

Turing Definability

Angsheng Li

Institute of Software
Chinese Academy of Sciences

Algorithm and Information Colloquium - 2010
20th, Jan., 2010

Abstract

We introduce the recent progress on Turing definability, and potential new structures of the Turing computations such as the structure of the approximation hierarchy under the bounded Turing (bT) reductions.

A common issue among algorithms, computational complexity and computability which we view as a unity, instead of different disciplines, is the problem of algorithmic new ideas.

In this talk, we would examine the algorithmic aspects from the viewpoint of computability.

Turing computation

A Turing machine consists of:

- a machine head,
- an input tape,
- a working tape, and
- an output tape.

A Turing machine is a triple (Q, Σ, Δ) with the following properties:

- (1) Q is a finite set of *states*, which contains an *initial state*, q_0 say, and a *halting state*, q_1 say.
- (2) Σ is a finite set of alphabet.
- (3) Δ is a finite set of rules of the form $(q, s; q', s', d)$.

Turing-Church thesis: A function is computable \iff it is computable by a Turing machine.

Enumeration

THEOREM. There is an algorithm to generate all Turing machines such as

$$\phi_0, \phi_1, \phi_2, \dots$$

- Universal Turing machine
a machine computes $f(x, y) = \phi_x(y)$
- oracle Turing machines, computing functionals
- Every computable function will be computed by infinitely many Turing machines.

Turing reducibilities

Given sets $A, B \subseteq \omega = \{0, 1, 2, \dots\}$, we say that A is *Turing reducible to* B , if there is a Turing machine which computes A when it is equipped with an oracle B . Write

$$A \leq_T B$$

or

$$A = \Phi^B$$

for some Turing functional Φ , i.e., a Turing machine which equips an oracle tape.

– this offers a partial ordering of the structures

Join operator

Given $A, B \subseteq \omega$, define the *join* of A and B by

$$A \oplus B = \{2x \mid x \in A\} \cup \{2y + 1 \mid y \in B\}.$$

- information can be coded together easily.
- this offers the possibility to study the structure of the Turing computations.

Turing jump operator

Turing's theorem: The halting problem is undecidable.

Define the *halting set* by

$$\emptyset' = \{\langle x, y \rangle \mid \phi_x(y) \downarrow\}.$$

Then:

$$\emptyset <_T \emptyset'.$$

For $A \subseteq \omega$, define

$$A' = \{\langle x, y \rangle \mid \Phi_x^A(y) \downarrow\}.$$

Then $A <_T A'$, giving an operator producing strictly harder problems.

Post's characterization

Let $A \subseteq \omega$, and \emptyset' be the halting problem.

Then:

$$A \leq_T \emptyset'$$



there is a computable function f such that for every $x \in \omega$,

$$A(x) = \lim_s f(x, s)$$

where we use A to denote the characteristic function of the language A .

High/low hierarchy

low_n : A set $X \leq_T \emptyset'$ is called low_n , if $X^{(n)} \equiv_T \emptyset^{(n)}$

This gives a lowness hierarchy:

$$L_1 \subset L_2 \subset L_3 \subset \dots$$

L_1 – called *low*, since it is close to \emptyset , and then all computable functions.

high_n : A set X is called high_n , if $X^{(n)} \equiv_T \emptyset^{(n+1)}$.
highness hierarchy:

$$H_1 \subset H_2 \subset H_3 \subset \dots$$

H_1 , called *high*, close to the halting problem.

Difference hierarchy

The *difference hierarchy*:

For every natural number n , and set $A \subseteq \omega$, we say that A is *n -computably enumerable* (n -c.e., for short), if there is a computable function f such that, for all x ,

- (i) $f(x, 0) = 0$,
- (ii) $\lim_s f(x, s) \downarrow = A(x)$, and
- (iii) $|\{s \mid f(x, s-1) \neq f(x, s)\}| \leq n$.

Let D_n be the set of all n -computably enumerable sets, for each n .

The structures

The high/low hierarch and the difference hierarchy offer a nice classification of functions computable relative to the halting problem.

- this builds the fundamentals of the local structures of Turing degrees
- it also provides both theoretical and technical resources for the study of the global structures of Turing degrees through the Turing jump operator

The questions

A set A is called *computably enumerable* (c.e., for short), if there is an algorithm to enumerate the elements of it.

