad \square$$

Course of values recursion

if g, τ, t are CRSF, then so is

$$f(x) := g(\{\langle u, f(u)\rangle : u \in \mathrm{tc}(x)\}, x)$$

provided $\tau(\cdot, x) : f(x) \preccurlyeq t(x)$ for all x.

CRSF and PTIME

view PTIME as a class of functions on binary strings $b_0 \cdots b_{n-1} \in \{0,1\}^*$.

Code by sets: $\nu(b_0 \cdots b_{n-1}) := \{i < n : b_i = 1\} \cup \{n\}$.

$F : V \to V$ represents $f : \{0,1\}^* \to \{0,1\}^*$ if

$$F(\nu(b_0 \cdots b_{n-1})) = \nu(f(b_0 \cdots b_{n-1}))$$

CRSF and PTIME

view PTIME as a class of functions on binary strings $b_0 \cdots b_{n-1} \in \{0,1\}^*$.

Code by sets: $\nu(b_0 \cdots b_{n-1}) := \{i < n : b_i = 1\} \cup \{n\}$.

$F : V \to V$ represents $f : \{0,1\}^* \to \{0,1\}^*$ if

$$F(\nu(b_0 \cdots b_{n-1})) = \nu(f(b_0 \cdots b_{n-1}))$$

Theorem

A string function is PTIME iff it is represented by some CRSF function.

CRSF and PTIME

view PTIME as a class of functions on binary strings $b_0 \cdots b_{n-1} \in \{0,1\}^*$.

Code by sets: $\nu(b_0 \cdots b_{n-1}) := \{i < n : b_i = 1\} \cup \{n\}$.

$F : V \to V$ represents $f : \{0,1\}^* \to \{0,1\}^*$ if
$F(\nu(b_0 \cdots b_{n-1})) = \nu(f(b_0 \cdots b_{n-1}))$

Theorem

A string function is PTIME iff it is represented by some CRSF function.

Proof idea $\subseteq$*:* simulate limited recursion on notation in CRSF.

Proof idea $\supseteq$*:* let $F(x)$ CRSF. Restrict to hereditarily finite x. Show
Mostowski graph of $x \mapsto$ Mostowski graph of $F(x)$
is PTIME.

Related work

Bellantoni-Cook 1992 recursion theoretic characterization of PTIME

idea: arguments have two sorts, normal and safe:

$f(0, \vec{x}/\vec{y}) = h(\vec{x}/\vec{y})$

$f(s_b(x), \vec{x}/\vec{y}) = g_b(x, \vec{x}/f(x, \vec{x}/\vec{y}), \vec{y}) \qquad b \in \{0, 1\}$

The derivable functions $f(\vec{x}/)$ are precisely PTIME.

Related work

Bellantoni-Cook 1992 recursion theoretic characterization of PTIME

idea: arguments have two sorts, normal and safe:

$f(0, \vec{x}/\vec{y}) = h(\vec{x}/\vec{y})$

$f(s_b(x), \vec{x}/\vec{y}) = g_b(x, \vec{x}/f(x, \vec{x}/\vec{y}), \vec{y}) \qquad b \in \{0, 1\}$

The derivable functions $f(\vec{x}/)$ are precisely PTIME.

Beckmann, Buss, Friedman 2015 define SRSF: safe recursive set functions

on HF, SRSF is alternating exponential time with poly alternations.

Related work

Bellantoni-Cook 1992 recursion theoretic characterization of PTIME

idea: arguments have two sorts, normal and safe:

$f(0, \vec{x}/\vec{y}) = h(\vec{x}/\vec{y})$

$f(s_b(x), \vec{x}/\vec{y}) = g_b(x, \vec{x}/f(x, \vec{x}/\vec{y}), \vec{y}) \qquad b \in \{0, 1\}$

The derivable functions $f(\vec{x}/)$ are precisely PTIME.

Beckmann, Buss, Friedman 2015 define SRSF: safe recursive set functions

on HF, SRSF is alternating exponential time with poly alternations.

