HW3

Graded

Student

Vidit Dharmendra Pokharna

Total Points

40 / 40 pts

Question 1

Q1

8 / 8 pts

✔ **– 0 pts** Correct

– 1 pt Minor Error

– 4 pts Major Error

– 6 pts Incorrect

Question 2

Q2

8 / 8 pts

✔ **– 0 pts** Correct

– 1 pt Minor error

– 2 pts Logic error

– 4 pts Incorrect Machine

– 6 pts Algorithm Approach

– 8 pts No answer

Question 3

Q3

8 / 8 pts

✔ **– 0 pts** Correct

– 2 pts Minor error

– 4 pts Major error

– 8 pts Completely wrong or no answer

1 should be a on the beginning here

Question 4

Q4

8 / 8 pts

✔ **– 0 pts** Correct

– 4 pts Major Error

– 1 pt Minor Error

– 2 pts Did not clear the tape to the right of x_i

– 4 pts Did not show how to count the number of 1s (value of i). Counting should be simulated on the TM

– 8 pts No answer

– 7 pts Question asks for the tape to be halted with just x_i on the tape, not just the pointer pointing to x_i

– 2 pts Explanation is not clear

– 8 pts Incorrect

Question 5

Q5

8 / 8 pts

✔ **– 0 pts** Correct

– 1 pt Very minor error

– 2 pts Minor Error

– 4 pts Major Error

– 6 pts Incorrect

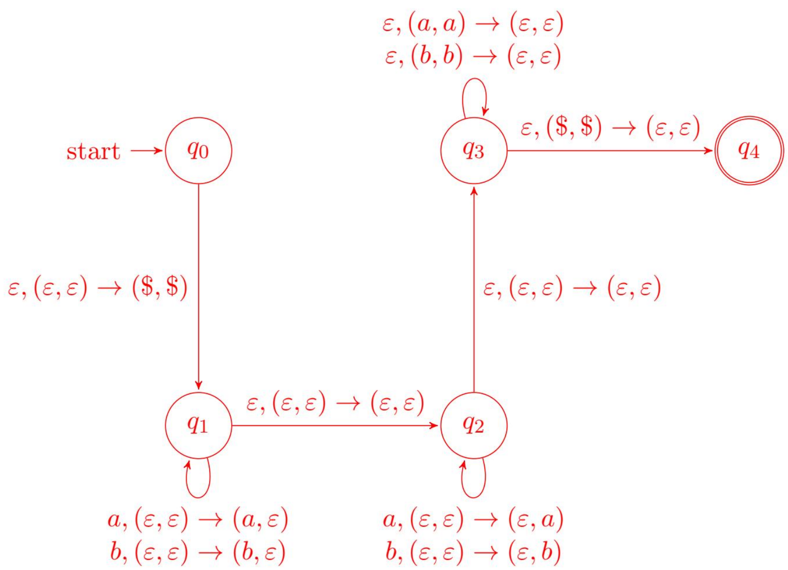
– 8 pts No answer

Questions assigned to the following page: 1 and 2

CS 4510 Automata and Complexity	02/15/2024
Homework 3: Turing Machines	
Vidit D. Pokharna	*Due:2/23/2024*

