A New Kind of Computability Theory on Reals

IMS-JSPS Joint Workshop in Mathematical Logic and Foundations of Mathematics

Longyun Ding

School of Mathematical Sciences
Nankai University

Sep 2nd, 2014
NUS, Singapore

Outline

Ordinary computability

Definition (Basic recursive functions)

(1) Zero function: $Z(n) = 0$,

(2) Successor function: $S(n) = n + 1$,

(3) Projection functions: $U_j^{(m)}(n_1, \cdots, n_m) = x_j$.

Definition (Recursive functions)

The minimal class of functions contains all basic recursive functions and closed under:

(1) Composition,

(2) Primitive recursion,

(3) Minimalization.

Unlimited register machine (URM)

Registers: $R_1, R_2, \cdots, R_k, \cdots$.
Program: a sequence of instructions $I_1, I_2, \cdots, I_s$.
Instruction:

(1) Zero(k): $R_k \leftarrow 0$,

(2) Succ(k): $R_k \leftarrow r_k + 1$ (for r_k stored in R_k),

(3) Copy(k_1, k_2): $R_{k_2} \leftarrow r_{k_1}$,

(4) Jump(k_1, k_2, j): if $r_{k_1} = r_{k_2}$, go to instruction number j, otherwise go to the next instruction.

Computation on ω^ω—Oracle URM

Instruction: Zero, Succ, Copy, Jump and

(5) Alpha(k): $R_k \leftarrow \alpha(r_k)$.

Given $\alpha \in \omega^\omega$ and $n \in \omega$, machine M outputs m. We say M computes $f_M : \omega^\omega \to \omega^\omega$ with

$$f_M(\alpha)(n) = m.$$

Fact

f_M is continuous.

BSS (Blum-Shub-Smale) machine on $\mathbb{R}$

(Sketch) A flowchart with:

(1) Exactly on input node. Having no incoming edge, and one outgoing edge. With a polynomial $\mathbb{R}^n \to \mathbb{R}^l$.

(2) At least one output node. Having no outgoing edge. With a polynomial $\mathbb{R}^l \to \mathbb{R}^m$.

(3) Computing node. Having at least one incoming edge, and one outgoing edge. With a polynomial $\mathbb{R}^l \to \mathbb{R}^l$.

(4) Branch node. Having at least one incoming edge, and two outgoing edges. With a polynomial $h : \mathbb{R}^l \to \mathbb{R}$ for determining which outgoing edge to go, one for $h > 0$ and another for $h \leq 0$.

Fact

If $f : \mathbb{R}^n \to \mathbb{R}^m$ is BSS computable, then $\mathbb{R}^n$ can be split into countably many pieces, and on every piece, f is a polynomial.

Generalized BSS machine (Kfufi-Stolboushkin-Uzhichin)

Let $\mathfrak{A}$ be a model on language L with domain A. A flowchart with:

(1) Exactly on input node. With an item sequence $(t_1, \cdots, t_l)$ with variables in $x_1, \cdots, x_m$.

(2) Output node. With an item sequence $(t_1, \cdots, t_n)$ with variables in $x_1, \cdots, x_l$.

(3) Computing node. With an item sequence $(t_1, \cdots, t_l)$ with variables in $x_1, \cdots, x_l$.

(4) Branch node. with an atomic formula $\psi(x_1, \cdots, x_l)$.

For original BSS machine, we take $A = \mathbb{R}$ and language

$$L = (+, \cdot, \geq 0, c_r)_{r \in \mathbb{R}}.$$

URM on a model

Instruction: Copy, Jump, and

(3) $\text{Const}_i(k)$: $R_k \leftarrow c_i$ where c_i is the explanation of a constant symbol.

(4) $\text{Func}_i(k_1, \cdots, k_n; k)$: $R_k \leftarrow f_i(r_{k_1}, \cdots, r_{k_n})$ where f_i is the explanation of a function symbol.

(5) $\text{Pred}_i(k_1, \cdots, k_n; j)$: Let X_i be the explanation of a predicate symbol. If $X_i(r_{k_1}, \cdots, r_(k_n))$ be true, go to instruction number j.

Theorem

For any model $\mathfrak{A}$ with domain A, a function $f : A^m \to A^n$ is generalized BSS machine computable iff it is URM computable.

