Undecidability of the Halting Problem

Deepak D’Souza

Department of Computer Science and Automation
Indian Institute of Science, Bangalore.

18 November 2013
Outline

1. Universal Turing machine
2. Halting Problem
3. Some corollaries
We can construct a TM $U$ that takes the encoding of a TM $M$ and its input $x$, and “interprets” $M$ on the input $x$.

$U$ accepts if $M$ accepts $x$, rejects if $M$ rejects $x$, and loops if $M$ loops on $x$. 
Encoding a TM as a $\{0, 1\}$-string

$0^n 1^m 1^k 1^s 1^t 1^r 1^u 1^v 1 0^p 1^a 1^q 1^b 10 1 0^p' 1^a' 1^q' 1^b' 100 \ldots 1 0^p'' 1^a'' 1^q'' 1^b'' 10$.

represents a TM $M$ with

- states $\{1, 2, \ldots, n\}$.
- Tape alphabet $\{1, 2, \ldots, m\}$.
- Input alphabet $\{1, 2, \ldots, k\}$ (with $k < m$).
- Start state $s \in \{1, 2, \ldots, n\}$.
- Accept state $t \in \{1, 2, \ldots, n\}$.
- Reject state $r \in \{1, 2, \ldots, n\}$.
- Left-end marker symbol $u \in \{k + 1, \ldots, m\}$.
- Blank symbol $v \in \{k + 1, \ldots, m\}$.
- Each string $0^p 1^a 1^q 1^b 10$ represents the transition $(p, a) \rightarrow (q, b, L)$.
Example encoding of TM and its input

Input is encoded as $0^a1^b1^c$ etc.

Exercise: What does the following TM do on input 001010?

Example encoding of a TM

000100001001000100010001000100
1 01000101000100 1 0100100100100 1 010101010.

[Assume accept and reject states are sink states]
Example encoding of TM and its input

Input is encoded as $0^a1^b1^c$ etc.

Exercise: What does the following TM do on input 001010?

Example encoding of a TM

00010000100101001000100010000 1 01000101000100 1 0100100100100 1 010101010.

[Assume accept and reject states are sink states]
How the universal Turing machine works

Use 3 tapes: for input $M \# x$, for current configuration, and for current state and position of head.

Repeat:
  - Execute the transition of $M$ applicable in the current config.

Accept if $M$ gets into $t$ state, Reject if $M$ gets into $r$ state.
Halting Problem for Turing machines

- Fix an encoding $enc$ of TM's as above.
- Define the language

$$\text{HP} = \{enc(M)\#enc(x) \mid M \text{ halts on } x\}.$$
Undecidability of HP

Theorem (Turing 1936)

*The language HP is not recursive.*
Proving undecidability of HP

Assume that we have a Turing machine $M$ which decides HP. Then we can compute the entries of the table below:

<table>
<thead>
<tr>
<th>$x$</th>
<th>$\epsilon$</th>
<th>0</th>
<th>1</th>
<th>00</th>
<th>01</th>
<th>10</th>
<th>11</th>
<th>000</th>
<th>001</th>
<th>010</th>
<th>011</th>
<th>111</th>
<th>\ldots</th>
</tr>
</thead>
<tbody>
<tr>
<td>$M_\epsilon$</td>
<td>L</td>
<td>H</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>H</td>
<td>H</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>\ldots</td>
</tr>
<tr>
<td>$M_0$</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>\ldots</td>
<td></td>
</tr>
<tr>
<td>$M_1$</td>
<td>H</td>
<td>H</td>
<td>L</td>
<td>H</td>
<td>L</td>
<td>H</td>
<td>H</td>
<td>L</td>
<td>H</td>
<td>L</td>
<td>H</td>
<td>\ldots</td>
<td></td>
</tr>
<tr>
<td>$M_{00}$</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>\ldots</td>
<td></td>
</tr>
<tr>
<td>$M_{01}$</td>
<td>L</td>
<td>H</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>H</td>
<td>H</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>\ldots</td>
<td></td>
</tr>
<tr>
<td>$M_{10}$</td>
<td>H</td>
<td>H</td>
<td>L</td>
<td>H</td>
<td>L</td>
<td>H</td>
<td>H</td>
<td>L</td>
<td>H</td>
<td>L</td>
<td>H</td>
<td>\ldots</td>
<td></td>
</tr>
<tr>
<td>$M_{11}$</td>
<td>L</td>
<td>H</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>H</td>
<td>H</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>\ldots</td>
<td></td>
</tr>
<tr>
<td>$M_{000}$</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>H</td>
<td>H</td>
<td>L</td>
<td>L</td>
<td>H</td>
<td>L</td>
<td>\ldots</td>
<td></td>
</tr>
<tr>
<td>\ldots</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

For each $x \in \{0, 1\}^*$ let $M_x$ denote the TM

- $M$, if $x$ is the encoding of TM $M$ with input alphabet 0, 1.
- $M_{\text{loop}}$ otherwise, where $M_{\text{loop}}$ is a one-state Turing machine that loops on all its inputs.

Table entry $(x, y)$ tells whether TM $M_x$ halts on the input $y$. Note that $y$ is an (unencoded) input in $\{0, 1\}^*$. 
A TM $N$ that behaves differently from all TM’s

- Let us assume we have a TM $M$ that decides HP.
- Then we can define a TM $N$ as follows: Given input $x \in \{0, 1\}^*$, it
  - runs as $M$ on $x\#\text{enc}(x)$.
  - If $M$ accepts (i.e. $M_x$ halts on $x$), goes to a new “looping” state $l$ and loops there.
  - If $M$ rejects (i.e. $M_x$ loops on $x$), goes to the accept state $t'$.
- $N$ essentially “complements the diagonal” of the table: Given input $x \in \{0, 1\}^*$ it halts iff $M_x$ loops on $x$.
- Consider $y = \text{enc}(N)$. Then $y$ cannot occur as any row of the table since the behaviour of $N$ differs from all rows in the table. This is a contradiction.
The constructed TM $N$ **complements** the diagonal of the table, and hence does not occur as any of the TM’s listed. This is not possible!
Complement of HP is not r.e.

Fact 1: If \( L \) and \( \overline{L} \) are both r.e. then \( L \) (and \( \overline{L} \)) must be recursive.

- Let \( M \) accept \( L \) and \( M' \) accept \( \overline{L} \).
- We can construct a total TM that simulates \( M \) and \( M' \) on given input, one step at a time.
- Accept if \( M \) accepts, Reject if \( M' \) accepts.

Fact 2: HP is recursively enumerable.

- Just run the universal TM \( U \) on input \( M\#x \); accept iff \( U \) halts (i.e. \( M \) accepts or rejects \( x \)).

Corollary

*The language \( \neg \text{HP} \) is not even recursively enumerable.*
Where HP lies

All languages over $A$

- RE
- Recursive
- CFL
- DCFL
- Regular

- $a^n b^n$
- $a^n b^n c^n$
- $\neg HP$
- $HP$
- $a^n b^n c^n$
- $a^n b^n c^n$
- $a^n b^n c^n$
- $a^n b^n$