How to write a 21st century proof

Leslie Lamport

A method of writing proofs is described that makes it harder to prove things that are not true. The method, based on hierarchical structuring, is simple and practical. The author’s twenty years of experience writing such proofs is discussed.

Source: lamport.azurewebsites.net/pubs/proof.pdf

Why should we read this?

- Structured proofs are just a method of displaying your proofs in a brief, yet rigorous way
- I will be doing structured proofs in this lesson, you can use this method of presenting proofs in the test!
Today we will learn...

- Computable functions
- Mapping reducible
- Mapping reducibility and decidability/undecidability
- Mapping reducibility and Turing recognition/unrecognition

Section 5.3.
Motivation

If $A$ is regular, then $X_A$ decidable.

$$X_A = \{ \langle D \rangle \mid D \text{ is a DFA} \land L(D) \cap A \neq \emptyset \}$$

**Proof.** If $A$ is regular, then let $C'$ be the DFA that recognizes $A$. Let intersect be the implementation of $\cap$ and $E_{DFA}$ the decider of $E_{DFA}$. The following is the decider of $X_A$.

```python
def X_A(D):
    return not E_DFA(intersect(C, D))
```

We reduced the problem of checking if $X_A$ is decidable in terms of checking if $E_{DFA}$.

Can we generalize this process?
Motivation

If $A$ is regular, then $X_A$ decidable.

$$X_A = \{ \langle D \rangle \mid D \text{ is a DFA} \land L(D) \cap A \neq \emptyset \}$$

Proof (2\textsuperscript{nd} try).

1. Let $L_1(D) = L(D) \cap A$ where $D$ is a DFA.
Motivation

If $A$ is regular, then $X_A$ decidable.

$$X_A = \{ \langle D \rangle \mid D \text{ is a DFA } \land L(D) \cap A \neq \emptyset \}$$

Proof (2\textsuperscript{nd} try).

1. Let $L_1(D) = L(D) \cap A$ where $D$ is a DFA.
2. For any $D$ we have that $L_1(D)$ is regular. (Proof?)
Motivation

If $A$ is regular, then $X_A$ decidable.

$$X_A = \{ \langle D \rangle \mid D \text{ is a DFA} \land L(D) \cap A \neq \emptyset \}$$

Proof (2nd try).

1. Let $L_1(D) = L(D) \cap A$ where $D$ is a DFA.
2. For any $D$ we have that $L_1(D)$ is regular. (Proof?)
3. Let $D_{DA}$ be the DFA that recognizes $L_1(D)$. (Proof?)
Motivation

If $A$ is regular, then $X_A$ decidable.

$$X_A = \{ \langle D \rangle \mid D \text{ is a DFA} \land L(D) \cap A \neq \emptyset \}$$

Proof (2nd try).

1. Let $L_1(D) = L(D) \cap A$ where $D$ is a DFA.
2. For any $D$ we have that $L_1(D)$ is regular. (Proof?)
3. Let $D_{DA}$ be the DFA that recognizes $L_1(D)$. (Proof?)
4. $\langle D \rangle \in X_A$ iff $\langle L_1(D) \rangle \in \overline{E_{DFA}}$ (Proof?)
Motivation

If $A$ is regular, then $X_A$ decidable.

$$X_A = \{ \langle D \rangle \mid D \text{ is a DFA} \land L(D) \cap A \neq \emptyset \}$$

Proof ($2^{nd}$ try).

1. Let $L_1(D) = L(D) \cap A$ where $D$ is a DFA.
2. For any $D$ we have that $L_1(D)$ is regular. (Proof?)
3. Let $D_{DA}$ be the DFA that recognizes $L_1(D)$. (Proof?)
4. $\langle D \rangle \in X_A$ iff $\langle L_1(D) \rangle \in \overline{E}_{\text{DFA}}$ (Proof?)
5. The test $\langle L_1(D) \rangle \in \overline{E}_{\text{DFA}}$ is decidable, and equivalent to testing $\langle D \rangle \in X_A$, so the latter is decidable?

†: Recall that if $A$ decidable, then $\overline{A}$ decidable (Lesson 21).
Mapping reducibility

Intuition

If we can establish an equivalence up to some function, then we can reduce a problem into another known problem solved in another language.

