Decidability

prepared by John Sieg, UMass Lowell
content largely from Michael Sipser’s Introduction to the Theory of Computation, second edition, Chapter 4

Overview

• Show a problem is decidable by constructing a Turing machine (algorithm) that decides it.
• Show that a problem is undecidable by proving that no Turing machine decides it.

Theorem 4.1: Acceptance Problem for DFAs

\[ A_{DFA} = \{ \langle B, w \rangle \mid B \text{ is a DFA that accepts input string } w \} \]

Theorem 4.1: \( A_{DFA} \) is a decidable language.

proof: The following TM decides \( A_{DFA} \):
M=“On input \( \langle B, w \rangle \), where \( B \) is a DFA:
1. Simulate \( B \) on input \( w \).
2. If the simulation ends in an accept state, \text{accept}. If it ends in a nonaccepting state, \text{reject}.”

Languages Related to \( A_{DFA} \)

Similarly, the following languages are decidable:

\[ A_{NFA} = \{ \langle B, w \rangle \mid B \text{ is a NFA that accepts input string } w \} \]

\[ A_{REX} = \{ \langle B, w \rangle \mid B \text{ is a regular expression that generates string } w \} \]
Theorem 4.4

\[ E_{\text{DFA}} = \{ \langle A \rangle \mid A \text{ is a DFA and } L(A) = \emptyset \} \]

**Theorem 4.4**: \( E_{\text{DFA}} \) is a decidable language.

**proof**: The following TM decides \( E_{\text{DFA}} \):

- **1.** Mark the start state of \( A \).
- **2.** Repeat until no new states get marked:
  - **3.** Mark any state with a transition from a marked state.
  - **4.** If no accept state is marked, accept. Otherwise, reject.

Theorem 4.7

\[ A_{\text{CFG}} = \{ \langle G, w \rangle \mid G \text{ is a CFG that generates string } w \} \]

**Theorem 4.7**: \( A_{\text{CFG}} \) is a decidable language.

**proof**: The following TM decides \( A_{\text{CFG}} \):

- **1.** Convert \( G \) to an equivalent Chomsky normal form grammar.
- **2.** List all derivations with \( 2n-1 \) steps, where \( n \) is length of \( w \) (except if \( n=0 \) ...).
- **3.** If any of these derivations generate \( w \), accept. Otherwise, reject.

Theorem 4.9: CFLs are Decidable

**Theorem 4.9**: Every CFL is decidable.

**proof**: Given a CFL \( A \), let \( G \) be a CFG for \( A \).

- **1.** Run TM \( S \) from Theorem 4.7 on \( \langle G, w \rangle \).
- **2.** If this machine accepts, accept. If it rejects, reject.

Counting

- **Definition 4.12**: Two sets have the same size if there is a one-to-one, onto function (bijection, correspondence) between them.

- **Definition 4.14**: A set is countable if it is finite or has the same size as \( \mathbb{N} \).

- **Example**: \( \{2, 4, 6, \ldots\} \) is countable.
- **Example**: \( \mathbb{Q} \) is countable.
- **Example**: Given alphabet \( \Sigma \), \( \Sigma^* \) is countable.
- **Example**: \( \mathbb{R} \) is uncountable.
Corollary 4.18

Corollary 4.18: Some languages are not Turing-recognizable.

proof: The set $B$ of infinite binary sequences is uncountable.

Let $L$ be the set of languages over alphabet $\Sigma$. Each language has a unique sequence called the characteristic sequence.

Ex. $\Sigma = \{a, b\}$ $L = \{a^n b^n \mid n \geq 0\}$

$\Sigma^* = \{\varepsilon, a, b, aa, ab, ba, bb, aab, \ldots\}

\chi_L = 1\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 1\ \ldots$

$B$ uncountable implies $L$ uncountable.

But the number of Turing machines is countable!

The Acceptance (Halting) Problem

Theorem 4.11: The language $A_{TM} = \{\langle M, w \rangle \mid M$ is a TM and $M$ accepts $w\}$ is undecidable.

proof: by contradiction. Assume $A_{TM}$ is decidable.

Suppose $H$ is a decider for $A_{TM}$.

$H(\langle M, w \rangle) = accept$ if $M$ accepts $w$

$reject$ if $M$ does not accept $w$

Construct $D$ as follows:

$D = \langle M, \langle M \rangle \rangle.$

1. Run $H$ on input $\langle M, \langle M \rangle \rangle$.

2. Output the opposite of what $H$ outputs.

The Acceptance Problem, cont.

$D(\langle M \rangle) = accept$ if $M$ does not accept $\langle M \rangle$

$reject$ if $M$ accepts $\langle M \rangle$

Now run $D$ on itself.

$D(\langle D \rangle) = accept$ if $D$ does not accept $\langle D \rangle$

$reject$ if $D$ accepts $\langle D \rangle$

Both possible cases lead to a contradiction.

A Turing Unrecognizable Language

A language is **co-Turing-recognizable** if it is the complement of a Turing-recognizable language.

Theorem 4.22: A language is decidable iff it is Turing-recognizable and co-Turing-recognizable.

Corollary 4.23: $A_{TM}$ is not Turing-recognizable.