CS5371
Theory of Computation

Lecture 13: Computability IV
(Undecidable Languages)

Objectives

- In this lecture, we investigate some undecidable languages
- We first introduce the diagonalization method, which is a powerful tool to show a language is undecidable
- Afterwards, we give examples of undecidable languages that are
  - Turing recognizable but not decidable
  - Non-Turing recognizable

Math Review: Countable Set

Let $N = \{1, 2, 3, 4, \ldots\}$ be the set of natural numbers. We say an infinite set A have the same size as N, if there exists a one-to-one correspondence

$$f: N \rightarrow A.$$

In other words, for each a in A, there is a unique x in N such that $f(x) = a$.

Definition: A set A is countable if $|A|$ is finite, or A has the same size as N

Countable Set?

Is the following a countable set?

1. $\mathbb{N}$
2. $\mathbb{Z}$ (the set of integers)
3. The set of positive odd numbers
4. Subset of a countable set
5. $\mathbb{Q}$ (the set of positive rational numbers)

A number is rational if it can be expressed n/m for some integers n and m

Countable Set? (2)

1. N --- Yes.

   Let f: N → N be f(x) = x

2. Z --- Yes.

   Let f:N → Z be

   f(x) = (x-1)/2 when x is odd

   f(x) = -x/2 when x is even

3. ODD = Set of +ve odd numbers --- Yes.

   Let f: N → ODD be f(x) = 2x - 1

Countable Set? (3)

4. Subset of a countable set -- Yes.

Let A be a countable set, B be A's subset.

Case 1: If |B| is finite, B is countable

Case 2: If |B| is infinite, A must be a countable set with the same size as N

Let f be a one-to-one correspondence from N to A. Based on f, we shall give a one-to-one correspondence g from N to A

Countable Set? (4)

Construction of g:

Since f is one-to-one correspondence, f^{-1} is well-defined. Also, we see that

$f^{-1}(b) = f^{-1}(b')$ if and only if $b = b'$

Thus, we can list the elements of B uniquely, such that b is before b' if

$f^{-1}(b) < f^{-1}(b')$

Let r_b be the rank of b in the list

Countable Set? (5)

Q1: What are the values of r_b?

Ans: r_b will range from 1, 2, 3, …

Q2: If $x \neq y$, can $r_x = r_y$?

Ans: No

Now, we define the function g, with

$$g(r_b) = b \qquad \text{for all } b \text{ in } B$$

which is a one-to-one correspondence from N to B

Countable Set? (6)

Why do we use the term countable??

For a countable set S, there will be a one-to-one correspondence f from N to S.

If f(k) = x, we call x the k^{th} element of S

To list out elements in S, we may list the 1st element, then the 2nd element, then the 3rd element, and so on.

(Just like counting sheep when we cannot sleep)

... and, we will not miss any element of S!

Countable Set? (7)

5. Q (Set of +ve rational numbers) -- Yes.

   We first prove the following set, Q′, is countable:

   Q′ = { "n/m" : n, m are +ve integers }

   where "n/m"=the string (not value) of n/m

Example elements of Q′ are: 1/2, 2/3, 2/4, 3/3, 18/2, … (Note that 1/2 and 2/4 are two distinct elements in Q′)

Countable Set? (8)

To see why Q′ is countable, let us find a systematic way to list out its elements

For sum = 2, 3, 4, …

List all "n/m" with n+m = sum and n, m > 0

Precisely, we list "(sum-1)/1",

then "(sum-2)/2",

… … … ,

then "2/(sum-2)",

then "1/(sum-1)"

Countable Set? (9)

Based on the above listing procedure, we will first list 1/1, then 2/1, then 1/2, then 3/1, then 2/2, then 1/3, then 4/1, and so on

We can see that each elements of Q' will be listed eventually… Thus, Q' is countable (what will be the one-to-one correspondence from N to Q'??)

Countable Set? (10)

Now, if we remove from Q' all but one strings that represent the same value (such as 1/2, 2/4, 3/6, … have the same value, but we keep only the one with smallest "sum"), the resulting set will be equivalent to Q.

Thus, Q is countable. (why??)

Uncountable Set Exists?

Theorem: The set of real numbers R′ in the range [0,1) is uncountable.

Proof: Assume on the contrary that R′ is countable. Then, there is some one-to-one correspondence f that maps N to R′.

Let x_k be the real number with $f(k) = x_k$. Consider x such that for every k, its k^{th} digit (after the decimal place) is equal to "the k^{th} digit of x_k" + 1 (mod 10).

Uncountable Set Exists? (2)

E.g.

$x_1 = f(1)$ 0.7182818284590452354…

$x_2 = f(2)$ 0.4426950408889634074…

$x_3 = f(3)$ 0.1415926535897932384 6…

$x_4 = f(4)$ 0.4142135623730950488 0…

$x_5 = f(5)$ 0.5000000000000000000…

$x_6 = f(6)$ 0.9999999999999999999…

⋮ ⋮

x = 0.852310…

Uncountable Set Exists? (3)

Now, there is something special about x

- Firstly, x is a real number in [0,1).

  By our assumption, there is some j such that $f(j) = x$
- However, by our construction of x, there is no j such that $f(j) = x$, because $f(j)$ will be different from x at the j^{th} digit
- Thus, a contradiction occurs (where??)
- We conclude that R' is uncountable

Uncountable Set Exists? (4)

Then, we also have

Theorem: The set of real numbers R is uncountable.

(Why??)

