Harvard CS 121 and CSCI E-207

Lecture 16: Undecidability

Harry Lewis

October 28, 2010

- Reading: Sipser §4.2, §5.1.

Motivation

- Goal: to find an explicit undecidable language

- By the Church–Turing thesis, such a language has a membership problem that cannot be solved by any kind of algorithm

- We know such languages exist, by a counting argument.
  - Every recursive language is decided by a TM
  - There are only countably many TMs
  - There are uncountably many languages

$\therefore$ Most languages are not recursive (or even r.e.)

Three basic facts on the recursive vs. r.e. languages

1. If L is recursive, then L is r.e.

   Proof:

Three basic facts on the recursive vs. r.e. languages

1. If L is recursive, then L is r.e.

   <u>Proof:</u>
   If M decides L, then a machine can accept L by running M, and then going into an infinite loop if M would have halted in the q_{accept} state.

2. If L is recursive then so is $\overline{L}$.

   <u>Proof:</u>

Three basic facts on the recursive vs. r.e. languages

1. If L is recursive, then L is r.e.

   Proof:
   If M decides L, then a machine can accept L by running M, and then going into an infinite loop if M would have halted in the q_{accept} state.

2. If L is recursive then so is $\overline{L}$.

   Proof:
   A machine can decide $\overline{L}$ by running M and then giving a “no” answer when M would give “yes” and vice versa.

3. L is recursive if and only if both L and $\overline{L}$ are r.e.

   Proof: . . .

Is every Turing-recognizable set decidable?

This would be true if there were an algorithm to solve

The Acceptance Problem:

Given a TM M and an input w, does M accept input w?

Formally, $\mathrm{A_{TM}} = \{\langle M, w\rangle : M \text{ accepts } w\}$.

Proposition: If $\mathrm{A_{TM}}$ is recursive, then every r.e. language is recursive.

"$\mathrm{A_{TM}}$ is the hardest r.e. language."

- $\mathrm{A_{TM}}$ is said to be *r.e.-complete*

A simplifying detail: every string represents some TM

- Let Σ be the alphabet over which TMs are represented (that is, $\langle M\rangle \in \Sigma^*$ for any TM M)
- Let $w \in \Sigma^*$
- if $w = \langle M\rangle$ for some TM M then w represents M
- Otherwise w represents some fixed TM M_0 (say the simplest possible TM).

Thm: A_{TM} is not recursive

- Look at A_{TM} as a table answering every question:

	w_0	w_1	w_2	w_3	
M_0	Y	N	N	Y	
M_1	Y	Y	N	N	(WLOG assume
M_2	N	N	N	N	every string w_i
M_3	Y	Y	Y	Y	encodes a TM M_i)

- Entry matching (M_i, w_j) is Y iff M_i accepts w_j
- If A_{TM} were recursive, then so would be the diagonal D and its complement.
  - $D = \{w_i : M_i$ accepts $w_i\}$.
  - $\overline{D} = \{w_i : M_i$ does not accept $w_i\}$.
- But $\overline{D}$ differs from every row, i.e. it differs from every r.e. language. $\Rightarrow\Leftarrow$.

Unfolding the Diagonalization

- Suppose for contradiction that A_{TM} were recursive.

- Then there is a TM M^* that decides
  $\overline{D} = \{\langle M \rangle : M$ does not accept $\langle M \rangle\}$.

  - $M^*(\langle N \rangle)$ runs the decider for A_{TM} on $\langle N, \langle N \rangle \rangle$ and does the opposite.

- Run M^* on its own description $\langle M^* \rangle$.

- Does it accept?

  M^* accepts $\langle M^* \rangle$

  $\Leftrightarrow \langle M^* \rangle \in \overline{D}$

  $\Leftrightarrow M^*$ does not accept $\langle M^* \rangle$.

- Contradiction!

Alan Mathison Turing (1912-1954)

24 Years Old when he published *On computable numbers . . .*

Some More Undecidable Problems About TMs

- The Halting Problem: Given M and w, does M halt on input w?

  Proof:

  Suppose $\text{HALT}_{\text{TM}} = \{\langle M, w\rangle : M \text{ halts on } w\}$ were decided by some TM H.

  Then we could use H to decide A_{TM} as follows.

  On input $\langle M, w\rangle$,

  - Modify M so that whenever it is about to go into q_{reject}, it instead goes into an infinite loop. Call the resulting TM M'.

  - Run $H(\langle M', w\rangle)$ and do the same.

  Note that M' halts on w iff M accepts w, so this is indeed a decider for A_{TM}. $\Rightarrow\Leftarrow$.

Undecidable Problems, Continued

- For a certain fixed M_0:

  Given w, does M_0 halt on input w?

Undecidable Problems, Continued

- For a certain fixed M_0:

  Given w, does M_0 halt on input w?

What about:

- For a fixed M_0 *and* a fixed w_0, does M_0 halt on input w_0?