Reducibility

prepared by John Sieg, UMass Lowell
content largely from Michael Sipser’s Introduction to
the Theory of Computation, second edition, Chapter 5

Overview

• Problem A is reducible to Problem B
  means that a solution to B can be used to
  solve A. (We’ll use the more precise term
  “mapping reducible” later.)
• If A is reducible to B and B is decidable,
  then A is decidable.
• If A is reducible to B and A is undecidable
  then B is undecidable.

Theorem 5.1: The (Real) Halting Problem

\[ \text{HALT}_{TM} = \{ \langle M, w \rangle \mid M \text{ is a TM that halts on input } w \} \]

**Theorem 5.1:** \( \text{HALT}_{TM} \) is undecidable.
**proof:** by contradiction. Assume there is a TM \( R \) that
decides \( \text{HALT}_{TM} \). Construct TM \( S \) to decide \( A_{TM} \) (from
Theorem 4.11):

\[ S=\text{“On input } \langle M, w \rangle \text{”, where } M \text{ is encoding of a TM:
1. Run TM } R \text{ on input } \langle M, w \rangle
2. If } R \text{ rejects, reject.}
3. If } R \text{ accepts, simulate } M \text{ on } w \text{ until it halts.}
4. If } M \text{ has accepted, accept; otherwise reject.”}
“Reject inputs not of form \( \langle M, w \rangle \). Always do similar
checks.

**Theorem 5.2**

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

**Theorem 5.2:** \( E_{TM} \) is undecidable.
**proof:** by contradiction. Assume there is a TM \( R \) that
decides \( E_{TM} \). Construct TM \( S \) to decide \( A_{TM} \).

Construct \( M_1 \) as follows:

\[ M_1 =\text{“On input } x\text{:
1. If } x \neq w, \text{ reject.
2. If } x = w, \text{ run } M \text{ on input } w \text{ and accept if } M \text{ does.”}

S=“On input \( \langle M,w \rangle \), where \( M \) is encoding of a TM:
1. Use description of \( M \) and \( w \) to construct \( M_r \).
2. Run \( R \) on input \( \langle M_r \rangle \)
3. If \( R \) accepts, reject; if \( R \) rejects, accept.”
Theorem 5.4

$EQ_{TM} = \{ \langle M_1, M_2 \rangle \mid M_1 \text{ and } M_2 \text{ are TMs and } L(M_1) = L(M_2) \}$

**Theorem 5.4**: $EQ_{TM}$ is undecidable.

**Sketch of proof**: by contradiction. Assume there is a TM $R$ that decides $EQ_{TM}$. Construct TM $S$ to decide $E_{TM}$.

$S$ = “On input $\langle M \rangle$, where $M$ is a TM:
1. Run $R$ on input $\langle M, M_1 \rangle$, where $M_1$ is a TM that rejects all inputs.
2. If $R$ accepts, accept; if $R$ rejects, reject.”

Computation Histories

- A **computation history** is a sequence $C_1, C_2, \ldots, C_m$ where $C_1$ is a start configuration and $C_m$ is accept or reject configuration.
- Accepting computation histories vs. rejecting computation histories.
- Example: $q_{start}abc\, dq_2 bc\, deq_3 c\, defq_{accept}$

Linear Bounded Automata

- A **linear bounded automaton** is a TM which is not allowed to move off the portion of the tape holding the input. If it tries to move off the input portion, it stays put instead.
- You may choose the tape alphabet to be $k$ times as large as the input alphabet, so the memory can be thought of as being $k$ times as large as the input size.

```
 a b c b c
```

Lemma 5.8

**Lemma 5.8**: Let $M$ be an LBA with $q$ states and $g$ symbols in the tape alphabet. There are exactly $qng^n$ distinct configurations of $M$ for a tape of length $n$.

**Sketch of proof**: There is a distinct configuration for each of $q$ states, each of $n$ R/W head positions, and each $g^n$ possible strings on the tape.
Theorem 5.9

\[ A_{LBA} = \{ \langle M, w \rangle \mid M \text{ is an LBA that accepts string } w \} \]

**Theorem 5.9:** \( A_{LBA} \) is decidable.

**proof:** The following algorithm decides \( A_{LBA} \).

\[ L = \text{"On input } \langle M, w \rangle, \text{ where } M \text{ is an LBA:} \]

1. Simulate \( M \) on \( w \) for \( q^n \) steps or until it halts.
2. If \( M \) has halted, accept if it has accepted and reject if it has rejected. If it has not halted, reject.”