Arai 2015 defines PCSF: predicatively computable set functions.

SRSF $\supsetneq$ PCSF$^+$ $\supseteq$ PCSF is PTIME on HF

Related work

Bellantoni-Cook 1992 recursion theoretic characterization of PTIME

idea: arguments have two sorts, normal and safe:

$f(0, \vec{x}/\vec{y}) = h(\vec{x}/\vec{y})$

$f(s_b(x), \vec{x}/\vec{y}) = g_b(x, \vec{x}/f(x, \vec{x}/\vec{y}), \vec{y}) \qquad b \in \{0, 1\}$

The derivable functions $f(\vec{x}/)$ are precisely PTIME.

Beckmann, Buss, Friedman 2015 define SRSF: safe recursive set functions on HF, SRSF is alternating exponential time with poly alternations.

Arai 2015 defines PCSF: predicatively computable set functions.

SRSF $\supsetneq$ PCSF$^+$ $\supseteq$ PCSF is PTIME on HF

Theorem The functions $f(\vec{x}/)$ in PCSF$^+$ are precisely CRSF.

Definability theoretic view

α-recursion theory: computable means Σ_1-definable on an admissible set.

An admissible set is a model of KP:

language $\{\in\}$

Extensionality $\forall u(u \in x \leftrightarrow u \in y) \rightarrow x = y$

Pair $\exists z \forall u(u \in z \leftrightarrow u = x \vee u = y)$

Union $\exists z \forall u(u \in z \leftrightarrow \exists v \in x \ u \in v)$

Δ_0-Separation $\exists z \forall u(u \in z \leftrightarrow u \in x \wedge \varphi(u, \vec{x}))$ for $\varphi \in \Delta_0$

Δ_0-Collection $\forall u \in x \exists v \ \varphi(u, v, \vec{x}) \rightarrow \exists z \forall u \in x \exists v \in z \ \varphi(u, v, \vec{x})$ for $\varphi \in \Delta_0$

Class Induction $\forall y(\forall u \in y \ \varphi(u, \vec{x}) \rightarrow \varphi(y, \vec{x})) \rightarrow \varphi(x, \vec{x})$ for all φ

Definability theoretic view

α-recursion theory: computable means Σ_1-definable on an admissible set.

An admissible set is a model of KP:

language $\{\in, \{\cdot,\cdot\}, \bigcup \cdot\}$

Extensionality $(\forall u(u \in x \leftrightarrow u \in y) \rightarrow x = y)$

Pair $(u \in \{x, y\} \leftrightarrow u = x \vee u = y)$ defining axiom

Union $(u \in \bigcup x \leftrightarrow \exists v \in x\ u \in v)$ defining axiom

Δ_0-Separation $\exists z \forall u(u \in z \leftrightarrow u \in x \wedge \varphi(u, \vec{x}))$ for $\varphi \in \Delta_0$

Δ_0-Collection $\forall u \in x \exists v\ \varphi(u, v, \vec{x}) \rightarrow \exists z \forall u \in x \exists v \in z\ \varphi(u, v, \vec{x})$ for $\varphi \in \Delta_0$

Class Induction $\forall y(\forall u \in y\ \varphi(u, \vec{x}) \rightarrow \varphi(y, \vec{x})) \rightarrow \varphi(x, \vec{x})$ for all φ

Theories for primitive recursive computation

KP_1 is KP with Induction restricted to Σ_1-formulas φ.

$f(\vec{x})$ is Σ_1-definable in KP_1 if there is a Σ_1-formula $\varphi(\vec{x}, y)$ such that

$V \models \forall \vec{x}\ \varphi(\vec{x}, f(\vec{x}))$

$\mathsf{KP}_1 \vdash \forall \vec{x}\ \exists^{=1} y\ \varphi(\vec{x}, y)$

Theories for primitive recursive computation

KP_1 is KP with Induction restricted to Σ_1-formulas φ.

