Lecture 6: Reductions

Mika Göös

EPFL School of Computer and Communication Sciences

Recall

Turing-Recognizable/Decidable Languages

A TM machine M recognizes a language $L \subseteq \Sigma^*$ iff for all inputs $w \in \Sigma^*$:

1. If $w \in L$ then M accepts w and
2. If $w \notin L$ then M either rejects w or never halts

Such languages are called (Turing)-Recognizable

A TM machine M decides a language $L \subseteq \Sigma^*$ iff for all inputs $w \in \Sigma^*$:

1. M halts on w, and
2. M accepts w iff $w \in L$

Such languages are called (Turing)-Decidable

Undecidable Languages

There are undecidable languages

- **Turing machines are countable:** enumerate all encodings

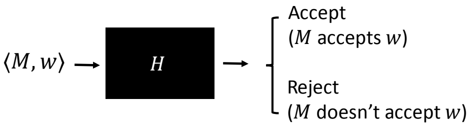
	$\langle M_1 \rangle$	$\langle M_2 \rangle$	$\langle M_3 \rangle$	$\langle M_4 \rangle$	$\langle M_5 \rangle$	$\langle M_6 \rangle$	...
M_1	A	∞	R	A	A	R	...
M_2	R	R	A	A	∞	A	...
M_3	R	∞	A	∞	R	R	...
M_4	A	∞	R	R	R	∞	...
M_5	∞	∞	A	A	A	A	...
M_6	R	A	R	∞	A	∞	...
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	
DIAG	R	A	R	A	R	A	

Let $DIAG = \{\langle M_i \rangle : M_i \text{ doesn't accept } \langle M_i \rangle\} = \{\langle M_2 \rangle, \langle M_4 \rangle, \langle M_6 \rangle, \ldots\}$

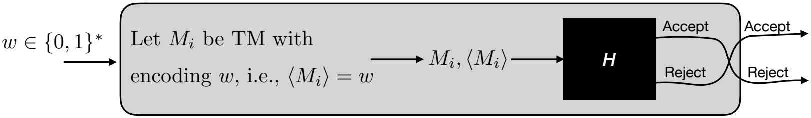
Then $DIAG \neq L(M_i)$ for all $i \in \mathbb{N}$. That is, *DIAG* is undecidable

Thm: $A_{TM} = \{\langle M, w\rangle : M \text{ is a TM and } M \text{ accepts } w\}$ is undecidable

Proof by contradiction Assume on the contrary that H is a decider for A_{TM}

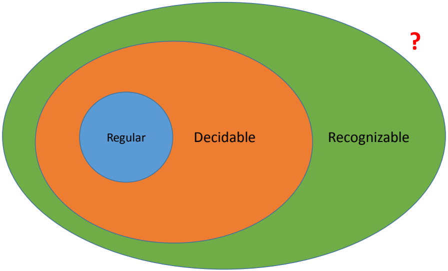


We construct a decider D for $DIAG = \{\langle M_i\rangle : M_i \text{ doesn't accept } \langle M_i\rangle\}$ using H:



Need to prove that D decides *DIAG*:

1. D halts on all inputs
2. D accepts $\langle M\rangle \iff M$ does not accept $\langle M\rangle \iff \langle M\rangle \in DIAG$



Unrecognizable languages?

Unrecognizable languages exist!

Thm: $\overline{A_{TM}}$ is not recognizable

Follows because A_{TM} is recognizable and since

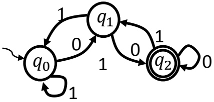
Thm: A language L is decidable iff it is recognizable and its complement is also recognizable

Reductions

A Reduction . . .

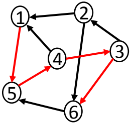
$NE_{DFA} \coloneqq \{\langle D \rangle | L(D) \neq \emptyset\}$

Input:

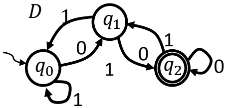


$L_{path} \coloneqq \{\langle G, v, w \rangle | v, w \in V(G)$ and $\exists$ a path from v to w in $G\}$

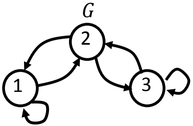
Input:



1,6

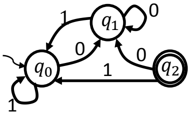


Accept ⇒ Accept

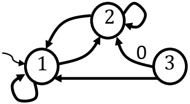


1,3

The reduction does not use the knowledge whether $\langle D \rangle$ is in the language NE_{DFA} or not!

