Computability Theory of Function Composition

Joey Lakerdas-Gayle

University of Waterloo

2025 CMS Winter Meeting
Logic in Canada IV
December 8, 2025

Setting: Partial functions

$$\begin{aligned}\textsc{Part} &= \{\text{partial functions } f \text{ from } \mathbb{N} \text{ to } \mathbb{N}\} \\ &= \left\{f : n \mapsto \Phi_e^X(n) \mid e \in \mathbb{N}, X \subseteq \mathbb{N}\right\}\end{aligned}$$

Setting: Partial functions

$$\begin{aligned}\textsc{Part} &= \{\text{partial functions } f \text{ from } \mathbb{N} \text{ to } \mathbb{N}\} \\ &= \left\{f : n \mapsto \Phi_e^X(n) \mid e \in \mathbb{N}, X \subseteq \mathbb{N}\right\}\end{aligned}$$

Everywhere, identify sets/oracles $X \subseteq \mathbb{N}$ with their characteristic function $\chi_X : \mathbb{N} \to \{0, 1\}$ so that $\mathcal{P}(\mathbb{N}) \subseteq \textsc{Part}$.

Setting: Partial functions

$$\begin{aligned}\textsc{Part} &= \{\text{partial functions } f \text{ from } \mathbb{N} \text{ to } \mathbb{N}\}\\ &= \left\{f : n \mapsto \Phi_e^X(n) \mid e \in \mathbb{N}, X \subseteq \mathbb{N}\right\}\end{aligned}$$

Everywhere, identify sets/oracles $X \subseteq \mathbb{N}$ with their characteristic function $\chi_X : \mathbb{N} \to \{0, 1\}$ so that $\mathcal{P}(\mathbb{N}) \subseteq \textsc{Part}$.

P-reducibility on $\textsc{Part}$ [see Cooper [1]]

A function f is *partial reducible* to a function g (written $f \leq_P g$) if the following equivalent conditions hold:

1. For all $X \subseteq \mathbb{N}$, if g is X-computable, then f is X-computable.
2. f can be computed using a "black box" computation of g.

Setting: Partial functions

$$\begin{aligned}\textsc{Part} &= \{\text{partial functions } f \text{ from } \mathbb{N} \text{ to } \mathbb{N}\}\\ &= \left\{f : n \mapsto \Phi_e^X(n) \mid e \in \mathbb{N}, X \subseteq \mathbb{N}\right\}\end{aligned}$$

Everywhere, identify sets/oracles $X \subseteq \mathbb{N}$ with their characteristic function $\chi_X : \mathbb{N} \to \{0, 1\}$ so that $\mathcal{P}(\mathbb{N}) \subseteq \textsc{Part}$.

P-reducibility on $\textsc{Part}$ [see Cooper [1]]

A function f is *partial reducible* to a function g (written $f \leq_P g$) if the following equivalent conditions hold:

1. For all $X \subseteq \mathbb{N}$, if g is X-computable, then f is X-computable.
2. f can be computed using a "black box" computation of g.

Also,

- $f \equiv_P g$ iff $f \leq_P g$ and $g \leq_P f$
- the equivalence classes of $\equiv_P$ are called *partial degrees*, $\mathcal{D}_P$

Setting: Partial functions

Partial reducibility ($\leq_P$) is the natural extension of Turing reducibility ($\leq_T$) to partial functions:

- $\leq_T$ is only defined on $\mathrm{PART} \times \mathcal{P}(\mathbb{N}) \subset \mathrm{PART}^2$.
- If $f \in \mathrm{PART}$ and $X \in \mathcal{P}(\mathbb{N})$, then $f \leq_T X \iff f \leq_P X$.

Function composition

How hard is it to compute $h \in \textsc{Part}$ **via composition**?

Function composition

How hard is it to compute $h \in \textsc{Part}$ **via composition**?

Definition (LG.)

For $h \in \textsc{Part}$,

$\mathrm{sp}(h) := \{(\mathbf{a}, \mathbf{b}) : \exists f \leq_P \mathbf{a},\ \exists g \leq_P \mathbf{b},\ g \circ f = h\} \subseteq \mathcal{D}_P^2$.

I.e., $(\mathbf{a}, \mathbf{b}) \in \mathrm{sp}(h)$ means that we can split a computation for h into an **a**-computable "preprocessing" step, followed by a **b**-computable "postprocessing" step.

Splitting reducibility

Recall that

$$\mathrm{sp}(h) = \{(\mathbf{a}, \mathbf{b}) : \exists f \leq_P \mathbf{a},\ \exists g \leq_P \mathbf{b},\ g \circ f = h\}$$

I.e., how hard it is to compute h via composition.

Splitting reducibility

Recall that

$$\mathrm{sp}(h) = \{(\mathbf{a}, \mathbf{b}) : \exists f \leq_P \mathbf{a},\ \exists g \leq_P \mathbf{b},\ g \circ f = h\}$$

I.e., how hard it is to compute h via composition.

Definition (LG.)