$f(\vec{x})$ is Σ_1-definable in KP_1 if there is a Σ_1-formula $\varphi(\vec{x}, y)$ such that

$V \models \forall \vec{x}\ \varphi(\vec{x}, f(\vec{x}))$

$\mathsf{KP}_1 \vdash \forall \vec{x}\ \exists^{=1} y\ \varphi(\vec{x}, y)$

Rathjen 1992

A function is PRSF iff it is Σ_1-definable in KP_1.

Theories for primitive recursive computation

KP_1 is KP with Induction restricted to Σ_1-formulas φ.

$f(\vec{x})$ is Σ_1-definable in KP_1 if there is a Σ_1-formula $\varphi(\vec{x}, y)$ such that

$V \models \forall \vec{x}\ \varphi(\vec{x}, f(\vec{x}))$

$\mathsf{KP}_1 \vdash \forall \vec{x}\ \exists^{=1} y\ \varphi(\vec{x}, y)$

Rathjen 1992

A function is PRSF iff it is Σ_1-definable in KP_1.

Parsons 1970

A function over $\mathbb{N}$ is primitive recursive iff it is Σ_1-definable in $I\Sigma_1$.

$I\Sigma_1$ is Robinson's Q plus

$\varphi(0, \vec{x}) \wedge \forall y(\varphi(y, \vec{x}) \to \varphi(y+1, \vec{x})) \to \varphi(x, \vec{x})$ for $\varphi \in \Sigma_1$

Theory for PTIME

Language of arithmetic plus $\lfloor x/2 \rfloor$, $|x| = \lceil \log(x+1) \rceil$, $x\#y = 2^{|x|\cdot|y|}$.

Δ_0^b-formulas have only sharply bounded quantifiers $\exists x \leq |t|, \forall x \leq |t|$

Theory for PTIME

Language of arithmetic plus $\lfloor x/2 \rfloor$, $|x| = \lceil \log(x+1) \rceil$, $x\#y = 2^{|x|\cdot|y|}$.

Δ_0^b-formulas have only sharply bounded quantifiers $\exists x \leq |t|, \forall x \leq |t|$

Σ_1^b-formulas have the form $\exists x \leq t$ Δ_0^b. 'there is x of polynomial length"

Σ_1^b-definable sets (over $\mathbb{N}$) are precisely those in NP.

Theory for PTIME

Language of arithmetic plus $\lfloor x/2 \rfloor$, $|x| = \lceil \log(x+1) \rceil$, $x\#y = 2^{|x|\cdot|y|}$.

Δ_0^b-formulas have only sharply bounded quantifiers $\exists x \leq |t|, \forall x \leq |t|$

Σ_1^b-formulas have the form $\exists x \leq t$ Δ_0^b. 'there is x of polynomial length"

Σ_1^b-definable sets (over $\mathbb{N}$) are precisely those in NP.

S_2^1 contains certain defining axioms for the symbols plus

$\varphi(0, \vec{x}) \wedge \forall y(\varphi(\lfloor y/2 \rfloor, \vec{x}) \rightarrow \varphi(y, \vec{x})) \rightarrow \varphi(x, \vec{x})$ for $\varphi \in \Sigma_1^b$

Theory for PTIME

Language of arithmetic plus $\lfloor x/2 \rfloor$, $|x| = \lceil \log(x+1) \rceil$, $x\#y = 2^{|x|\cdot|y|}$.

Δ_0^b-formulas have only sharply bounded quantifiers $\exists x \leq |t|, \forall x \leq |t|$

Σ_1^b-formulas have the form $\exists x \leq t$ Δ_0^b. 'there is x of polynomial length"

Σ_1^b-definable sets (over $\mathbb{N}$) are precisely those in NP.

S_2^1 contains certain defining axioms for the symbols plus

$$\varphi(0,\vec{x}) \wedge \forall y(\varphi(\lfloor y/2 \rfloor, \vec{x}) \rightarrow \varphi(y,\vec{x})) \rightarrow \varphi(x,\vec{x}) \quad \text{for } \varphi \in \Sigma_1^b$$

Buss 1986 A function is PTIME iff it is Σ_1^b-definable in S_2^1.

