Reducibility, part 3

April 17, 2020
Mid Term 2 is graded and back (yay!), ....and class did well (yay!!!)
Observations on mid term 2 solutions

Problem 1
• The vast majority nailed this!

Problem 2
• FMMs
  -- forget to read from input
  -- forget to handle (, )
  -- forget to push in reverse order

Problem 3
• Mostly fine, level of detail varied

Problem 4
• Generally, majority struggled
• FMM: treat w as part of input (it is not)

FMM: frequently made mistake

Problem 1: CFG
• 1.a) show that the grammar has two leftmost derivations
• 1.b) remove one rule to make the grammar unambiguous
• 1.c) show after your fix in b that this sentence has one derivation

Problem 2: CFG → PDA
• JH-Lisp

Problem 3: Give implementation-level description of TMs that decide languages
• 3.a) w contains twice as many Fs as Ts
• 3.b) w does not contain twice as many Fs as Ts

Problem 4: show that A is decidable
• A = \{<R> | R is a reg exp describing a language containing at least one string w that has 000 as a substring\}
Homework 7 is out, Due by Friday April 24

- **Problem 1** (20 pts)
  - **Topic**: show that collection of decidable languages is closed under the operations concat., star, compl.
  - **Lecture(s)**: 3/25
  - **Book**: sec 3.1

- **Problem 2** (20 pts)
  - **Topic**: Hilbert’s tenth problem, decidability of the problem in the single-variable polynomial case
  - **Lecture(s)**: 3/30
  - **Book**: sec 3.3

- **Problem 3** (20 pts)
  - **Topic**: show that a given language on DFA \(L(A) = \Sigma^*\) is decidable
  - **Lecture(s)**: 4/1, 4/3
  - **Book**: sec 4.1

- **Problem 4** (20 pts)
  - **Topic**: uncountable sets, diagonalization
  - **Lecture(s)**: 4/6
  - **Book**: sec 4.2

- **Problem 5** (10 pts)
  - **Topic**: co-Turing-recognizable languages
  - **Lecture(s)**: 4/8
  - **Book**: sec 4.2

- **Problem 6** (10 pts)
  - **Topic**: Magic: The Gathering
  - **Lecture(s)**: 4/1, 4/3, 4/6, 4/8
  - **Book**: sec 4.1, 4.2
  - + real-life
Now let us get to “reductions”…

Decidable:

- $A_{DFA}$ (acceptance)
- $A_{NFA}$
- $A_{REX}$
- $E_{DFA}$ (emptiness)
- $EQ_{DFA}$ (equivalence)
- $A_{CFG}$
- $E_{CFG}$
- $EQ_{CFG}$

Undecidable:

- $A_{TM}$
- $HALT_{TM}$
- $E_{TM}$
- $EQ_{TM}$
- $REGULAR_{TM}$
- $CFL_{TM}$
- $DECIDABLE_{TM}$

Diagonalization

Rice’s theorem
Reductions via computation histories

- The computation history method is an important technique for proving that $A_{\text{TM}}$ is reducible to certain languages.

- Often useful when the problem to be shown undecidable involves testing for the existence of something.
  - E.g. undecidability of Hilbert’s tenth problem, testing for the existence of integral roots in a polynomial.

- We defined computation history in last lecture and used it to show a decidable problem; today we will use it to show an example of an undecidable problem.
Let $M$ be a Turing machine and $w$ an input string. An accepting computation history for $M$ on $w$ is a sequence of configurations, $C_1, C_2, \ldots, C_l$, where $C_1$ is the start configuration of $M$ on $w$, $C_l$ is an accepting configuration of $M$, and each $C_i$ legally follows from $C_{i-1}$ according to the rules of $M$. A rejecting computation history for $M$ on $w$ is defined similarly, except that $C_l$ is a rejecting configuration.
A *linear bounded automaton* is a restricted type of Turing machine wherein the tape head isn’t permitted to move off the portion of the tape containing the input. If the machine tries to move its head off either end of the input, the head stays where it is—in the same way that the head will not move off the left-hand end of an ordinary Turing machine’s tape.

An LBA is a TM with a limited amount of memory.
Let:

$$A_{\text{LBA}} = \{ \langle M, w \rangle \mid M \text{ is an lBA that accepts string } w \}.$$ 

Lemma:

Let $M$ be an lBA with $q$ states and $g$ symbols in the tape alphabet. There are exactly $qng^n$ distinct configurations of $M$ for a tape of length $n$.

