CS 181 Midterm

WILLIAM CULVER RANDALL

TOTAL POINTS

223 / 302

QUESTION 1

Question 1: FFAs 130 pts

1.1 1a 27 / 30

✓ **- 3 pts** **Weak explanation**

Showing that two strings are either correctly accepted or rejected by the machine does not count as a full explanation. An explanation should consist of general intuition about the machine.

1.2 1b 20 / 20

✓ **- 0 pts** **Correct**

1.3 1c 49 / 50

✓ **- 1 pts** **FFAs are nondeterministic, so there may be multiple computational paths. You want to consider the execution on some accepting path, not "the" path.**

1.4 1d 5 / 30

✓ **- 25 pts** **Cannot assume that the permutation w of x is equal to x. You have to either show a contradiction for all permutations w of x or prove that the only valid permutation w of x for this PL is equal to x.**

QUESTION 2

Question 2: TFAs 120 pts

2.1 2a 20 / 20

✓ **- 0 pts** **Correct**

2.2 2b 20 / 20

✓ **- 0 pts** **Correct**

2.3 2c 30 / 30

✓ **- 0 pts** **Correct**

2.4 2d 50 / 50

✓ **- 0 pts** **Correct: transition modification clearly described and explained, accepting states modified.**

QUESTION 3

3 Extra Credit: FPDA 0 / 50

+ 50 pts Correct

- 10 pts No explanation

+ 50 pts Correct, except that your machine doesn't accept the empty string.

+ 45 pts Partial Credit: Correct, except that your machine does not accept strings with an equal number of 0's and 1's in the first half. Recall that 0 is the tie-breaker for majority.

+ 45 pts Partial Credit: Correct, except that if there are an equal number of 0's and 1's in the first half, your machine can accept if the second half consists of entirely 1's. Recall that 0 is the tie-breaker for majority.

+ 10 pts Partial credit: concrete construction reflects a high level idea with potential.

+ 10 pts Partial credit: Correct intuition, but no concrete construction

✓ **+ 0 pts** **No attempt/I don't know/vague idea only**

+ 0 pts Incorrect

QUESTION 4

4 Extra Credit: Automadoggos 2 / 2

✓ **- 0 pts** **Excellent!**

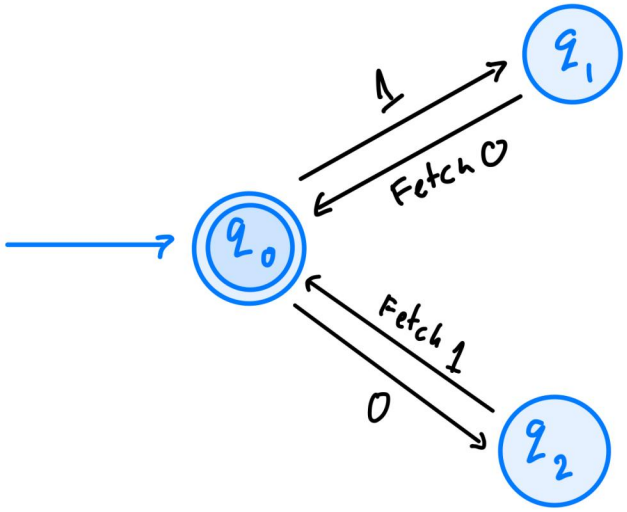
QUESTION 5

5 Honor Code Agreement 0 / 0

✓ **- 0 pts** Correct

1. **FFA**

(a) L_{EQ}



For this FFA if I input a string $s = 1001 \in L_{EQ}$ into the FFA it will start at q_0 then move to q_1 on 1 then it will fetch the 0 and it will repeat with the remaining input '10'.
This FFA will reject $s = 11 \notin L_{EQ}$ because it will get stuck at q_1.