Theory for PTIME

Language of arithmetic plus $\lfloor x/2 \rfloor$, $|x| = \lceil \log(x+1) \rceil$, $x\#y = 2^{|x|\cdot|y|}$.

Δ_0^b-formulas have only sharply bounded quantifiers $\exists x \leq |t|, \forall x \leq |t|$

Σ_1^b-formulas have the form $\exists x \leq t$ Δ_0^b. 'there is x of polynomial length"

Σ_1^b-definable sets (over $\mathbb{N}$) are precisely those in NP.

S_2^1 contains certain defining axioms for the symbols plus

$\varphi(0,\vec{x}) \wedge \forall y(\varphi(\lfloor y/2 \rfloor, \vec{x}) \rightarrow \varphi(y,\vec{x})) \rightarrow \varphi(x,\vec{x})$ for $\varphi \in \Sigma_1^b$

Buss 1986 A function is PTIME iff it is Σ_1^b-definable in S_2^1.

Definability S_2^1 proves the defining equations of Cobham.

Witnessing If $S_2^1 \vdash \exists y\ \varphi(\vec{x}, y)$, then $S_2^1 \vdash \varphi(\vec{x}, f(\vec{x}))$ for some $f \in$ PTIME.

Theory $\mathbf{KP}_1^{\prec}$

language $\in,\ 0,\ 1,\ \bigcup x,\ \{x,y\},\ x\times y,\ \mathrm{tc}(x),\ x\odot y,\ x\odot^{-1} y,\ x\#y$

idea sharply bounded quantification corresponds to $\exists x\in t, \forall x\in t$

Theory $\mathbf{KP}_1^{\preccurlyeq}$

language $\in,\ 0,\ 1,\ \bigcup x,\ \{x,y\},\ x\times y,\ \mathrm{tc}(x),\ x\odot y,\ x\odot^{-1} y,\ x\#y$

idea sharply bounded quantification corresponds to $\exists x\in t, \forall x\in t$

Δ_0-formulas have only sharply bounded quantifiers

$\Sigma_1^{\preccurlyeq}$-formulas have the form $\exists x \preccurlyeq t\ \Delta_0$ for t a #-term

Theory $\mathbf{KP}_1^{\preccurlyeq}$

language $\in,\ 0,\ 1,\ \bigcup x,\ \{x,y\},\ x\times y,\ \mathrm{tc}(x),\ x\odot y,\ x\odot^{-1} y,\ x\#y$

idea sharply bounded quantification corresponds to $\exists x\in t, \forall x\in t$

Δ_0-formulas have only sharply bounded quantifiers

$\Sigma_1^{\preccurlyeq}$-formulas have the form $\exists x \preccurlyeq t\ \Delta_0$ for t a #-term

Extensionality $\forall u(u\in x \leftrightarrow u\in y)\rightarrow x=y$

defining axioms for the symbols

Δ_0-Separation $\exists z\forall u(u\in z\leftrightarrow u\in x\wedge\varphi(u,\vec{x}))$ for $\varphi\in\Delta_0$

Δ_0-Collection $\forall u\in x\exists v\ \varphi(u,v,\vec{x})\rightarrow\exists z\forall u\in x\exists v\in z\ \varphi(u,v,\vec{x})$ for $\varphi\in\Delta_0$

$\Sigma_1^{\preccurlyeq}$-Induction $\forall y(\forall u\in y\ \varphi(u,\vec{x})\rightarrow\varphi(y,\vec{x}))\rightarrow\varphi(x,\vec{x})$ for all $\varphi\in\Sigma_1^{\preccurlyeq}$

Theory $\mathbf{KP}_1^{\preccurlyeq}$

language $\in,\ 0,\ 1,\ \bigcup x,\ \{x,y\},\ x\times y,\ \mathrm{tc}(x),\ x\odot y,\ x\odot^{-1} y,\ x\#y$

idea sharply bounded quantification corresponds to $\exists x\in t, \forall x\in t$

