Asymptotic notions of computability

Tiago Royer

The University of Chicago
Department of Computer Science

November 7, 2023

https://cs.uchicago.edu/~royer/seminar.pdf

Intuition

"Definition"

A Turing machine M **solves** a problem P
if **for every** instance x of P,
M halts on x with the correct answer.

"Definition"

A Turing machine M **asymptotically solves** a problem P
if **for almost every** instance x of P,
M halts on x with the correct answer.

Density Definition

Definition

The **upper density** of a subset A of $\{0,1\}^*$ is the limit

$$\limsup_{n\to\infty} \frac{|\{x \in A : |x| = n\}|}{2^n}.$$

A is **sparse** if $d(A) = 0$ and **dense** if its complement is sparse.

Density Definition

Definition

The **upper density** of a subset A of $\{0,1\}^*$ is the limit

$$\limsup_{n\to\infty} \frac{|\{x \in A : |x| = n\}|}{2^n}.$$

A is **sparse** if $d(A) = 0$ and **dense** if its complement is sparse.

Sparsity is equivalent to

$$|\{x \in A : |x| = n\}| = o(2^n)$$

Coarse and Generic computability

Definition

A set A is **coarsely computable**
if there exists a Turing machine M such that
$M(x)\downarrow$ for all x and the set

$$\{x \mid M(x) = A(x)\}$$

is dense.

Definition

A set A is **generically computable**
if there exists a Turing machine M such that
$M(x)\downarrow$ implies $M(x) = A(x)$ and the set

$$\{x \mid M(x)\downarrow\}$$

is dense.

Examples

Example

Every computable set is both coarsely and generically computable.

Example

The set

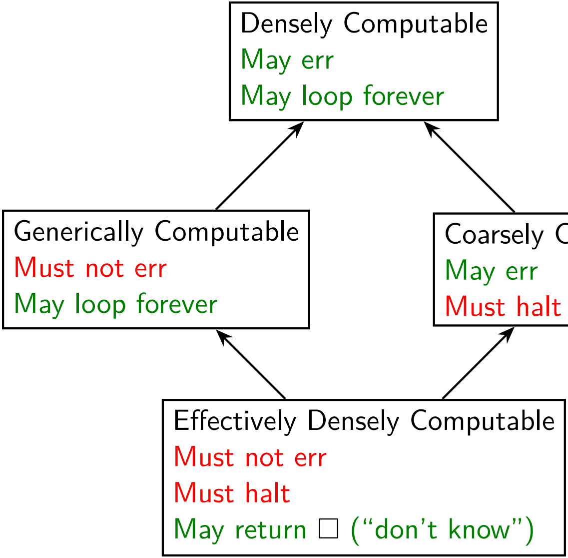
$$A = \{0^n \mid n \in \mathsf{HaltingProblem}\}$$

is not computable, but it is both coarsely and generically computable.

Example

Post's Correspondence Problem is not computable, but it is both coarsely and generically computable.

Four Horsemen of Asymptotic Computability



Coarse Reducibility

Definition

A is a **coarse approximation** of B if $A \triangle B$ is sparse.

Definition

A set A is **coarsely reducible** to a set B (denoted $A \leq_{\mathrm{c}} B$) if there's a Turing machine M such that, for every coarse approximation C of B, the set M^C is a coarse approximation of A.

Minimal Pairs

Definition

A pair of sets A and B form a **minimal pair** for Turing reducibility if neither A nor B are computable,
but if $C \leq_{\mathrm{T}} A$ and $C \leq_{\mathrm{T}} B$,
then C is computable.

Theorem (1950's)

There exists a minimal pair for the Turing degrees.

Minimal Pairs

Theorem (Hirschfeldt, Jockusch, Kuyper, Schupp, 2016)

There are measure-1 minimal pairs for coarse reducibility.

Theorem (Astor, Hirschfeldt, Jockusch, 2019)

There are measure-1 minimal pairs for dense reducibility.

Theorem (Hirschfeldt, 2020)

There exists a minimal pair for generic reducibility.

Theorem (R)

There are only measure-0 many minimal pairs for generic reducibility.

Open Problem

Are there minimal pairs for effective dense reducibility?

