Proving Languages not Decidable I

- Earlier we proved that A_{TM} (i.e., the acceptance problem of TM) is undecidable
- Many other problems are not decidable either
- But how to prove that?
- We will introduce a technique called reducibility

Reducibility I

- Reduction: converting the first problem to the second so the second can be used to solve the first
- Example: finding your way around a city can be solved by having a map

  The problem is reduced to obtaining a map
- We say A reduces to B if we can use a solution to B to solve A
- In the earlier example,

  A: getting around a city
  B: obtaining a map

Reducibility II

- A mathematical example:

  A: measuring the area of a rectangle

  B: measuring the length/width
- When A is reducible to B, solving A is not harder than B
- The reason is that a solution to B gives a solution to A
- Therefore,

$$B \text{ decidable} \Rightarrow A \text{ decidable}$$

  and

$$A \text{ undecidable} \Rightarrow B \text{ undecidable}$$

E_{TM} Undecidable I

- Consider

$$E_{\mathrm{TM}} = \{\langle M \rangle \mid M : \text{ a TM and } L(M) = \emptyset\}$$

- We prove that it is undecidable
- Idea: we do the proof by contradiction
- Assume E_{TM} is decidable. Then there is a decider R
- From R we construct a decider S for A_{TM}. This causes a contradiction because A_{TM} is undecidable
- For A_{TM}, the input is $\langle M, w \rangle$. Under such an input, we design S to be as follows

E_{TM} Undecidable II

1. Design a TM M_1 such that

$$L(M_1) \neq \emptyset \Leftrightarrow M \text{ accepts } w \tag{1}$$

2. Run R on input $\langle M_1 \rangle$
3. If R accepts, *reject*; if R rejects, *accept*

- We discuss the design of M_1 later
- We see that if R accepts, then $L(M_1) = \emptyset$ implies that M rejects w
- Thus for input $\langle M, w \rangle$, in a finite number of steps we know if M accepts w or not

E_{TM} Undecidable III

- Then A_{TM} is decidable, a contradiction
- How to design M_1?
- M_1 takes input x and have
  1. If $x \neq w$, *reject*
  2. If $x = w$, run M on input w and *accept* if M does

E_{TM} Undecidable IV

- Clearly,

$$L(M_1) = \emptyset \text{ or } \{w\}$$

  We see that

$$M \text{ accepts } w \Rightarrow L(M_1) = \{w\} \neq \emptyset$$
$$L(M_1) \neq \emptyset \Rightarrow M \text{ accepts } w$$

  Thus the condition (1) is satisfied
- M_1 takes w as part of its description
- This is fine as we can design a machine related to a specific string