Computational Theory
Uncomputability

Dr Russ Ross

Dixie State University—Computer and Information Technologies

Fall 2016

Adapted from notes by Harry Lewis
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 that such languages exist, by a counting argument:
  - Every decidable language is decided by a TM
  - There are only countably many TMs
  - There are uncountably many languages
  ∴ Most languages are not decidable (or even Turing-recognizable)
Reminder: Three basic facts on decidability vs. recognizability

1. If $L$ is decidable, then $L$ is r.e.
   **Proof:**
   If $M$ decides $L$, then a machine can recognize $L$ by running $M$, and then going into an infinite loop if $M$ would have halted in the $q_{reject}$ 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, $A_{TM} = \{\langle M, w \rangle : M \text{ accepts } w \}$.

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

“$A_{TM}$ is the hardest r.e. language.”

$A_{TM}$ is said to be *r.e.-complete*
A simplifying detail: every string represents some TM

- Let $\Sigma$ 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 decidable

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

<table>
<thead>
<tr>
<th></th>
<th>$w_0$</th>
<th>$w_1$</th>
<th>$w_2$</th>
<th>$w_3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$M_0$</td>
<td>$Y$</td>
<td>$N$</td>
<td>$N$</td>
<td>$Y$</td>
</tr>
<tr>
<td>$M_1$</td>
<td>$Y$</td>
<td>$Y$</td>
<td>$N$</td>
<td>$N$</td>
</tr>
<tr>
<td>$M_2$</td>
<td>$N$</td>
<td>$N$</td>
<td>$N$</td>
<td>$N$</td>
</tr>
<tr>
<td>$M_3$</td>
<td>$Y$</td>
<td>$Y$</td>
<td>$Y$</td>
<td>$Y$</td>
</tr>
</tbody>
</table>

(WLOG assume every string $w_i$ encodes a TM $M_i$)

- Entry matching $(M_i, w_j)$ is $Y$ iff $M_i$ accepts $w_j$
Thm: $A_{TM}$ is not decidable

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

<table>
<thead>
<tr>
<th></th>
<th>$w_0$</th>
<th>$w_1$</th>
<th>$w_2$</th>
<th>$w_3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$M_0$</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>$M_1$</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>$M_2$</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>$M_3$</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
</tbody>
</table>

(WLOG assume every string $w_i$ encodes a TM $M_i$)

- Entry matching $(M_i, w_j)$ is Y iff $M_i$ accepts $w_j$

- If $A_{TM}$ were decidable, then so would be the diagonal $D$ and its complement.
  - $D = \{ w_i : M_i \text{ accepts } w_i \}$
  - $\overline{D} = \{ w_i : M_i \text{ does not accept } w_i \}$

- 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 $D = \{\langle M \rangle : M \text{ does not accept } \langle M \rangle \}$.
  - $M^*(\langle N \rangle)$ 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?

  $\begin{align*}
  M^* \text{ accepts } \langle M^* \rangle \\
  \iff \langle M^* \rangle \in \overline{D} \oplus \\
  \iff M^* \text{ does not accept } \langle M^* \rangle
  \end{align*}$

- Contradiction!
Alan Mathison Turing

Alan Mathison Turing (1912–1954)

24 Years Old when he published *On computable numbers* ...
The Halting Problem

Given $M$ and $w$, does $M$ halt on input $w$?

Proof:
Suppose $HALT_{TM} = \{\langle M, w \rangle : M \text{ halts on } w \}$ 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 $q_{\text{reject}}$, 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 $A_{TM}$. $\Rightarrow \Leftarrow$. 

Dr. Russ Ross (Dixie State University)
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 $w_0$?
Further Undecidable Problems

Given \( M \), does \( M \) halt on the empty string?

**Proof** by reduction:

Suppose \( M_1 \) decided \( \{\langle M \rangle : M \text{ halts on } \varepsilon\} \).
Then \( M_1 \) could be used to decide \( \text{HALT}_{TM} \):

Given \( \langle M, w \rangle \),

Construct \( \langle M_w \rangle \), where \( M_w \) is a TM that writes \( w \) on the empty tape and then runs \( M \).