(b) L_F

$$x = 1222211000 \in L(F)$$

1222200011

1222210001

1202020211

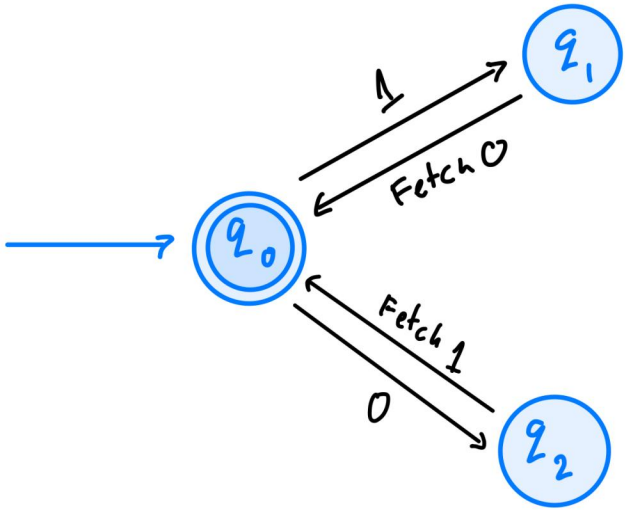
1.1 1a 27 / 30

✓ **- 3 pts** **Weak explanation**

Showing that two strings are either correctly accepted or rejected by the machine does not count as a full explanation. An explanation should consist of general intuition about the machine.

1. **FFA**

(a) L_{EQ}



For this FFA if I input a string $s = 1001 \in L_{EQ}$ into the FFA it will start at q_0 then move to q_1 on 1 then it will fetch the 0 and it will repeat with the remaining input '10'.
This FFA will reject $s = 11 \notin L_{EQ}$ because it will get stuck at q_1.

(b) L_F

$$x = 1222211000 \in L(F)$$

1222200011

1222210001

1202020211

1.2 1b 20 / 20

✓ - 0 pts Correct

(c) **Fetching Pumping Lemma**

Claim:
For all Fetching languages L, there exists an $n \in \mathbb{N}$ s.t. for all $x \in L$ with $|x| \geq n$, there exists strings a,b,c s.t. $w = abc \in L$ where w is a permutation of x and $|ab| \leq n$, $|b| > 0$, and $\forall i \geq 0, ab^i c \in L$.
Proof:
Given L is a Fetching Language
Let the FFA $M_F = (Q, q_0, \Sigma, \delta, F)$ solve the language L.
Let n be the number of states in M_F.

$$n = |Q|$$

Let w be a permutation of x in which we can replace all Fetch transitions with a regular transition (where "regular" refers to transitions which operate on the immediate input character). For example, from 1b the permutation of $x = 1222211000$ where $w = 1202020211$. Using this permutation of x, w, and running that through the FFA in 1b is the same as replacing each fetch transition with a normal transition in an NFA.

$$\textbf{Let} \text{ this string } w = abc \in L \text{ and } |x| = |w| \geq n.$$

Let $q_0, q_1, \ldots, q_{|w|}$ be the states in order that M_F visits when processing the word w.
When we process **a** we go from state

$$q_0 \text{ to state } q_{|a|}$$

and when we process **b** we will visit

$$q_{|a|} \to q_{|a|+1} \to \ldots \to q_{|a|+|b|}$$

When we process **c** M_F will go from state

$$q_{|a|+|b|} \text{ to } q_{|w|}$$

Because of the **pigeonhole principle** I have more states visited than I have states in the machine.
Since there are $|w| \geq n$ states we need to visit, there exists an $i, j \in \mathbb{N}$ such that $|a| \leq i < j \leq |ab| \leq n$ where $q_i = q_j$. This means that there is a point in processing the input w where a loop occurs.
This means for every $k \geq 0$, M_F will have the computation path of the following when processing $ab^k c$

$$q_0 \text{ to state } q_{|a|}$$
$$q_{|a|} \to q_{|a|+1} \to \ldots \to q_{|a|+|b|} \text{ which happens k times}$$
$$q_{|a|+|b|} \text{ to } q_{|w|}$$

Because $q_{|w|} \in F$ we can say that $ab^i c \in L \forall i \geq 0$. Therefore our claim holds.

□

1.3 1c **49 / 50**

✓ **- 1 pts** **FFAs are nondeterministic, so there may be multiple computational paths. You want to consider the execution on some accepting path, not "the" path.**

(d) L_{count}

i. Assume for contradiction that L_{count} is Fetching.
Now we can apply the pumping lemma about Fetching languages from question 1c.
PL gives us some $p \in \mathbb{N}$.

ii. Given a string $x \in L_{count}$ where w is the identity permutation of x s.t.
$w = 0^p 1^p \in L_{count}$.

iii. PL tells us that w can be partitioned into a, b, and c s.t.

$$w = abc \in L_{count}$$

and a, b, and c will satisfy these 3 conditions.