- all decidable sets are computably enumerable,
- the halting problem is computably enumerable, and
- the halting problem is complete for all c.e. sets under Turing reductions, and $\emptyset <_T \emptyset'$.

Post (1944): Is there any c.e. set X such that

$$\emptyset <_T X <_T \emptyset'?$$

Finite injury argument

Friedberg (1957), Muchnik (1956):

There exists a computably enumerable set X such that

$$\emptyset <_T X <_T \emptyset'.$$

- *finite injury method* were introduced in this proof.
- *ordering* is essential in computing, one must not neglect

Proof - I

We build c.e. A, B to satisfy for all e ,

$$\mathcal{R}_{2e}: A \neq \Phi_e^B,$$

$$\mathcal{R}_{2e+1}: B \neq \Phi_e^A.$$

To satisfy \mathcal{R}_{2e} , we design the following *strategy*.

1. Choose a *fresh* number, a say.
2. Wait for a stage, s say, at which

$$\Phi_e^B(a) \downarrow = 0.$$

Then:

- enumerate a into A ,
- restrain all elements $\leq \text{use}(\Phi_e^B(a)[s])$ from entering B .

Proof - II

If step 2 occurs, then

$$\Phi_e^B(\mathbf{a}) \downarrow = 0 \neq 1 = A(\mathbf{a}),$$

otherwise, then

$$\Phi_e^B(\mathbf{a}) \neq 0 = A(\mathbf{a}).$$

In either case, \mathcal{R}_{2e} is satisfied.

Proof - III

The problem is how to preserve the computation $\Phi_e^B(a)[s] \downarrow = 0$ once step 2 occurred.

The idea is to fix a *priority ranking of all requirements* such as

$$\mathcal{R}_0, \mathcal{R}_1, \mathcal{R}_2, \mathcal{R}_3, \dots$$

as they are listed.

During the course of the construction, each strategy cannot be injured by any strategy working on lower priority ranking requirements.

So the strategy for every requirement can only be injured finitely many times.

Splitting

Sacks (1963) *splitting*:

For any c.e. set X , if X is undecidable, then there are disjoint sets A_0 , and A_1 satisfying:

- (1) $A_0 \cup A_1 = X$,
- (2) $A_i <_T X$ for each $i = 0, 1$.

Robinson 1971 *low splitting*:

splitting can be done above any low (low_1) sets.

– a nice application of the finite injury method.

Density – infinite injury

Sacks (1964) density:

For any c.e. sets X, Y , if $X <_T Y$, then there is a c.e. set A such that

$$X <_T A <_T Y.$$

- *infinite injury method* was invented in this proof.
- why is it possible of infinite injury?
 - every strategy opens window infinitely often, and there are infinitely many times at which all strategies with higher priority ranking open window
 - a strategy will guess the outcomes of the strategies of higher priority ranking requirements, and the one guessing right will be successful

Priority tree argument

A *strategy* is an algorithm, or a computable procedure, which will try to satisfy some requirement, and which may have different *outcomes* eventually.

We define a *priority ordering* among the *outcomes*, and arrange all strategies of the requirements on nodes of a tree, *the priority tree*, T say.

Let $\alpha \in T$ be a strategy.

We ensure that

- (i) if no injury occurs, and α has infinitely many chances to act, then α is successful to satisfy its requirement,
- (ii) α knows the outcomes for all nodes $\beta \subset \alpha$, so can avoid injury from these β 's
- (iii) α will cancel any possible injury from nodes to the right of α
- (iv) for any $\beta \supset \alpha$, β acts only if α allows, so β does not injury α .

True Path

Then:

α may be injured by strategies to the left of α .

Let p be the infinite path such that

(a) for every $\alpha \in p$, there are only finitely many times some $\beta <_L \alpha$ acts

(b) every $\alpha \in p$ is visited infinitely often.

p exists, called the *true path*, denoted by TP.

So a node on the true path can be injured only finitely many times, and eventually satisfies its requirement.

Minimal pairs

Lachlan 1966 *minimal pair*.

