

# CS 373: Theory of Computation

Manoj Prabhakaran

Mahesh Viswanathan

Fall 2008

---

# 1 Reductions

## Mapping Reductions

**Definition 1.** A function  $f : \Sigma^* \rightarrow \Sigma^*$  is *computable* if there is some Turing Machine  $M$  that on every input  $w$  halts with  $f(w)$  on the tape.

**Definition 2.** A *reduction* (a.k.a mapping reduction/many-one reduction) from a language  $A$  to a language  $B$  is a computable function  $f : \Sigma^* \rightarrow \Sigma^*$  such that

$$w \in A \text{ if and only if } f(w) \in B$$

In this case, we say  $A$  is *reducible* to  $B$ , and we denote it by  $A \leq_m B$ .

---

## Reductions and Recursive Enumerability

**Proposition 3.** *If  $A \leq_m B$  and  $B$  is r.e., then  $A$  is r.e.*

*Proof.* Let  $f$  be a reduction from  $A$  to  $B$  and let  $M_B$  be a Turing Machine recognizing  $B$ . Then the Turing machine recognizing  $A$  is

On input  $w$

  Compute  $f(w)$

  Run  $M_B$  on  $f(w)$

  Accept if  $M_B$  accepts, and reject if  $M_B$  rejects  $\square$

**Corollary 4.** *If  $A \leq_m B$  and  $A$  is not r.e., then  $B$  is not r.e.*

---

## Reductions and Decidability

**Proposition 5.** *If  $A \leq_m B$  and  $B$  is decidable, then  $A$  is decidable.*

*Proof.* Let  $f$  be a reduction from  $A$  to  $B$  and let  $M_B$  be a Turing Machine *deciding*  $B$ . Then a Turing machine that decides  $A$  is

On input  $w$

  Compute  $f(w)$

  Run  $M_B$  on  $f(w)$

  Accept if  $M_B$  accepts, and reject if  $M_B$  rejects  $\square$

**Corollary 6.** *If  $A \leq_m B$  and  $A$  is undecidable, then  $B$  is undecidable.*

---

## 1.1 Examples

### The Halting Problem

**Proposition 7.** *The language  $HALT = \{\langle M, w \rangle \mid M \text{ halts on input } w\}$  is undecidable.*

*Proof.* Recall  $A_{TM} = \{\langle M, w \rangle \mid w \in L(M)\}$  is undecidable. Will give reduction  $f$  to show  $A_{TM} \leq_m HALT \implies HALT$  undecidable.

Let  $f(\langle M, w \rangle) = \langle N, w \rangle$  where  $N$  is a TM that behaves as follows:

On input  $x$

Run  $M$  on  $x$

If  $M$  accepts then halt and accept

If  $M$  rejects then go into an infinite loop

$N$  halts on input  $w$  if and only if  $M$  accepts  $w$ . i.e.,  $\langle M, w \rangle \in A_{TM}$  iff  $f(\langle M, w \rangle) \in HALT$   $\square$

---

### Emptiness of Turing Machines

**Proposition 8.** *The language  $E_{TM} = \{\langle M \rangle \mid L(M) = \emptyset\}$  is not r.e.*

*Proof.* Recall  $L_d = \{\langle M \rangle \mid \langle M \rangle \notin L(M)\}$  is not r.e.

$L_d$  is reducible to  $E_{TM}$  as follows. Let  $f(\langle M \rangle) = \langle N \rangle$  where  $N$  is a TM that behaves as follows:

On input  $x$

Run  $M$  on  $\langle M \rangle$  for  $|x|$  steps

Accept  $x$  only if  $M$  accepts  $\langle M \rangle$  within  $|x|$  steps

Observe that  $L(N) = \emptyset$  if and only if  $M$  does not accept  $\langle M \rangle$  if and only if  $\langle M \rangle \in L_d$ .  $\square$

---

### Checking Regularity

**Proposition 9.** *The language  $REGULAR = \{\langle M \rangle \mid L(M) \text{ is regular}\}$  is undecidable.*

*Proof.* We give a reduction  $f$  from  $A_{TM}$  to  $REGULAR$ . Let  $f(\langle M, w \rangle) = \langle N \rangle$ , where  $N$  is a TM that works as follows:

On input  $x$

If  $x$  is of the form  $0^n 1^n$  then accept  $x$

else run  $M$  on  $w$  and accept  $x$  only if  $M$  does

If  $w \in L(M)$  then  $L(N) = \Sigma^*$ . If  $w \notin L(M)$  then  $L(N) = \{0^n 1^n \mid n \geq 0\}$ . Thus,  $\langle N \rangle \in REGULAR$  if and only if  $\langle M, w \rangle \in A_{TM}$   $\square$

## 2 Rice's Theorem

### Checking Properties

Given  $\langle M \rangle$

Does $L(M)$ contain $\langle M \rangle$ ?	} Undecidable
Is $L(M)$ non-empty?	
Is $L(M)$ empty?	
Is $L(M)$ infinite?	} Undecidable
Is $L(M)$ finite?	
Is $L(M)$ co-finite (i.e., is $\overline{L(M)}$ finite)?	
Is $L(M) = \Sigma^*$ ?	

