CS 373: Theory of Computation

Gul Agha Mahesh Viswanathan

Fall 2010

---

1 Reductions

1.1 Definitions and Observations

Mapping Reductions

Definition 1. A function $f : \Sigma^* \to \Sigma^*$ 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 $f : \Sigma^* \to \Sigma^*$ 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` □

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` □

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

1.2 Examples

The Halting Problem

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

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

Emptiness of Turing Machines

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

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

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

```
On input x
 Run M on ⟨M⟩ for |x| steps
 Accept x only if M accepts ⟨M⟩ within |x| steps
```

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

Checking Regularity

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

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

```
On input x
 If x is of the form 0^n1^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 $L(N) = \{0^n1^n \mid n \geq 0\}$. Thus, $\langle N\rangle \in$ REGULAR if and only if $\langle M, w\rangle \in A_{\text{TM}}$ □

Checking Equality

Proposition 10. $EQ_{\text{TM}} = \{\langle M_1, M_2\rangle \mid L(M_1) = L(M_2)\}$ *is not r.e.*

Proof. We will give a reduction f from E_{TM} to EQ_{TM}. Let M_1 be the Turing machine that on any input, halts and rejects i.e., $L(M_1) = \emptyset$. Take $f(M) = \langle M, M_1 \rangle$.

Observe $M \in E_{\text{TM}}$ iff $L(M) = \emptyset$ iff $L(M) = L(M_1)$ iff $\langle M, M_1 \rangle \in \text{EQ}_{\text{TM}}$. □

---

2 Rice's Theorem

2.1 Properties

Checking Properties

Given M

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

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

Properties

Definition 11. A property of languages is simply a set of languages. We say L *satisfies* the property $\mathbb{P}$ if $L \in \mathbb{P}$.

Definition 12. For any property $\mathbb{P}$, define language $L_{\mathbb{P}}$ to consist of Turing Machines which accept a language in $\mathbb{P}$:

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

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

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

Trivial Properties

Definition 13. 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 14. 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 $L_{\mathbb{P}} = \emptyset$.

2.2 Main Theorem

Rice's Theorem

Proposition 15. *If $\mathbb{P}$ is a non-trivial property, then $L_{\mathbb{P}}$ is undecidable.*

- Thus $\{M \mid L(M) \in \mathbb{P}\}$ 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 16.

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

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

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_{TM} to $L_{\mathbb{P}}$.*
- Then, since A_{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_{TM}, to N 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 $N \in L_{\mathbb{P}}$. □

Proof of Rice's Theorem

Proof (contd). The reduction f maps $\langle M, w\rangle$ to N, 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 $L(N) = \emptyset$. □

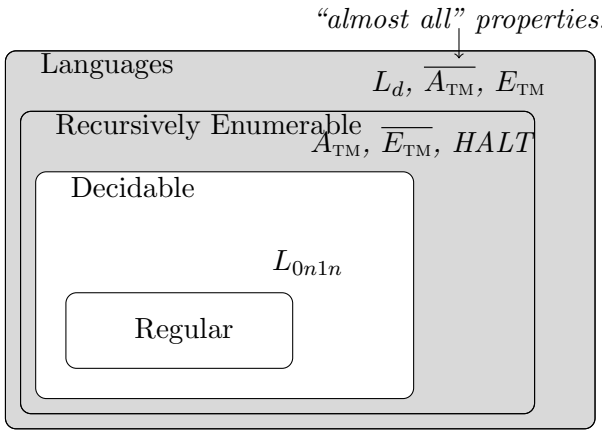
Rice's Theorem

Recap

Every non-trivial property of r.e. languages is undecidable

- Rice's theorem says nothing about properties of Turing machines
- Rice's theorem says nothing about whether a property of languages is recurisvely enumerable or not.

Big Picture . . . again


"almost all" properties!
Languages
L_d, $\overline{A_{\text{TM}}}$, E_{TM}
Recursively Enumerable
A_{TM}, $\overline{E_{\text{TM}}}$, $HALT$
Decidable
L_{0n1n}
Regular