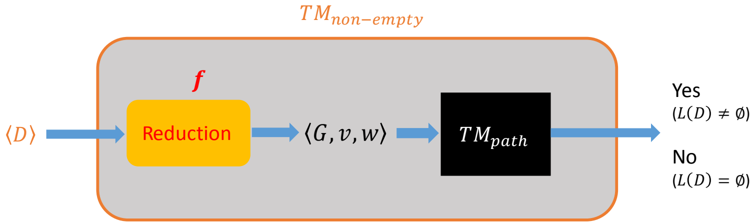


Reject ⇒ Reject



1,3

A reduction . . .



$f : \{\langle D \rangle \mid \text{D is a DFA}\} \rightarrow \{\langle G, v, w \rangle \mid G \text{ is a directed graph, } v, w \in V(G)\}$

$\langle D \rangle$ can be in NE_{DFA} or not

There might or might not be a path from v to w

Informally

Reducibility Use knowledge about complexity of one language to reason about the complexity of other in an easy way

Reduction Way to show that to solve one problem (A), it is sufficient to solve another problem (B)

A reduces to B. . .

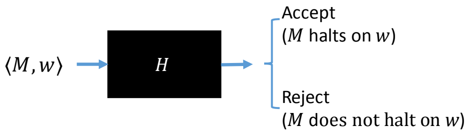
- Many examples (e.g. calculate area of rectangle reduces to calculate its width and height)
- Reductions quantify relative hardness of problems
  - If problem B is easy then problem A is easy too
  - If problem A is hard then problem B is hard too

Some Examples

$$HALT = \{\langle M, w\rangle \mid M \text{ halts on input } w\}$$

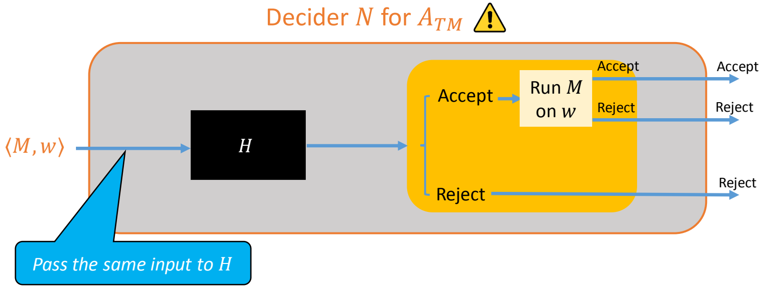
Theorem: $HALT$ is undecidable

Proof: Assume on the contrary that $HALT$ is decided by H



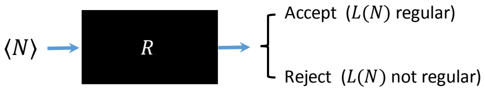
Construct a decider for $A_{TM} = \{\langle M, w\rangle : M \text{ accepts } w\}$

Decider N for A_{TM}



Theorem: $REG_{TM} = \{\langle N \rangle \mid L(N) \text{ is regular}\}$ is undecidable

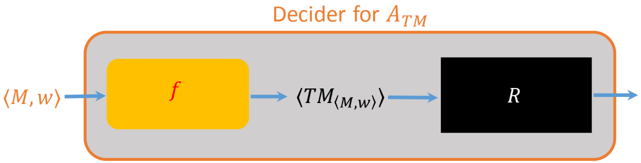
Proof: Assume on the contrary that REG_{TM} is decided by R



Can we decide A_{TM} using R?

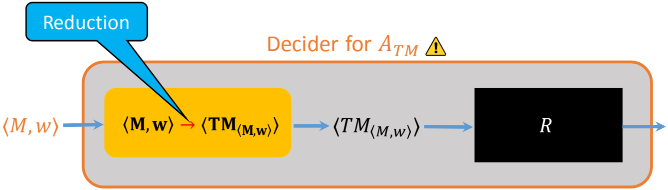
We'd like: given $\langle M, w \rangle$, construct a TM $TM_{\langle M,w \rangle}$ such that

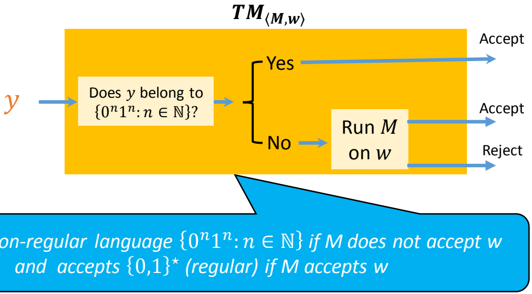
- If $\langle M, w \rangle \in A_{TM} \Rightarrow L(\langle TM_{\langle M,w \rangle} \rangle)$ is *regular*
- If $\langle M, w \rangle \notin A_{TM} \Rightarrow L(\langle TM_{\langle M,w \rangle} \rangle)$ is *non-regular* (for example $\{0^n 1^n \mid n \in \mathbb{N}\}$)



Input of A_{TM} : $\langle M, w \rangle$ $\xrightarrow{f}$ Input of R: $\langle N \rangle$

