Intro to Theory of Computation

LECTURE 20

CS
464

Last time

- Mapping reductions
- Computation history method

Today

- Computation history method
- Recursion theorem

Homework 8 due
Practice exam 2 out

Sofya Raskhodnikova

I-clicker question (frequency: AC)

A language L is Turing-recognizable $\Leftrightarrow$ $L \leq_m A_{TM}$

A. **Only the $\Rightarrow$ direction is true.**

B. **Only the $\Leftarrow$ direction is true.**

C. **Both directions are true.**

D. **Neither direction is true.**

I-clicker question (frequency: AC)

A two-dimensional automaton (2DIM-DFA) takes an $m \times n$ rectangle as input, for any $m, n \geq 2$. The boundary squares contain #; internal squares contain symbols from alphabet Σ. The transition function $\delta: Q \times (\Sigma \cup \{\#\}) \to Q \times \{L, R, U, D\}$ indicates the next state and head movement (left, right, up, down). How many distinct configurations does a 2DIM-DFA have on a given input?

A. $(m-2)(n-2)|Q|$

B. $mn|Q|$

C. $mn|Q| \cdot |\Sigma|$

D. $mn|Q| \cdot |\Sigma|^{(m-2)(n-2)}$

E. None of the above.

#	#	#	#	#	#	#
#	0	0	1	0	0	#
#	0	1	1	1	0	#
#	0	1	1	1	1	#
#	0	0	1	0	1	#
#	#	#	#	#	#	#

I-clicker question (frequency: AC)

Let $A_{\text{2DIM-DFA}} = \{\langle D, x\rangle \mid \text{D is a 2DIM-DFA and } D \text{ accepts } x\}$

1. Can a 2DIM-DFA loop?
2. Is $A_{\text{2DIM–DFA}}$ decidable?

A. YES to both.

B. NO to both.

C. YES to 1, NO to 2.

D. NO to 1, YES to 2.

Last time

We used a computation history method to show that E_{LBA} is undecidable.

$$\boxed{C_0 \# C_1 \# \ldots \# C_\ell}$$

starting configuration

accepting configuration

ALL_{PDA} is undecidable

$ALL_{PDA} = \{ \langle P \rangle \mid P$ is a PDA that accepts all strings $\}$

- We can use the computation history method to show that ALL_{PDA} is undecidable.
- It follows that ALL_{CFG} is undecidable.
- It follows that EQ_{CFG} is undecidable.

Problems in language theory

A_{DFA} decidable	A_{CFG} decidable	A_{TM} **un**decidable
E_{DFA} decidable	E_{CFG} decidable	E_{TM} **un**decidable
EQ_{DFA} decidable	EQ_{CFG} **un**decidable	EQ_{TM} **un**decidable

Post Correspondence Problem

Domino: $\left[\frac{a}{ab}\right]$ **. Top and bottom are strings.**

Input: collection of dominos.

$$\left[\frac{aa}{aba}\right], \left[\frac{ab}{aba}\right], \left[\frac{ba}{aa}\right], \left[\frac{abab}{b}\right]$$

Match: list of some of the input dominos (repetitions allowed) with top = bottom

$$\left[\frac{ab}{aba}\right], \left[\frac{aa}{aba}\right], \left[\frac{ba}{aa}\right], \left[\frac{aa}{aba}\right], \left[\frac{abab}{b}\right]$$

Problem: determine if a match exists.

POST={input with a match} is undecidable.

Recursion Theorem

Making TMs that can obtain their own descriptions

TM

with applications to ~~computer~~ viruses

A TM P_w that prints w

There is a computable function q that on input w outputs $\langle P_w \rangle$, where P_w is a TM that prints w.

> P_w= ``Erase input.
> 1. Print w and halt.''

TM computing q:

``On input w,

1. Print $\langle Pw \rangle$ and halt.''

TM S that prints $\langle S \rangle$

Theorem. There is a TM S that erases input, prints $\langle S \rangle$ and halts.

Proof:

- $q(w) = \langle P_w \rangle$

A$=P_{\langle B \rangle}$

B =``On input $\langle M \rangle$, where M is a TM,

1. Compute $\langle P_{\langle M \rangle} \rangle = q(\langle M \rangle)$.
2. Construct TM S':

   $\boldsymbol{S'}$ =``Erase input.
   1. Run $P_{\langle M \rangle}$.
   2. Run M.
   3. Halt.''

3. Output S' and halt.''

S =``Erase input.

1. Run TM A.
2. Run TM B.
3. Halt.''

In English

- Write this sentence.

- Write two copies of the following, the second one in quotes:

  ``Write two copies of the following, the second one in quotes:''

Recursion Theorem

If there is a TM T that computes a function $t(\langle R\rangle, w)$ then there is a TM R that computes $r(w) = t(\langle R\rangle, w)$.

Punchline: *``Obtain your own description''* is a valid step in an algorithmic description of a TM.

R =``Erase input.
1. Run TM A.
2. Run TM B.
3. Run TM T.
4. Halt.''

A=$P_{\langle BT\rangle}$

Application of recursion theorem

- A TM M is **minimal** if there is no TM equivalent to M that has a shorter description than $\langle M \rangle$.
- $MIN_{TM} = \{\langle M \rangle \mid M \text{ is minimal TM}\}$.
- Show that MIN_{TM} is not Turing-recognizable.