CSE 311: Foundations of Computing I

Homework 8 (due Wednesday, November 26)

You will use (website coming soon!) to submit your answers for this questions 0, 1, 2, and 4 on this homework. And you can check your final result to question 3 using the website. Consequently, we expect your write-ups will be rather short this week.

The extra good news is it is actually possible to verify correctness of your answers this time; so, like with the regular expressions questions, if you get it correct on the website, you are ensured full credit.

0. Doing Fun Automata (25 points)

For each of the following, create a *DFA* that recognizes exactly the language given.

(a) [5 Points] The set of all binary strings that end with 0 and have even length, or start with 1 and have odd length.

(b) [5 Points] The set of all binary strings that have a 1 in every even-numbered position counting from the start of the string with the start of the string counting as position 1.

(c) [5 Points] The set of all binary strings that contain at least two 1's.

(d) [5 Points] The set of all binary strings that contain at most two 0's. Use different state labels from the ones you used in the previous part.

(e) [5 Points] Combine the machines from the previous two parts to produce a machine that recognizes the set of all binary strings that contain at least two 1's *or* at most two 0's.

1. Now For Another (15 points)

For each of the following, create an *NFA* that recognizes exactly the language given.

(a) [5 Points] The set of all binary strings that start with two one's **or** end with two one's.

(b) [10 Points] The set of all binary strings that start with two one's **and** end with two one's.

2. Lock Me Twice, Shame On You (15 points)

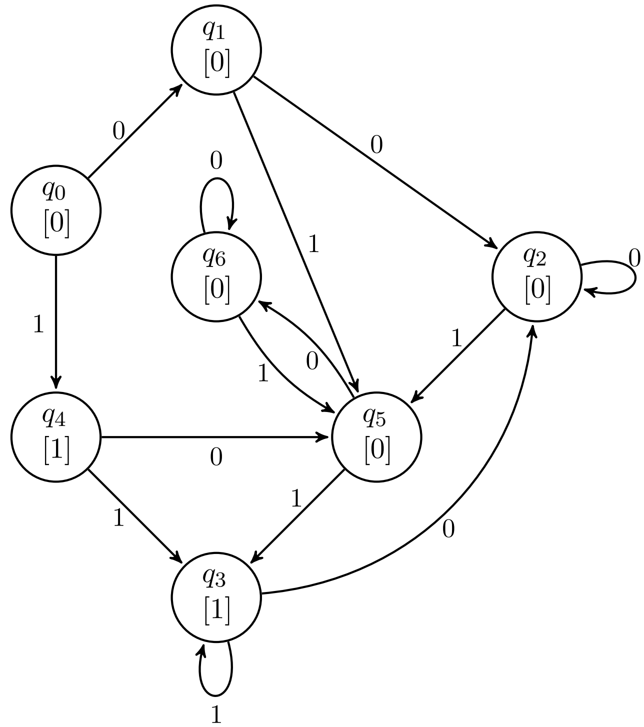
Design a finite state machine for a digital combination lock with the following behavior:

- The inputs are **reset**, **enter**, 1, 2, 3.
- The correct combination is 311.
- The outputs are **unlock** and **error**.
- If 3, then 1, then 1, then **enter** are pressed, in order, with nothing in between or before, then the lock should **unlock**.
- In any other case, when **enter** is pressed, (except after reset) it should **error**.
- If the lock has **error**ed, it should keep producing an **error** state until **reset** is pressed.
- After **reset** is pressed, the lock should act as if nothing were pressed.

3. Minimi (10 points)

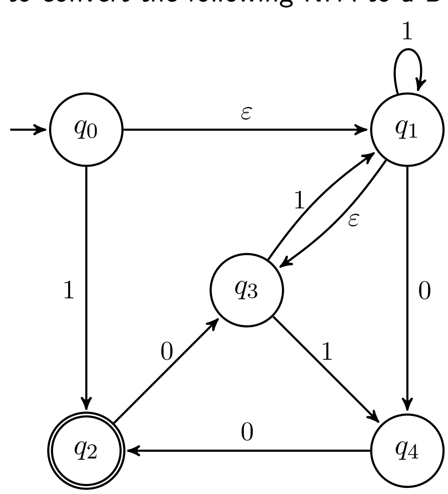
Use the algorithm for minimization that we discussed in class to minimize the following automaton. Be sure to write down every step of the algorithm and circle the groups at every step.

We will provide a way to check your final answer (via a website), but you should also turn the individual steps in on paper.



4. You Don't Have to Sign an NDA (10 points)

Use the construction from lecture to convert the following NFA to a DFA.



5. $a + d + d \neq regular$ (25 points)

(a) [10 Points] Let $\Sigma = \{1, +, =\}$.
Define $L_1 = \{x + y = z \ : \ x, y, z \in \{1\}^*$ and $(x)_1 + (y)_1 = (z)_1\}$. That is, L_1 is all the strings of true statements of the form $x + y = z$ where $+$ and $=$ are characters and x, y, and z are strings of 1's interpreted as unary numbers. For example, "$111 + 11 = 11111$" $\in L_1$, because $3 + 2 = 5$. But, "$111 + 11 = 11$" $\notin L_1$, because $3 + 2 \neq 2$.

Prove that L_1 is not a regular language.

(b) [15 Points] Let $\Sigma = \{0, 1, +, =\}$.
Define $L_2 = \{x + y = z \ : \ x, y, z \in \{0, 1\}^*$ and $(x)_2 + (y)_2 = (z)_2\}$. That is, L_2 is the same idea as L_1, except the numbers are interpreted as binary instead of unary. For example, "$111 + 11 = 1000$" $\in L_2$, because $5 + 3 = 8$. But, "$111 + 11 = 11$" $\notin L_2$, because $7 + 3 \neq 3$.

Prove that L_2 is not a regular language.

6. EXTRA CREDIT: 8 and 9 (-NoValue- points)

The last two extra credits will both be programming questions. As a result, they will both come out in the next few days, and they will be due on the very last day of the quarter. Our hope is that you really enjoy these assignments, and we intend to be significantly more forthcoming with hints than usual to accomplish this goal.

Furthermore, since they are programming questions, we will release auto-graders for both of them some time next week!