Then run \( M_1 \) on input \( \langle M_w \rangle \)

\( M_1 \) accepts \( \langle M_w \rangle \) \iff \( M_w \) halts on \( \varepsilon \) \iff \( M \) halts on \( w \)

But \( \text{HALT}_{TM} \) is undecidable. \( \Rightarrow \) \( \Leftarrow \)
"Co-X"

- For any property $X$ that a set might have, a set $S$ is **co-$X$** iff $\overline{S}$ has property $X$.
- For example, a co-finite set of natural numbers is a set that is missing on a finite number of elements.
- A co-regular language is . . . ?
- A co-recursive language is . . . ?
- What about a co-CF language?
- Proved last time:
  - A language is recursive if and only if it is both r.e. and co-r.e.
Non-r.e. Languages

**Theorem:** The following languages are not r.e.:

- $\overline{A_{TM}} = \{ \langle M, w \rangle : M \text{ does not accept } w \}$
- $\overline{HALT_{TM}} = \{ \langle M, w \rangle : M \text{ does not halt on } w \}$
- $\overline{HALT_{TM}^{\varepsilon}} = \{ \langle M \rangle : M \text{ does not halt on } \varepsilon \}$

**Proof:** If these languages were r.e. then $A_{TM}$, $HALT_{TM}$, and $HALT_{TM}^{\varepsilon}$ would be both r.e. and co-r.e. and hence recursive.
Is it possible to determine, given a TM $M$, whether $M$ accepts a finite or infinite set?

Let $A_{\text{infinite}} = \{ \langle M \rangle : L(M) \text{ is infinite} \}$. Is $A_{\text{infinite}}$ recursive?
Is it possible to determine, given a TM $M$, whether $M$ accepts a finite or infinite set?

- Let $A_{\text{infinite}} = \{ \langle M \rangle : L(M) \text{ is infinite} \}$. Is $A_{\text{infinite}}$ recursive?

- Suppose $M_2$ decides $A_{\text{infinite}}$. To decide $A_{\text{TM}}$, given $\langle M \rangle$ and $\langle w \rangle$, construct $\langle M^*_w \rangle$ so that
  - If $M$ accepts $w$ then $M^*_w$ accepts its input, regardless of what it is, and
  - If $M$ does not accept $w$ then $M^*_w$ runs forever.

- Then run $M_2$ on input $\langle M^*_w \rangle$.

- $L(M^*_w)$ is either $\Sigma^*$ (and therefore infinite) or $\emptyset$ (and therefore finite) depending on whether or not $M$ accepts $w$.

Reduce $A_{\text{TM}}$ to $A_{\text{infinite}}$; since $A_{\text{TM}}$ is undecidable, so is $A_{\text{infinite}}$. 

Dr Russ Ross  (Dixie State University)
Reading: Sipser Ch. 5
Formalizing the Notion of Reduction

- $L_1$ “reduces” to $L_2$ if we can use a “black box” for $L_2$ to build an algorithm for $L_1$.

- A function $f : \Sigma_1^* \rightarrow \Sigma_2^*$ is **computable** if there is a Turing machine that for every input $w \in \Sigma_1^*$, $M$ halts with just $f(w)$ on its tape.

- A **(mapping) reduction** of $L_1 \subseteq \Sigma_1^*$ to $L_2 \subseteq \Sigma_2^*$ is a computable function
  $$f : \Sigma_1^* \rightarrow \Sigma_2^*$$
  such that, for any $w \in \Sigma_1^*$,
  $$w \in L_1 \text{ iff } f(w) \in L_2$$

  We write $L_1 \leq_m L_2$. 

**Lemma:** If $L_1 \leq_m L_2$, then

- if $L_2$ is decidable (resp., r.e.), then so is $L_1$;
- if $L_1$ is undecidable (resp., non-r.e.), then so is $L_2$. 

![Diagram showing reductions and decidability](image)
Examples of Reductions from Last Lecture

▶ For every Turing-recognizable $L$, $L \leq_m A_{TM}$.

▶ $A_{TM} \leq_m HALT_{TM}$.

▶ $HALT_{TM} \leq_m HALT_{TM}^{\varepsilon}$. 
Rice’s Theorem

Informally: *every* (nontrivial) property of Turing-recognizable languages is undecidable.

