Foundations of Computability Theory (5)

Nan Fang

Institute of Software, Chinese Academy of Sciences

Peking University

2025 Autumn Semester

bounded Turing reducibilities

Definition 5.1

1. Let f be any function. A set A is *f-bounded Turing (f-bT-) reducible* to a set B if A is Turing reducible to B via a Turing functional where the use function is bounded by f, i.e., if there is a number e such that $A = \Phi_e^B$ and $\varphi_e^B(x) \leq f(x) + 1$ for all $x \geq 0$.
2. A set A is *bounded-Turing* (bT-) reducible or *weak-truth-table* (wtt-) reducible to a set B ($A \leq_{\mathrm{bT}} B$ or $A \leq_{\mathrm{wtt}} B$) if A is f-bT-reducible to B for some computable function f.
3. A set A is *identity-bounded-Turing* (ibT-) reducible to a set B ($A \leq_{\mathrm{ibT}} B$) if A is f-bT-reducible to B for the identity function $f(x) = x$.
4. A set A is $(\mathrm{i}+k)$-bT-*reducible* to a set B ($A \leq_{(\mathrm{i}+k)\mathrm{bT}} B$) if A is f-bT-reducible to B for $f(x) = x + k$.
5. A set A is *computable Lipschitz* (cl-) reducible to a set B ($A \leq_{\mathrm{cl}} B$) if A is $(\mathrm{i}+k)$-bT-reducible to B for some $k \geq 0$.

Clearly, for any sets $A, B \subseteq \mathbb{N}$, and for any $k \geq 0$,

$$A \leq_{\mathrm{ibT}} B \quad \Rightarrow \quad A \leq_{(\mathrm{i}+k)\mathrm{bT}} B \quad \Rightarrow \quad A \leq_{(\mathrm{i}+k+1)\mathrm{bT}} B \quad \Rightarrow \quad A \leq_{\mathrm{cl}} B \quad \Rightarrow \quad A \leq_{\mathrm{wtt}} B.$$

Theorem 5.2

Let A be a noncomputable set.

(i) *For $A+1 = \{x+1 : x \in A\}$, $A+1 <_{\mathrm{ibT}} A$ and $A+1 =_{(\mathrm{i}+1)\mathrm{bT}} A$.*

(ii) *For $2A = \{2x : x \in A\}$, $2A <_{\mathrm{cl}} A$ and $2A =_{\mathrm{wtt}} A$.*

(Note that, for c.e. A, the sets $A+1$ and $2A$ are also c.e.)

Proof.

We give the proof of (i). The proof of (ii) is similar.
Obviously, $A+1 \leq_{\mathrm{ibT}} A$ and $A \leq_{(\mathrm{i}+1)\mathrm{bT}} A+1$. So it suffices to show that $A \not\leq_{\mathrm{ibT}} A+1$. For a contradiction assume that $A \leq_{\mathrm{ibT}} A+1$ and fix an ibT-functional Φ such that $A = \Phi^{A+1}$.
Note that to compute $A(x)$, the functional Φ may query the oracle $A+1$ on numbers up to x.
These queries can be replaced by queries to A on numbers up to $x-1$. Thus, there is a functional $\widehat{\Phi}$ such that $A = \widehat{\Phi}^A$ and on input x all queries of $\widehat{\Phi}^A(x)$ (if any) are less than x. Such set are called *self-reducible*. It is easy to see that self-reducible sets are computable. □

Corollary 5.3

For any noncomputable c.e. set A there are noncomputable c.e. sets B and $\widehat{B}$ such that

