# CS301: Theory of Computation
# Section 5.1 - Reducibility in Language Theory

Daniel White

Gettysburg College
Computer Science

Spring 2024

We explored one example last chapter of a problem that is computationally unsolvable. *i.e.* $A_{\mathrm{TM}}$ is undecidable.

Here we investigate more unsolvable problems in language theory using the method of **reduction**.

Reduction is a way of converting one problem to another in such a way that a solution to the second problem can be used to solve the first problem.

When problem $A$ is reducible to $B$, solving $A$ cannot be harder than solving $B$ since giving a solution to $B$ results in a solution to $A$.

**Theorem (pg. 216)**

$A$ is reducible to $B$ and $B$ is decidable $\rightarrow$ $A$ is decidable
$A$ is undecidable and reducible to $B$ $\rightarrow$ $B$ is undecidable

We have already determined the undecidability of $A_{\text{TM}}$, and will use this to solve the **halting problem**, where we ask if the language $H_{\text{TM}} = \{\langle M, w\rangle \mid M \text{ is a TM and } M \text{ halts on input } w\}$ is decidable. We will answer in the negative.

**exercise:** Phrase the halting problem in "normal" programming terms.

## Theorem 5.1 (pg. 216)

$H_{\text{TM}} = \{\langle M, w\rangle \mid M \text{ is a TM and } M \text{ halts on input } w\}$ is undecidable.

*Proof.* By way of contradiction, suppose that $H_{\text{TM}}$ is decidable by Turing machine $R$. Let us now construct a TM $S$ which decides $A_{\text{TM}}$ by using $R$ as a subroutine.

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

1. Run TM $R$ on input $\langle M, w\rangle$.
2. If $R$ rejects, *reject*.
3. Otherwise . . .

**exercise:** Finish this proof.

homework: 5.10

## Theorem 5.2 (pg. 217)

$E_{TM} = \{\langle M \rangle \mid M \text{ is a TM and } L(M) = \varnothing\}$ is undecidable.

*Proof.* Given a string $w$, define TM $M_w$ on input $x$ as

1. If $x \neq w$, *reject*.
2. If $x = w$, run $M$ on input $w$ and *accept* if $M$ does.

By way of contradiction, suppose that $E_{TM}$ is decidable by Turing machine $R$. Let us now construct a TM $S$ which decides $A_{TM}$ by using $R$ and $M_w$ as a subroutine. On input $\langle M, w \rangle$:

1. Construct the TM $M_w$ described above.
2. Run $R$ on input $\langle M_w \rangle$.
3. ...

**exercise:** Finish this proof.

**Theorem 5.4 (pg. 220)**

$Q_{\mathrm{TM}} = \{\langle M_1, M_2\rangle \mid M_1, M_2 \text{ are TMs with } L(M_1) = L(M_2)\}$ is undecidable.

*Proof.* By way of contradiction ...

**exercise:** Finish this proof.

**exercise:** Phrase the problem of the decidability of $Q_{\mathrm{TM}}$ in "normal" programming terms.

$\boxed{\text{homework: Thm 5.3}}$

## Definition 5.5 (pg. 221)

Let $M$ be a TM and $w$ an input string. An **accepting computation history** for $M$ on $w$ is a sequence of configurations

$$C_1, C_2, \ldots C_\ell,$$

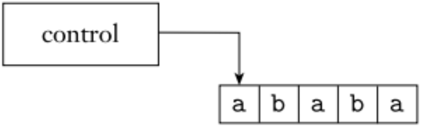
where $C_1$ is the start configuration, $C_\ell$ is an accepting configuration, and each $C_i$ follows legally from $C_i - 1$ according to the rules of $M$.

A **rejecting computation history** for $M$ on $w$ is defined similarly, except that $C_\ell$ is a rejecting configuration.

Note that computation histories are finite sequences since they are defined only when a machine halts. *(Hence, we can provide such histories to another TM.)*

## Definition 5.6 (pg. 221)

A **linear bounded automaton (LBA)** is a restricted type of Turing machine wherein the tape head isn't permitted to move off of the portion of the tape containing the input.




If the machine tries to move its head off either end of the input, the head stays where it is instead.

## Lemma 5.8 (pg. 222)

Let $M$ be a LBA with $q$ states and $g$ symbols in the tape alphabet. There are exactly $qng^n$ distinct configurations of $M$ given an input string of length $n$.

## Theorem 5.9 (pg. 222)

$A_{\text{LBA}}$ is decidable.

*Proof.* Let $M$ be a LBA and $w$ a string. Define the decider $L$ so that on input $\langle M, w \rangle$:

1. Simulate $M$ on $w$ for $qng^n$ steps or . . .
2. . . .

**exercise:** Finish this proof.

## Theorem 5.10 (pg. 223)

$E_{\text{LBA}}$ is undecidable.

*Proof.* By way of contradiction, suppose $E_{\text{LBA}}$ is decided by some TM, $R$. As an intermediate step, we'll construct a LBA $B_{M,w}$ that verifies whether a given string is an accepting computational history for TM $M$ on input string $w$. Now build a decider $S$ for $A_{\text{TM}}$. On input $\langle M, w\rangle$:

1. Construct LBA $B_{M,w}$ from $M$ and $w$ as described above.
2. Run $R$ on input $\langle B\rangle$.
3. ...

**exercise:** Why is $B$ an LBA?
**exercise:** $B$ implementation details?
**exercise:** Finish this proof.

homework: Thm 5.13