

CS21 Decidability and Tractability

Lecture 13
February 4, 2009

February 4, 2009

CS21 Lecture 13

1

Outline

- reductions
- more undecidable problems
 - computation histories
 - surprising contrasts between decidable/undecidable
 - Rice's Theorem
 - Post Correspondence Problem

February 4, 2009

CS21 Lecture 13

2

Reductions

- A better option:
 - to prove **NEW** is decidable, show how to transform it into a known decidable problem **OLD** so that solution to **OLD** can be used to solve **NEW**.
 - to prove **NEW** is undecidable, show how to transform a known undecidable problem **OLD** into **NEW** so that solution to **NEW** can be used to solve **OLD**.
- called a **reduction**

February 4, 2009

CS21 Lecture 13

3

Another example

- Try to prove undecidable:
 - $E_{TM} = \{ \langle M \rangle : L(M) = \emptyset \}$
- which problem should we **reduce from**?
 - $HALT = \{ \langle M, w \rangle : M \text{ halts on input } w \}$
 - $A_{TM} = \{ \langle M, w \rangle : M \text{ accepts input } w \}$
- Some things we can do:
 - check if $\langle M \rangle \in E_{TM}$
 - construct another TM M' and check if $\langle M' \rangle \in E_{TM}$

February 4, 2009

CS21 Lecture 13

4

Another example

- Construct TM M' :
 - on input x , if $x \neq w$, then reject
 - else simulate M on x , and accept if M does.
- on input $\langle M, w \rangle$
 - construct M' from description of M
 - check if $\langle M' \rangle \in E_{TM}$
 - if no, M must accept w ; **ACCEPT**
 - if yes, M cannot accept w ; **REJECT**



February 4, 2009

CS21 Lecture 13

5

Another example

- Preceding reduction proved:

Theorem: E_{TM} is undecidable.

Proof (recap):

- suppose E_{TM} is decidable
- we showed how to use E_{TM} to decide A_{TM}
- conclude A_{TM} is decidable. Contradiction.

February 4, 2009

CS21 Lecture 13

6

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

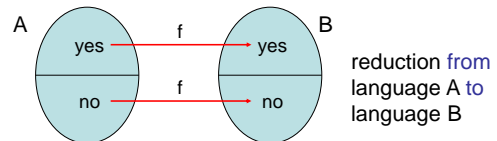
February 4, 2009

CS21 Lecture 13

7

Definition of reduction

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

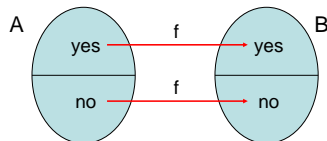


February 4, 2009

CS21 Lecture 13

8

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.

February 4, 2009

CS21 Lecture 13

9

Definition of reduction

- Notation: “A many-one reduces to B” is written

$$A \leq_m B$$

- “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

February 4, 2009

CS21 Lecture 13

10

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.

February 4, 2009

CS21 Lecture 13

11

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.

February 4, 2009

CS21 Lecture 13

12

Using reductions

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

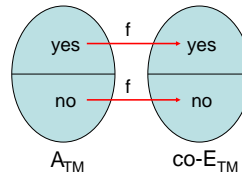
Proof:

- 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

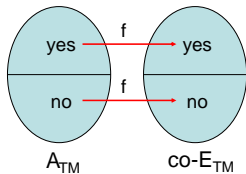
- Showed E_{TM} undecidable. Consider:

$$co-E_{TM} = \{ \langle M \rangle : L(M) \neq \emptyset \}$$



- $f(\langle M, w \rangle) = \langle M' \rangle$ 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



- $f(\langle M, w \rangle) = \langle M' \rangle$ 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 $\langle M, w \rangle \in A_{TM}$ then $f(M, w) \in co-E_{TM}$
- no maps to no?
 - if $\langle M, w \rangle \notin A_{TM}$ then $f(M, w) \notin co-E_{TM}$

Undecidable problems

Theorem: The language

$$REGULAR = \{ \langle M \rangle : M \text{ is a TM and } L(M) \text{ is regular} \}$$

is undecidable.

Proof:

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

Undecidable problems

Proof:

- $f(\langle M, w \rangle) = \langle M' \rangle$ described below

on input x:

- if x has form $0^n 1^n$, accept
- else simulate M on w and accept x if M accepts

- is f computable?
- YES maps to YES?
 - $\langle M, w \rangle \in A_{TM} \Rightarrow f(M, w) \in REGULAR$
- NO maps to NO?
 - $\langle M, w \rangle \notin A_{TM} \Rightarrow f(M, w) \notin REGULAR$

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.

February 4, 2009

CS21 Lecture 13

19

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

February 4, 2009

CS21 Lecture 13

20

Dec. and undec. problems

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

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

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

February 4, 2009

CS21 Lecture 13

21

Dec. and undec. problems

Theorem: A_{LBA} is decidable.

Proof:

- input $\langle M, w \rangle$ 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

February 4, 2009

CS21 Lecture 13

22

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 $f(\langle M, w \rangle)$ produce?
- Idea:
 - produce LBA B that accepts exactly the **accepting computation histories** of M on input w

February 4, 2009

CS21 Lecture 13

23

Dec. and undec. problems

Proof:

- $f(\langle M, w \rangle) = \langle B \rangle$ described below

on input x , check if x has form

$$\#C_1\#C_2\#C_3\#\dots\#C_k\#$$

- 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?

$$\langle M, w \rangle \in co-A_{TM} \Rightarrow f(\langle M, w \rangle) \in E_{LBA}$$

• NO maps to NO?

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

February 4, 2009

CS21 Lecture 13

24

Dec. and undec. problems

- two problems regarding Context-Free Grammars:
 - does a CFG generate all strings:
 $ALL_{CFG} = \{ \langle G \rangle : G \text{ is a CFG and } L(G) = \Sigma^* \}$
 - CFG emptiness:
 $E_{CFG} = \{ \langle G \rangle : G \text{ is a CFG and } L(G) = \emptyset \}$
- Both decidable? both undecidable? one decidable?

February 4, 2009

CS21 Lecture 13

25