There exist c.e. sets A , and B with the following properties:

(1) $A >_T \emptyset$ and $B >_T \emptyset$,

(2) For any function f , if

$f \leq_T A$, and $f \leq_T B$, then f is computable.

– a new presentation of infinite injury, called *priority tree method* was invented in this proof.

Lachlan Nonsplitting

Lachlan 1975 *nonsplitting*:

There exists two c.e. sets A and B satisfying the following properties:

(1) $A <_T B$, and

(2) for any c.e. sets X, Y , if $A \leq_T X$, $Y \leq_T B$, and $B \equiv_T X \oplus Y$, then either $X \equiv_T B$ or $Y \equiv_T B$.

– a \emptyset''' -priority tree method was invented in this proof, which was called a “monster method”.

– why?

– a finite injury on the true path

Harrington Nonsplitting

Let \emptyset' be the halting problem.

Harrington 1980:

\emptyset' is not splittable over some c.e. set $A <_T \emptyset'$.

– this is a real understanding of Lachlan's monster method, by introducing some rules, called *golden rules* for priority tree argument.

Cooper-Li Nonsplitting

Cooper and Li, 2002:

\emptyset' is not splittable over some low_2 set.

– this complements with the Robinson low splitting.

Major subdegree problem

Lachlan 1967:

Given c.e. sets $A <_T B$, we say that A has a *major subdegree* of B , if for any c.e. set X ,

$$X \oplus A \equiv_T \emptyset' \iff X \oplus B \equiv_T \emptyset'.$$

Q. Does every c.e. set non-computable, and incomplete, has a major subdegree?

Definable ideals

Cooper non-cupping 1974:

There exists a c.e. A , satisfying:

- (1) $A >_T \emptyset$, and
- (2) for any c.e. X ,

$$X \oplus A \equiv_T \emptyset' \iff X \equiv_T \emptyset'.$$

- such an A has a major subdegree, partially answer Lachlan's question,
- the Turing degrees of all such A form an *ideal* in the structure of the c.e. Turing degrees.

Continuity of cupping

Ambos-Spies, Lachlan, Soare, 1993:

For any c.e. sets X, Y , if:

$$- X <_T \emptyset', Y <_T \emptyset',$$

$$- \emptyset' \equiv_T X \oplus Y,$$

then there is a c.e. Z such that

$$(1) Z <_T X,$$

$$(2) Z \oplus Y \equiv_T \emptyset'.$$

Continuity of capping

Harrington and Soare 1989:

For any c.e. X, Y , if

– $\emptyset <_T X, \emptyset <_T Y$,

– $X \wedge Y \equiv_T \emptyset$,

then there is a c.e. Z with

(1) $X <_T Z$, and

(2) $Z \wedge Y \equiv_T \emptyset$.

Seetapun's theorem

Seetapun 1991:

The dual of the Lachlan's major subdegree problem has a positive solution.

The major subdegree theorem

Cooper, Li, 2008:

For any c.e. $B \not\equiv_T \emptyset, \emptyset'$, there exists a c.e. A with

(1) $A <_T B$, and

(2) for any c.e. X ,

$$A \oplus X \equiv_T \emptyset' \iff B \oplus X \equiv_T \emptyset'.$$

– answering Lachlan's question.

Δ_2^0 degrees

The Turing degrees of all functions approximated by computable function, or the limit theory of the computable functions. That is:

All X with $X \leq \emptyset'$.

In Δ_2^0 sets, for any $X \not\equiv_T \emptyset$, there is an A such that

$$\emptyset' \equiv_T A \oplus X \equiv_T A'$$

n-c.e. degrees

Let $n > 1$.

For any n -c.e. set $X \not\equiv_T \emptyset$, there is an n -c.e. set A such that

$$\emptyset' \equiv_T A \oplus X \equiv_T A'.$$

– how about if $n = 1$? No – by Cooper noncupping

Join theorem for c.e. degrees

Jockusch, Li and Yang, 2004:

For any c.e. set X , if $X \not\equiv_T \emptyset$, then there is a c.e. set a with the following property:

$$\emptyset'' \equiv_T (A \oplus X)' \equiv_T A''.$$

– the best possibility by Cooper 1974

Cupping

Let $n > 1$ be a natural number.

