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 $HALT_{TM}$
- reduce $A_{TM}$ to $E_{TM}$

Computable functions

Deciding a language vs. computation

Def. A function $f : \Sigma^* \rightarrow \Sigma^*$ 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.

Reducibility and Decidability

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

Proof: immediate, by contradiction from above theorem.

Revisiting Examples: Halting

Reducing $A_{TM}$ to $HALT_{TM}$

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

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 $(M, M_\emptyset)$ from $(M)$.

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 $(M, w)$ and constructs $(M_w)$

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

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

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 $A_{TM} \leq_m E_{TM}$, then $\overline{A_{TM}} \leq_m \overline{E_{TM}}$.

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

⇒ Mapping reductions may not exist! sensitive to complementation

An Exercise with Complements

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

Complement the reduction relation:

$\overline{A} \leq_m \overline{\overline{A}}$, thus $\overline{A} \leq_m A$.

Since $A$ is Turing-recognizable, so is $A$.

Therefore, $A$ is decidable.

Reduction for Recognizers

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

**Corollary** If $A \leq_m B$ and $A$ is not Turing-recognizable then $B$ is not Turing-recognizable.

Sometimes, using complement may help:

$A \leq_m B$ equivalent to $\overline{A} \leq_m \overline{B}$

To prove $B$ not recognizable, we might prove $A_{TM} \leq_m \overline{B}$

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

1. Reduce $A_{TM}$ to $E_{Q_{TM}}$

On input $(M, w)$, construct two machines:

$M_\emptyset$: rejects any input

$M_w$: ignore input, run $M$ on $w$, report result

$f((M, w)) = (M_\emptyset, M_w)$

$M_w$ 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 $\overline{E_{Q_{TM}}}$

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

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