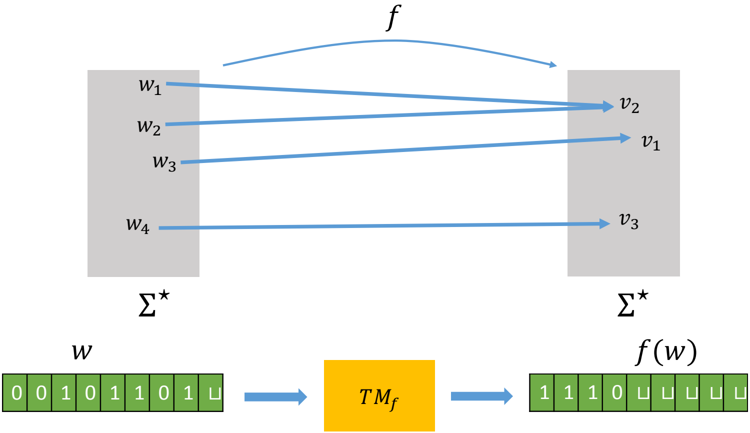
Theorem: $REG_{TM} = \{\langle N \rangle \mid L(N) \text{ is regular}\}$ is undecidable





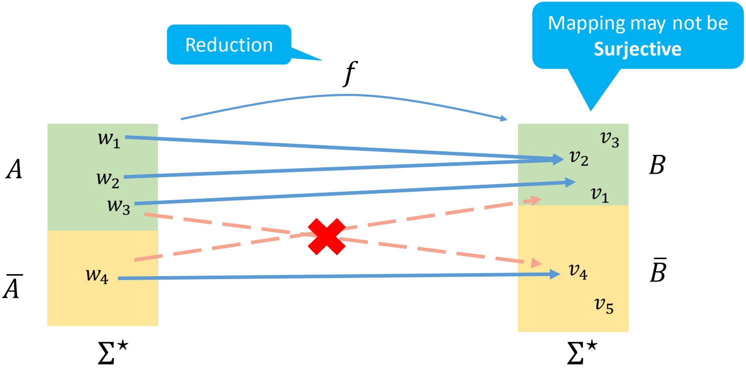
Formalising reductions

Reductions Part 1: Computability



Definition: A function $f : \Sigma^* \to \Sigma^*$ is a *computable function* if some TM M, on every input w halts with just $f(w)$ on its tape

Reductions Part 2: Correctness



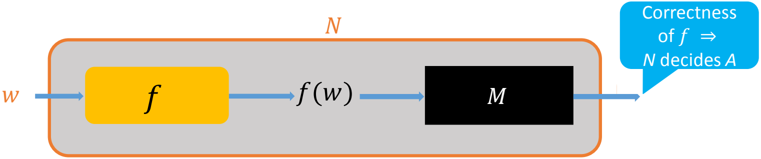
Definition: Language A is **mapping reducible** to language B, written $A \leq_m B$, if there is a computable function $f : \Sigma^* \to \Sigma^*$, such that for every $w \in \Sigma^*$:

$$w \in A \Leftrightarrow f(w) \in B$$

Theorem: If $A \leq_m B$ and B is decidable, then A is decidable

Proof:

- Assume that M is a decider for B and f is a reduction from A to B
- Let N be a TM as follows:



- $N =$ "On input w :

Computability of f ⇒ N is a decider

1. Compute $f(w)$
2. Run M on input $f(w)$ and output whatever M outputs"

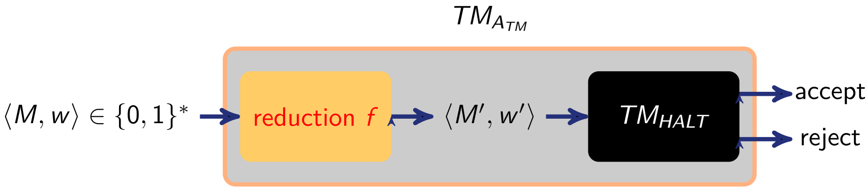
Corollary: If $A \leq_m B$ and A is undecidable, then B is undecidable

Examples of Reductions

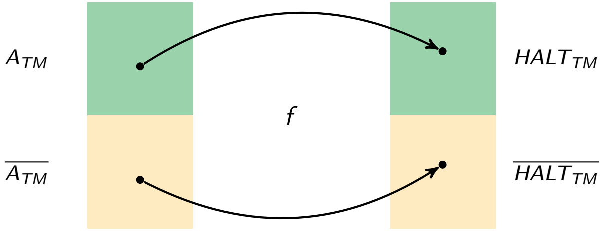
$$HALT_{TM} = \{\langle M, w\rangle \mid M \text{ halts on } w\}$$
$$A_{TM} = \{\langle M, w\rangle \mid M \text{ accepts } w\}$$

Theorem: $A_{TM} \leq_m HALT_{TM}$

Proof idea:



Define computable f such that



Theorem: $A_{TM} \leq_m HALT$ ($\Rightarrow HALT$ is undecidable)

- Let us define a function f as follows
- Given an input $x = \langle M, w \rangle$, return $f(x) = \langle M', w \rangle$, where
- M' ="On input y :

  f has to write down the code of M' only.
  It does not run M'!
  (M' might loop for some inputs.)