Δ_0-formulas have only sharply bounded quantifiers

$\Sigma_1^{\preccurlyeq}$-formulas have the form $\exists x \preccurlyeq t\ \Delta_0$ for t a #-term

Extensionality $\forall u(u\in x \leftrightarrow u\in y)\to x=y$

defining axioms for the symbols

Δ_0-Separation $\exists z\forall u(u\in z\leftrightarrow u\in x\wedge\varphi(u,\vec{x}))$ for $\varphi\in\Delta_0$

Δ_0-Collection $\forall u\in x\exists v\ \varphi(u,v,\vec{x})\to\exists z\forall u\in x\exists v\in z\ \varphi(u,v,\vec{x})$ for $\varphi\in\Delta_0$

$\Sigma_1^{\preccurlyeq}$-Induction $\forall y(\forall u\in y\ \varphi(u,\vec{x})\to\varphi(y,\vec{x}))\to\varphi(x,\vec{x})$ for all $\varphi\in\Sigma_1^{\preccurlyeq}$

Intuition $\mathbf{KP}_1^{\preccurlyeq}$ is to $\mathbf{KP}_1$ as S_2^1 is to $I\Sigma_1$.

Goal prove Definability and Witnessing wrt CRSF.

The formula $x \preccurlyeq y$

is $\exists e\ e : x \preccurlyeq y$ where $e : x \preccurlyeq y$ is

> e is the graph of a (multi-valued) embedding of x into y
> i.a.w, $u \mapsto \{v : \langle u, v\rangle \in e\}$ is an embedding of x into y.

The formula $x \preccurlyeq y$

is $\exists e\ e : x \preccurlyeq y$ where $e : x \preccurlyeq y$ is

e is the graph of a (multi-valued) embedding of x into y
i.a.w, $u \mapsto \{v : \langle u, v\rangle \in e\}$ is an embedding of x into y.

i.e. the Δ_0-formula

$$e \subseteq \mathsf{tc}(x) \times \mathsf{tc}(y)$$

$$\wedge\ \forall u \in \mathsf{tc}(x)\ \exists v \in \mathsf{tc}(y)\ \langle u, v\rangle \in e$$

$$\wedge\ \forall u, u' \in \mathsf{tc}(x)\ \forall v \in \mathsf{tc}(y)\ (u \neq u' \wedge \langle u, v\rangle \in e \to \langle u', v\rangle \notin e)$$

$$\wedge\ \forall u, u' \in \mathsf{tc}(x)\ \forall v' \in \mathsf{tc}(y)\ (u \in u' \wedge \langle u', v'\rangle \in e \to \exists v \in \mathsf{tc}(v')\ \langle u, v\rangle \in e).$$

Definability

$\Sigma_1^{\preccurlyeq}$-expansion of $\mathsf{KP}_1^{\preccurlyeq}$:

Add certain defining axioms $\varphi(\vec{x}, f(\vec{x}))$ for new symbols $f(\vec{x})$ where

$\varphi(\vec{x}, y) \in \Sigma_1^{\preccurlyeq}$, $t(\vec{x})$ #-term

$\mathsf{KP}_1^{\preccurlyeq} \vdash \exists^{\leq 1} y \ \varphi(\vec{x}, y)$

$\mathsf{KP}_1^{\preccurlyeq} \vdash \forall \vec{x} \ \exists y \preccurlyeq t(\vec{x}) \ \varphi(\vec{x}, y)$

Definability

$\Sigma_1^{\preccurlyeq}$-expansion of $\mathsf{KP}_1^{\preccurlyeq}$:

Add certain defining axioms $\varphi(\vec{x}, f(\vec{x}))$ for new symbols $f(\vec{x})$ where

