COMPSCI 501: Formal Language Theory

Lecture 18: Mapping Reducibility

Marius Minea
marius@cs.umass.edu

University of Massachusetts Amherst

4 March 2019

Reducibility so far

Reducibility: a tool to show undecidability

Reducing A to B: use solution for B to solve A

If B is decidable, can decide A (using reduction)
If A undecidable, B also undecidable

Examples:
reduce A_{TM} to $HALT_{\mathsf{TM}}$
reduce A_{TM} to E_{TM}

Computable functions

Deciding a language vs. computation

Def. A function $f : \Sigma^* \to \Sigma^*$ is a **computable function** if some Turing machine M, on input w, *halts* with just $f(w)$ on tape.

If such a Turing machine exists, what language does it decide?

$L = \{\langle w, f(w)\rangle\}$. Why ?

Useful also to formalize transformation of machine descriptions.

Mapping Reducibility

Def. A language A is **mapping reducible** to language B (written $A \leq_{\mathsf{m}} B$) if there is a *computable function* $f : \Sigma^* \to \Sigma^*$ where for every w, $w \in A \Leftrightarrow f(w) \in B$

strings in A are mapped to strings in B
strings not in A are mapped to strings not in B

Can answer whether $w \in A$ by *testing* whether $f(w) \in B$

$A \leq_{\mathsf{m}} B$ equivalent to $\overline{A} \leq_{\mathsf{m}} \overline{B}$

$\leq_{\mathsf{m}}$ is transitive (why?): $A \leq_{\mathsf{m}} B \wedge B \leq_{\mathsf{m}} C \to A \leq_{\mathsf{m}} C$

Reducibility and Decidability

Theorem If $A \leq_{\mathsf{m}} B$ and B is decidable, then A is decidable.

Let M be a decider for B, and f the reduction.

Decider for A does as expected – on input w:
compute $f(w)$
run decider M on $f(w)$ and give same answer

Corollary If $A \leq_{\mathsf{m}} B$ and A is undecidable then B is undecidable.

Proof: immediate, by contradiction from above theorem.

Revisiting Examples: Halting

Reducing A_{TM} to $HALT_{\mathsf{TM}}$
$\langle M, w\rangle \in A_{\mathsf{TM}}$ *iff* $\langle M, w\rangle \in HALT_{\mathsf{TM}}$

Construct the machine M':
On arbitrary input x:
if M accepts, *accept*
if M rejects, enter a loop
implicit: if M loops, so will M'

$f(\langle M, w\rangle) = \langle M', w\rangle$ (same word w)

Dealing with improper input in mapping reductions:
if input not in A (invalid encoding), output something not in B.

Reducing E_{TM} to EQ_{TM}

Idea was: reduce emptiness test to comparison with simple TM $M_\emptyset$ that has empty language.

Reduction simply needs to construct pair $\langle M, M_\emptyset \rangle$ from $\langle M \rangle$

Reducing A_{TM} to E_{TM}

Recall construction: TM M_w that accepts at most the string w
rejects everything else, then calls original recognizer M

Can define function f that takes $\langle M, w \rangle$ and constructs $\langle M_w \rangle$

But: M accepts w iff $L(M_w)$ is *not empty*.

So we have reduced A_{TM} to the **complement** $\overline{E_{\mathsf{TM}}}$

Since complement preserves decidability, proof goes through.

But we don't have a mapping reduction!

Mapping reduction from A_{TM} to E_{TM} ?

We've seen a *reduction*, but could there be a *mapping reduction*?

If $A_{\mathsf{TM}} \leq_{\mathsf{m}} E_{\mathsf{TM}}$, then $\overline{A_{\mathsf{TM}}} \leq_{\mathsf{m}} \overline{E_{\mathsf{TM}}}$.

But $\overline{E_{\mathsf{TM}}}$ is Turing-recognizable (why?)
which would mean $\overline{A_{\mathsf{TM}}}$ recognizable (false).

$\Rightarrow$ Mapping reductions may not exist!
sensitive to complementation

An Exercise with Complements

If A is Turing-recognizable and $A \leq_{\mathsf{m}} \overline{A}$, then A is decidable.

Complement the reduction relation:

$$\overline{A} \leq_{\mathsf{m}} \overline{\overline{A}}, \text{ thus } \overline{A} \leq_{\mathsf{m}} A.$$

Since A is Turing-recognizable, so is $\overline{A}$.

Therefore, A is decidable.

Reduction for Recognizers

Theorem If $A \leq_{\mathsf{m}} B$ and B is Turing-recognizable, then A is Turing-recognizable.

Corollary If $A \leq_{\mathsf{m}} B$ and A is not Turing-recognizable then B is not Turing-recognizable.

Sometimes, using complement may help:

$A \leq_{\mathsf{m}} B$ equivalent to $\overline{A} \leq_{\mathsf{m}} \overline{B}$

To prove B not recognizable, we might prove $A_{\mathsf{TM}} \leq_{\mathsf{m}} \overline{B}$

EQ_{TM} not Turing-recognizable nor co-recognizable

1. Reduce A_{TM} to EQ_{TM}

On input $\langle M, w \rangle$, construct two machines:
$M_\emptyset$: rejects any input
M_w: ignore input, run M on w, report result
$f(\langle M, w \rangle) = \langle M_\emptyset, M_w \rangle$

M_w accepts everything or nothing, depending on M's run on w.

M_w *not equivalent* to $M_\emptyset$ precisely when M accepts w.

2. Reduce A_{TM} to $\overline{EQ_{\mathsf{TM}}}$

Same construction, with M_{all} (accepts everything) instead of $M_\emptyset$.

M_w *equivalent* to M_{all} precisely when M accepts w.