Example

4. (Mapping-reducibility): \( \langle D \rangle \in X_A \iff \langle L_1(D) \rangle \in \overline{E}_{DFA} \)

5. (Decidability): The test \( \langle L_1(D) \rangle \in \overline{E}_{DFA} \) is decidable, and equivalent to testing \( \langle D \rangle \in X_A \), so the latter is decidable?

We will now implement a framework on reducibility
Mapping reducibility
Computable function

Definition 5.17

We say that

\[ f : \Sigma^* \rightarrow \Sigma^* \]

is a **computable** function if there exists a Turing Machine \( M \) that when given \( w \) halts and results in \( f(w) \) on its tape.

**Intuition**

This is a **total** function (terminates for all inputs) encoded in terms of a Turing Machine.
Mapping reducible

Definition 5.20

Language $A$ is **mapping reducible** to language $B$, notation $A \leq_m B$ if there is a computable function $f$, where for every $w$,

$$w \in A \iff f(w) \in B$$

What can we do with mapping reducible?

- Convert membership testing in $A$ into membership testing in $B$
Example

\[ X_A = \{ \langle D \rangle \mid D \text{ is a DFA} \land L(D) \cap A \neq \emptyset \} \]

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

We show that \( X_A \leq_m E_{DFA} \).

1. Given a DFA \( D \) let \( L_1(D) \) be the DFA that recognizes \( L(D) \cap A \), where \( A \) is regular.

2. We show that \( X_A \leq_m E_{DFA} \).
Example

\[ X_A = \{ \langle D \rangle \mid D \text{ is a DFA} \land L(D) \cap A \neq \emptyset \} \]

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

We show that \( X_A \leq_m \overline{E_{\text{DFA}}} \).

1. Given a DFA \( D \) let \( L_1(D) \) be the DFA that recognizes \( L(D) \cap A \), where \( A \) is regular.

2. We show that \( X_A \leq_m \overline{E_{\text{DFA}}} \)
   - Show: If \( \langle D \rangle \in X_A \), then \( \langle L_1(D) \rangle \in \overline{E_{\text{DFA}}} \).
     1. \( L(D) \cap A \neq \emptyset \) (assumption)
     2. \( \langle L_1(D) \rangle \in \overline{E_{\text{DFA}}} \) (by 2.1)
   - Show: If \( \langle L_1(D) \rangle \in \overline{E_{\text{DFA}}} \), then \( \langle D \rangle \in X_A \).
     1. \( \langle D \rangle \in X_A \) by \( L_1(D) \neq \emptyset \) (assumption)
1. Show that $\leq_m$ is a reflexive relation.
2. Show that $\leq_m$ is a transitive relation.
Motivation

If \( A \) is regular, then \( X_A \) decidable.

**Proof (2\textsuperscript{nd} try).**

1. Let \( L_1(D) = L(D) \cap A \) where \( D \) is a DFA.
2. For any \( D \) we have that \( L_1(D) \) is regular. *(Proof?)*
3. Let \( D \) be the DFA that recognizes \( L_1 \). *(Proof?)*
4. \( X_A \leq_m \overline{E}_{\text{DFA}} \) (Before: \( \langle D \rangle \in X_A \iff \langle L_1(D) \rangle \in \overline{E}_{\text{DFA}} \))
5. The test \( \langle L_1(D) \rangle \in \overline{E}_{\text{DFA}} \) is decidable, and equivalent to testing \( \langle D \rangle \in X_A \), so the latter is decidable?

We will now generalize the (5) step
Decidability on membership reducible

Theorem 5.22

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

Proof in cogumbreiro/turing.
Completing the running example

If $A$ regular, then $X_A = \{\langle D \rangle \mid D$ is a DFA $\land L(D) \cap A \neq \emptyset\}$ decidable.

Proof ($3^{rd}$ try).

1. For any $D$ we have that $L_{DA} = L(D) \cap A$ is regular, since:
   - For any DFA $D$ we have that $L(D) \cap A$ is regular, since regular langs are closed for $\cap$, $L(D)$ is regular (def of reg langs), and $A$ is regular (assumption).

2. $X_A \leq_m \overline{E}_{DFA}$, by Slide 9

3. $\overline{E}_{DFA}$ is decidable, by Lemma R.4 and $E_{DFA}$ decidable (Theorem 4.4)

4. $X_A$ is decidable, by Theorem 5.22, $\overline{E}_{DFA}$ is decidable, and $X_A \leq_m \overline{E}_{DFA}$ (2)

Lemma R.4 (Lesson 21). If $A$ decidable, then $\overline{A}$ decidable.
Undecidability on membership reducible

Corollary 5.23

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

Proof.
Undecidability on membership reducible

Corollary 5.23

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

Proof.