Theorem 5.10

\[ E_{LBA} = \{ \langle M \rangle \mid M \text{ is an LBA and } L(M) = \emptyset \} \]

**Theorem 5.10:** \( E_{LBA} \) is undecidable.

**sketch of proof:** by contradiction. Assume there is a TM \( R \) that decides \( E_{LBA} \). Construct TM \( S \) to decide \( A_{TM} \).

Given \( \langle M, w \rangle \), a TM can construct another TM \( B \) that checks whether its input is an accepting computation history of \( M \) on \( w \):

\[
B \rightarrow \ldots \# \ x \ b_1 \ a \ b \# \ x \ x \ q_1 \ b \# \ldots
\]

Post Correspondence Problem

Given a set of dominoes

\[ P = \left\{ \begin{bmatrix} t_1 \end{bmatrix}, \begin{bmatrix} t_2 \end{bmatrix}, \ldots, \begin{bmatrix} t_k \end{bmatrix} \right\} \]

is there a match, i.e., a sequence \( i_1, i_2, \ldots, i_m \) such that

\[ t_1 \ t_2 \ \ldots \ t_m = b_{i_1} \ b_{i_2} \ \ldots \ b_{i_m} \]

Not all the dominoes need to be used. Dominoes may be used more than once.
Post Correspondence Problem Example

Given dominoes

\[ P1 = \left\{ \frac{a}{bb} , \frac{a}{a} , \frac{bb}{cc} , \frac{ca}{a} \right\} \]

Here's a match:

\[ \frac{a}{ab} \quad \frac{bb}{cc} \quad \frac{ca}{a} \]

Theorem 5.15

\[ PCP = \{ (P) \mid P \text{ is an instance of the Post correspondence problem with a match} \} \]

**Theorem 5.15**: \( PCP \) is undecidable.

**Sketch of proof**: by contradiction

Show that \( A_{TM} \) can be reduced to \( PCP \).

Show first that \( A_{TM} \) can be reduced to \( MP_{PCP} \),
where \( MP_{PCP} \) is like \( PCP \), except that a specific tile is required to be first.

It's then easy to show map a solution \( P \) of \( MP_{PCP} \)
on to a solution \( P \) of \( PCP \) (omitted here).

Theorem 5.15, cont'd

Given \( M = (Q, \Sigma, I, \delta, q_{accept}, q_{reject}) \) and

\( w = w_1 \ldots w_n \) to "simulate" \( A_{TM} \) on \( (M, w) \):

Part 1: Put \( \frac{\#}{\#q_0w_1w_2\ldots w_n\#} \) in \( P' \) as the
domino that must occur first in any match.

Part 2: For every \( a, b \in I \) and every \( q, r \in Q \) where \( q \neq q_{reject} \),
if \( \delta(q, a) = (r, b, L) \), put \( \frac{ca}{ar} \) into \( P' \).

Part 3: For every \( a, b, c \in I \) and every \( q, r \in Q \) where \( q \neq q_{reject} \),
if \( \delta(q, a) = (r, b, L) \), put \( \frac{ca}{ar} \) into \( P' \).

Part 4: For every \( a \in I \) put \( \frac{a}{a} \) into \( P' \).

Part 5: Put \( \frac{\#}{\#} \) and \( \frac{\#}{\#} \) into \( P' \).
Theorem 5.15, cont’d

Part 6: For every $a \in \Gamma$
put $\left[ \begin{array}{c} a \text{ Accept} \\ \text{Accept} \end{array} \right]$ and $\left[ \begin{array}{c} \text{Accept} a \\ \text{Accept} \end{array} \right]$ into $\mathcal{P}$.

Part 7: Put $\left[ \begin{array}{c} \text{Accept} \\ \# \# \end{array} \right]$ into $\mathcal{P}$.

Now we must argue that the resulting $\text{MPCP}$ has a match iff $M$ accepts $w$ (omitted here).

Computable Functions

Definition 5.17: A function $f: \Sigma^* \rightarrow \Sigma^*$ is a **computable function** if some TM $M$, on every input $w$, halts with just $f(w)$ on its tape.

Ex. $f(n) = n+1$, $f\langle \langle n, m \rangle \rangle = n+m$, $f\langle \langle n, m \rangle \rangle = n \mod m$

Ex. $f\langle \langle M \rangle \rangle = M$ where $M$ is a non-deterministic TM and $M'$ is the equivalent deterministic TM generated by construction in Theorem 3.16.

Ex. $f\langle \langle M, w \rangle \rangle = B$, the LBA of Theorem 5.10 that checks whether its input is an accepting computation history of $M$ on $w$

Formalizing Reducibility

Definition 5.20: Language $A$ is **mapping reducible** (**many-one reducible**) to language $B$, written $A \leq_m B$, if there is a computable function $f: \Sigma^* \rightarrow \Sigma^*$, where for every $s$,
\[ s \in A \text{ iff } f(s) \in B. \]

The function $f$ is called a **reduction** of $A$ to $B$.