1. *$A \leq_{\mathrm{cl}} B$ but $A \not\leq_{\mathrm{ibT}} B$ and*
2. *$A \leq_{\mathrm{wtt}} \widehat{B}$ but $A \not\leq_{\mathrm{cl}} \widehat{B}$.*

truth-table reducibilities

Definition 5.4

1. A *truth-table reduction* (tt-*reduction* for short) is a pair (g, h) of computable functions $g : \omega \to \omega^*$ (the selector function) and $h : \omega \times \{0,1\}^* \to \{0,1\}$ (the evaluation function). A set A is tt-*reducible* to a set B via (g, h) if for all numbers x

$$A(x) = h(x, B(y_0), \ldots, B(y_n)) \quad \text{where } g(x) = \langle y_0, \ldots, y_n \rangle.$$

   And A is tt-*reducible* to B ($A \leq_{tt} B$) if A is tt-reducible to B via some tt-reduction (g, h).
2. Let $k \geq 1$. A truth-table reduction (g, h) is a truth-table reduction of *norm* k or a k-tt-*reduction* for short if the selection function is of type $g : \omega \to \omega^k$. A is k-tt-*reducible* to B if there is a k-tt-reduction (g, h) such that A is tt-reducible to B via (g, h).
3. A truth-table reduction (g, h) is a *bounded truth-table* (btt) reduction if (g, h) is a k-tt-reducible for some $k \geq 1$. A is btt-*reducible* to B if A is k-tt-reducible to B for some $k \geq 1$.

truth-table reducibilities

Clearly, for any sets $A, B \subseteq \mathbb{N}$, and for any $k \geq 1$,

$$A \leq_1 B \quad \Rightarrow \quad A \leq_\mathrm{m} B \quad \Rightarrow \quad A \leq_{1-\mathrm{tt}} B \quad \Rightarrow \quad A \leq_{\mathrm{k}-\mathrm{tt}} B \quad \Rightarrow$$
$$A \leq_{(\mathrm{k}+1)-\mathrm{tt}} B \quad \Rightarrow \quad A \leq_\mathrm{btt} B \quad \Rightarrow \quad A \leq_\mathrm{tt} B \quad \Rightarrow \quad A \leq_\mathrm{wtt} B.$$

The failure of the converse of the second implication is witnessed by the fact that

$$\overline{K} \not\leq_\mathrm{m} K \quad \text{and} \quad \overline{K} \leq_{1-\mathrm{tt}} K.$$

Later, we will introduce methods to show that the converses of all other implications do not hold either in general.

characterization for tt-reducibility

Theorem 5.5

For any sets A and B, $A \leq_{\mathrm{tt}} B$ if and only if there is a total Turing functional Ψ such that $A \leq_{\mathrm{T}} B$ via Ψ, i.e., $A = \Psi^B$.

Proof.

"$\Rightarrow$:" Fix (g,h) such that $A \leq_{\mathrm{tt}} B$ via (g,h). Then this reduction can be simulated by a total oracle machine M as follows. First M^X on input x first computes $g(x) = \langle y_0, \dots, y_n \rangle$ (without using oracle X), then, by querying the oracle about $y_0, \dots, y_n$, M^X produces $X(y_0), \dots, X(y_n)$, and, finally, M^X computes $h(x, X(y_0), \dots, X(y_n))$ (without using the oracle).

"$\Leftarrow$:" Assume that $A = \Psi^B$ where the Turing functional Ψ is total. Then, for any x, the computation tree $CT_\Psi(x)$ of Ψ on input x is finite. So the depth $f(x) = \mathrm{depth}(CT_\Psi(x))$ of $CT_\Psi(x)$ is finite and computable from x. Moreover, $f(x)$ is a bound on the length of the computations $\Psi^X(x)$ for any oracle X (hence a bound on the oracle queries in any of the computations $\Psi^X(x)$ (for $X \subseteq \omega$). So if we let

$$g(x) = \langle 0, 1, 2, \dots, f(x) \rangle \quad \text{and} \quad h(x, i_0, \dots, i_{f(x)}) = \Psi^{\{y \leq f(x) : i_y = 1\}}(x)$$

then g and h are computable and $A \leq_{\mathrm{tt}} B$ via (g,h). $\square$

comparing the strong reducibilities

Theorem 5.6

For noncomputable sets A and B the following and in general (up to transitivity) only the following relations hold.