Theorem:

$A_{\text{LBA}}$ is decidable.
Let:

\[ E_{\text{LBA}} = \{ \langle M \rangle | M \text{ is an lBA where } L(M) = \emptyset \} \]

Theorem:

\[ E_{\text{LBA}} \text{ is undecidable.} \]
Proof idea

- The proof is reduction from $A_{TM}$
  - We show that if $E_{LBA}$ were decidable, $A_{TM}$ would also be decidable
- Suppose that $E_{LBA}$ is decidable
  - How can we use this assumption to decide $A_{TM}$?
- For a TM $M$ and an input $w$, we can determine whether $M$ accepts $w$ by constructing a certain LBA $B$ and then testing whether $L(B)$ is empty
- The language that $B$ recognizes comprises all accepting computation histories for $M$ on $w$
  - If $M$ accepts $w$, this language contains one string and so is nonempty
  - If $M$ does not accept $w$, this language is empty
  - If we can determine whether $B$’s language is empty, clearly we can determine whether $M$ accepts $w$
Proof idea

- Now we describe how to construct $B$ from $M$ and $w$
- Note that we need to show more than the mere existence of $B$
- We have to show how a TM can obtain a description of $B$, given descriptions of $M$ and $w$
- Note that we construct $B$ only to feed its description into the presumed $E_{LBA}$ decider, but not to run $B$ on some input
Proof idea

- We construct $B$ to accept its input $x$ if $x$ is an accepting computation history for $M$ on $w$

- We assume that the accepting computation history is presented as a single string with the configurations separated from each other by the # symbol

\[ \text{#} \quad C_1 \quad \text{#} \quad C_2 \quad \text{#} \quad C_3 \quad \text{#} \quad \cdots \quad \text{#} \quad C_l \quad \text{#} \]
Proof idea

- The LBA $B$ works as follows
  - When it receives an input $x$, $B$ is supposed to accept if $x$ is an accepting computation history for $M$ on $w$
  - First, $B$ breaks up $x$ according to the delimiters into strings $C_1, C_2, \ldots C_l$
  - Then $B$ determines whether the $C_i$’s satisfy the three conditions of an accepting computation history.
    1. $C_1$ is the start configuration for $M$ on $w$
    2. Each $C_{i+1}$ legally follows from $C_i$
    3. $C_l$ is an accepting configuration for $M$
  - By inverting the decider’s answer, we obtain the answer to whether $M$ accepts $w$
  - Thus we can decide $A_{TM}$, a contradiction
Putting it together: the proof

**Proof** Now we are ready to state the reduction of $A_{TM}$ to $E_{LBA}$. Suppose that TM $R$ decides $E_{LBA}$. Construct TM $S$ to decide $A_{TM}$ as follows.

$S = \text{“On input } \langle M, w \rangle, \text{ where } M \text{ is a TM and } w \text{ is a string:
  1. Construct lBA } B \text{ from } M \text{ and } w \text{ as described in the proof idea.}
  2. Run } R \text{ on input } \langle B \rangle.
  3. If } R \text{ rejects, accept; if } R \text{ accepts, reject.”}

If $R$ accepts $\langle B \rangle$, then $L(B) = \emptyset$. Thus, $M$ has no accepting computation history on $w$ and $M$ doesn’t accept $w$. Consequently, $S$ rejects $\langle M, w \rangle$. Similarly, if $R$ rejects $\langle B \rangle$, the language of $B$ is nonempty. The only string that $B$ can accept is an accepting computation history for $M$ on $w$. Thus, $M$ must accept $w$. Consequently, $S$ accepts $\langle M, w \rangle$. Figure 5.12 illustrates lBA $B$. 
One last result, $\text{ALL}_{\text{CFG}}$

Let:

$$\text{ALL}_{\text{CFG}} = \{ \langle G \rangle | G \text{ is a CFG and } L(G') = \Sigma^* \}.$$ 

Theorem:

$\text{ALL}_{\text{CFG}}$ is undecidable.

We state this result without proof, and it is enough for our purposes.
Another look at our growing list...

Decidable:

- $A_{DFA}$ (acceptance)
- $A_{NFA}$
- $A_{REX}$
- $E_{DFA}$ (emptiness)
- $EQ_{DFA}$ (equivalence)
- $A_{CFG}$
- $E_{CFG}$
- $EQ_{CFG}$
- $A_{LBA}$

Undecidable:

- $A_{TM}$
- $HALT_{TM}$
- $E_{TM}$
- $EQ_{TM}$
- $REGULAR_{TM}$
- $CFL_{TM}$
- $DECIDABLE_{TM}$
- $E_{LBA}$
- $ALL_{CFG}$

Rice's theorem

reduction
diagonalization

computation histories