

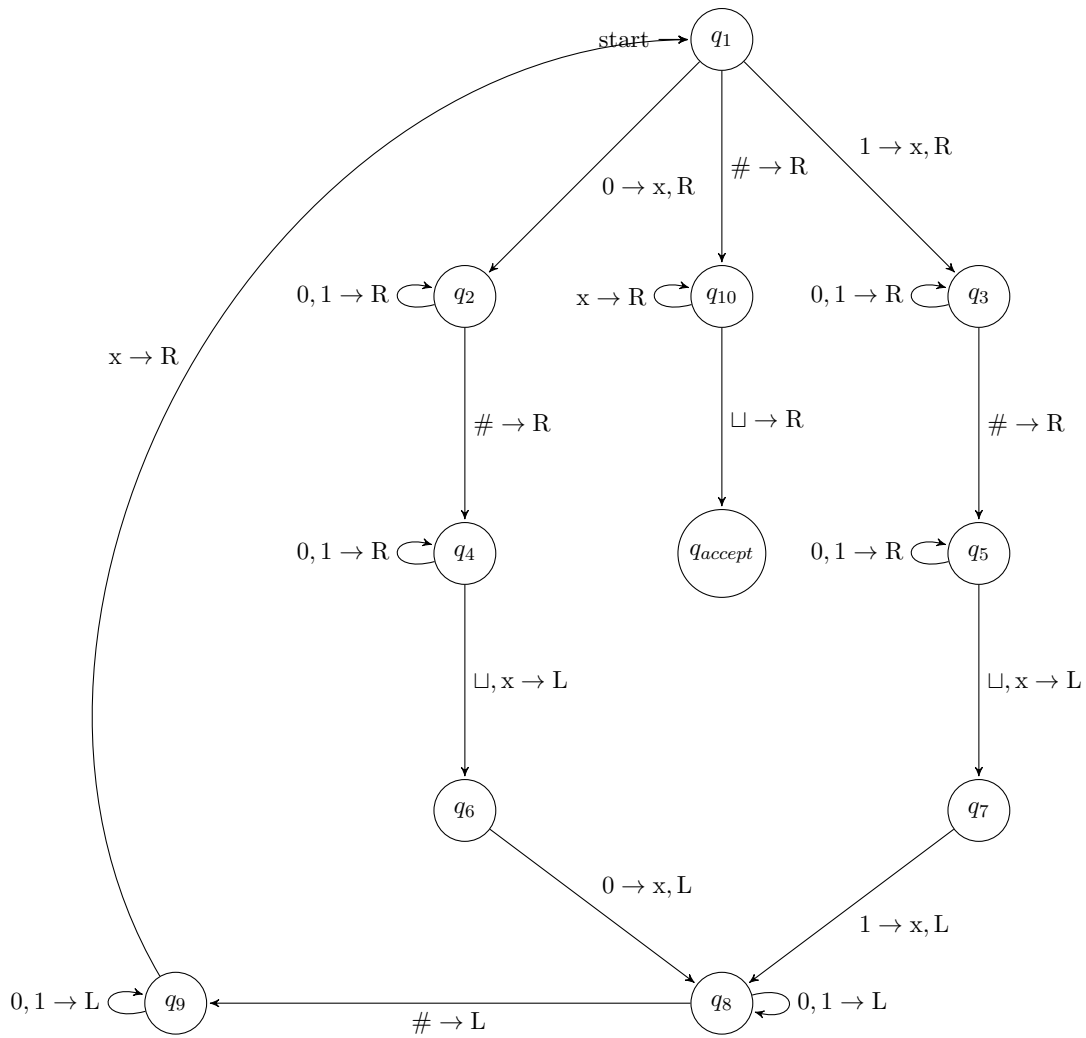
Homework 5 – Due Thursday, March 5, 2026 at 11:59 PM

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. Collaboration is not allowed on problems marked “INDIVIDUAL.”

Problems There are 5 required problems.

1. (**Low-Level to Implementation-Level**) The following page illustrates the state diagram of a Turing machine using input alphabet $\Sigma = \{0, 1, \#\}$ and tape alphabet $\Gamma = \{0, 1, \#, x, \sqcup\}$.