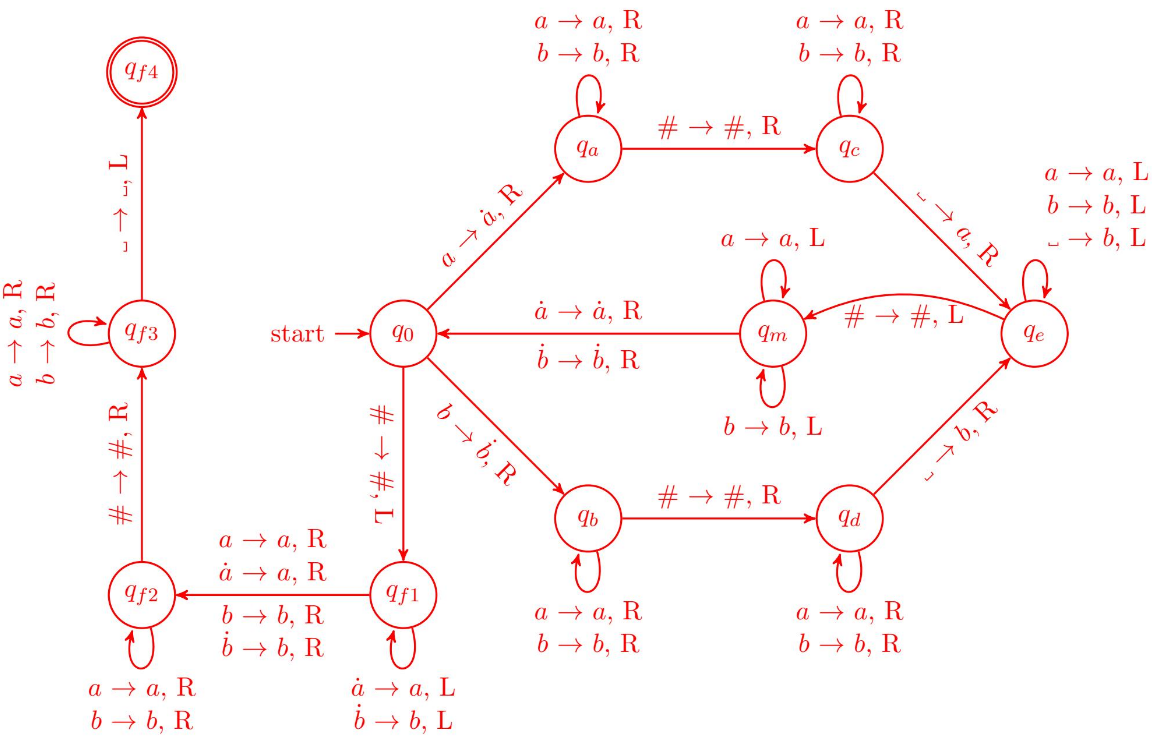
You should submit a typeset or *neatly* written pdf on Gradescope. The grading TA should not have to struggle to read what you've written; if your handwriting is hard to decipher, you will be asked to typeset your future assignments. Four bonus points if you use LaTeX, and our template. You may collaborate with other students, but any written work should be your own.

1. Suppose we have a push down automata with two stacks. It has nondeterministic transitions of the form `read`, (`pop1`,`pop2`) $\to$ (`push1`,`push2`). It reads from the input, pops from both stacks, and pushes to both stacks simultaneously. Give the state diagram of this PDA with two stacks to decide $\{ww \mid w \in \Sigma^*\}$. (Hint: What allowed a single stack PDA to decide ww^R but not ww?)



2. Give the state diagram of a Turing machine which begins with $w\#$ on its tape for any $w \in \Sigma^*$ and halts with $w\#w$ on its tape. The input alphabet is $\Sigma = \{a, b\}$ and the tape alphabet may be $\Gamma = \{a, b, \dot{a}, \dot{b}, \#, \llcorner\!\lrcorner\}$. This Turing machine performs the copy ability. We gave a decision version of this in class. You are giving the computation version.

Questions assigned to the following page: 3 and 2



3. Give the sequence of configurations of your state diagram from question 2 beginning from $q_0aba\#$. Each on a new line please. (Hint, why not write a program for this)

$q_0aba\#$
$\dot{a}q_aba\#$
$\dot{a}bq_aa\#$
$\dot{a}baq_a\#$
$\dot{a}ba\#q_c\text{\textvisiblespace}$
$\dot{a}ba\#aq_e$
$\dot{a}ba\#q_ea$
$\dot{a}baq_e\#a$
$\dot{a}bq_ma\#a$
$\dot{a}q_mba\#a$
$q_m\dot{a}ba\#a$
$\dot{a}q_0ba\#a$
$\dot{a}\dot{b}q_ba\#a$
$\dot{a}\dot{b}aq_b\#a$
$\dot{a}\dot{b}a\#q_da$
$\dot{a}\dot{b}a\#aq_d\text{\textvisiblespace}$
$\dot{a}\dot{b}a\#abq_e$
$\dot{a}\dot{b}a\#aq_eb$
$\dot{a}\dot{b}a\#q_eab$
$\dot{a}\dot{b}aq_e\#ab$