Which of these properties can be decided? None! By *Rice's Theorem*

---

### Properties

**Definition 10.** A property of languages is simply a set of languages. We say  $L$  satisfies the property  $\mathbb{P}$  if  $L \in \mathbb{P}$ .

**Definition 11.** For any property  $\mathbb{P}$ , define language  $L_{\mathbb{P}}$  to consist of Turing Machines which accept a language in  $\mathbb{P}$ :

$$L_{\mathbb{P}} = \{\langle M \rangle \mid L(M) \in \mathbb{P}\}$$

Deciding  $L_{\mathbb{P}}$ : deciding if a language represented as a TM satisfies the property  $\mathbb{P}$ .

- *Example:*  $\{\langle M \rangle \mid L(M) \text{ is infinite}\}$ ;  $E_{\text{TM}} = \{\langle M \rangle \mid L(M) = \emptyset\}$
- *Non-example:*  $\{\langle M \rangle \mid M \text{ has 15 states}\}$   $\leftarrow$  This is a property of TMs, and not languages!

---

### Trivial Properties

**Definition 12.** A property is trivial if either it is not satisfied by any r.e. language, or if it is satisfied by all r.e. languages. Otherwise it is non-trivial.

*Example 13.* Some trivial properties:

- $\mathbb{P}_{\text{ALL}} =$  set of all languages
- $\mathbb{P}_{\text{R.E.}} =$  set of all r.e. languages
- $\overline{\mathbb{P}}$  where  $\mathbb{P}$  is trivial
- $\mathbb{P} = \{L \mid L \text{ is recognized by a TM with an even number of states}\} = \mathbb{P}_{\text{R.E.}}$

Observation. For any trivial property  $\mathbb{P}$ ,  $L_{\mathbb{P}}$  is decidable. (Why?) Then  $L_{\mathbb{P}} = \Sigma^*$  or  $L_{\mathbb{P}} = \emptyset$ .

---

### Rice's Theorem

**Proposition 14.** *If  $\mathbb{P}$  is a non-trivial property, then  $L_{\mathbb{P}}$  is undecidable.*

- Thus  $\{\langle M \rangle \mid L(M) \in \mathbb{P}\}$  is not decidable (unless  $\mathbb{P}$  is trivial)

We cannot algorithmically determine any interesting property of languages represented as Turing Machines!

### Properties of TMs

Note. Properties of TMs, as opposed to those of languages they accept, may or may not be decidable.

*Example 15.*

$$\left. \begin{array}{l} \{\langle M \rangle \mid M \text{ has 193 states}\} \\ \{\langle M \rangle \mid M \text{ uses at most 32 tape cells on blank input}\} \\ \{\langle M \rangle \mid M \text{ halts on blank input}\} \end{array} \right\} \text{Decidable}$$

$$\left. \begin{array}{l} \{\langle M \rangle \mid \text{on input 0011 } M \text{ at some point writes the} \\ \text{symbol \$ on its tape}\} \end{array} \right\} \text{Undecidable}$$

### Proof of Rice's Theorem

#### Rice's Theorem

If  $\mathbb{P}$  is a non-trivial property, then  $L_{\mathbb{P}}$  is undecidable.

*Proof.* • Suppose  $\mathbb{P}$  non-trivial and  $\emptyset \notin \mathbb{P}$ .

– (If  $\emptyset \in \mathbb{P}$ , then in the following we will be showing  $L_{\overline{\mathbb{P}}}$  is undecidable. Then  $L_{\mathbb{P}} = \overline{L_{\overline{\mathbb{P}}}}$  is also undecidable.)

- Recall  $L_{\mathbb{P}} = \{\langle M \rangle \mid L(M) \text{ satisfies } \mathbb{P}\}$ . *We'll reduce  $A_{\text{TM}}$  to  $L_{\mathbb{P}}$ .*
- Then, since  $A_{\text{TM}}$  is undecidable,  $L_{\mathbb{P}}$  is also undecidable. □

### Proof of Rice's Theorem

*Proof (contd).* Since  $\mathbb{P}$  is non-trivial, at least one r.e. language satisfies  $\mathbb{P}$ . i.e.,  $L(M_0) \in \mathbb{P}$  for some TM  $M_0$ .

*Will show a reduction  $f$  that maps an instance  $\langle M, w \rangle$  for  $A_{\text{TM}}$ , to  $\langle N \rangle$  such that*

- If  $M$  accepts  $w$  then  $N$  accepts the same language as  $M_0$ .
  - Then  $L(N) = L(M_0) \in \mathbb{P}$
- If  $M$  does not accept  $w$  then  $N$  accepts  $\emptyset$ .
  - Then  $L(N) = \emptyset \notin \mathbb{P}$

Thus,  $\langle M, w \rangle \in A_{\text{TM}}$  iff  $\langle N \rangle \in L_{\mathbb{P}}$ . □

---

### Proof of Rice's Theorem

*Proof (contd).* The reduction  $f$  maps  $\langle M, w \rangle$  to  $\langle N \rangle$ , where  $N$  is a TM that behaves as follows:

On input  $x$

Ignore the input and run  $M$  on  $w$

If  $M$  does not accept (or doesn't halt)

then do not accept  $x$  (or do not halt)

If  $M$  does accept  $w$

then run  $M_0$  on  $x$  and accept  $x$  iff  $M_0$  does.