For $h_1, h_2 \in \textsc{Part}$,

$$h_1 \leq_{sp} h_2 \iff \mathrm{sp}(h_1) \supseteq \mathrm{sp}(h_2)$$

I.e., any pair of oracles that can compute functions that compose to h_2 can also compute functions that compose to h_1.

Equivalently, $h_1 \leq_{sp} h_2$ iff whenever $h_2 = g_2 \circ f_2$, there are $f_1 \leq_P f_2$ and $g_1 \leq_P g_2$ for which $h_1 = g_1 \circ f_1$.

Splitting reducibility

Proposition (LG.)

For every c.e. set A, there are c.e. sets $X, Y \subset \mathbb{N}$ such that $A \leq_T X \oplus Y$, but for all $f \leq_T X$ and $g \leq_T Y$, $g \circ f \neq A$.

Splitting reducibility

Proposition (LG.)

For every c.e. set A, there are c.e. sets $X, Y \subset \mathbb{N}$ such that $A \leq_T X \oplus Y$, but for all $f \leq_T X$ and $g \leq_T Y$, $g \circ f \neq A$.

Using $\leq_{sp}$ and writing $B = X \oplus Y$:

Corollary

For every c.e. set A, there is a c.e. set B such that $A \leq_T B$, but $A \not\leq_{sp} B$.

Splitting reducibility

Proposition (LG.)

For every c.e. set A, there are c.e. sets $X, Y \subset \mathbb{N}$ such that $A \leq_T X \oplus Y$, but for all $f \leq_T X$ and $g \leq_T Y$, $g \circ f \neq A$.

Using $\leq_{sp}$ and writing $B = X \oplus Y$:

Corollary

For every c.e. set A, there is a c.e. set B such that $A \leq_T B$, but $A \not\leq_{sp} B$.

Taking $A = \emptyset'$:

Corollary

There is a c.e. $B \equiv_T \emptyset'$ with $\emptyset' \not\leq_{sp} B$.

So computing via composition is not captured by $\leq_T$ (or $\leq_P$).

Splitting reducibility

Observation

$\leq_{sp}$ is a finer reducibility than $\leq_P$. I.e., $h_1 \leq_{sp} h_2 \implies h_1 \leq_P h_2$.

Proof.

Suppose $h_1 \leq_{sp} h_2$. Since $\mathrm{id}_{\mathbb{N}} \circ h_2 = h_2$, $(\deg_P(h_2), \mathbf{0}) \in \mathrm{sp}(h_2) \subseteq \mathrm{sp}(h_1)$. So there is $f \leq_P h_2$ and $g \leq_P \mathbf{0}$ with $g \circ f = h_1$, meaning that $h_1 \leq_P h_2$. $\square$

Splitting reducibility

Observation

$\leq_{sp}$ is a finer reducibility than $\leq_P$. I.e., $h_1 \leq_{sp} h_2 \implies h_1 \leq_P h_2$.

Proof.

Suppose $h_1 \leq_{sp} h_2$. Since $\mathrm{id}_{\mathbb{N}} \circ h_2 = h_2$, $(\deg_P(h_2), \mathbf{0}) \in \mathrm{sp}(h_2) \subseteq \mathrm{sp}(h_1)$. So there is $f \leq_P h_2$ and $g \leq_P \mathbf{0}$ with $g \circ f = h_1$, meaning that $h_1 \leq_P h_2$. $\square$

So every *sp*-degree is contained in a *P*-degree.

Question

For a partial degree $\mathbf{a}$*, what can we say about* $\mathbf{a}/\equiv_{sp}$*?*

From earlier, there is a c.e. $B \equiv_T \emptyset'$ with $\emptyset' \not\leq_{sp} B$. So $\mathbf{0}'_P/\equiv_{sp}$ has at least two distinct elements.

Splitting reducibility

Observation

$\leq_{sp}$ is a finer reducibility than $\leq_P$. I.e., $h_1 \leq_{sp} h_2 \implies h_1 \leq_P h_2$.

Proof.

Suppose $h_1 \leq_{sp} h_2$. Since $\mathrm{id}_{\mathbb{N}} \circ h_2 = h_2$, $(\deg_P(h_2), \mathbf{0}) \in \mathrm{sp}(h_2) \subseteq \mathrm{sp}(h_1)$. So there is $f \leq_P h_2$ and $g \leq_P \mathbf{0}$ with $g \circ f = h_1$, meaning that $h_1 \leq_P h_2$. $\square$

So every *sp*-degree is contained in a P-degree.

Question

For a partial degree $\mathbf{a}$, *what can we say about* $\mathbf{a}/\equiv_{sp}$*?*

From earlier, there is a c.e. $B \equiv_T \emptyset'$ with $\emptyset' \not\leq_{sp} B$. So $\mathbf{0}'_P/\equiv_{sp}$ has at least two distinct elements.

In fact, we will see that $\mathbf{0}'_P/\equiv_{sp}$ has a **greatest** element.

Uniform computable categoricity

Question: How hard is it to compute isomorphisms between arbitrary computable copies of $(\mathbb{N}, S)$?

I.e. copies $(\mathbb{N}, S_0) \cong (\mathbb{N}, S)$ where $S_0 : \mathbb{N} \to \mathbb{N}$ is computable.