1. $B$ is decidable, by contradiction.
2. $A$ is decidable, by Theorem 5.22 and $A \leq_m B$ (assumption) and $B$ decidable (1)
3. We reach a contradiction: $A$ is decidable (2) and undecidable (assumption).

(1) $H_0: A \leq_m B$
$H_1: \sim$ Decidable $A$
\hline
$\sim$ Decidable $B$
(2) $H_0: A \leq_m B$
$H_1: \sim$ Decidable $A$
\hline
$H_2: $ Decidable $B$
\hline
False
(3) $H_0: A \leq_m B$
$H_1: \sim$ Decidable $A$
\hline
$H_2: $ Decidable $B$
\hline
$H_3: $ Decidable $A$
\hline
False
Exercise 5.24

- $A_{TM} \leq_m HALT_{TM}$ holds$^\dagger$.
- Show that $HALT_{TM}$ is undecidable.
Exercise 5.24

- $A_{TM} \leq_m \text{HALT}_{TM}$ holds$^\dagger$.
- Show that $\text{HALT}_{TM}$ is undecidable.

1. Apply Corollary 5.23 since $A_{TM}$ is undecidable (Theorem 4.11) and $A_{TM} \leq_m \text{HALT}_{TM}$ (hypothesis).

$^\dagger$ Proof in cogumbreiro/turing.
Theorem 5.28

If \( A \leq_m B \) and \( B \) is recognizable, then \( A \) is recognizable.

Exercise

- \( A_{TM} \leq_m HALT_{TM} \)

Show that \( A_{TM} \) is recognizable via mapping reducibility.
Theorem 5.28

If \( A \leq_m B \) and \( B \) is recognizable, then \( A \) is recognizable.

Exercise

- \( A_{TM} \leq_m HALT_{TM} \)

Show that \( A_{TM} \) is recognizable via mapping reducibility.

Proof.

1. Give a program that recognizes \( HALT_{TM} \) (homework!)
2. \( A_{TM} \), by Theorem 5.28, \( A_{TM} \leq_m HALT_{TM} \), and (1).
Corollary 5.29

If $A$ is unrecognizable and $A \leq_m B$, then $B$ is unrecognizable.

Theorem R.1

If $A \leq_m B$, then $\overline{A} \leq_m \overline{B}$.

Exercise

Show that $\overline{\text{HALT}}_{TM}$ is unrecognizable.
Corollary 5.29

If $A$ is unrecognizable and $A \leq_m B$, then $B$ is unrecognizable.

Theorem R.1

If $A \leq_m B$, then $\overline{A} \leq_m \overline{B}$.

Exercise

Show that $\overline{HALT}^{TM}$ is unrecognizable.

Proof.

1. $\overline{A}^{TM} \leq_m \overline{HALT}^{TM}$, by Theorem R.1 and $A^{TM} \leq_m HALT^{TM}$ (exercise 5.24)

2. $\overline{HALT}^{TM}$ is unrecognizable, by Corollary 5.29, $\overline{A}^{TM} \leq_m \overline{HALT}^{TM}$ (1), and $\overline{A}^{TM}$ is unrecognizable (Corollary 4.23)
Extra proofs
Decidability on membership reducible

Theorem 5.22

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

Proof.
Decidability on membership reducible

Theorem 5.22

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

Proof.

1. $B$ is decidable, so let $M_B$ be its decider.
2. Let $M_A$ be a turing machine defined as: $M_A(w) = M_B(f(w))$
   Run $M_B$ with input $f(w)$. If $M_B$ accepts, $M_A$ accepts. If $M_B$ rejects, $M_A$ rejects.
3. Correctness: Prove that $L(M_A) = A$ (next slide)
4. Termination: Prove that $M_A$ halts for every input:
Decidability on membership reducible

Theorem 5.22

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

Proof.