Outline

Computational structure

Definition

We say $\mathfrak{B} = (B, S, f_i)_{i \in I}$ is a computational structure if

1. $\omega \subseteq B$,
2. $S(n) = n + 1$ for $n \in \omega$,
3. $f_i = f_i^{(m_i)}$ be an m_i-ary partial function on B.

If $\mathfrak{A}$ is a model with domain A, we can put $B = \omega \cup A$, replace each constant by a constant function, and replace each predicate by its character function.

$\mathfrak{B}$-recursive functions

Definition

The minimal class of functions contains all f_i, S, Zero function, and Projection functions, and closed under:

(1) Composition,

(2) Primitive recursion with respect on ω,

(3) Minimalization: $f(x) = \mu n[h(n, x) = 0]$ where $h(n, x) \in \omega$ for all $n \in \omega, x \in B$.

$\mathfrak{B}$-URM

Instruction: Zero, Succ, copy, Jump, and

(5) $\mathrm{Func}_i(k_1, \cdots, k_{m_i}; k)$: $R_k \leftarrow f_i(r_{k_1}, \cdots, r_{k_{m_i}})$.

Theorem

All $\mathfrak{B}$-recursive functions are $\mathfrak{B}$-URM computable.

Codable computational structures

Definition

If there are $\mathfrak{B}$-recursive functions $c : B^2 \to B$ and $l, r : B \to B$ such that $l(c(x, y)) = x$ and $r(c(x, y)) = y$, we say $\mathfrak{B}$ is codable.

Theorem

If $\mathfrak{B}$ is codable, all $\mathfrak{B}$-URM computable functions are $\mathfrak{B}$-recursive.

Theorem

If $\mathfrak{B}$ is codable, there exists an universal $\mathfrak{B}$-recursive function $U(y, x)$ such that, for any $\mathfrak{B}$-recursive function $f(x)$, there is $y_0 \in B$,

$$U(y_0, x) = f(x), \quad (\forall x \in B).$$

Furthermore, if there are only countably many f_i in $\mathfrak{B}$, then y_0 can always be in ω.

Type 2 computational structures

$\mathfrak{pf}B$ be the set of all partial functions $\alpha : \mathrm{dom}(f) \subseteq B \to B$.

Definition

Let $\omega \subseteq B$, $C = B \cup \mathfrak{pf}B$. For $i \in I$, $f_i = f_i^{(m_i,n_i)}$ is from $B^{m_i} \times (\mathfrak{pf}B)^{n_i}$ to B. We say $\mathfrak{C} = (C, S, f_i)_{i\in I}$ is a type 2 computational structure.

Two Zero functions: $Z(x) = 0$ and $Z^2(\alpha) = 0$ for $x \in B$ and $\alpha \in \mathfrak{pf}B$.
Two kind of Projections:

$$U_j^{(m,n)}(x_1, \cdots, x_m, \alpha_1, \cdots, \alpha_n) = x_j,$$

$$U_{j,k}^{(m,n)}(x_1, \cdots, x_m, \alpha_1, \cdots, \alpha_n) = \alpha_k(x_j).$$

$\mathfrak{C}$-recursive functions

If $q : B^2 \to B$, then $\lambda zq : B \to \mathfrak{pf}B$ as

$$\lambda zq(z, x)(y) = q(y, x).$$

Definition ($\mathfrak{C}$-recursive functions)

The minimal class of functions contains all f_i, S, two Zero functions, and two kind of Projection functions, and closed under

1. Composition, such as $f(x) = h(g(x), \lambda zq(z, x))$.
2. Primitive recursion with respect on ω, such as

$$\begin{cases} f(0, x) = g(x), \\ f(n + 1, x) = h(n, x, \lambda zf(n, z)). \end{cases}$$

3. Minimalization.

$\mathfrak{C}$-URM

Registers: countably many type 0 registers for $x \in B$ and countably many type 1 registers for $\alpha \in \mathfrak{pf}B$. Instruction: Zero, Succ, Copy, Jump, all involved only type 0 registers;

(5) $\mathrm{Func}_i(l_1, \cdots, l_{m_i}, k_1, \cdots, k_{n_i}; k)$: $R_k \leftarrow f_i(r_{l_1}, \cdots, r_{l_{m_i}}, \gamma_{k_1}, \cdots, \gamma_{k_{n_i}})$.

