Reductions

Let A and B be languages over the same alphabet Σ.

An *algorithm* that transforms:

- Strings in A to strings in B.
- Strings not in A to strings not in B.

B is decidable $\Rightarrow$ A is decidable.

A is undecidable $\Rightarrow$ B is undecidable.

One way to show a problem B to be undecidable is to reduce an undecidable problem A to B.

Reductions

A Turing machine M *computes* a function f if:

M halts on all inputs.

On input x it writes $f(x)$ on the tape and halts.

Such a function f is called a *computable* function.

Examples: increment, addition, multiplication, shift.

Any algorithm with output is computing a function.

Reductions

Let A and B be languages over Σ.

A is *reducible* to B if and only if:

- there exists a computable function $f : \Sigma^* \longrightarrow \Sigma^*$ such that
- for all $w \in \Sigma^*$, $w \in A \Leftrightarrow f(w) \in B$.

Notation: $A \leq_m B$.

FACT: $A \leq_m B \Leftrightarrow \overline{A} \leq_m \overline{B}$.

$w \in A \Leftrightarrow f(w) \in B$ is equivalent to $w \notin A \Leftrightarrow f(w) \notin B$.

Reductions

To Re-iterate:

1. **Construction**: $f(w)$ from w by an algorithm.
2. **Correctness**: $w \in A \Leftrightarrow f(w) \in B$.

An Example involving DFAs

$EQ_{\text{DFA}} = \{A, B \mid A, B \text{ are DFAs and } L(A) = L(B)\}$.

$E_{\text{DFA}} = \{A \mid A \text{ is a DFA and } L(A) = \emptyset\}$.

A *reduction machine* on input A, B two DFAs:

1. Constructs the DFA A' such that $L(A') = \overline{L(A)}$.
2. Constructs the DFA B' such that $L(B') = \overline{L(B)}$.
3. Constructs the DFA M_1 such that $L(M_1) = L(A) \cap L(B')$.
4. Constructs the DFA M_2 such that $L(M_2) = L(A') \cap L(B)$.
5. Constructs the DFA C such that $L(C) = L(M_1) \cup L(M_2)$.
6. Outputs C.

Correctness:

- Suppose $L(A) = L(B)$. Then, $L(C) = \emptyset$.
- Suppose $L(A) \neq L(B)$. Then, $L(C) \neq \emptyset$.

That is, $EQ_{\text{DFA}} \leq_m E_{\text{DFA}}$.

An Example involving CFGs

A$LLCFG$ $=$ $\{G \mid G$ is a CFG and $L(G) = \Sigma^*\}$.

$EQ_{\mathrm{CFG}} = \{G, H \mid G, H$ are CFGs and $L(G) = L(H)\}$.

A *reduction machine* on input G a context-free grammar with alphabet Σ:

1. Constructs a CFG H with rules of the form $S' \leftarrow aS' \mid \epsilon$, for all $a \in \Sigma$.
2. Outputs (G, H).

$L(H) = \Sigma^*$.

Correctness:

- Suppose G generates all strings in Σ^*. Then, $L(G) = L(H)$.
- Suppose G does not generate some string in Σ^*. Then $L(G) \neq L(H)$.

That is, A$LLCFG \leq_m EQ_{\mathrm{CFG}}$.

The Halting problem

$HALT_{\mathrm{TM}} = \{M, w \mid M$ is a DTM and M halts on $w\}$.

The reduction machine outputs a DTM that loops whenever M reaches the rejecting state.

On input M, w:

1. Constructs the following machine M':

   Read input x.

   Simulate M on x.

   If M accepts, halt and accept.

   If M halts and rejects, enter a loop.

2. Outputs M', w .

That is, $A_{\mathrm{TM}} \leq_m HALT_{\mathrm{TM}}$.

$HALT_{\mathrm{TM}}$ is undecidable since A_{TM} is undecidable.

The Halting problem

Three machines:

- A reduction machine that is a DTM.
- Input to the reduction machine is M, w, where M is a DTM.
- Output of the reduction machine is M', w, where M' is a DTM.

The Halting problem

The output DTM M' has the input M hard-coded into it.

Let $M = (Q, \Sigma, \Gamma, \delta, q_s, q_a, q_r)$. What is M'?

$M' = (Q', \Sigma, \Gamma, \delta', q_s, q_a, {q_r}')$:

$Q' = Q \cup \{q_l, {q_r}'\}$.