1. $B$ is decidable, so let $M_B$ be its decider.
2. Let $M_A$ be a turing machine defined as: $M_A(w) = M_B(f(w))$
   Run $M_B$ with input $f(w)$. If $M_B$ accepts, $M_A$ accepts. If $M_B$ rejects, $M_A$ rejects.
3. **Correctness:** Prove that $L(M_A) = A$ (next slide)
4. **Termination:** Prove that $M_A$ halts for every input: $M_A$ just runs $M_B$, which halts for every input.

Our goal is show that there exists a Turing machine that decides $A$, so we must prove that it does recognize $A$ (correctness) and that it decides $A$ (termination).
Decidability on membership reducible

Theorem 5.22

Proof (Continuation). Show that $L(M_A) = A$.

We do a case analysis on the result of executing $M_A$ with input $w$ and show that $w$ is (not) in $A$: 
Decidability on membership reducible

Theorem 5.22

Proof (Continuation). Show that $L(M_A) = A$.

We do a case analysis on the result of executing $M_A$ with input $w$ and show that $w$ is (not) in $A$:

- If $M_A$ accepts some $w$ we must show that $w \in A$. From $M_B$, we get that $f(w) \in L(B)$, thus, from Def 5.20, we have $w \in A$. 

Decidability on membership reducible

Theorem 5.22

Proof (Continuation). Show that $L(M_A) = A$.

We do a case analysis on the result of executing $M_A$ with input $w$ and show that $w$ is (not) in $A$:

- If $M_A$ accepts some $w$ we must show that $w \in A$. From $M_B$, we get that $f(w) \in L(B)$, thus, from Def 5.20, we have $w \in A$.
- If $M_A$ rejects some $w$ we must show that $w \notin A$. If reject, then $f(w) \notin L(B)$, thus, from Def 5.20, we have $w \notin A$. 


Example 5.24

$HALT_{TM}$ is undecidable

Proof.

1. We show that $A_{TM} \leq_m HALT_{TM}$ with $f$, where $f(\langle M, w \rangle) = \langle M', w \rangle$ and $M'$ runs $M(w)$ if $M$ rejects, then loop, otherwise accept.

2. Since $A_{TM}$ is undecidable, then $HALT_{TM}$ is undecidable (Corollary 5.23).

Unfold Def 5.20:

$$\langle M, w \rangle \in A_{TM} \iff f(\langle M, w \rangle) \in HALT_{TM}$$
Example 5.24

$HALT_{TM}$ is undecidable

Proof.

1. We show that $A_{TM} \leq_m HALT_{TM}$ with $f$, where $f(\langle M, w \rangle) = \langle M', w \rangle$ and $M'$ runs $M(w)$ if $M$ rejects, then loop, otherwise accept.

2. Since $A_{TM}$ is undecidable, then $HALT_{TM}$ is undecidable (Corollary 5.23).

Unfold Def 5.20:

$\langle M, w \rangle \in A_{TM} \iff f(\langle M, w \rangle) \in HALT_{TM}$

Step 1: $\langle M, w \rangle \in A_{TM} \implies f(\langle M, w \rangle) \in HALT_{TM}$

Step 2: $f(\langle M, w \rangle) \in HALT_{TM} \implies \langle M, w \rangle \in A_{TM}$
Example 5.24

$HALT_{TM}$ is undecidable

Recall that:

\[ HALT_{TM} = \{ \langle M, w \rangle \mid M \text{ is a TM and } M \text{ halts on input } w \} \]

and that $M'$ runs $M(w)$ if $M$ reject, then loop, otherwise accept.

**Proof (continuation).**

**Step 1.** $\langle M, w \rangle \in A_{TM} \implies f(\langle M, w \rangle) \in HALT_{TM}$. 
Example 5.24

**HALT**

**TM** is undecidable

Recall that:

\[ HALT_{TM} = \{ \langle M, w \rangle \mid M \text{ is a TM and } M \text{ halts on input } w \} \]

and that \( M' \) runs \( M(w) \) if \( M \) reject, then loop, otherwise accept.

**Proof (continuation).**

**Step 1.** \( \langle M, w \rangle \in A_{TM} \implies f(\langle M, w \rangle) \in HALT_{TM}. \)

- Since \( \langle M, w \rangle \in A_{TM}, \) then \( M \) accepts \( w. \)
Example 5.24

$HALT_{TM}$ is undecidable

Recall that:

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

and that $M'$ runs $M(w)$ if $M$ reject, then loop, otherwise accept.