Arslanov, 1985:

For any n -c.e. $X \not\equiv_T \emptyset$, there is a 2-c.e. A such that

$$A \oplus X \equiv_T \emptyset'.$$

– this is different from the c.e. sets, i.e., the 1-c.e. sets, by
Cooper 1974

Diamond lattice

Downey, 1989:

There exist 2-c.e. X, Y , both $\not\equiv_T \emptyset$ with

(1) $X \wedge Y \equiv_T \emptyset$,

(2) $X \oplus Y \equiv_T \emptyset'$.

– different from c.e. case, by Lachlan 1966.

Non-density

Cooper, Harrington, Lachlan, Lempp, Soare, 1991:

There exists a 2-c.e. set $\not\equiv_T \emptyset'$ which has maximal n -c.e. degrees for all n .

– this is not true for c.e., by Sacks density theorem.

Li-Yi cupping theorem

Li, Yi, 1999:

There exist incomplete 2-c.e. sets A_0 , and A_1 , for that for any n -c.e. $X \not\equiv_T \emptyset$, one of the (1) and (2) below holds:

(1) $A_0 \oplus X \equiv_T \emptyset'$,

(2) $A_1 \oplus X \equiv_T \emptyset'$.

– extending both Arslanov 1985, and Downey 1989.

Splitting

Cooper, 1990:

For any 2-c.e. $X \not\equiv_T \emptyset$, there exists 2-c.e. A_0 and A_1 such that

(1) $A_0, A_1 <_T X$

(2) $X \equiv_T A_0 \oplus A_1$.

– splitting holds in 2-c.e.

Turing approximations

Cooper, Li, 2002:

For any c.e. B , 2-c.e. A , if

$B <_T A$, then there exist 2-c.e. X_0, X_1 with

(1) $B \leq_T X_0, X_1 <_T A$, and

(2) $A \equiv_T X_0 \oplus X_1$.

Open question. Are the c.e. degrees definable in the 2-c.e. degrees?

Open questions

Lachlan's major sub-degree problem in n -c.e. degrees, for every $n > 1$.

- major sub-degree problem leads to a wide topic, which represents a large body of interesting research in c.e. Turing degrees
- nothing is known for n -c.e. degrees in this topic, deserved to be developed

bT-reductions

Given A, B , we say that A is *bounded Turing reducible* (bT-reducibility) to B , if there is a Turing functional Φ say, such that

- $A = \Phi^B$, and
- the use function of Φ , ϕ say, is bounded by some computable function.

Write

$$A \leq_{\text{bT}} B$$

Continuity of capping

Brodhead, A. Li, W. Li, 2008:

The dual of the major sub-degree theorem holds for c.e. sets under the bT-reductions.

Definable filters

A. Li, W. Li, Pan, Tang, 2009:

There exists a c.e. set A, B , such that

(1) $\emptyset <_{bT} A <_{bT} \emptyset'$, and

(2) For any c.e. X ,

$$B \oplus X \equiv_{bT} \emptyset' \iff A \leq_{bT} X.$$

– the major sub-degree theorem fails to hold for c.e. sets under bT-reductions.

n-c.e. bT-degrees

Let $n > 1$.

Nothing non-trivial is known for the *n*-c.e. sets under the bT-reductions.

– a new theme is called for

Methods

Computability treats a Turing machine as a *black box*, fortunately the priority methods compensate this weakness, and build a well developed theory.

New directions in computability

Structures of PSPACE under logspace reductions
– called for

Major open problems in complexity

Hartmanis built the foundation of CC directly based on Turing machines, treating a Turing machine as a *black box*.

Consequently,

1. *ordering*, a fate problem, is simply neglected in the study of current CC
2. a relativization obstacle exists in every nontrivial problem,
3. this leaves all problems open, including the most important ones:
 - Sequential or parallel computing
 - time or space
 - deterministic or non-deterministic computations.

Conjecture: Not all the three major open problems above are equally hard, the easiest one could be solved in the near future

Approaches to major open problems

1. algebraic + probability + new reductions with coding or error correcting codes
2. **white box approach** of Turing computations – local decision + game theoretical approach

These two approaches naturally avoid the relativization obstacles in computational complexity

THANK YOU!