HW5 - Variations on a Turing Machine

COMP361 — Suhas Arehalli

Spring 2025

Instructions

As with previous homeworks,

- Deadlines are posted on the course website.
- Solutions should be turned in as a typeset pdf, preferably using LaTeX. I recommend a tool like Overleaf.
- Assignments are individual, and you should not share solutions with other students. However, you are free to discuss problems within the bounds of the academic integrity policy in the syllabus, and should indicate in your assignment the people you got assistance from (including instructors, peers, and preceptors!).
- Proofread your work before submission, and consult the proof style guide to make sure your submissions are both correct and clear.
- Graded problems will be indicated by an asterisk (*) next to the problem number. You are still expected to solve all of the problems.

Questions

1. ***(Sipser 3.21)* Consider a **Queue Automaton**, a model like a Push-Down Automaton but using a queue data structure for external memory rather than a stack.

   For ease, we can formalize the model more than Sipser does, using TM/PDA-like conventions, in case that helps your understanding of the machine. Remember that our descriptions for conversions are meant to be *high-level*, though they should be precise enough that a non-malicious reader can accurately formalize what you describe.

   A **Queue Automaton** can be defined as a 6-tuple $(Q, \Sigma, \Gamma, \delta, q_0, F)$ with a finite set of states Q, input alphabet Σ, queue alphabet Γ with special symbol $\$ \in \Gamma$, a transition function $\delta : Q \times \Sigma_\varepsilon \times \Gamma_\varepsilon \to Q \times \Gamma_\varepsilon$.

   A configuration of a queue automaton is defined as a pair $(q, v, s) \in Q \times \Sigma^* \times \Gamma^*$, where q represents the current state, v represents the remaining input, and s represents the contents of the queue. A string w is accepted by a queue automaton $M = (Q, \Sigma, \Gamma, \delta, q_0, F)$ iff $\exists (r_1, v_1, s_1), \ldots, (r_k, v_k, s_k)$ such that

1. $(r_1, v_1, s_1) = (q_0, w\$, \varepsilon)$
2. For all choices of i, $\exists a, b \in \Gamma_\varepsilon$, $c \in \Sigma_\varepsilon$ and $s' \in \Gamma^*$ such that
   - $\delta(r_i, c, a) = (r_{i+1}, b)$,
   - $v_i = cv_{i+1}$,
   - $s_i = as'$, and
   - $s_{i+1} = s'b$
3. $r_k \in F$

i.e., we have a machine that consumes it's input (but can choose to make ε-transitions!) — equivalent to the unidirectional input tape Sipser describes — and can optionally queue and dequeue from a queue. Again, the exact formal description is just to make sure you can verify exactly what the limitations of the machine might be if you're unsure.

(a) Show a Turing Machine can simulate a Queue Automaton. That is, given a Queue Automaton M, prove we can design a TM that recognizes $L(M)$.

(b) We'll show that the Queue Automaton can simulate a Turing machine in a few steps. First, consider this scheme to use a queue to represent a TM's tape: Recall that we represent a TM's configuration as a triple u, q, v where the tape's contents consist of uv and the head points to the first symbol of v. Let's have our queue represent this with a queue string $s = v\$u$, so that the head of our queue corresponds to the head of our tape, and the \$ symbol demarcates the beginning of our tape. Suppose a TM wants to make a transition from state q_i to q_j that reads a symbol a, writes a symbol b, and moves to the right. Describe how you would implement this operation using a queue automaton instead of a stack. Be careful of edge cases.

(c) Now consider a similar transition, but you move to the left rather than the right. This will be trickier! Feel free to modify our scheme for encoding a tape into a queue if helpful. ***HINT**:* Try and find a way to make enough simulated rightward moves that you end up in the same configuration you'd be in if you had made a leftward move. IF you're stuck, experiment!

(d) Note that a Queue Automaton, as defined here, consumes its input while a TM begins with its input on its tape. Describe how a Queue Automaton might consume it's input such that it begins in an appropriate configuration to start simulating TM transitions.

2. *(Sipser 3.20)* Consider the **SR-TM**, a Turing Machine that replaces leftward head movements with a stay-put operation. i.e., $\delta : Q \times \Gamma \to Q \times \Gamma \times \{S, R\}$.

   (a) This machine is weaker than a regular Turing Machine. Explore the limitations of this model by thinking through what it can use it's tape memory for. What class of languages does this type of model recognize?

   (b) Prove this by showing a less expressive model of computation can simulate this machine.