**Rice’s Theorem:** Let $\mathcal{P}$ be any subset of the class of r.e. languages such that $\mathcal{P}$ and its complement are both nonempty. Then the language $L_\mathcal{P} = \{ \langle M \rangle : L(M) \in \mathcal{P} \}$ is undecidable.

Thus, given a TM $M$, it is undecidable to tell if

- $L(M) = \emptyset$,
- $L(M)$ is regular,
- $|L(M)| = \infty$, etc.
Proof of Rice’s Theorem

- We will reduce $L_\epsilon$ to $L_P$.
- Suppose without loss of generality that $\emptyset \notin P$.
- Pick any $L_0 \in P$ and say $L_0 = L(M_0)$.
- Define $f(\langle M \rangle) = \langle M' \rangle$, where $M'$ is a TM that on input $w$,
  - first simulates $M$ on input $\epsilon$
  - then simulates $M_0$ on input $w$
- **Claim:** $f$ is a mapping reduction from $L_\epsilon$ to $L_P$.
- Since $L_\epsilon$ is undecidable, so is $L_P$. 
Recursion Theory over $\mathbb{N}$

- We have presented this theory as a theory of languages.
- Classically it is treated as a theory of sets of numbers.
- The two are equivalent since strings can be converted to numbers (treating strings as numerals, for example) and v.v.
- So it makes sense to say “The set of primes is recursive.”
Recursive functions

- A function $f$ is recursive if $f$ is computable
- e.g. if there is a TM that always leaves $f(w)$ on the tape when started with input $w$
- Similarly we can speak of a recursive function from numbers to numbers
- Thm: A set is r.e. iff it is the range of a recursive function or is $\emptyset$
Reading: Sipser Ch. 5
There is no algorithm to determine, given any grammar \( G \) and any string \( w \), whether \( w \in L(G) \).

Proof: Suppose there were such a decision procedure. Then we could use it to solve the halting problem:

Given \( M \) and \( w \), to determine if \( M \) halts on input \( w \), construct a grammar \( G \) such that \( L(M) = L(G) \) and determine if \( w \in L(G) \).

Since the halting problem is unsolvable, so is this problem.

There is a particular grammar \( G_0 \) for which this problem is unsolvable: namely, the grammar for the universal TM.
Two-Counter Machines

- A counter machine can add and subtract 1 from its registers and check if they are zero

- **Theorem:** The halting problem is unsolvable even for 2-counter machines.

- **Proof:**
  1. One TM tape to two pushdown stores
  2. One pushdown store to two counters
  3. Four counters to two counters
An Undecidable Problem about Context Free Grammars

**Theorem:** It is undecidable to determine, given CFGs $G_1$ and $G_2$, whether $L(G_1) \cap L(G_2) = \emptyset$.

**Proof:** Reduction from $\{\langle G, w \rangle : G$ is a general grammar generating $w\}$

- Given $\langle G, w \rangle$, we can construct grammars $G_1$, $G_2$ such that:
  
  $L(G_1) = \{C_1\#D_1^R\#C_2\#D_2^R\#\cdots\#C_n\#D_n^R : n \geq 1, \text{ and for each } i, C_i \Rightarrow_G D_i\}$

  $L(G_2) = \{S\#C_2^R\#C_2\#C_3^R\#\cdots\#C_n^R\#C_n\#w^R : n \geq 1, \text{ and the } C_i \text{ are arbitrary strings}\}$

- $G_1$ generates pairs of unrelated one-step derivations

- $G_2$ has $S$ and $w$ at beginning and end, and in between pairs match (even positions reversed)
Intersection of CFLs, continued

- Any string in $L(G_1)$ or $L(G_2)$ has an odd number of #s
- Any string in $L(G_1) \cap L(G_2)$ is a derivation of $w$ in $G$ (every other intermediate string is reversed)
- So $L(G_1) \cap L(G_2)$ is nonempty iff $w$ is derivable in $G$
- So $\langle G, w \rangle \mapsto \langle G_1, G_2 \rangle$ is a reduction from general grammar derivability to $\{\langle G_1, G_2 \rangle : L(G_1) \cap L(G_2) \neq \emptyset \}$.