$$
\begin{array}{ccccccc}
 & & & & & & A \leq_{\mathrm{T}} B \\
 & & & & & & \Uparrow \\
 & & & & A \leq_{\mathrm{tt}} B & \Rightarrow & A \leq_{\mathrm{wtt}} B \\
 & & & & \Uparrow & & \Uparrow \\
 & & & & A \leq_{\mathrm{btt}} B & & A \leq_{\mathrm{cl}} B \\
 & & & & \Uparrow & & \Uparrow \\
A \leq_{1} B & \Rightarrow & A \leq_{\mathrm{m}} B & \Rightarrow & A \leq_{1-\mathrm{tt}} B & & A \leq_{\mathrm{ibT}} B
\end{array}
$$

computably approximable sets and ω-c.e. sets

Definition 5.7

A set A is *computably approximable (c.a.)* if there is a uniformly computable sequence of sets $\{A_s\}_{s\in\omega}$ such that for all x,

$$A(x) = \lim_s A_s(x).$$

We call $\{A_s\}_{s\in\omega}$ a computable approximation for A.

Definition 5.8

1. Let f be any function. A set A is *f-c.e.* if it is computably approximable via a computable approximation $\{A_s\}_{s\in\omega}$ such that for all x,

$$A_0(x) = 0 \quad \& \quad |\{s : A_s(x) \neq A_{s+1}(x)\}| \leq f(x).$$

2. For $n \geq 0$, a set A is *n-c.e.* if it is f-c.e. for the constant function $f(x) = n$.
3. A set A is *ω-c.e.* if it is f-c.e. for some computable function f.

Clearly, the only 0-c.e. set is $\emptyset$, 1-c.e. sets are exactly the c.e. sets. It is proven that the following proper inclusions hold (see Odifreddi 1989).

$$\text{computable} \subset \text{c.e.} \subset \text{2-c.e.} \subset \text{3-c.e.} \subset \cdots \subset \omega\text{-c.e.} \subset \text{c.a.}$$

Shoenfield limit lemma

Lemma 5.9 (Shoenfield limit lemma)

A set A is c.a. if and only if $A \leq_{\mathrm{T}} \emptyset'$.

Proof.

"$\Leftarrow$": Fix a Turing functional Φ_e such that $A = \Phi_e^{\emptyset'}$. Then the required computable approximation is given by $A_s = \{x < s : \Phi_e^{\emptyset'}(x)[s] = 1\}$.

"$\Rightarrow$": We define a c.e. set C such that $A \leq_{\mathrm{T}} C$. This is sufficient because as a c.e. set, $C \leq_{\mathrm{m}} \emptyset'$, which implies $C \leq_{\mathrm{T}} \emptyset'$.

$$\text{If } A_s(x) \neq A_{s+1}(x) \text{ we put } \langle x, i\rangle \text{ into } C_{s+1}\text{, where } i \text{ is least such that } \langle x, i\rangle \notin C_s.$$

To see that $A \leq_T C$, on input x, using the oracle C compute the least i such that $\langle x, i\rangle \notin C$. If i is even then $A(y) = A_0(y)$, otherwise $A(y) = 1 - A_0(y)$. □

The set C in the proof is called the *change set* because it records the changes of the computable approximation.

characterizations of ω-c.e. sets

Theorem 5.10

A is ω-c.e. $\Leftrightarrow A \leq_{\mathrm{wtt}} \emptyset' \Leftrightarrow A \leq_{\mathrm{tt}} \emptyset'$.

Proof.

"$A \leq_{\mathrm{wtt}} \emptyset' \Rightarrow A$ is ω-c.e.": Suppose that $A \leq_{\mathrm{wtt}} \emptyset'$ via a functional Φ_e with computable use bound f. To show that A is ω-c.e., as before let $A_s = \{x < s : \Phi_e^{\emptyset'}(x)[s] = 1\}$. Since $\Phi_e^{\emptyset'}(x)[s]$ only becomes undefined when a number less than $f(x)$ enters $\emptyset'$, the number of changes of $A_s(x)$ is bounded by $2f(x)$.