A. ab is of the form $0^x 0^y$ because length of ab

$$|ab| = x + y \leq p$$

B. Since b is non empty, $|b| > 0$, b contains at least one 0.

C. Apply condition A to see that

$$0^x 0^{y \cdot i} 1^p \in L_{count} \forall i \geq 0$$

This is a contradiction on the definition of L_{count} for $i > 1$ and $i = 0$.

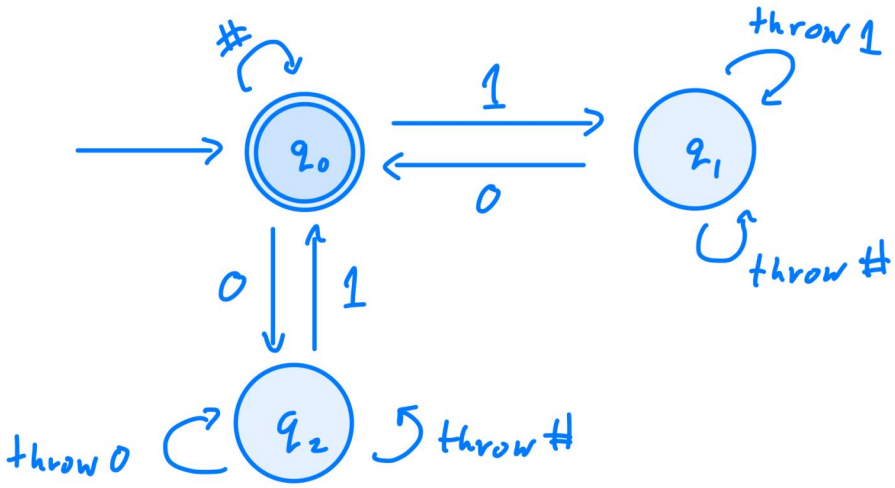
$$\therefore L_{count} \text{ is not Fetching.}$$

1.4 1d **5 / 30**

✓ **- 25 pts** Cannot assume that the permutation w of x is equal to x. You have to either show a contradiction for all permutations w of x or prove that the only valid permutation w of x for this PL is equal to x.

2. **TFA**

(a) L_{EQ}



For example we can try a string $w = 1100 \in l_{EQ}$ which will be turned into 1100#.

$$
\begin{aligned}
1100\#, q_0 \\
100\#, q_1 \\
00\#1, q_0 \\
0\#1, q_0 \\
\#1, q_2 \\
1\#, q_2 \\
\#, q_0 \\
\varepsilon, q_0 \; accepts
\end{aligned}
$$

Another string $w = 0010 \notin l_{EQ}$ will be turned into 0010#.

$$
\begin{aligned}
0010\#, q_0 \\
010\#, q_2 \\
10\#0, q_2 \\
0\#0, q_0 \\
\#0, q_2 \\
0\#, q_2 \\
\vdots
\end{aligned}
$$

goes on to infinity and therefore rejects.
This construction will always take input in 0 and 1 pairs so it will be left with only # at q_0 if the string is in L_{EQ}.