$\varphi(\vec{x}, y) \in \Sigma_1^{\preccurlyeq}$, $t(\vec{x})$ #-term

$\mathsf{KP}_1^{\preccurlyeq} \vdash \exists^{\leq 1} y \ \varphi(\vec{x}, y)$

$\mathsf{KP}_1^{\preccurlyeq} \vdash \forall \vec{x} \ \exists y \preccurlyeq t(\vec{x}) \ \varphi(\vec{x}, y)$

There exists such an expansion $\mathsf{KP}_1^{\preccurlyeq}(L_{\mathsf{crsf}})$ to language L_{crsf} such that

- for all $h(y_1, \ldots, y_r), g_1(\vec{x}), \ldots, g_r(\vec{x}) \in L_{\mathsf{crsf}}$ there is $f(\vec{x}) \in L_{\mathsf{crsf}}$ such that

  $\mathsf{KP}_1^{\preccurlyeq}(L_{\mathsf{crsf}}) \vdash f(\vec{x}) = h(g_1(\vec{x}), \ldots, g_r(\vec{x}))$

Definability

$\Sigma_1^{\preccurlyeq}$-expansion of $\mathsf{KP}_1^{\preccurlyeq}$:

Add certain defining axioms $\varphi(\vec{x}, f(\vec{x}))$ for new symbols $f(\vec{x})$ where

$\varphi(\vec{x}, y) \in \Sigma_1^{\preccurlyeq}$, $t(\vec{x})$ #-term

$\mathsf{KP}_1^{\preccurlyeq} \vdash \exists^{\leq 1} y \ \varphi(\vec{x}, y)$

$\mathsf{KP}_1^{\preccurlyeq} \vdash \forall \vec{x} \ \exists y \preccurlyeq t(\vec{x}) \ \varphi(\vec{x}, y)$

There exists such an expansion $\mathsf{KP}_1^{\preccurlyeq}(L_{\mathsf{crsf}})$ to language L_{crsf} such that

- for all $h(y_1, \ldots, y_r), g_1(\vec{x}), \ldots, g_r(\vec{x}) \in L_{\mathsf{crsf}}$ there is $f(\vec{x}) \in L_{\mathsf{crsf}}$ such that

  $\mathsf{KP}_1^{\preccurlyeq}(L_{\mathsf{crsf}}) \vdash f(\vec{x}) = h(g_1(\vec{x}), \ldots, g_r(\vec{x}))$

- for all $g(z, x), \tau(u, x) \in L_{\mathsf{crsf}}$, #-terms $t(x)$ there is $f(x) \in L_{\mathsf{crsf}}$ such that

  $\mathsf{KP}_1^{\preccurlyeq}(L_{\mathsf{crsf}}) \vdash \tau(\cdot, x) : g(f''(x), x) \preccurlyeq t(x) \to f(x) = g(f''(x), x)$

  where $f''(x) \in L_{\mathsf{crsf}}$ is such that $\mathsf{KP}_1^{\preccurlyeq}(L_{\mathsf{crsf}}) \vdash f''(x) = \{f(y) : y \in x\}$.

Witnessing fails

Witnessing Question if $\varphi(\vec{x}, y) \in \Delta_0$ and

$$\mathsf{KP}_1^{\prec}(L_{\mathsf{crsf}}) \vdash \exists y \ \varphi(\vec{x}, y),$$

then there is a witnessing function $f(\vec{x}) \in L_{\mathsf{crsf}}$ such that

$$\mathsf{KP}_1^{\prec}(L_{\mathsf{crsf}}) \vdash \varphi(\vec{x}, f(\vec{x})) \ ?$$

Witnessing fails

Witnessing Question if $\varphi(\vec{x}, y) \in \Delta_0$ and

$$\mathsf{KP}_1^{\prec}(L_{\mathsf{crsf}}) \vdash \exists y \ \varphi(\vec{x}, y),$$

then there is a witnessing function $f(\vec{x}) \in L_{\mathsf{crsf}}$ such that

$$\mathsf{KP}_1^{\prec}(L_{\mathsf{crsf}}) \vdash \varphi(\vec{x}, f(\vec{x})) \ ?$$

Counterexample

$\mathsf{KP}_1^{\prec} \vdash \exists y(x \neq 0 \to y \in x)$

Witnessing function C satisfies $(x \neq 0 \to C(x) \in x)$.