Minimal Degrees

Definition

A sets A has a **minimal degree** for Turing reducibility if A is not computable, but if $C \leq_{\mathrm{T}} A$, then either $A \leq_{\mathrm{T}} C$ or C is computable.

Theorem (1950's)

There exists a minimal Turing degree.

Minimal Degrees

Theorem (R)

There are minimal degrees for coarse reducibility.

Theorem (R)

There are minimal degrees for dense reducibility.

Open Problem

Are there minimal degrees for generic and effective dense reducibility?

Requirements

Theorem (R)

There are minimal degrees for coarse reducibility.

A has minimal coarse degree if we satisfy:

$$R_e : \Phi_e^A \text{ total} \Rightarrow \Phi_e^A \text{ is either coarsely computable or } A \leq_{\mathrm{c}} \Phi_e^A.$$

Requirements

Theorem (R)

There are minimal degrees for coarse reducibility.

A has minimal coarse degree if we satisfy:

$$R_e : \Phi_e^A \text{ total} \Rightarrow \Phi_e^A \text{ is either coarsely computable or } A \leq_c \Phi_e^A.$$

The intuition: build a sequence of trees $T_0 \supseteq T_1 \supseteq T_2 \cdots$
and pick a path $A \in \bigcap_i [T_i]$.
T_e will ensure R_e.

Back to Turing degrees: e-splittings

Definition

A string σ in a tree T is e-**splitting**
if there exist $\tau_0, \tau_1 \in T$ with $\tau_0, \tau_1 \succcurlyeq \sigma$ and some x such that

$$\Phi_e^{\tau_0}(x)\downarrow, \Phi_e^{\tau_1}(x)\downarrow, \text{ and } \Phi_e^{\tau_0}(x) \neq \Phi_e^{\tau_1}(x).$$

Back to Turing degrees: e-splittings

Definition

A string σ in a tree T is e-**splitting**
if there exist $\tau_0, \tau_1 \in T$ with $\tau_0, \tau_1 \succcurlyeq \sigma$ and some x such that

$$\Phi_e^{\tau_0}(x)\downarrow, \Phi_e^{\tau_1}(x)\downarrow, \text{ and } \Phi_e^{\tau_0}(x) \neq \Phi_e^{\tau_1}(x).$$

Let $A \in [T]$ (i.e. A is a path in T). Assume T computable.
If every string in T is e-splitting, then $A \leq_{\mathrm{T}} \Phi_e^A$;
If no string in T is e-splitting, then Φ_e^A is partial computable.

(e,k)-splittings

Definition

A string σ in a tree T is (e,k)-**splitting**
if there exist $\tau_0, \tau_1 \in T$ with $\tau_0, \tau_1 \succcurlyeq \sigma$ and some n such that
if $|x| = n$ then $\Phi_e^{\tau_0}(x)\downarrow$ and $\Phi_e^{\tau_1}(x)\downarrow$, and

$$\frac{|\{x : |x| = n \wedge \Phi_e^{\tau_0}(x) \neq \Phi_e^{\tau_1}(x)\}|}{2^n} > 2^{-k}.$$

(e, k)-splittings

Definition

A string σ in a tree T is (e, k)-**splitting**
if there exist $\tau_0, \tau_1 \in T$ with $\tau_0, \tau_1 \succcurlyeq \sigma$ and some n such that
if $|x| = n$ then $\Phi_e^{\tau_0}(x)\downarrow$ and $\Phi_e^{\tau_1}(x)\downarrow$, and

$$\frac{|\{x : |x| = n \wedge \Phi_e^{\tau_0}(x) \neq \Phi_e^{\tau_1}(x)\}|}{2^n} > 2^{-k}.$$

Let $A \in [T]$ (i.e. A is a path in T). Assume T computable.
If every string in T is (e, k)-splitting, then $A \leq_{\mathrm{c}} \Phi_e^A$;
If no string in T is e-splitting, then Φ_e^A is coarsely computable **up to precision** 2^{-k}.

Joe Miller to the rescue!

Theorem (Joe Miller)

Suppose there's a computable sequence $e_0, e_1, \ldots$ of indices such that Φ_{e_i} computes the set B with precision 2^{-i}. Then B is coarsely computable.

Joe Miller to the rescue!