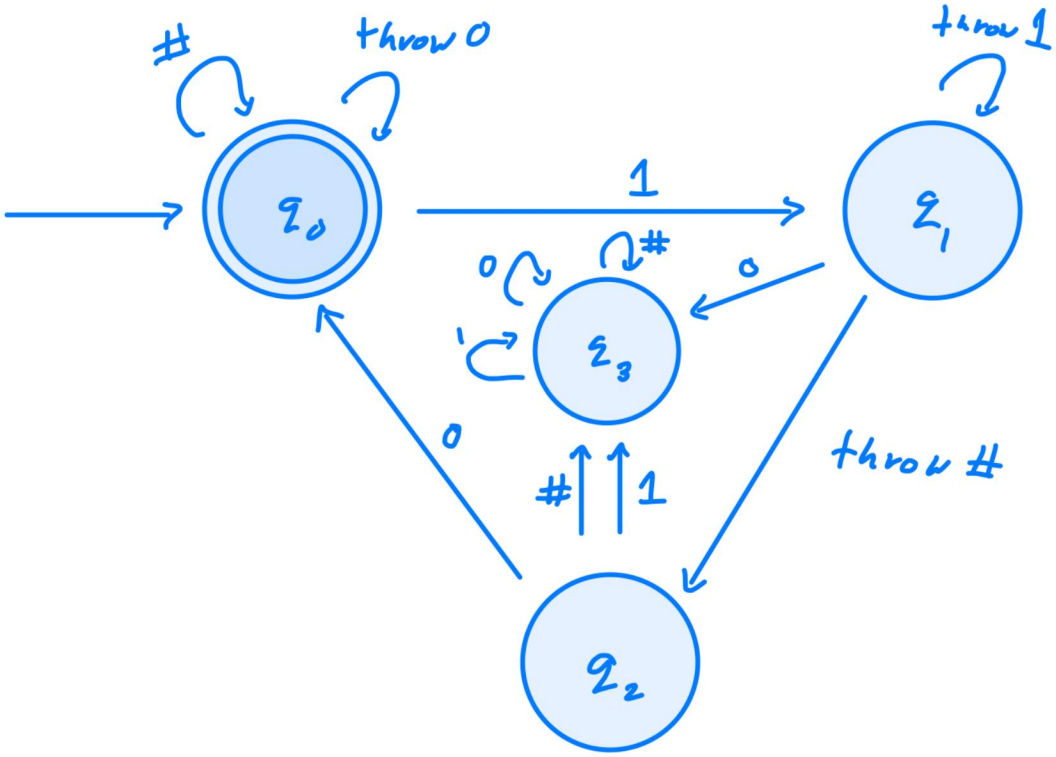
2.1 2a **20 / 20**

✓ **- 0 pts** Correct

(b) **L(T)**

$$L(T) = 1^*223^*$$

(c) L_{count}



This construction will accept only if there are the same number of 0's as there are 1's given that the 0's are consecutive and come before the consecutive 1's. Sanity check strings such as "1" and "0" both fail by ending in the dead state q_3 or with an infinite computation path which means the string does not ever accept. While "01" accepts by going from state 0 to 1 to 2 and accepts in state 0.