Questions assigned to the following page: 3 and 4

$\dot{a}\dot{b}q_m a\#ab$

$\dot{a}q_m \dot{b}a\#ab$

$\dot{a}\dot{b}q_0 a\#ab$

$\dot{a}\dot{b}\dot{a}q_a\#ab$

$\dot{a}\dot{b}\dot{a}\#q_c ab$

$\dot{a}\dot{b}\dot{a}\#aq_c b$

$\dot{a}\dot{b}\dot{a}\#abq_c \text{\textvisiblespace}$

$\dot{a}\dot{b}\dot{a}\#abaq_e$

$\dot{a}\dot{b}\dot{a}\#abq_e a$

$\dot{a}\dot{b}\dot{a}\#aq_e ba$

$\dot{a}\dot{b}\dot{a}\#q_e aba$

$\dot{a}\dot{b}\dot{a}q_e\#aba$

$\dot{a}\dot{b}q_m\dot{a}\#aba$

$\dot{a}\dot{b}\dot{a}q_0\#aba$

$\dot{a}\dot{b}q_{f1}\dot{a}\#aba$

$\dot{a}q_{f1}\dot{b}a\#aba$

$q_{f1}\dot{a}ba\#aba$

$aq_{f2}ba\#aba$

$abq_{f2}a\#aba$

$abaq_{f2}\#aba$

$aba\#q_{f3}aba$

$aba\#aq_{f3}ba$

$aba\#abq_{f3}a$

$aba\#abaq_{f3}\text{\textvisiblespace}$

$aba\#abq_{f4}a\text{\textvisiblespace}$

4. Give a high level, detailed, description of a Turing machine which computes the projection function $U(n, i, x_1, x_2, ..., x_n) = x_i$. Do not give a state diagram. The Turing machine must begin with $1^n\#1^i\#x_1\#...\#x_n$ and halt with just x_i on its tape left-shifted fully. If this was psuedocode, the Turing machine would compute the following algorithm:

```
def U(n, i, A[]):
    return A[i]
```

Initialization: The Turing machine begins by marking the start of the tape. It replaces the first "1" with a special symbol, say "S", to denote the start. This marking is crucial for the machine to find its way back to the beginning of the tape.

Counting to i: The machine moves right, clearing any "1"s it encounters until it reaches the first "#". This marks the end of the 1^n sequence. Upon encountering the first "#", the machine enters a new state to handle the 1^i sequence. For each "1" in the 1^i sequence that is followed by another "1", the machine crosses out the "1" and continues moving right until it encounters a "#", which it then deletes. This action signifies that one element has been bypassed, and the machine continues clearing everything until the next "#".

Questions assigned to the following page: 4 and 5

Locating x_i: After each "1" in the 1^i sequence is crossed out and the subsequent "#" is cleared, the machine moves left to find the last crossed-out "1" and repeats the bypass process, until it encounters a "1" that is not immediately followed by another "1" but a "#". This indicates the i^{th} element is next. The machine then changes its logic, moving right past the "#" (clearing it), and continues until it finds the next "#", marking the start of x_i.

Retrieving x_i: Upon clearing the "#" that precedes x_i, the machine carefully moves over x_i, without altering it, until it encounters the next "#", which signifies the end of x_i. The machine clears this "#" to indicate the end of the retrieval phase.

Clearing Remaining Elements: The machine then proceeds to clear all subsequent characters until the tape is empty, effectively removing x_{i+1} to x_n.

Shifting x_i **Left**: The machine moves back to the start marker "S" and begins the process of left-shifting x_i. It does this by relocating each character of x_i to the leftmost position available, starting from the "S" marker, until x_i is fully left-shifted and alone on the tape.

Final State: The Turing machine halts when x_i is the only content left on the tape, fully left-shifted, with the rest of the tape cleared.

5. Prove that the computable functions are closed under composition.

Consider two computable functions f and g, with corresponding Turing machines S and T that compute these functions, respectively. Assume that the states of S and T are disjoint. Let Q_S be the set of states of S, with q_{S0} as the initial state and q_{SH} as the halting state. Similarly, let Q_T be the set of states of T, with q_{T0} and q_{TH} as its initial and halting states, respectively. Let δ_S and δ_T be the transition functions for S and T.

To prove that the composition $g \circ f$ is computable, we construct a Turing machine $S \circ T$ as follows:

- The set of states $Q_{S \circ T}$ is the union of the states of S without its halting state and the states of T: $Q_{S \circ T} = (Q_S \setminus \{q_{SH}\}) \cup Q_T$.
- The initial state of $S \circ T$ is the initial state of S: $q_{S \circ T,0} = q_{S0}$.
- The halting state of $S \circ T$ is the halting state of T: $q_{S \circ T,H} = q_{TH}$.
- The transition function $\delta_{S \circ T}$ includes the transitions of S and T, with the transition from q_{SH} in S modified to go to q_{T0} in $S \circ T$.

This construction ensures that after S halts, instead of stopping, the machine transitions to the initial state of T and continues computation. Thus, $S \circ T$ effectively computes f and then g, which by the definition of computable functions, means that the composition $g \circ f$ is computable. This shows that the set of computable functions is closed under composition. ■