Homework 5 – Due Tuesday, October 19, 2021 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 3 required problems and 1 bonus problem.

1. (**Midterm feedback survey**) Please fill out the mid-semester feedback form here `https://forms.gle/quetnL595uJjQWoBA` 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. (**Insert-only TM**) An *insert-only* Turing machine (ITM) is the same as a basic (deterministic) one-tape Turing machine, but instead of writing a new symbol to the current cell under the tape head, it inserts a new cell with that symbol to the immediate left of the current cell. It can also move the tape head while leaving the tape content unchanged.

   Here are some examples of how an ITM could work.

   - If an ITM is in configuration $867p309$ and its transition function specifies $\delta(p, 3) = (q, 5, \mathtt{R})$, then the next configuration will be $86753q09$. (The ITM reads symbol 3, inserts symbol 5, then moves to the right.)
   - If an ITM is in configuration $867r309$ and its transition function specifies $\delta(r, 3) = (s, 5, \mathtt{L})$, then the next configuration will be $867s5309$. (The ITM reads symbol 3, inserts symbol 5, then moves to the left.)
   - If an ITM is in configuration $867s5309$ and its transition function specifies $\delta(s, 5) = (t, *, \mathtt{L})$, then its next configuration will be $86t75309$. Here, "writing" the $*$ symbol indicates that the TM should move without modifying the tape.

   (a) Modify Definition 3.3 in Sipser to give the formal definition of a insert-only TM. Most of it should stay the same, but pay special attention to the syntax of the transition function (item 4), which should handle the special $*$ symbol that is not part of the tape alphabet.

   (b) Show that insert-only TMs are *no more* powerful than basic TMs. That is, use an implementation-level simulation argument to show that every language recognizable by an insert-only TM is also Turing-recognizable.

   (c) Show that insert-only TMs are *at least* as powerful as basic TMs. That is, use an implementation-level simulation argument to show that every Turing-recognizable language is also recognizable by an insert-only TM.

   Hint: You may want to enlarge the tape alphabet of your ITM to include another special symbol that tells it to ignore some of the tape content.

Parts (a) and (b) together show that insert-only TMs are equivalent to basic TMs: They exactly recognize the class of Turing-recognizable languages.

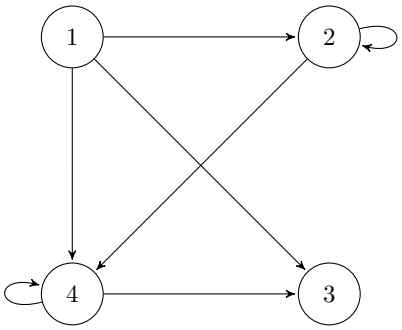
3. (**Nondeterministic TMs**)

   (a) Given a basic Turing machine M recognizing a language L, give an implementation-level description of a **nondeterministic** (multi-tape) TM recognizing $SYM(L) = \{xy \mid yx \in L\}$. Briefly explain why your construction works. Note: You may have done this last week in Problem 4b 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. The construction should be a lot simpler!

   (b) To present a (directed) graph G as input to a Turing machine, one needs to provide a string *encoding* it, denoted by $\langle G \rangle$. One convenient such encoding is the flattened adjacency matrix representation: A graph G on vertices $1, 2, \ldots, n$ can be encoded as $\langle G \rangle = \#w_1\#w_2\#\ldots\#w_n\#$ where each $w_i \in \{0,1\}^n$ with $w_{i,j} = 1$ if there is an edge from i to j in G, and $w_{i,j} = 0$ if there is not an edge.

      i. Draw the directed graph on 3 vertices that is encoded by #100#110#001#.

      ii. What is the encoding $\langle H \rangle$ of the following graph H?



   (c) A (directed) *triangle* in a directed graph consists of three vertices i, j, k, such that there is an edge from i to j, an edge from j to k, and an edge from k to i. Thus, a digraph G contains a triangle if and only if there exist i, j, k such that $w_{i,j} = w_{j,k} = w_{k,i} = 1$ in its encoding.

   Give an implementation-level description of a **nondeterministic** (multi-tape) TM deciding the language $L = \{\langle G \rangle \mid$ there exists a triangle in directed graph $G\}$. Briefly explain why your construction works. Again, your solution must use nondeterminism in a meaningful way.

   Hint: Recall that a nondeterministic TM is a decider if it halts on every input, on every computation branch.

4. (**Bonus problem**) Let A be a Turing-recognizable language which is not decidable. (We will prove later in the course that such languages exist.) Consider a TM M that recognizes A. Prove that there are infinitely many input strings on which M loops forever. If you need to construct a TM to solve this problem, you can give a high-level description.