Theorem (Joe Miller)

Suppose there's a $\emptyset'$-computable sequence $e_0, e_1, \ldots$ of indices such that Φ_{e_i} computes the set B with precision 2^{-i}.
Then B is coarsely computable.

Strategy

- Set $T_0 =$ perfect binary tree,
  $T_{\langle e,k\rangle+1} =$ subtree of $T_{\langle e,k\rangle}$ aiming to be (e,k)-splitting
- Pick $A \in \bigcap_{e,k}[T_{\langle e,k\rangle}]$
- Fixed e:
  - If some $T_{\langle e,k\rangle+1}$ is (e,k)-splitting, then $A \leq_{\mathrm{nc}} \Phi_e^A$.
  - If no $T_{\langle e,k\rangle+1}$ is (e,k)-splitting, then we can approximate Φ_e^A.

Strategy

- Set $T_0 =$ perfect binary tree,
  $T_{\langle e,k\rangle+1} =$ subtree of $T_{\langle e,k\rangle}$ aiming to be (e,k)-splitting
- Pick $A \in \bigcap_{e,k}[T_{\langle e,k\rangle}]$
- Fixed e:
  - If some $T_{\langle e,k\rangle+1}$ is (e,k)-splitting, then $A \leq_{\mathrm{nc}} \Phi_e^A$.
  - If no $T_{\langle e,k\rangle+1}$ is (e,k)-splitting, then we can approximate Φ_e^A.

Problem: the trees are not computable,
so the sets below A are coarsely computable relative to $\emptyset^{(\omega)}$...

Down to $\emptyset''''$

Let's do the construction by stages.

- Set $T^0_{\langle e,k\rangle}$ = perfect binary tree,
  $T^{s+1}_{\langle e,k\rangle+1}$ = subtree of $T^s_{\langle e,k\rangle+1}$ aiming to be (e,k)-splitting but only querying computations that finish within s steps
- Define $T^*_{\langle e,k\rangle} = \lim_s T^s_{\langle e,k\rangle}$,
  pick $A \in \bigcap_{e,k}[T_{\langle e,k\rangle}]$
- Now each $T^*_{\langle e,k\rangle}$ is $\emptyset'$-computable:
  - We get $A \leq_{\mathrm{T}} \emptyset''$
  - A' computes a sequence of $\emptyset'$-approximations to sets below A
  - so sets below A have $\emptyset''''$-computable approximations

Down to $\emptyset'''$

We don't need the whole $T^*_{\langle e,k\rangle}$, just a large enough subtree of it.

- Let's force $T^s_{\langle e,k\rangle}$ to change as little as possible.
- $T^{s+1}_{\langle e,k\rangle+1}$ searches for τ_0, τ_1 in $T^{s+1}_{\langle e,k\rangle}$.
- Pick the earliest pair found and don't change it for any $t > s$
  - unless we find an (e, k)-splitting pair
- Once there are no more changes on $T^*_{\langle e,k\rangle}$ along A, we can compute all strings in $T^*_{\langle e,k\rangle}$ extending this prefix of A.
  - A' computes a sequence of **computable** approximations to sets below A
  - so sets below A have $\emptyset'''$-computable approximations

Down to $\emptyset''$

We can interleave the construction of $T^s_{\langle e,k\rangle}$ and A.

- Once T^s_s is defined, let $\sigma_s =$ some string in T^s_s
- Force T^s_t, for $t > s$, to include σ_s
- Set $A = \lim_s \sigma_s$.
- Now $A \leq_{\mathrm{T}} \emptyset'$;
  - A' computes a sequence of computable approximations to sets below A
  - so sets below A have $\emptyset''$-computable approximations

Down to $\emptyset'$

Finally, do permitting to make A low

- Fix some low noncomputable c.e. set C
- Only allow changes between $T^{s}_{\langle e,k\rangle+1}$ and $T^{s+1}_{\langle e,k\rangle+1}$ if C permits it
- Now $A \leq_{\mathrm{T}} C$
  - so sets below A have $\emptyset'$-computable approximations
  - so the approximation theorem applies.

Asymptotic notions of computability

Tiago Royer

The University of Chicago
Department of Computer Science

November 7, 2023

https://cs.uchicago.edu/~royer/seminar.pdf