**Proof (continuation).**

**Step 1.** $\langle M, w \rangle \in A_{TM} \implies f(\langle M, w \rangle) \in HALT_{TM}$.

- Since $\langle M, w \rangle \in A_{TM}$, then $M$ accepts $w$.
- Thus, $M'$ halts, and therefore $\langle M', w \rangle \in HALT_{TM}$
Example 5.24

\( HALT_{TM} \) is undecidable

Recall that:

1. \( HALT_{TM} = \{ \langle M, w \rangle \mid M \text{ is a TM and } M \text{ halts on input } w \} \)
2. \( M' \) runs \( M(w) \) if \( M \) reject, then loop, otherwise accept.

Proof (continuation).

Step 2. We have \( f(\langle M, w \rangle) = \langle M', w \rangle \in HALT_{TM} \) and must show \( \langle M, w \rangle \in A_{TM} \).
Example 5.24

$HALT_{TM}$ is undecidable

Recall that:

1. $HALT_{TM} = \{ \langle M, w \rangle \mid M \text{ is a TM and } M \text{ halts on input } w \}$
2. $M'$ runs $M(w)$ if $M$ reject, then loop, otherwise accept.

Proof (continuation).

Step 2. We have $f(\langle M, w \rangle) = \langle M', w \rangle \in HALT_{TM}$ and must show $\langle M, w \rangle \in A_{TM}$.

- Since $f(\langle M, w \rangle) \in HALT_{TM}$ and (1), then $M'$ halts.
Example 5.24

$HALT_{TM}$ is undecidable

Recall that:

1. $HALT_{TM} = \{ \langle M, w \rangle \mid M \text{ is a TM and } M \text{ halts on input } w \}$
2. $M'$ runs $M(w)$ if $M$ reject, then loop, otherwise accept.

Proof (continuation).

**Step 2.** We have $f(\langle M, w \rangle) = \langle M', w \rangle \in HALT_{TM}$ and must show $\langle M, w \rangle \in A_{TM}$.

- Since $f(\langle M, w \rangle) \in HALT_{TM}$ and (1), then $M'$ halts.
- Thus, $M'$ accepts,
Example 5.24

$HALT_{TM}$ is undecidable

Recall that:

1. $HALT_{TM} = \{ \langle M, w \rangle \mid M \text{ is a TM and } M \text{ halts on input } w \}$
2. $M'$ runs $M(w)$ if $M$ reject, then loop, otherwise accept.

Proof (continuation).

Step 2. We have $f(\langle M, w \rangle) = \langle M', w \rangle \in HALT_{TM}$ and must show $\langle M, w \rangle \in A_{TM}$.

- Since $f(\langle M, w \rangle) \in HALT_{TM}$ and (1), then $M'$ halts.
- Thus, $M'$ accepts, and since $M'$ only accepts when $M$ accepts $w$, we conclude our proof.
Theorem 5.28

If $A \leq_m B$ and $B$ is recognizable, then $A$ is recognizable.

Detailed proof.

We must show that there exists some $M_A$ that recognizes $A$. 
Theorem 5.28

If \( A \leq_m B \) and \( B \) is recognizable, then \( A \) is recognizable.

**Detailed proof.**

We must show that there exists some \( M_A \) that recognizes \( A \).

1. Let \( M_A(w) = M_B(f(w)) \).
   That is, machine \( M_A \) given \( w \) computes \( f(w) \) and **accepts**.

2. Show that \( L(M_A) = A \).
   - **Step 1:** If \( M_A \) accepts \( w \), then \( w \in A \).
   - **Step 2:** If \( w \in A \), then \( M_A \) accepts \( w \).
Theorem 5.28

If \( A \leq_m B \) and \( B \) is recognizable, then \( A \) is recognizable.

**Proof (Step 1).**

<table>
<thead>
<tr>
<th>Hypothesis</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>H1</td>
<td>( M_A ) accepts ( w )</td>
</tr>
<tr>
<td>H2</td>
<td>( w \in A \iff f(w) \in B )</td>
</tr>
<tr>
<td>H3</td>
<td>( M_B ) recognizes ( B )</td>
</tr>
<tr>
<td>H4</td>
<td>( M_A(w) = M_B(f(w)) )</td>
</tr>
</tbody>
</table>

**Goal:** show that \( w \in A \)

1. Since (H1) \( M_A \) accept \( w \) and H4, we have that \( M_B \) accepts \( f(w) \).
Theorem 5.28

If $A \leq_m B$ and $B$ is recognizable, then $A$ is recognizable.

