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, … } 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. N
2. Z (the set of integers)
3. The set of positive odd numbers
4. Subset of a countable set
5. 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, and B be its subset. (Case 1) If |A| is finite, |B| is finite, so that B is countable. (Case 2) If |A| is infinite, let f: N → A be a one-to-one correspondence from N to A.

(Case 2.1) If |B| is finite, then B is countable. (Case 2.2) Else, we shall give a one-to-one correspondence g: N→B:

Countable Set? (4)

Construction of g:

For each element b in B, let r_b (rank of b) be the number of elements of x in B with $f^{-1}(x) \leq f^{-1}(b)$. (Note: f^{-1} is well defined, as f is one-to-one correspondence.) Precisely,

$r_b = | \{ x : x \text{ in } B \text{ and } f^{-1}(x) \leq f^{-1}(b) \} |$.

Intuitively, by considering the values $f^{-1}(b_1)$, $f^{-1}(b_2)$, …, of each element b_1, b_2, … in B, if $f^{-1}(b)$ is k^{th} smallest among all f^{-1}, $r_b = k$

Countable Set? (5)

Q1: What are the values of r_b?

Ans: r_b will range from 1, 2, 3, … (Since the element with k^{th} smallest f^{-1} value will have r_b=k)

Q2: For two different elements x and y in B, can $r_x = r_y$? Ans: No (why??)

Now, we set up the function g, with $g(r_b)$ = b for all b in B. We can see that g: N → B and g is a one-to-one correspondence

Countable Set? (6)

Why do we use the term countable??

For a countable (infinite) set S, let $f: N \rightarrow S$ be the one-to-one correspondence. 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 you cannot sleep) Interestingly, we will not miss any element in S!

Countable Set? (7)

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

   Before we prove Q is countable, let us prove the following set, Q′, is countable:

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

   where "n/m" denote the string (not the 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)

6. To see why Q' is countable, let us find a systematic way to list out its elements one by one

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 in R such that $f(k) = x_k$. Consider a number x such that 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.

$x_1 = f(1)$ 0.71828182845904523 54…

$x_2 = f(2)$ 0.44269504088896340 74…

$x_3 = f(3)$ 0.14159265358979323 846…

$x_4 = f(4)$ 0.41421356237309504 880…

$x_5 = f(5)$ 0.50000000000000000 000…

$x_6 = f(6)$ 0.99999999999999999 99…

⋮ ⋮

$x = 0.852310\ldots$

Uncountable Set Exists? (3)

Now, there is something special about x

- Firstly, x is a real number, so that by our assumption, there is some j such that $f(j) = x$.
- On the other hand, by our construction of x, there is no j such that $f(j) = x$, because the real number $f(j)$ will be different from x at the j^{th} decimal place
- 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??)

Uncountable Set Exists? (5)

- In the proof of R' (the set of real numbers in [0,1)) is not countable, we assume a one-to-one correspondence f from N to R' first, but then construct a real number x and 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.

Non-Turing Recognizable (2)

(Part 1) The set of all TMs is countable:

- It is sufficient if we can show the set E of encoding of TMs is countable (as there is one-to-one correspondence between TM and its encoding)
- We observe that for any finite Σ, the set of strings in Σ^* is countable (we first count those strings of length 0, then those strings of length 1, then strings of length 2, and so on)
- Each TM can be encoded as a string in some finite Σ. Thus, E is a subset of Σ^* → countable

Non-Turing Recognizable (3)

(Part 2) The 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, …
- To show Part 2, it is sufficient if we can show the set of languages S whose strings are from {0,1}*, is uncountable (what is the relationship between S and B???)
- We now prove the above statement using diagonalization technique

Non-Turing Recognizable (4)

- Suppose on the contrary that S is countable. Then there is a one-to-one correspondence f that maps N to S
- Call 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 as follows:
  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 such that f(j) = s. Thus, a contradiction occurs, so that S is uncountable

Acceptance by TM

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

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

Theorem: A_{TM} is undecidable

Acceptance by TM (2)

Proof: We use the diagonalization technique again. Suppose on the contrary that A_{TM} is decidable. Let H be the corresponding decider. In other words, on input ⟨M, w⟩, H accepts the input if M accepts w, and H rejects the input if M does not accept w.

- Let us construct a decider D as follows:

D = "On input ⟨M⟩, where M is a TM

Step 1. Run H on input ⟨M, ⟨M⟩ ⟩

Step 2. If H accepts, D rejects. Else, D accepts"

Acceptance by TM (3)

- In other words,

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

  D(⟨M⟩) = reject if M accepts ⟨M⟩
- What happens when D is given the input ⟨D⟩?

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

  D(⟨D⟩) = reject if D accepts ⟨D⟩
- Thus, the output of D(⟨D⟩) is not accept, and is not reject. This implies that D is not a decider
- Contradiction occurs (D is a decider, D is not a decider) → we conclude A_{TM} is undecidable

Where is the diagonalization?

- In the construction of D (assuming H exists)
- Precisely, if H exists, we can complete the table below:

	$\langle M_1 \rangle$	$\langle M_2 \rangle$	$\langle M_3 \rangle$	$\langle M_4 \rangle$	$\cdots$
M_1	accept	reject	accept	accept	
M_2	accept	reject	reject	reject	$\cdots$
M_3	reject	accept	reject	accept	
M_4	reject	reject	reject	accept	
$\vdots$			$\vdots$		

- Set D reject accept accept reject $\cdots$

Property of Decidable Language

Theorem: Let L be a language, and L^c be its complement. (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 and L^c are Turing-recognizable, let M_1 be the TM that recognizes L, and M_2 be the TM that recognizes L^c. We construct a decider D for L (thus proving (2)) as follows:

D = "On input w,

Step 1. Run both M_1 and M_2 on input w in parallel (that is, D takes turn to simulate one step of each machine)

Step 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 non-Turing recognizable.

Why??

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
  - The idea of relating 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
  - For example, 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