## Monday - Memorial Day

No class today.

## Wednesday

Recall: $A$ is **mapping reducible to** $B$, written $A \leq_m B$, means there is a computable function $f : \Sigma^* \to \Sigma^*$ such that *for all* strings $x$ in $\Sigma^*$,

$$x \in A \qquad \text{if and only if} \qquad f(x) \in B.$$

True or False: $\overline{A_{TM}} \leq_m \overline{HALT_{TM}}$

True or False: $HALT_{TM} \leq_m A_{TM}$.

**Theorem** (Sipser 5.28): If $A \leq_m B$ and $B$ is recognizable, then $A$ is recognizable.

**Proof**:

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

*Strategy*:

(i) To prove that a recognizable language $R$ is undecidable, prove that $A_{TM} \leq_m R$.

(ii) To prove that a co-recognizable language $U$ is undecidable, prove that $\overline{A_{TM}} \leq_m U$, i.e. that $A_{TM} \leq_m \overline{U}$.

$$E_{TM} = \{\langle M \rangle \mid M \text{ is a Turing machine and } L(M) = \emptyset\}$$

Example string in $E_{TM}$ is _____ . Example string not in $E_{TM}$ is _____ .

$E_{TM}$ is decidable / undecidable and recognizable / unrecognizable .

$\overline{E_{TM}}$ is decidable / undecidable and recognizable / unrecognizable .

**Claim**: _____ $\leq_m \overline{E_{TM}}$.

**Proof**: Need computable function $F : \Sigma^* \to \Sigma^*$ such that $x \in A_{TM}$ iff $F(x) \notin E_{TM}$. Define

$F =$ " On input $x$,

1. Type-check whether $x = \langle M, w \rangle$ for some TM $M$ and string $w$. If so, move to step 2; if not, output
2. Construct the following machine $M'_x$:

3. Output $\langle M'_x \rangle$."

Verifying correctness:

| Input string | Output string |
|---|---|
| $\langle M, w \rangle$ where $w \in L(M)$ | |
| $\langle M, w \rangle$ where $w \notin L(M)$ | |
| $x$ not encoding any pair of TM and string | |

## Review: Week 9 Wednesday

Please complete the review quiz questions on Gradescope about mapping reductions.

**Pre class reading for next time**: Introduction to Chapter 7.

## Friday

Recall: $A$ is **mapping reducible to** $B$, written $A \leq_m B$, means there is a computable function $f : \Sigma^* \to \Sigma^*$ such that *for all* strings $x$ in $\Sigma^*$,

$$x \in A \qquad \text{if and only if} \qquad f(x) \in B.$$

$$EQ_{TM} = \{\langle M, M' \rangle \mid M \text{ and } M' \text{ are both Turing machines and } L(M) = L(M')\}$$

Example string in $EQ_{TM}$ is _____ . Example string not in $EQ_{TM}$ is _____ .

$EQ_{TM}$ is decidable / undecidable and recognizable / unrecognizable .

$\overline{EQ_{TM}}$ is decidable / undecidable and recognizable / unrecognizable .

To prove, show that _____ $\leq_m EQ_{TM}$ and that _____ $\leq_m \overline{EQ_{TM}}$.

Verifying correctness:

| Input string | Output string |
|---|---|
| $\langle M, w \rangle$ where $M$ halts on $w$ | |
| $\langle M, w \rangle$ where $M$ loops on $w$ | |
| $x$ not encoding any pair of TM and string | |

In practice, computers (and Turing machines) don't have infinite tape, and we can't afford to wait unboundedly long for an answer. "Decidable" isn't good enough - we want "Efficiently decidable".

For a given algorithm working on a given input, how long do we need to wait for an answer? How does the running time depend on the input in the worst-case? average-case? We expect to have to spend more time on computations with larger inputs.

A language is **recognizable** if _____

A language is **decidable** if _____

A language is **efficiently decidable** if _____

A function is **computable** if _____

A function is **efficiently computable** if _____

Definition (Sipser 7.1): For $M$ a deterministic decider, its **running time** is the function $f : \mathbb{N} \to \mathbb{N}$ given by

$$f(n) = \text{max number of steps } M \text{ takes before halting, over all inputs of length } n$$

Definition (Sipser 7.7): For each function $t(n)$, the **time complexity class** $TIME(t(n))$, is defined by

$$TIME(t(n)) = \{L \mid L \text{ is decidable by a Turing machine with running time in } O(t(n))\}$$

An example of an element of $TIME(1)$ is

An example of an element of $TIME(n)$ is

Note: $TIME(1) \subseteq TIME(n) \subseteq TIME(n^2)$

Definition (Sipser 7.12) : $P$ is the class of languages that are decidable in polynomial time on a deterministic 1-tape Turing machine

$$P = \bigcup_k TIME(n^k)$$

*Compare to exponential time: brute-force search.*

Theorem (Sipser 7.8): Let $t(n)$ be a function with $t(n) \geq n$. Then every $t(n)$ time deterministic multitape Turing machine has an equivalent $O(t^2(n))$ time deterministic 1-tape Turing machine.

## Review: Week 9 Friday

Please complete the review quiz questions on Gradescope about complexity.

**Pre class reading for next time**: Skim Chapter 7.