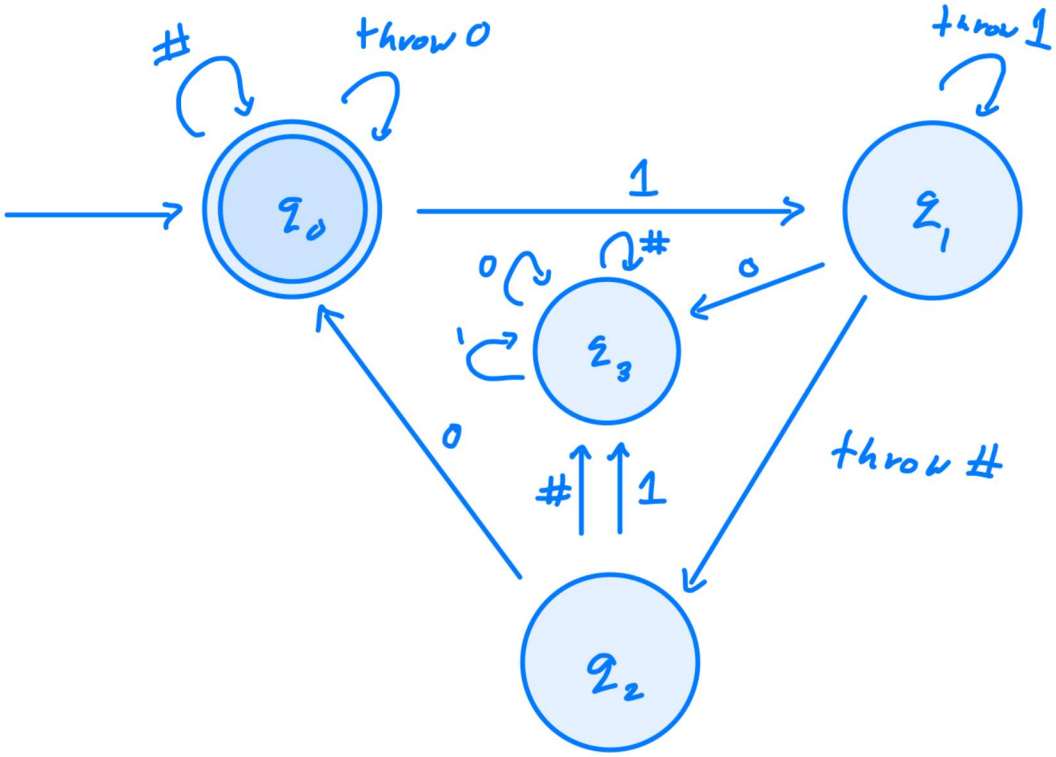
2.2 2b 20 / 20

✓ - 0 pts Correct

(b) **L(T)**

$$L(T) = 1^*223^*$$

(c) L_{count}



This construction will accept only if there are the same number of 0's as there are 1's given that the 0's are consecutive and come before the consecutive 1's. Sanity check strings such as "1" and "0" both fail by ending in the dead state q_3 or with an infinite computation path which means the string does not ever accept. While "01" accepts by going from state 0 to 1 to 2 and accepts in state 0.

2.3 2c 30 / 30

✓ - 0 pts Correct

(d) **Fetch = Thrown**

Claim:
If L is Fetching then L is Thrown.
Proof:
Let L be Fetching so that there exists a FFA $M_f = (Q, q_0, \Sigma, \delta, F)$ which solves L.
If L is also Thrown there must exist a TFA $M_t = (Q', q_0', \Sigma', \delta', F')$ which also solves L.

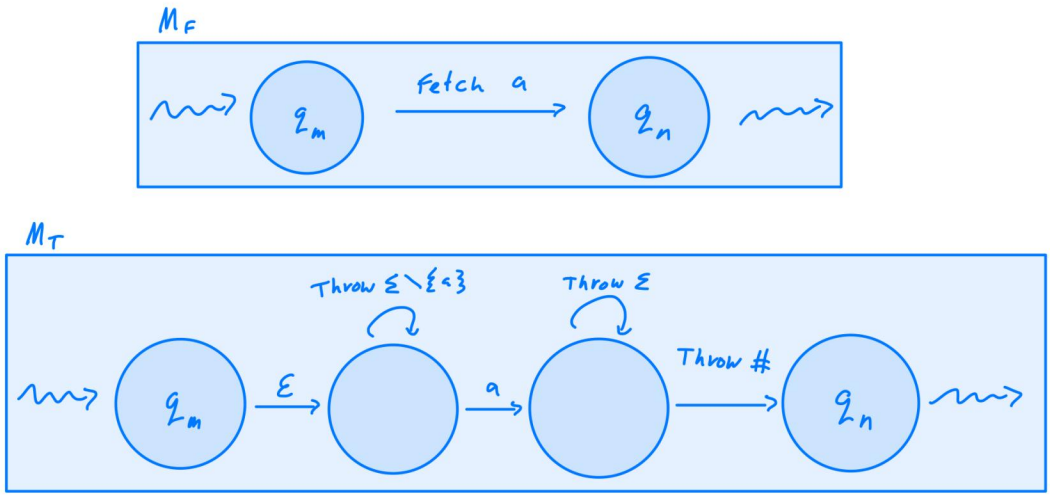
$$Q' = Q \cup \{\text{set of states needed to change each Fetch into a Throw}\} \cup \{q_f\}$$
$$q_0' = q_0$$
$$\Sigma' = \Sigma \cup \{\#\} \cup \{\varepsilon\}$$

For our delta function it is the same for all transitions except those that are defined as any tuple such as $(\{\text{Throw}\}, \sigma \in \Sigma')$.

$$\delta'(q \in Q', \sigma \in \Sigma') = \delta(q \in Q', \sigma \in \Sigma')$$

See the Figure below to see how you implement a Fetch transitions with Throw transitions.

$$\delta'(q \in Q, (\{\text{Throw}\}, \sigma \in \Sigma'))$$



From this figure we can see that for any Fetch transition starting at $q_m \in Q$ we need to define two more states which we use to implement the Fetch transition with Throw transitions. First we Throw on the entire alphabet except on the symbol which we want to Fetch. Then we transition on the symbol which we want to Fetch, and after that we throw on the entire alphabet except #. Finally we throw # and proceed to the next state $q_n \in Q$.
On Fetch transitions we will essentially have to create

$$F' = q_f \text{ where } \delta'(p \in F, \#) = q_f$$

Therefore by proof by construction if L is Fetching then L is Thrown.