Uniform computable categoricity

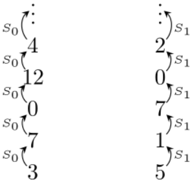
Question: How hard is it to compute isomorphisms between arbitrary computable copies of $(\mathbb{N}, S)$?

I.e. copies $(\mathbb{N}, S_0) \cong (\mathbb{N}, S)$ where $S_0 : \mathbb{N} \to \mathbb{N}$ is computable.

(Partial) Answer: There are always computable isomorphisms, but you need at least $\emptyset'$ to find them!

Proof (sketch).

Use $\emptyset'$ to find the "minimum" element of both copies. Then use the minimum elements and the two computable successor relations to compute the isomorphism. $\square$



Uniform computable categoricity

Question: How hard is it to compute isomorphisms between arbitrary computable copies of $(\mathbb{N}, S)$?
I.e. copies $(\mathbb{N}, S_0) \cong (\mathbb{N}, S)$ where $S_0 : \mathbb{N} \to \mathbb{N}$ is computable.

(Partial) Answer: There are always computable isomorphisms, but you need at least $\emptyset'$ to find them!

Also: You can also *computably* find $\emptyset'$-computable isomorphisms.

Fact

Let $\mathcal{S} \subset \textsc{Part}$ *be the functions that find minimums for* $(\mathbb{N}, S)$.

Uniform computable categoricity

Question: How hard is it to compute isomorphisms between arbitrary computable copies of $(\mathbb{N}, S)$?
I.e. copies $(\mathbb{N}, S_0) \cong (\mathbb{N}, S)$ where $S_0 : \mathbb{N} \to \mathbb{N}$ is computable.

(Partial) Answer: There are always computable isomorphisms, but you need at least $\emptyset'$ to find them!

Also: You can also *computably* find $\emptyset'$-computable isomorphisms.

Fact

Let $\mathcal{S} \subset \textsc{Part}$ *be the functions that find minimums for* $(\mathbb{N}, S)$.

You can $\mathbf{a}$*-computably find* $\mathbf{b}$*-computable isomorphisms between copies of* $(\mathbb{N}, S)$ *iff* $(\mathbf{a}, \mathbf{b}) \in sp(h)$ *for some* $h \in \mathcal{S}$.

Uniform computable categoricity

Theorem (LG.)

There is a function $\zeta \in \mathcal{S}$ and c.e. sets X, Y such that

1. $X \oplus Y \equiv_P \zeta \equiv_P \emptyset'$,

Uniform computable categoricity

Theorem (LG.)

There is a function $\zeta \in \mathcal{S}$ and c.e. sets X, Y such that

1. $X \oplus Y \equiv_P \zeta \equiv_P \emptyset'$,
2. *for all* $h \in \mathcal{S}$, $(\deg_P(X), \deg_P(Y)) \notin sp(h)$,
   - *so you cannot X-computably find Y-computable isomorphisms between computable copies of $(\mathbb{N}, S)$.*

Uniform computable categoricity

Theorem (LG.)

There is a function $\zeta \in \mathcal{S}$ and c.e. sets X, Y such that

1. $X \oplus Y \equiv_P \zeta \equiv_P \emptyset'$,
2. *for all* $h \in \mathcal{S}$, $(\deg_P(X), \deg_P(Y)) \notin sp(h)$,
   - *so you cannot X-computably find Y-computable isomorphisms between computable copies of* $(\mathbb{N}, S)$.
3. *for all* $f \leq_T \emptyset'$, $f \leq_{sp} \zeta$.
   - *so* $\deg_{sp}(\zeta)$ *is the greatest element of* $\mathbf{0}'_P / \equiv_{sp}$.

Questions

Question

Are there $\mathbf{a}, \mathbf{b} < \mathbf{0}'_P$ *with* $(\mathbf{a}, \mathbf{b}) \in sp(\zeta)$*?*
(There are $\mathbf{a} \not\geq_P \mathbf{0}'_P$ *and* $\mathbf{b} <_T \mathbf{0}'_P$ *[LG.]).*

Question

Is there any P-degree $\mathbf{a}$ *for which* $\mathbf{a}/\equiv_{sp}$ *has only one element?*
(Yes, for sets; follows from [Ladner & Sasso, 1975]).

Question

$Th(\mathcal{D}_{sp}, \leq)$*?* $Aut(\mathcal{D}_{sp}, \leq)$*?*

Question

Does there exist a function h *for which* $(\mathbf{a}, \mathbf{b}) \in sp(h)$ *iff* $\mathbf{a} \geq_T h$ *or* $\mathbf{b} \geq_T h$*? (No for* $h \equiv_P \emptyset'$*).*

Thank you!

S. B. Cooper. Enumeration reducibility, nondeterministic computations and relative computability of partial functions. In *Recursion Theory Week*, pages 57–110, Berlin, Heidelberg, 1990. Springer Berlin Heidelberg.

R. Ladner and L. Sasso. The weak truth table degrees and recursively enumerable sets. *Ann. Math. Logic*, 4:429–448, 1975.