CS 332: *Theory of Computation* — Prof. Mark Bun
Boston University — October 25, 2022

Homework 6 – Due Thursday, October 28, 2022 at 11:59 PM

Reminder Collaboration is permitted, but you must write the solutions *by yourself without assistance*, and be ready to explain them orally to the course staff if asked. You must also identify your collaborators and write "Collaborators: none" if you worked by yourself. Getting solutions from outside sources such as the Web or students not enrolled in the class is strictly forbidden.

Note You may use various generalizations of the Turing machine model we have seen in class, such as TMs with two-way infinite tapes, stay-put, or multiple tapes. If you choose to use such a generalization, state clearly and precisely what model you are using.

Problems There are 4 required problems.

1. (**Midterm feedback survey**) Please fill out the mid-semester feedback form here `https://forms.gle/WsL8GHRSw2X9LntD7` to let us know what's working, what isn't working, and what we can do to improve. (You'll earn a bit of participation credit for acknowledging that you completed it in response to this question, but the survey is anonymous, so it's on the honor system.)

2. (**Nondeterministic Turing machines**)

   (a) Give a high-level description of a **nondeterministic** (multi-tape) TM recognizing the following language over $\{0, 1, \#\}$: $\{s\#a_1\#a_2\# \dots \#a_n \mid$ where n is a positive integer, $a_1, \dots, a_n$ are binary integers such that some subset of $a_1, \dots, a_n$ sums to exactly $s\}$.

   Note: It is possible to do this with a deterministic TM, but we want to give you practice with the concept of nondeterminism. So your solution must use an NTM's ability to nondeterministically guess in a meaningful way.

   (b) Given a Turing machine M, give a high-level description of a **nondeterministic** (multi-tape) TM recognizing $(L(M))^*$. Again, your solution must use nondeterminism in a meaningful way.

   (c) Explain why part (b) implies that the Turing-recognizable languages are closed under star.

   (d) Explain (briefly) how you would modify your previous construction and its analyses to show that the **decidable** languages are closed under star.

   Hint: Recall that a nondeterministic TM is a decider if it halts on every input, on every computation branch. The class of languages decided by NTMs is exactly the class of decidable languages.

3. (**Code as data**) The goal of this problem is to help you get comfortable with the idea of Turing machines taking descriptions of other Turing machines as input. Consider the following description of a three-tape TM H.

> **Algorithm 1:** $H(\langle M, w\rangle)$
>
> **Input** : Encoding of a basic single-tape TM M and a string $w \in \{\mathrm{A}, \mathrm{B}, \ldots, \mathrm{Z}\}^*$
>
> 1. Copy the the string w to tape 2.
> 2. Repeat the following three steps forever:
> 3. Simulate M for one step on tape 2.
> 4. Erase the contents of tape 3. Copy the contents of tape 2 to tape 3, and check if the substring "TURING" appears on tape 3. If it does, *accept*. Otherwise, continue.
> 5. If M halts (in either an accept or reject state), *reject*. Otherwise, continue.

(a) Let M_1 be the following (uninteresting) TM. Is $\langle M_1, \varepsilon\rangle \in L(H)$? Explain why or why not.

> **Algorithm 2:** $M_1(x)$
>
> **Input** : String $x \in \{\mathrm{A}, \ldots, \mathrm{Z}\}^*$
>
> 1. Write "ALANTURINGWASHERE" to the tape and *reject*.

(b) Let M_2 be the following TM. Is $\langle M_2, \mathrm{LAMBDA}\rangle \in L(H)$? Explain why or why not.

> **Algorithm 3:** $M_2(x)$
>
> **Input** : String $x \in \{\mathrm{A}, \ldots, \mathrm{Z}\}^*$
>
> 1. Scan the input string x left-to-right, replacing every "A" with "U" and every "I" with "O". *Accept* if the last symbol of x is "Z".
>   Otherwise, *reject*.

(c) Is $\langle M_2, \mathrm{PUTARINGONIT}\rangle \in L(H)$? Explain why or why not.

(d) What is the language $L(H)$ recognized by H?

(e) Is H a decider for the language $L(H)$? Explain why or why not.

4. ($\mathrm{SUB}_{\mathrm{DFA,REX}}$) Consider the following computational problem: Given a DFA D and a regular expression R, is the language recognized by D a subset of the language generated by R?

(a) Formulate this problem as a language $\mathrm{SUB}_{\mathrm{DFA,REX}}$.

(b) Show that $\mathrm{SUB}_{\mathrm{DFA,REX}}$ is decidable.

Hint: Following the examples in Sipser Chapter 4.1, you may assume that the procedures we've seen in class for converting back and forth between automata and regular expressions can be implemented on Turing machines.