Diagonalization Method

- In the proof of R′ is uncountable, what we do are the following:
  1. assume a one-to-one f from N to R′
  2. construct x (in R′) based on f
  3. show that x cannot correspond to any number in N
- The technique is called diagonalization (x is constructed by choosing a different value for each digit along the "diagonal")

Non-Turing Recognizable

Theorem: Some language are non-Turing recognizable.

Proof: We are going to show that (1) the set of all TMs is countable, but (2) the set of all languages is uncountable.

Combining, there must be some language which is non-Turing recognizable, as each TM can recognize only one language.

The set of all TMs is countable

- It is sufficient to show the set E of encoding of TMs is countable (as each TM has distinct encoding)
- Fact: For any finite Σ, the set of strings in Σ^* is countable

  Proof: first count strings of length 0, then strings of length 1, then strings of length 2, …
- Each TM can be encoded as a string in Σ^*. Thus, E is a subset of Σ^* ➔ countable

Set of all languages is uncountable

- Let B be the set of all binary strings

  Note: B is countable, and we label the elements of B by $b_1, b_2, b_3, \ldots$

- To show Part 2, it is sufficient if we can show the set of languages S whose strings are from B is uncountable

  (what is the relationship between S and B???)

We now prove the above statement using diagonalization technique

Set of all languages is uncountable

- Suppose on the contrary S is countable
  ➔ there is a one-to-one f from N to S

  f(k): the k^{th} element of S, denoted by s_k

  (Keep in mind that each element of S is a subset of B)
- Let us construct an element s of S :

  If $b_k \in s_k$, then s does not contain b_k

  If $b_k \notin s_k$, then s contains b_k
- Then, there is no j with f(j) = s.
- A contradiction occurs ➔ S is uncountable

Acceptance by TM

- We will later give an example of a non-Turing recognizable language
- Let us now focus on Turing-recognizable languages, and show that among them, some are undecidable
- Let A_{TM} be the language

  $\{ \langle M, w \rangle \mid M \text{ is a TM that accepts } w \}$

Theorem: A_{TM} is undecidable

Acceptance by TM (2)

Proof: By diagonalization technique again.

Suppose on contrary that A_{TM} is decidable.
Let H be the corresponding decider.

That is, on input ⟨M, w⟩, H accepts if M accepts w, and H rejects if M does not accept w

Let us construct a decider D as follows:

D = "On input ⟨M⟩, where M is a TM

1. Run H on input ⟨M, ⟨M⟩ ⟩
2. If H accepts, D rejects. Else, D accepts"

Acceptance by TM (3)

- Since H halts on all inputs, D halts on all inputs ➔ D is a decider
- A closer look on D :

  D(⟨M⟩) = accept if M not accepts ⟨M⟩

  D(⟨M⟩) = reject if M accepts ⟨M⟩

- What if D is given the input ⟨D⟩?

  D(⟨D⟩) = accept if D not accepts ⟨D⟩

  D(⟨D⟩) = reject if D accepts ⟨D⟩

Acceptance by TM (3)

- Thus, the output of D(⟨D⟩) must not be accept, and must not be reject (what can that be??)

  ➔ D is not a decider

- A contradiction occurs
- We conclude A_{TM} is undecidable

Where is the diagonalization?

- In the construction of D, we assume H exists, we can complete the table below:

	$\langle M_1 \rangle$	$\langle M_2 \rangle$	$\langle M_3 \rangle$	$\langle M_4 \rangle$	…
M_1	accept	reject	accept	accept	
M_2	accept	reject	reject	reject	…
M_3	reject	accept	reject	accept	
M_4	reject	reject	reject	accept	
⋮			⋮		
Set D	reject	accept	accept	reject	…

Property of Decidable Language

Theorem: Let L^c denote the complement of L.

(1) If L is decidable, both L and L^c are Turing-recognizable.

(2) If L and L^c are Turing-recognizable, L is decidable.

Proof of (1): L is decidable, so that L is Turing-recognizable (why?). Also, L is decidable implies L^c is decidable (why?). Thus, L^c is Turing-recognizable.

Property of Decidable Language

Proof of (2): If L, L^c are Turing-recognizable,

let M_1 = TM that recognizes L, and

M_2 = TM that recognizes L^c

We construct a decider D for L:

D = "On input w,

1. Run both M_1 and M_2 on input w in parallel
   (D takes turn to simulate one step of each machine)
2. If M_1 accepts, D accepts.
   If M_2 accepts, D rejects"

Quick Quiz: Why is D a decider?

Non-Turing Recognizable Language (example)

Theorem: The complement of A_{TM} is not Turing-recognizable.

Fact 1: A_{TM} is Turing-recognizable (why?).

Fact 2: A_{TM} is undecidable.

Fact 3: If complement of A_{TM} is Turing-recognizable, then A_{TM} is decidable.

What we have learnt so far

- A_{DFA}, A_{NFA}, A_{RE}, A_{CFG}, E_{DFA}, E_{CFG}, EQ_{DFA} are decidable languages
- TM is more powerful than CFG
- A_{TM} is undecidable
- The complement of A_{TM} is not Turing recognizable

Next Time

- Reducibility
  - To relate the solutions of two problems
  - If a solution to a problem B can be used to give a solution to a problem *A*, it seems that *A* cannot be harder than B
  - E.g., B = solving quintic (degree 5) equation,
    A = solving quadratic (degree 2) equation
  - This idea is useful in showing many other results in computability theory and later in complexity theory