Proof (Step 1).

| **Hypothesis** |
|-----------------|-----------------|
| H1 $M_A$ accepts $w$ |
| H2 $w \in A \iff f(w) \in B$ |
| H3 $M_B$ recognizes $B$ |
| H4 $M_A(w) = M_B(f(w))$ |

Goal: show that $w \in A$

1. Since (H1) $M_A$ accept $w$ and H4, we have that $M_B$ accepts $f(w)$.
2. From $M_B$ accepts $f(w)$ and H3, we get $f(w) \in B$. 
Theorem 5.28

If $A \leq_m B$ and $B$ is recognizable, then $A$ is recognizable.

Proof (Step 1).

<table>
<thead>
<tr>
<th>Hypothesis</th>
</tr>
</thead>
<tbody>
<tr>
<td>H1</td>
</tr>
<tr>
<td>H2</td>
</tr>
<tr>
<td>H3</td>
</tr>
<tr>
<td>H4</td>
</tr>
</tbody>
</table>

Goal: show that $w \in A$

1. Since (H1) $M_A$ accept $w$ and H4, we have that $M_B$ accepts $f(w)$.
2. From $M_B$ accepts $f(w)$ and H3, we get $f(w) \in B$.
3. Since $f(w) \in B$ and H2, then $w \in A$. 

Mapping reducibility
Theorem 5.28

If $A \leq_m B$ and $B$ is recognizable, then $A$ is recognizable.

**Proof. (Step 2)**

<table>
<thead>
<tr>
<th><strong>Hypothesis</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>H0 $w \in A$</td>
</tr>
<tr>
<td>H1 $w \in A \iff f(w) \in B$</td>
</tr>
<tr>
<td>H2 $M_B$ recognizes $B$</td>
</tr>
<tr>
<td>H3 $M_A(w) = M_B(f(w))$</td>
</tr>
</tbody>
</table>

**Goal:** show that $M_A$ accepts $w$. 
Theorem 5.28

If $A \leq_m B$ and $B$ is recognizable, then $A$ is recognizable.

**Proof. (Step 2)**

<table>
<thead>
<tr>
<th>Hypothesis</th>
</tr>
</thead>
<tbody>
<tr>
<td>H0 $w \in A$</td>
</tr>
<tr>
<td>H1 $w \in A \iff f(w) \in B$</td>
</tr>
<tr>
<td>H2 $M_B$ recognizes $B$</td>
</tr>
<tr>
<td>H3 $M_A(w) = M_B(f(w))$</td>
</tr>
</tbody>
</table>

**Goal:** show that $M_A$ accepts $w$.

1. From (H1) $w \in A$ and H2, we have that $f(w) \in B$.
2. From $f(w) \in B$ and H3, we have that $M_B$ accepts $f(w)$
3. From $M_B$ accepts $f(w)$ and H4, we have that $M_A$ accepts $w$. 
Theorem 5.28

If $A \leq_m B$ and $B$ is recognizable, then $A$ is recognizable.

Detailed proof.

We must show that there exists some $M_A$ that recognizes $A$. 
Theorem 5.28

If $A \leq_m B$ and $B$ is recognizable, then $A$ is recognizable.

**Detailed proof.**

We must show that there exists some $M_A$ that recognizes $A$.

1. Let $M_A(w) = M_B(f(w))$.
   That is, machine $M_A$ given $w$ computes $f(w)$ and accepts.

2. Show that $L(M_A) = A$.
   - **Step 1:** If $M_A$ accepts $w$, then $w \in A$.
   - **Step 2:** If $w \in A$, then $M_A$ accepts $w$. 

CS420  Mapping reducibility  Lecture 26  Tiago Cogumbreiro  28 / 35
Homework 8
EXAMPLE  5.24

In Theorem 5.1 we used a reduction from $A_{TM}$ to prove that $HALT_{TM}$ is undecidable. This reduction showed how a decider for $HALT_{TM}$ could be used to give a decider for $A_{TM}$. We can demonstrate a mapping reducibility from $A_{TM}$ to $HALT_{TM}$ as follows. To do so, we must present a computable function $f$ that takes input of the form $\langle M, w \rangle$ and returns output of the form $\langle M', w' \rangle$, where

$$\langle M, w \rangle \in A_{TM} \text{ if and only if } \langle M', w' \rangle \in HALT_{TM}.$$ 

The following machine $F$ computes a reduction $f$.

$$F = \text{“On input } \langle M, w \rangle:\$$

1. Construct the following machine $M'$.
   $$M' = \text{“On input } x:\$$
   1. Run $M$ on $x$.
   2. If $M$ accepts, accept.
   3. If $M$ rejects, enter a loop.”