  1. Run M on y;
  2. **If M rejects y, enter infinite loop. If M accepts y, accept y.**
- Check that f is computable

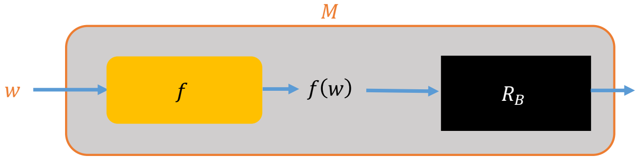
- $\Rightarrow$ If $\langle M, w \rangle \in A_{TM}$ then M' halts on w.
  Thus, $\langle M', w \rangle \in HALT$
- $\Leftarrow$ If $\langle M, w \rangle \notin A_{TM}$ then either (1) will never halt or M rejects w and (2) will ensure no halting. Thus, $\langle M', w \rangle \notin HALT$

Theorem: $A_{TM} \leq_m REG_{TM}$ ($\Rightarrow REG_{TM}$ is undecidable)

- Let us define a function f as follows
- Given an input $x = \langle M, w \rangle$, return $f(x) = \langle M' \rangle$, where
- $M' =$"On input y :
  1. if $y \in B = \{0^n 1^n : n \geq 0\}$, then accept y
  2. Run M on w, and accept y iff M accepts w"
- Check that f is computable

- $\Rightarrow$ If $\langle M, w \rangle \in A_{TM}$ then M' accepts all inputs.
  Thus, $L(M') = \{0, 1\}^*$ is regular — $\langle M' \rangle \in REG_{TM}$
- $\Leftarrow$ If $\langle M, w \rangle \notin A_{TM}$ then M does not accept w.
  Thus $L(M') = B$ is non-regular — $\langle M' \rangle \notin REG_{TM}$

Theorem: If $A \leq_m B$ and B is recognizable then A is recognizable



Definition of M

R_B is a recognizer for B

$A \leq_m B$

M accepts w $\iff$ R_B accepts $f(w)$ $\iff$ $f(w) \in B$ $\iff$ $w \in A$

$\Rightarrow$ ***M* is a recognizer for *A***

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

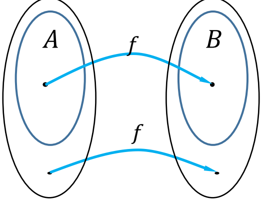
$$EQ_{TM} = \{\langle M_1, M_2\rangle \mid M_1, M_2 \text{ are TMs s.t. } L(M_1) = L(M_2)\}$$

Theorem: $\overline{A_{TM}} \leq_m EQ_{TM}$ ($\Rightarrow EQ_{TM}$ is unrecognizable)

- Let us define a function f as follows
- Given an input $x = \langle M, w\rangle$, return $f(x) = \langle M_1, M_2\rangle$, where
- M_1 =“On input y :
  1. Run M on w (ignore the input y)
  2. If M accepts then accept, else enter an infinite loop
- M_2 =“On input y :
  1. Reject y

- $\Rightarrow$ If $\langle M, w\rangle \in \overline{A_{TM}}$ then M_1 loops on all inputs. Thus, $L(M_1) = \emptyset = L(M_2)$ — $\langle M_1, M_2\rangle \in EQ_{TM}$
- $\Leftarrow$ If $\langle M, w\rangle \notin \overline{A_{TM}}$ then M accepts w and hence M_1 accepts every string. Thus $L(M_1) = \Sigma^* \neq \emptyset = L(M_2)$ — $\langle M_1, M_2\rangle \notin EQ_{TM}$

Summary



Definition: A function $f : \Sigma^* \to \Sigma^*$ is a *computable function* if some TM M, on every input w halts with just $f(w)$ on its tape

Definition: Language A is **mapping reducible** to language B, written $A \leq_m B$, if there is a computable function $f : \Sigma^* \to \Sigma^*$, such that for every $w \in \Sigma^*$:

$$w \in A \Leftrightarrow f(w) \in B$$

f is called a **reduction** from A to B

Theorem: If $A \leq_m B$ and B is decidable (recognizable), then A is decidable (recognizable)

Corollary: If $A \leq_m B$ and A is undecidable (unrecognizable) then B is undecidable (unrecognizable)

Next week: Recap of reductions!