More About Mapping Reducibility

Theorem 5.22: If $A \leq_m B$ and $B$ is decidable, then $A$ is decidable.

Corollary 5.23: If $A \leq_m B$ and $A$ is undecidable, then $B$ is undecidable.

Theorem 5.28: If $A \leq_m B$ and $B$ is Turing-recognizable, then $A$ is Turing-recognizable.

Corollary 5.23: If $A \leq_m B$ and $A$ is not Turing-recognizable, then $B$ is not Turing-recognizable.
Theorem 5.22

**Theorem 5.22**: If \( A \leq_m B \) and \( B \) is decidable, then \( A \) is decidable.

![Diagram]

To decide whether \( s \in A \), compute \( f(A) \) and check whether \( f(A) \in B \). Here is a decider for \( B \), given a decider \( M_A \) for \( A \):

\[
M_B = \text{"On input } s:\n\begin{enumerate}
\item Compute } f(A); \\
\item Run } M_A \text{ on } f(A) \text{ and output whatever } M_A \text{ outputs."}
\]

Example 5.24

Theorem 5.1 used a reduction \( f \) from \( A_{TM} \) to \( HALT_{TM} \). In that theorem’s proof, it was claimed that a decider for \( HALT_{TM} \) could be used to build a decider for \( A_{TM} \).

Formally, we must show \( A_{TM} \leq_m HALT_{TM} \).
Then we can apply Corollary 5.23.

Example 5.24, cont’d

\( A_{TM} \leq_m HALT_{TM} ? \)
This machine \( F \) computes a reduction \( f : \)
\[
F = \text{"On input } \langle M, w \rangle * \text{, where } M \text{ is a TM:} \\
1. Construct the following machine } M':\n\begin{enumerate}
\item \text{"On input } x:\n\begin{enumerate}
\item Run } M \text{ on } x. \\
\item If } M \text{ accepts, accept.} \\
\item If } M \text{ rejects, enter a loop."}
\end{enumerate}
\item Output } \langle M', w \rangle ." \\
*Reject inputs not of form } \langle M, w \rangle .
\]

Example 5.24, cont’d

More formal proof of Theorem 5.1:

We show \( HALT_{TM} \) is undecidable by reducing a known undecidable problem to it. Specifically, we show that \( A_{TM} \leq_m HALT_{TM} \).

This machine \( F \) computes a reduction \( f : \)
\[
F = \text{"See previous slide. } >> \\
\text{Now show that } s \in A_{TM} \text{ iff } f(s) \in HALT_{TM} .
\]
This means that \( s \) (the input of \( F \)) is of form \( \langle M, w \rangle \) where \( M \) accepts \( s \) iff \( f(s) \) (the output of \( F \)) is of form \( \langle M', w \rangle \) where \( M' \) halts on \( w \). Obvious.
Example 5.25

The proof of Theorem 5.15 used two reductions:
\[ A_{TM} \leq_m MPCP \text{ and } MPCP \leq_m PCP. \] Because \( \leq_m \) is transitive, we have \( A_{TM} \leq_m PCP \).

What is the reduction \( f \) for \( A_{TM} \leq_m MPCP \)?

\( f \) maps each \( \langle M, w \rangle \) onto a set of dominoes as described in the proof.

Example 5.27

Theorem 5.2 used a reduction \( f \) from \( A_{TM} \) to \( E_{TM} \).

\( f \) sends each \( \langle M, w \rangle \) to a \( M' \) such that \( M \) accepts \( w \) iff \( M' \) does not recognize \( \emptyset \), i.e., \( M' \notin E_{TM} \). Function \( f \) is computed by TM \( S \) given in the proof of theorem 5.2.

Thus \( \overline{E_{TM}} \) is undecidable. Decidability is closed under complementation, so \( E_{TM} \) is also undecidable.

Rice’s Theorem
(Exercise 5.28)

Rice’s Theorem: Let \( P \) be a language consisting of TM descriptions such that

- \( P \) is non-trivial, i.e., it contains some but not all TM descriptions
- \( P \) is a property of the TM’s language, i.e., if \( L(M_1) = L(M_2) \), then \( \langle M_1 \rangle \in P \) iff \( \langle M_2 \rangle \in P \)

\( P \) is undecidable.

Consequences of Rice’s Theorem

Every non-trivial property of TM’s is undecidable, including:

- Is \( \epsilon \in L(M) \)?
- Is \( s \in L(M) \) for a given string?
- Is \( L(M) \) infinite?
- Is \( L(M) \) regular?

But don’t use Rice’s Theorem in homework or quizzes unless directed to do so!