Notice that indeed if  $M$  accepts  $w$  then  $L(N) = L(M_0)$ . Otherwise  $L(N) = \emptyset$ . □

---

## 3 Closure Properties

### 3.1 Closure of Decidable Languages

#### Closure of Decidable Languages

**Proposition 16.** *Decidable languages are closed under union, intersection, complementation, concatenation, kleene star, and inverse homomorphism*

*Proof.* Given TMs  $M_1, M_2$  that decide languages  $L_1, L_2$

- A TM that decides  $L_1 \cup L_2$ : on input  $x$ , run  $M_1$  and  $M_2$  on  $x$ , and accept iff either accepts. (Similarly for intersection.)
- A TM that decides  $\overline{L_1}$ : On input  $x$ , run  $M_1$  on  $x$ , and accept if  $M_1$  rejects, and reject if  $M_1$  accepts.

□

---

#### Closure of Decidable Languages

*Proof (contd).* • A TM to decide  $L_1 L_2$ : On input  $x$ , for each of the  $|x| + 1$  ways to divide  $x$  as  $yz$ : run  $M_1$  on  $y$  and  $M_2$  on  $z$ , and accept if both accept. Else reject.

- A TM to decide  $L_1^*$ : On input  $x$ , if  $x = \epsilon$  accept. Else, for each of the  $2^{|x|-1}$  ways to divide  $x$  as  $w_1 \dots w_k$  ( $w_i \neq \epsilon$ ): run  $M_1$  on each  $w_i$  and accept if  $M_1$  accepts all. Else reject.

- A TM to decide  $h^{-1}(L_1)$ : On input  $x$ , compute  $h(x)$  and run  $M_1$  on  $h(x)$ ; accept iff  $M_1$  accepts.

□

## Closure of Decidable Languages

**Proposition 17.** *Decidable languages are not closed under homomorphism*

*Proof.* We will show a decidable language  $L$  and a homomorphism  $h$  such that  $h(L)$  is undecidable

- Let  $L = \{xy \mid x \in \{0,1\}^*, y \in \{a,b\}^*, x = \langle M, w \rangle, \text{ and } y \text{ encodes an integer } n \text{ such that the TM } M \text{ on input } w \text{ will halt in } n \text{ steps}\}$
- $L$  is decidable: can simply simulate  $M$  on input  $w$  for  $n$  steps
- Consider homomorphism  $h$ :  $h(0) = 0, h(1) = 1, h(a) = h(b) = \epsilon$ .
- $h(L) = \text{HALT}$  which is undecidable.

□

## 3.2 Closure of Recursively Enumerable Languages

### Closure of Recursively Enumerable Languages

**Proposition 18.** *R.e. languages are closed under union, intersection, concatenation, kleene star, homomorphism and inverse homomorphism.*

*Proof.* Given TMs  $M_1, M_2$  that recognize languages  $L_1, L_2$

- A TM that recognizes  $L_1 \cup L_2$ : on input  $x$ , run  $M_1$  and  $M_2$  on  $x$  *in parallel*, and accept iff either accepts. (Similarly for intersection; but no need for parallel simulation)

□

### Closure of Recursively Enumerable Languages

*Proof (contd).* • A TM to recognize  $L_1L_2$ : On input  $x$ , do *in parallel*, for each of the  $|x| + 1$  ways to divide  $x$  as  $yz$ : run  $M_1$  on  $y$  and  $M_2$  on  $z$ , and accept if both accept. Else reject.

- A TM to recognize  $L_1^*$ : On input  $x$ , if  $x = \epsilon$  accept. Else, do *in parallel*, for each of the  $2^{|x|-1}$  ways to divide  $x$  as  $w_1 \dots w_k$  ( $w_i \neq \epsilon$ ): run  $M_1$  on each  $w_i$  and accept if  $M_1$  accepts all. Else reject.

- A TM to recognize  $h^{-1}(L_1)$ : On input  $x$ , compute  $h(x)$  and run  $M_1$  on  $h(x)$ ; accept iff  $M_1$  accepts.
- A TM to recognize  $h(L_1)$ : On input  $x$ , start going through all strings  $w$ , and if  $h(w) = x$ , start executing  $M_1$  on  $w$ , using *dovetailing* to interleave with other executions of  $M_1$ . Accept if any of the executions accepts.

□

## Closure of Recursively Enumerable Languages

**Proposition 19.** *R.e. languages are not closed under complementation.*

*Proof.* We saw that  $A_{TM}$  is r.e. but  $\overline{A_{TM}}$  is not.

□

Also we saw the following:

**Proposition 20.**  *$L$  is decidable iff  $L$  and  $\overline{L}$  are both r.e.*

## The Big Picture