□

2.4 2d 50 / 50

✓ **- 0 pts** Correct: transition modification clearly described and explained, accepting states modified.

3 Extra Credit: FPDA 0 / 50

+ **50 pts** Correct

- **10 pts** No explanation

+ **50 pts** Correct, except that your machine doesn't accept the empty string.

+ **45 pts** Partial Credit: Correct, except that your machine does not accept strings with an equal number of 0's and 1's in the first half. Recall that 0 is the tie-breaker for majority.

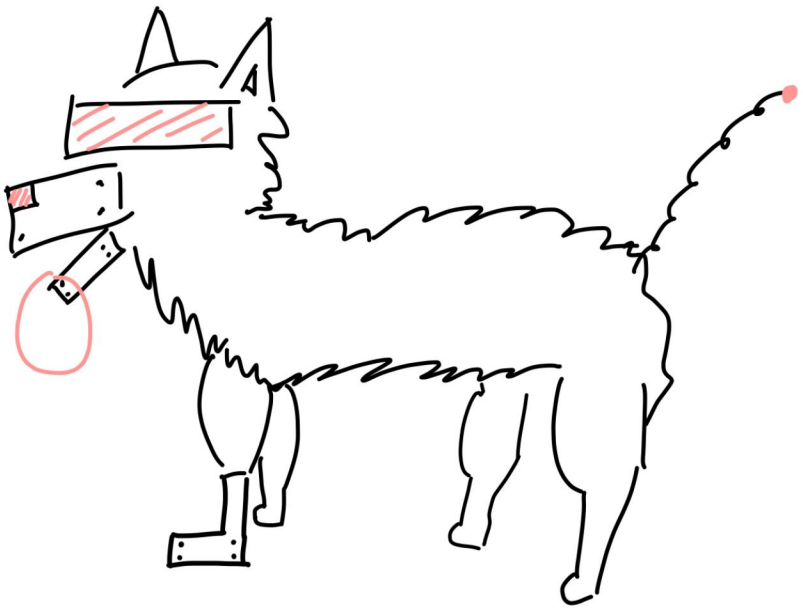
+ **45 pts** Partial Credit: Correct, except that if there are an equal number of 0's and 1's in the first half, your machine can accept if the second half consists of entirely 1's. Recall that 0 is the tie-breaker for majority.

+ **10 pts** Partial credit: concrete construction reflects a high level idea with potential.

+ **10 pts** Partial credit: Correct intuition, but no concrete construction

✓ + **0 pts** **No attempt/I don't know/vague idea only**

+ **0 pts** Incorrect



4 Extra Credit: Automadoggos 2 / 2

✓ **- 0 pts** **Excellent!**

Student Name: William Randall

Student ID: 805167986

CS181 Midterm W21

Please **handwrite** the following honor code agreement and sign and date in the spaces provided.

Honor Code Agreement: I promise and pledge my honor that during the exam period, I did not and will not talk to any person about CS 181 material except for the professor or the TA, nor will I refer to any material except for the class textbook, my own class notes, and material provided by the professor and TAs over the course of the class. I will abide by the CS181 Honor Code. If I have any questions about the honor code, I will ask the professor or the TAs.

I promise and pledge my honor that during the exam period, I did not and will not talk to any person about CS 181 material except for the professor or the TA, nor will I refer to any material except for the class textbook, my own class notes, and materials provided by the professor and TAs over the course of the class. I will abide by the CS181 Honor code. If I have any questions about the honor code, I will ask the professor or the TAs

Signature: William Randall

Date: Feb 11, 2021

5 Honor Code Agreement 0 / 0

✓ - 0 pts Correct