

So far we've reduced problems to each other via an oracle that gives us (often undecidable) answers. For example, given an oracle (a decider) for E_{TM} , we created a decider for A_{TM} . Since $A_{TM} \notin DEC$, it follows that $E_{TM} \notin DEC$. This is a Turing reduction, denoted as $A_{TM} \leq_T E_{TM}$.

In other words if $A \leq_T B$, then A is decidable relative to B . From this, we get the following theorem.

Thm) If $A \leq_T B$ and B is decidable, then A is decidable.

It is very tempting to think $A \leq_T B \Rightarrow B \in RE \Rightarrow A \in RE$.

After all, if you can map (accepting) instances of A to (accepting) instances of B , then if B is recognizable, surely A must be too.

There is, however, hidden additional structure there. If you map $x \in A$ to $f(x) \in B$, then you also map $x \notin A$ to $f(x) \notin B$.

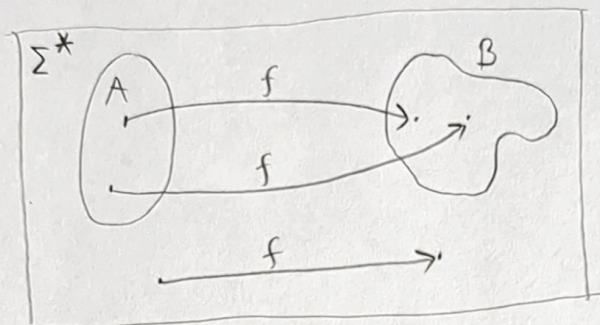
This is key! A general Turing reduction does not have this property. For example, we've shown that $A_{TM} \leq_T E_{TM}$, but $A_{TM} \in RE$ and $E_{TM} \notin RE$. The reduction we performed is more honestly written as $A_{TM} \leq_T \overline{E_{TM}}$. Note $\overline{E_{TM}} \in RE$.

Let's formalize this notion.

A function $f: \Sigma^* \rightarrow \Sigma^*$ is computable if there exists a TM M such that for all inputs $w \in \Sigma^*$, $M(w)$ halts with exactly $f(w)$ on its tape.

A language A is mapping reducible to a language B , written $A \leq_m B$, if there is a computable function $f: \Sigma^* \rightarrow \Sigma^*$ such that for every input $w \in \Sigma^*$, $w \in A \Leftrightarrow f(w) \in B$.

The function f is called the reduction from A to B .



On an intuitive level, when we write a reduction $A \leq B$, it means that A is no harder than B or, vice versa, B is at least as hard as A .

Mapping reductions give us a bit more information than Turing reductions.

Thm) If $A \leq_m B$ and $B \in \text{DEC}$, then $A \in \text{DEC}$.

Pf) Let D be a decider for B . Then $D(f(w))$ decides A .

Cor) If $A \leq_m B$ and $A \notin \text{DEC}$, then $B \notin \text{DEC}$.

We can recover old theorems via mapping reductions.

Thm) $\text{HALT}_{\text{TM}} \notin \text{DEC}$

Pf) Define the TM M' to be

- $N_{M'} =$ "On input w
- 1) Run $M(w)$
 - 2) Accept if $M(w)$ accepts.
 - 3) Loop forever"

Clearly $N_{M'}(w)$ halts iff $M(w)$ accepts. So if we define the obviously computable function $f(\langle M, w \rangle) = \langle M', w \rangle$ (improperly formatted strings are left unmodified), then $\langle M, w \rangle \in \text{A}_{\text{TM}}$ iff $\langle M', w \rangle \in \text{HALT}_{\text{TM}}$.

So $A_{TM} \leq_m \text{HALT}_{TM}$, but $A_{TM} \notin \text{DEC}$, hence $\text{HALT}_{TM} \notin \text{DEC}$. \square

But wait! There's more we can learn from $A_{TM} \leq_m \text{HALT}_{TM}$.

Thm) If $A \leq_m B$, then $\bar{A} \leq_m \bar{B}$.

Pf) The same reduction yields the result. \square

Thm) If $A \leq_m B$ and $B \in \text{RE}$, then $A \in \text{RE}$.

Pf) Identical to the DEC case except we have recognizers instead of deciders (this is what we would have hoped \leq_T would do). \square

Cor) If $A \leq_m B$ and $A \notin \text{RE}$, then $B \notin \text{RE}$.

Cor) If $A \leq_m B$ and $B \in \text{co-RE}$, then $A \in \text{co-RE}$.

Pf) $A \leq_m B \Rightarrow \bar{A} \leq_m \bar{B}$. Since $B \in \text{co-RE}$, $\bar{B} \in \text{RE} \Rightarrow \bar{A} \in \text{RE} \Rightarrow A \in \text{co-RE}$. \square

Cor) If $A \leq_m B$ and $A \notin \text{co-RE}$, then $B \notin \text{co-RE}$.

These theorems will show up again later in time/space resource restricted reductions, but first an example.

Thm) $\text{EQ}_{TM} \notin \text{RE}$ and $\text{EQ}_{TM} \notin \text{co-RE}$.

Pf) We give two reductions: $A_{TM} \leq_m \overline{\text{EQ}_{TM}}$ and $A_{TM} \leq_m \text{EQ}_{TM}$.

Consider the TMs R and $N_{M,w}$, where

$R =$ "On input w ,
 1) Reject"

$N_{M,w} =$ "On input v ,

1) Run $M(w)$

2) Accept if $M(w)$ accepts and reject otherwise."

Then $L(R) = \emptyset$ and $L(N_{M,w}) = \begin{cases} \emptyset & M(w) \text{ does not accept} \\ \Sigma^* & M(w) \text{ accepts.} \end{cases}$

So if we define the computable function

$$f(v) = \begin{cases} \langle R, R \rangle & v \neq \langle M, w \rangle \\ \langle R, N_{M,w} \rangle & v = \langle M, w \rangle, \end{cases}$$

then clearly $v \in A_{TM} \iff f(v) \in \overline{EQ_{TM}}$.

So $A_{TM} \leq \overline{EQ_{TM}}$, but $A_{TM} \notin \text{co-RE}$, so $\overline{EQ_{TM}} \notin \text{co-RE}$, hence $EQ_{TM} \notin \text{RE}$.

Now let the TM T be

$T =$ "On input w ,
1) Accept."

So $L(T) = \Sigma^*$, and we define the computable function

$$f(v) = \begin{cases} v & v \neq \langle M, w \rangle \\ \langle T, N_{M,w} \rangle & v = \langle M, w \rangle. \end{cases}$$

Then we have $v \in A_{TM} \iff f(v) \in EQ_{TM}$.

So $A_{TM} \leq_m EQ_{TM}$, but $A_{TM} \notin \text{co-RE}$, so $EQ_{TM} \notin \text{co-RE}$.

□

A few final notes. For a class of languages \mathcal{C} , we say a language A is \mathcal{C} -HARD if $\forall B \in \mathcal{C}, B \leq_m A$. If A is also a member of \mathcal{C} , then A is \mathcal{C} -COMPLETE. ($A \in \mathcal{C}$ and $A \in \mathcal{C}$ -HARD $\Rightarrow A \in \mathcal{C}$ -COMPLETE).

Any language $A \notin \text{RE}$ is necessarily RE-HARD. Similarly, $A \notin \text{co-RE}$ implies $A \in \text{co-RE-HARD}$. To see why this is the case has to do with the arithmetic hierarchy.