(6) $\mathrm{Value}(k_1, k_2; k)$: $R_k \leftarrow \gamma_{k_2}(r_{k_1})$.

(7) $\mathrm{Call}\{P\}(l_1, \cdots, l_m, k_1, \cdots, k_n; k)$:
$R^1_k \leftarrow \lambda z f_P(z, r_{l_1}, \cdots, r_{l_m}, \gamma_{k_1}, \cdots, \gamma_{k_n})$, where f_P is computed by program P.

Codable

Theorem

Assume $\mathfrak{C}$ is codable. Then f is $\mathfrak{C}$-recursive iff it is $\mathfrak{C}$-URM computable. And there exist universal $\mathfrak{C}$-recursive functions.

higher type computability theory

Omit.

Outline

E^2 and F^2

Assume $B = \omega$, then $C = \omega \cup \mathfrak{pf}\omega$.

$$E^2(\alpha) = \begin{cases} 0, & \alpha \in \omega^\omega \& \forall n(\alpha(n) = 0), \\ 1, & \alpha \in \omega^\omega \& \exists n(\alpha(n) > 0), \\ \uparrow, & \alpha \notin \omega^\omega. \end{cases}$$

$$F^2(\alpha) = \begin{cases} 0, & \forall n(\alpha(n) = 0 \text{ or } \uparrow), \\ 1, & \exists n(\alpha(n) > 0). \end{cases}$$

Fact

$E^2 = F^2 \upharpoonright \omega^\omega$.

We will consider $\mathfrak{C}_1 = (C, S, E^2)$ and $\mathfrak{C}_2 = (C, S, F^2)$.

Call levels

Definition (Call level of programs)

If no Call instruction appeared in M, then $\mathrm{Level}(M) = 0$, otherwise,

$$\mathrm{Level}(M) = 1 + \max\{\mathrm{Level}(P) : \mathrm{Call}(P) \text{ appeared in } M\}.$$

Fact

$f : \omega^\omega \to \omega$ is Level 0 computable iff it is Oracle URM computable, iff f is Σ^0_1-recursive, i.e., $\mathrm{Graph}(f) = \{(x, n) \in \omega^\omega \times \omega : f(x) = n\}$ is Σ^0_1.

Fact

$f : \omega^\omega \to \omega$ is Level 1 computable in E^2 iff it is M-S machine computable.

Computable in F^2 and E^2

Theorem

$f : \omega^\omega \to \omega$ is Level n computable in F^2 iff it is Σ^0_{n+1}-recursive, i.e., $\mathrm{Graph}(f)$ is Σ^0_{n+1}. Particularly, if f is total, $\mathrm{Graph}(f)$ is Δ^0_{n+1}.

Theorem

$f : \omega^\omega \to \omega$ is Level n computable in E^2 iff there are a Σ^0_{n+1}-recursive function g and a Π^0_{n+1} subset D of ω^ω such that $f = g \upharpoonright D$. Particularly, if f is total, $\mathrm{Graph}(f)$ is Δ^0_{n+1}.

M-S machine computable

Corollary

$f : \omega^\omega \to \omega$ is M-S machine computable iff $f = g \upharpoonright D$ where g is Σ^0_2-recursive and D is Π^0_2. Particularly, if f is total, $\mathrm{Graph}(f)$ is Δ^0_2.

Remark: If we use a type 1 register as output register, the type 2 URM machine can compute a function from $\omega^\omega \to \omega^\omega$.

Theorem (Normal form of M-S machine computable function)

Let $f : \omega^\omega \to \omega^\omega$. f is M-S machine computable iff

$$f(\alpha) = h(\alpha, \mu n[R(\alpha, n)]) \upharpoonright D,$$

in which h is Σ^0_1 recursive, R is Δ^0_2, and D is Π^0_2.

Corollaries

Theorem

χ_A is M-S machine computable iff A is Δ^0_2.

Theorem

If f is M-S machine computable, then $\mathrm{dom}(f) = D_1 \cap D_2$ where D_1 is Σ^0_2 and D_2 is Π^0_2.

Theorem

If f is M-S machine computable and $\mathrm{dom}(f)$ is Π^0_2, then there is a dense open subset U of $\mathrm{dom}(f)$ such that $f \upharpoonright U$ is continuous.

The end

Thank you!