Define δ' as follows:

For all states p, for all states $q \neq q_r$, for all symbols a, b, for all $D \in \{L, R, S\}$:

if $\delta(p, a) = (q, b, D)$ then $\delta'(p, a) = (q, b, D)$.

For all states p, for all symbols a, b, for all $D \in \{L, R, S\}$:

if $\delta(p, a) = (q_r, b, D)$ then $\delta'(p, a) = (q_l, b, S)$.

For all symbols a, include the transition $\delta'(q_l, a) = (q_l, a, S)$.

M' is constructed from M by the reduction machine.

$$\overline{E_{\mathrm{TM}}} = \{M \mid M \text{ is a TM and } L(M) \neq \emptyset\}$$

A reduction machine on input M, w :

1. Constructs the following machine M':

   Read input x.

   If $x \neq w$ then reject.

   If $x = w$, Simulate M on w.

   If M accepts, halt and accept.

2. Outputs M'.

Correctness:

- If M accepts w then $L(M')$ is not empty.
- If M does not accept w then $L(M')$ is empty.

That is, $A_{\mathrm{TM}} \leq_m \overline{E_{\mathrm{TM}}}$.

$$\overline{E_{\mathrm{TM}}} = \{M \mid M \text{ is a TM and } L(M) \neq \emptyset\}$$

$A_{\mathrm{TM}} \leq_m \overline{E_{\mathrm{TM}}}$.

This implies that $\overline{E_{\mathrm{TM}}}$ is undecidable.

This in turn implies that E_{TM} is undecidable.

At least one of them must be not recognizable.

$\overline{E_{\mathrm{TM}}}$ is recognizable.

E_{TM} is not recognizable.

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

EQ_{TM} is undecidable.

Is $\overline{EQ_{\mathrm{TM}}}$ recognizable?

See the text book.

Language is Context-free

$$CONTEXT-FREE_{\mathrm{TM}} \;=\; \{M \mid M \text{ is a TM and } L(M) \text{ is context} - \text{free}\}$$

A reduction machine on input M, w:

1. Constructs the following machine M':

   Read input x.

   If x has the form $0^n 1^n 2^n$ then halt and accept.

   If x is not of this form, simulate M on w.

   If M accepts, halt and accept.

2. Outputs M'.

Correctness

- $L(M')$ is Σ^* if M accepts w.
- $L(M')$ is not context-free if M does not accept w.

$A_{\mathrm{TM}} \leq_m CONTEXT-FREE_{\mathrm{TM}}$.

$CONTEXT-FREE_{\mathrm{TM}}$ is undecidable since A_{TM} is undecidable.

Language is Not Regular

$$\overline{REGULAR_{\mathrm{TM}}} = \{M \mid M \text{ is a TM and } L(M) \text{ is NOT regular}\}$$

A reduction machine on input M, w:

1. Constructs the following machine M':

   Read input x.

   If x is not of the form 0^n1^n then halt and reject.

   If x is of this form, simulate M on w.

   If M accepts, halt and accept.

2. Outputs M'.

Correctness:

- $L(M')$ is $\emptyset$ if M does not accept w.
- $L(M')$ is not regular if M accepts w.

$A_{\mathrm{TM}} \leq_m \overline{REGULAR_{\mathrm{TM}}}$.

$\overline{REGULAR_{\mathrm{TM}}}$ is undecidable since A_{TM} is undecidable.

Not Recognizable

$A_{\mathrm{TM}} \leq_m REGULAR_{\mathrm{TM}} \implies \overline{A_{\mathrm{TM}}} \leq_m \overline{REGULAR_{\mathrm{TM}}}$.

$\overline{REGULAR_{\mathrm{TM}}}$ is not recognizable.

$A_{\mathrm{TM}} \leq_m \overline{REGULAR_{\mathrm{TM}}} \implies \overline{A_{\mathrm{TM}}} \leq_m REGULAR_{\mathrm{TM}}$

$REGULAR_{\mathrm{TM}}$ is not recognizable.

Properties of Reductions

1. $A \leq_m B$ and $B \leq_m C \implies A \leq_m C$.

2. $A \leq_m B \implies \overline{A} \leq_m \overline{B}$

3. $A \leq_m B$ and B is decidable $\implies A$ is decidable.

4. Let A be recognizable. Then, $A \leq_m \overline{A}$ if and only if A and $\overline{A}$ are decidable.

5. $A \leq_m B$ and B is recognizable $\implies A$ is recognizable.