The notation “ $a \rightarrow R$ ” is shorthand for “ $a \rightarrow a, R$.” The reject state and transitions to the reject state are not shown. Whenever the TM tries to read a character for which there is no explicit transition that means that the TM goes to the reject state. Use the convention that the head moves right in each of these transitions to the reject state.



- Give the sequences of configurations that this TM M enters when given as input strings (i) $010\#100$, (ii) $10\#01$, and (iii) $0\#\#0$. Use the same representation for your configurations as we did in lecture 9.
- Give an implementation-level description of the Turing machine described by this state diagram. Hint: The machine is similar to Example 3.9 in Sipser.
- What is the language decided by M ?

2. (Implementation-Level to Low-Level)

- Give a state diagram of a TM whose implementation-level description is below. The input alphabet of this TM is $\{a, b\}$ and the tape alphabet is $\{a, b, x, \sqcup\}$.

Input : String w

1. Read the first symbol on the input string. If it is a blank symbol, *accept*. If it is an a , then erase this a (i.e., replace it with a \sqcup), move the head right, and go to step 5. If it is a b , then erase this b , move the head right, and go on to the next step.
 2. Continue moving the head right until a blank symbol is found. Replace this blank symbol with a b , move the head left, and go to step 6.
 3. Move the head right until either an a or a blank symbol is found. If it is a blank symbol, *accept*. If it is a then cross out this a (i.e., replace it with an x), move the head left, and go on to the next step.
 4. Repeatedly move the head left until the blank symbol is found. After it is found move the head one cell to the right and go on to the next step.
 5. Continue moving the head right until a b or a blank symbol is found. If a blank symbol is found, *reject*. Otherwise, cross out the b that was found, move the head one cell left, and go on to the next step.
 6. Continue moving the head left until the blank symbol is found. When it is found, move the head right and go back to step 3.
- (b) Give the sequences of configurations that your TM enters when given as input strings (i) *abbab*, and (ii) *baaab*.
- (c) What language is decided by the TM from part (a)?
3. (**Programming TMs**) Construct Turing machines that decide the following languages. That is, the machines should always halt after a finite number of steps on every input, and accept a string w if and only if w is in the given language. Implement your TMs in the following environment: <http://morphett.info/turing/turing.html>. Your solution should contain:
- (i) An implementation-level description of your code.
 - (ii) Code that we can copy from your submission and run directly on that website. (Please add comments and make your code as readable as possible.)

Your TM should enter a state that is clearly marked “accept” or “reject” when it is done. The final tape content does not matter; it does not need to draw an emoji or anything else on its tape.

- (a) $L_1 = \{w \in \{0, 1\}^* \mid \text{Every even position of } w \text{ is a } 0\}$.

Hint: You could solve this problem with a DFA. How would you simulate that DFA using a TM?

- (b) $L_2 = \{0^n 1^m \mid m, n \geq 0 \text{ and } m \text{ is a multiple of } n\}$.
4. (**Recognizability vs. Decidability**) Recall the high-level description of a TM recognizer for the language $\{\langle p \rangle \mid p \text{ is a two-variable integer polynomial and there exists } x, y \text{ such that } p(x, y) = 0\}$ that we described in class.
- Explain in a few sentences what is **wrong** about the following attempt to construct a TM decider for the same language:
5. (**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

Input : Encoding of polynomial p

1. For every possible setting of (x, y) to a pair of integer values:
2. Evaluate $p(x, y)$. If it equals 0, *accept*.

Input : Encoding of polynomial p

1. Found = False
2. For every possible setting of (x, y) to a pair of integer values:
3. If $p(x, y) = 0$, set Found = True.
4. If (Found = True), *accept*. Otherwise, *reject*.

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 $\delta(p, 3) = (q, 5, \mathbf{R})$, 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 $\delta(r, 3) = (s, 5, \mathbf{L})$, 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 $\delta(s, 5) = (t, *, \mathbf{L})$, 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 signals 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.