Verifying computations is easier than carrying them out!
**Tiling**: Given a finite set of patterns for square tiles:

![Tile Patterns]

Is it possible to tile the whole plane with tiles of these patterns in such a way that the abutting edges match?

**Theorem**: Tiling is undecidable.
Tiling, continued

Variant of tiling: fix the tile at the origin and ask whether the first quadrant can be tiled (easier to show undecidability).

Proof by reduction from $\overline{L_\varepsilon}$:

1. $\langle M \rangle \mapsto f$ sets of tiles so that:
   - $M$ does not halt on $\varepsilon$ $\iff$ $f(\langle M \rangle)$ tiles the first quadrant.

2. View computation of $M$ as “tableau”, filling first quadrant with elements of $C = Q \cup \Gamma$, each row being a configuration of $M$.

3. Computation valid iff every $2 \times 3$ window consistent with transition function of $M$ (and bottom row is correct initial configuration).

4. Each tile represents a $2 \times 3$ window of tableau. Edge colors force consistency with neighbors on overlap.
Diophantine Equations

These are equations like

\[ x^3 y^3 + 13xyz = 4u^2 - 22 \]

The coefficients and the exponents have to be integers. (No variables in the exponents!)

The question is whether the equation can be satisfied (made true) by substituting integers for the variables—this is known as Hilbert’s 10th problem.
Diophantus of Alexandria (200–284 AD)

“God gave him his boyhood one-sixth of his life, One twelfth more as youth while whiskers grew rife; And then yet one-seventh ere marriage begun; In five years there came a bouncing new son. Alas, the dead child of master and sage After attaining half the measure of his father’s life chill fate took him. After consoling his fate by the science of numbers for four years, he ended his life.”

Other problems concerning triangular arrays, etc., gave rise to quadratic equations.

Fermat’s statement of his “Last Theorem” was in the margin of his copy of Diophantus.
10. Determination of the Solvability of a Diophantine Equation.

Given a diophantine equation with any number of unknown quantities and with rational integral numerical coefficients: *To devise a process according to which it can be determined by a finite number of operations whether the equation is solvable in rational integers.*

**Thm:** (Matiyasevish, 1970): Hilbert’s 10th problem is unsolvable.
Relation to Gödel’s Incompleteness Theorem

- Axiom systems for mathematics, e.g.
  - Peano arithmetic—attempt to capture properties of \( \mathbb{N} \)
    - E.g. mathematical induction:
      
      If \( P[0] \)
      
      and, for all \( n, P[n] \Rightarrow P[n + 1] \),
      
      then for all \( n, P[n] \)
  
  - Zermelo-Frankel-Choice set theory (ZFC)—enough for all of modern mathematics

- Proofs of theorems from these axioms defined by (simple) rules of mathematical logic.
The Decision Problem (for Mathematics)

- **Entscheidungsproblem** is German for “Decision Problem”
- **The** Decision Problem is the problem of determining whether a mathematical statement is provable
- **Proposition:** Set of provable theorems is Turing-recognizable.
- **Is it decidable?**
Theorem [Church, Turing]: Set of provable theorems is undecidable.

Proof sketch:
- Reduce from $\text{HALT}^\varepsilon_{TM}$
- $\langle M \rangle \leftrightarrow$ mathematical statement $\phi_M = "M \text{ halts on } \varepsilon"$
- $M \text{ halts on } \varepsilon \Rightarrow \phi_M \text{ has a proof.}$
- $M \text{ does not halt on } \varepsilon \Rightarrow \phi_M \text{ not true.}$
Incompleteness of Mathematics

- **Gödel’s Incompleteness Theorem**: Some true statement is not provable.

**Proof sketch:**
- For every statement $\phi$, either $\phi$ or $\neg\phi$ is true.
- Suppose all true statements provable.
  - For all statements $\phi$, exactly one of $\phi$ and $\neg\phi$ is provable.
  - Set of provable theorems is r.e. and co-r.e.
  - Set of provable theorems recursive.
- Contradiction.

- See Sipser Chapter 6 for more on this & other advanced topics on computability theory.