This Global Choice (GC). It implies (AC), not provable in ZF.

Adding (GC)

$\mathsf{KPC}_1^{\preccurlyeq}$: add (GC) to $\mathsf{KP}_1^{\preccurlyeq}$

Get expansion $\mathsf{KPC}_1^{\preccurlyeq}(L^C_{\mathsf{crsf}}) \supseteq \mathsf{KP}_1^{\preccurlyeq}(L_{\mathsf{crsf}})$.

Definability theorem holds for $\mathsf{KPC}_1^{\preccurlyeq}(L^C_{\mathsf{crsf}})$.

Adding (GC)

$\mathsf{KPC}_1^{\preccurlyeq}$: add (GC) to $\mathsf{KP}_1^{\preccurlyeq}$

Get expansion $\mathsf{KPC}_1^{\preccurlyeq}(L_{\mathsf{crsf}}^C) \supseteq \mathsf{KP}_1^{\preccurlyeq}(L_{\mathsf{crsf}})$.

Definability theorem holds for $\mathsf{KPC}_1^{\preccurlyeq}(L_{\mathsf{crsf}}^C)$.

Witnessing Theorem if $\varphi(\vec{x}, y) \in \Delta_0$ and

$$\mathsf{KPC}_1^{\preccurlyeq}(L_{\mathsf{crsf}}^C) \vdash \exists y\ \varphi(\vec{x}, y),$$

then there is a witnessing function $f(\vec{x}) \in L_{\mathsf{crsf}}^C$ such that

$$\mathsf{KPC}_1^{\preccurlyeq}(L_{\mathsf{crsf}}^C) \vdash \varphi(\vec{x}, f(\vec{x})).$$

Partial conservativity

$\mathsf{KPC}_1^{\preccurlyeq}(L_{\mathsf{crsf}}^C) \supseteq \mathsf{KP}_1^{\preccurlyeq}(L_{\mathsf{crsf}})$

(AC) shows that extension is not conservative.

Partial conservativity

$\mathsf{KPC}_1^{\prec}(L^C_{\mathsf{crsf}}) \supseteq \mathsf{KP}_1^{\prec}(L_{\mathsf{crsf}})$

(AC) shows that extension is not conservative.

Theorem

Assume $\mathsf{KPC}_1^{\prec}(L^C_{\mathsf{crsf}}) \vdash \psi$, an L_{crsf}-formula.

Then ψ is provable in $\mathsf{KP}_1^{\prec}(L_{\mathsf{crsf}})$ plus the following local choice principles:

Partial conservativity

$\mathsf{KPC}_1^{\prec}(L^C_{\mathsf{crsf}}) \supseteq \mathsf{KP}_1^{\prec}(L_{\mathsf{crsf}})$

(AC) shows that extension is not conservative.

Theorem

Assume $\mathsf{KPC}_1^{\prec}(L^C_{\mathsf{crsf}}) \vdash \psi$, an L_{crsf}-formula.

Then ψ is provable in $\mathsf{KP}_1^{\prec}(L_{\mathsf{crsf}})$ plus the following local choice principles:

Well-Ordering

$$\forall x \exists \alpha \exists y \ (\ y : \alpha \xrightarrow{\text{bij}} x \ \wedge \ \forall \beta, \gamma \in \alpha \ (y(\beta) \in y(\gamma) \to \beta \in \gamma)\ )$$

Δ_0-Dependent Choice for $\varphi \in \Delta_0$

$$\forall x \exists y \varphi(x, y, \vec{x}) \to \forall \alpha \ \exists z \ (\mathsf{Fct}(z) \wedge \mathsf{dom}(z) = \alpha \wedge \forall \beta \in \alpha \ \varphi(z \restriction \beta, z(\beta), \vec{x}))$$

xiexie