"A is ω-c.e. $\Rightarrow A \leq_{\mathrm{tt}} \emptyset'$": Suppose that A is ω-c.e. via the computable approximation $\{A_s\}_{s\in\omega}$ and the computable function g bounding the number of changes. Let C be the change set as in previous proof. By definition, $\min\{i : \langle x, i\rangle \notin C\} \leq g(x)$. Then the reduction of A to C given in previous proof can be carried out by computing a truth-table from the input x and evaluating it on the answers to oracle questions to C. Hence $A \leq_{\mathrm{tt}} C \leq_{\mathrm{m}} \emptyset'$. □

low and high sets

For any c.a. set A, $\emptyset' \leq_T A' \leq_T \emptyset''$. We call A *low* if A has the least possible jump and *high* if A has the greatest possible jump.

Definition 5.11

A c.a. set A is *low* if $A' = \emptyset'$ and A is *high* if $A' = \emptyset''$. Similarly, a Turing degree $\mathbf{a}$ is *low* (*high*) if it contains a low (high) set.

Note that lowness and highness are degree invariant. So in a low (*high*) degree all sets are low (*high*). Moreover, (by the Jump Theorem) any computable set is low, any set Turing equivalent to $\emptyset'$ is high, and no low set is high (and vice versa). By considering iterated jumps, the lowness and highness notions can be generalized.

Definition 5.12

A c.a. set A is *low*$_n$ if $A^{(n)} = \emptyset^{(n)}$ and A is *high*$_n$ if $A^{(n)} = \emptyset^{(n+1)}$. Similarly, a Turing degree $\mathbf{a}$ is *low*$_n$ (*high*$_n$) if it contains a low$_n$ (high$_n$) set.

low and high sets

We let $\mathbf{L}_n$ ($\mathbf{H}_n$) be the class of the low$_n$ (high$_n$) Turing degrees and let $\mathbf{L} = \mathbf{L}_1$ ($\mathbf{H} = \mathbf{H}_1$) be the class of the low (high) degrees. Moreover, we let $\mathbf{L}_0 = \{\mathbf{0}\}$ and $\mathbf{H}_0 = \{\mathbf{0}'\}$. Note that

$$\{\mathbf{0}\} = \mathbf{L}_0 \subseteq \mathbf{L} = \mathbf{L}_1 \subseteq \mathbf{L}_2 \subseteq \mathbf{L}_3 \subseteq \cdots, \tag{1}$$

$$\{\mathbf{0}'\} = \mathbf{H}_0 \subseteq \mathbf{H} = \mathbf{H}_1 \subseteq \mathbf{H}_2 \subseteq \mathbf{H}_3 \subseteq \cdots, \tag{2}$$

$$\tag{3}$$

and

$$\bigcup_{n\geq 0} \mathbf{L}_n \cap \bigcup_{n\geq 0} \mathbf{H}_n = \emptyset. \tag{4}$$

In fact the low-high hierarchies are proper and there are “intermediate” c.e. degrees, i.e., c.e. degrees $\mathbf{a}$ which are neither low$_n$ nor high$_n$ for any $n \geq 0$.

Theorem 5.13 (Sacks’ Jump Theorem)

For any $n \geq 0$, $\mathbf{L}_n \subsetneq \mathbf{L}_{n+1}$ and $\mathbf{H}_n \subsetneq \mathbf{H}_{n+1}$. Moreover, there is a c.e. degree $\mathbf{a}$ such that $\mathbf{0}^{(n)} <_{\mathrm{T}} \mathbf{a}^{(n)} <_{\mathrm{T}} \mathbf{0}^{(n+1)}$ for all $n \geq 0$.