2. Output $\langle M', w \rangle$.”
Exercise 5.24 (Proof)

Use Corollary 5.23.

1. \( A_{TM} \leq_{m} HALT_{TM} \) (next slide)

2. \( A_{TM} \) is undecidable by Theorem 4.11 (Lesson 21)

**Theorem** example_5_24:
~ Decidable HALT_tm.

**Proof.**
- apply reducible_undecidable with \((A:=A_{tm})\).
  - apply A_tm_red_HALT_tm.
  - apply a_tm_undecidable.

Qed.
Exercise 5.24 (Proof)

The mapping function is given in the book. We separate the construction of $M'$ into its own function just so we can prove theorems more simply.

$F = "On input \langle M, w \rangle:\n\begin{enumerate}
1. Construct the following machine $M'$. $M' = "On input $x$:\n   \begin{enumerate}
   1. Run $M$ on $x$.
   2. If $M$ accepts, accept.
   3. If $M$ rejects, enter a loop."
2. Output $\langle M', w \rangle."$
\end{enumerate}"
\end{enumerate}"

(* Construct the following machine *)

Definition $A_{tm\_looper} M := Build (\n\begin{array}{l}
fun x \Rightarrow (* On input $x$: *)
mlet r \leftarrow Call M x in (* 1. Run $M$ on $x$ *)
if r then ACCEPT (* 2. If $M$ accepts, accept *)
else LOOP). (* 3. If $M$ rejects, enter a loop. *)
\end{array}\n)$

Definition $A_{tm\_to\_HALT\_tm} p :=\n\begin{array}{l}
(* On input $\langle M, w \rangle$ *)
let (M, w) := decode_machine_input p in (* 1. Construct the following machine $M'$ *)
let M' := $A_{tm\_looper} M w$ in (* 2. Output $\langle M', w \rangle$ *)
$\langle M', w \rangle$.
\end{array}\n$
Exercise 5.24 (Proof)

1. Show that if $F(w) \in HALT_{TM}$, then $w \in A_{TM}$.
2. Show that if $w \in A_{TM}$, then $F(w) \in HALT_{TM}$.

**Theorem** A_tm_red_HALT_tm: $A_{tm} \leq_{m} HALT_{tm}$.

**Proof.**
- apply reducible_iff with A_tm_to_HALT_tm.
  - split; intros.
  - apply A_tm_red_HALT_tm_1; auto.
  - apply A_tm_red_HALT_tm_2; auto.

Qed.
Exercise 5.24 (Proof)

Structure of the proof

1. Simplify assumption $w \in A_{TM}$ (with inversion theorem)
   - We get that $w = \langle M, i \rangle$ for some $M$ and $i$
   - We get that $M$ accepts $i$

2. Show $F(\langle M, i \rangle) \in HALT_{TM}$ (with constructor _def theorem)
   - $F(\langle M, i \rangle) = \langle \text{looper}(M), i \rangle$
   - Show that $\text{looper}(M)$ does not loop when $M$ accepts

Lemma $A_{tm\_red\_HALT\_tm\_1}$:
for all $w$,
$A_{tm\_to\_HALT\_tm\_w}$.
Exercise 5.24 (Proof)

1. Simplify assumption $F(w) \in HALT_{TM}$ (by inverting with \_inv)
   - Obtain $F(w) = \langle M', i \rangle$ for some $M'$ and $i$ (invert)
     - Obtain that $w = \langle M, i \rangle$ for some $M$
     - Obtain that $M' = looper(M)$
   - Obtain that $M'$ does not loop with $i$
     - Thus, $M$ accepts $i$

2. Show $w \in A_{TM}$ (construct goal using \_def theorem)
   - Since $M$ accepts $i$ and $w = \langle M, i \rangle$

Lemma A_tm_red_HALT_tm_2:
   forall w,
   HALT_tm (A_tm_to_HALT_tm w) ->
   A_tm w.