CS21 Decidability and Tractability

Lecture 13
February 2, 2024

1

Outline

- reductions
- many-one reductions
- undecidable problems
  - computation histories
  - surprising contrasts between decidable/undecidable
- Rice's Theorem

2

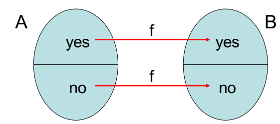
Definition of reduction

- Can you reduce co-HALT to HALT?
- We know that HALT is RE
- Does this show that co-HALT is RE?
  - recall, we showed co-HALT is not RE
- our current notion of reduction cannot distinguish complements

3

Definition of reduction

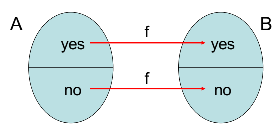
- More refined notion of reduction:
  - "many-one" reduction (commonly)
  - "mapping" reduction (book)



reduction from language A to language B

4

Definition of reduction



- function f should be computable

Definition: $f : \Sigma^* \rightarrow \Sigma^*$ is computable if there exists a TM M_f such that on every $w \in \Sigma^*$ M_f halts on w with f(w) written on its tape.

5

Definition of reduction

- Notation: "A many-one reduces to B" is written

$$A \le_m B$$

  - "yes maps to yes and no maps to no" means:

$w \in A$ maps to $f(w) \in B$ & $w \notin A$ maps to $f(w) \notin B$

- B is at least as "hard" as A
  - more accurate: B at least as "expressive" as A

6

Using reductions

Definition: $A \leq_m B$ if there is a computable function f such that for all w

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

Theorem: if $A \leq_m B$ and B is decidable then A is decidable

Proof:

– decider for A: on input w, compute f(w), run decider for B, do whatever it does.

Using reductions

- Main use: given language NEW, prove it is undecidable by showing OLD $\leq_m$ NEW, where OLD known to be undecidable
  - proof by contradiction
  - if NEW decidable, then OLD decidable
  - OLD undecidable. Contradiction.
- common to reduce in wrong direction.
- review this argument to check yourself.

Using reductions

Theorem: if $A \leq_m B$ and B is RE then A is RE

Proof:

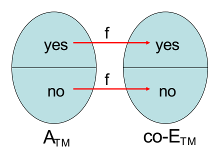
– TM for recognizing A: on input w, compute f(w), run TM that recognizes B, do whatever it does.

- Main use: given language NEW, prove it is not RE by showing OLD $\leq_m$ NEW, where OLD known to be not RE.

Many-one reduction example

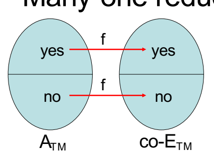
- Showed E_{TM} undecidable. Consider:

$$\text{co-}E_{TM} = \{<M> : L(M) \neq \emptyset\}$$



- f(<M, w>) = <M'> where M' is TM that
  - on input x, if $x \neq w$, then reject
  - else simulate M on x, and accept if M does
- f clearly computable

Many-one reduction example



- f(<M, w>) = <M'> where M' is TM that
  - on input x, if $x \neq w$, then reject
  - else simulate M on x, and accept if M does
- f clearly computable

- yes maps to yes?
  - if $<M, w> \in A_{TM}$ then $f(M, w) \in \text{co-}E_{TM}$
- no maps to no?
  - if $<M, w> \notin A_{TM}$ then $f(M, w) \notin \text{co-}E_{TM}$

Undecidable problems

Theorem: The language

REGULAR = {<M>: M is a TM and L(M) is regular}

is undecidable.

Proof:

- reduce from A_{TM} (i.e. show $A_{TM} \leq_m$ REGULAR)
- what should f(<M, w>) produce?

Undecidable problems

Proof:

– $f(<M, w>) = <M'>$ described below

on input x:

• if x has form 0^n1^n, accept

• else simulate M on w and accept x if M accepts

• is f computable?

• YES maps to YES?

$<M, w> \in A_{TM} \Rightarrow$ $f(M, w) \in$ REGULAR

• NO maps to NO?

$<M, w> \notin A_{TM} \Rightarrow$ $f(M, w) \notin$ REGULAR

Dec. and undec. problems

- the boundary between decidability and undecidability is often quite delicate
  - seemingly related problems
  - one decidable
  - other undecidable
- We will see two examples of this phenomenon next.

Computation histories

- Recall configuration of a TM: string uqv with $u,v \in \Gamma^*$, $q \in Q$
- The sequence of configurations M goes through on input w is a computation history of M on input w
  - may be *accepting*, or *rejecting*
  - reserve the term for halting computations
  - nondeterministic machines may have several computation histories for a given input.

Linear Bounded Automata

LBA definition: TM that is prohibited from moving head off right side of input.

- machine prevents such a move, just like a TM prevents a move off left of tape

- How many possible configurations for a LBA M on input w with $|w| = n$, m states, and $p = |\Gamma|$?
  - counting gives: mnp^n

Dec. and undec. problems

- two problems we have seen with respect to TMs, now regarding LBAs:
  - LBA acceptance:

    $A_{LBA} = \{<M, w> : \text{LBA M accepts input w}\}$
  - LBA emptiness:

    $E_{LBA} = \{<M> : \text{LBA M has } L(M) = \emptyset\}$
- Both decidable? both undecidable? one decidable?

Dec. and undec. problems

Theorem: A_{LBA} is decidable.

Proof:

- input <M, w> where M is a LBA
- key: only mnp^n configurations
- if M hasn't halted after this many steps, it must be looping forever.
- simulate M for mnp^n steps
- if it halts, accept or reject accordingly,
- else reject since it must be looping

Dec. and undec. problems

Theorem: E_{LBA} is undecidable.

Proof:

- reduce from co-A_{TM} (i.e. show co-$A_{TM} \leq_m E_{LBA}$)
- what should f(<M, w>) produce?
- Idea:
  - produce LBA B that accepts exactly the accepting computation histories of M on input w

February 2, 2024 CS21 Lecture 13 19

19

Dec. and undec. problems

Proof:

- f(<M, w>) = <B> described below

on input x, check if x has form

$\#C_1\#C_2\#C_3\#\ldots\#C_k\#$

- check that C_1 is the start configuration for M on input w
- check that $C_i \Rightarrow^1 C_{i+1}$
- check that C_k is an accepting configuration for M

- is B an LBA?
- is f computable?
- YES maps to YES?

  <M, w> $\in$ co-$A_{TM} \Rightarrow$ f(M, w) $\in E_{LBA}$
- NO maps to NO?

  <M, w> $\notin$ co-$A_{TM} \Rightarrow$ f(M, w) $\notin E_{LBA}$

February 2, 2024 CS21 Lecture 13 20

20