**CSE491/596, Fall 2021** **Problem Set 1** **Due Wed. Sept. 22**

**Lectures and Reading**. This week will complete the cycle of showing that the formalisma of DFAs, NFAs, and regular expressions all recognize the same class of languages. The class is standardly denoted by REG for the class of *regular* languages. Then lectures will move into the Myhill-Nerode technique for showing that certain other languages are *non*-regular and for showing that certain sets of strings must all be processed to distinguished states by a DFA. So read Debray's notes up through the end of section 3 on page 11, in tandem with the slides notes called "Extra notes for Weeks 2–3" on the course webpage. Then also read the "Myhill-Nerode" handout on the course webpage (which was also originally used in CSE396).

The assignment will be by *online submission* using CSE Autograder. I will revise this with submission details.

(1) Let positive natural numbers be written in standard binary notation but with the leading '1' removed: $\epsilon$ stands for 1, 0 for 10 = 2, 1 for 11 = 3, 00 for 100 = 4, and so on. This gives a 1-to-1 correspondence between $\{0,1\}^*$ and $\mathbb{N}^+$. Under this correspondence, design a deterministic finite automaton $M_6$ that recognizes the language $L_6$ of strings denoting integers that are 1 less than a multiple of 6:

$$L_6 = \{01, 011, 0001, \dots\} \qquad \text{standing for} \qquad \{5, 11, 17, \dots\}.$$

You may design $M_6$ directly or as a combination of smaller machines, but especially in the former case, you should provide "code comments" explaining the meaning and functions of the states and showing why $L(M_6) = L_6$. (Note typo fix from earlier version that added $\epsilon$ to $L_6$, which you *could* handle with an extra state but it's cleaner not to have it. 24 pts. total)

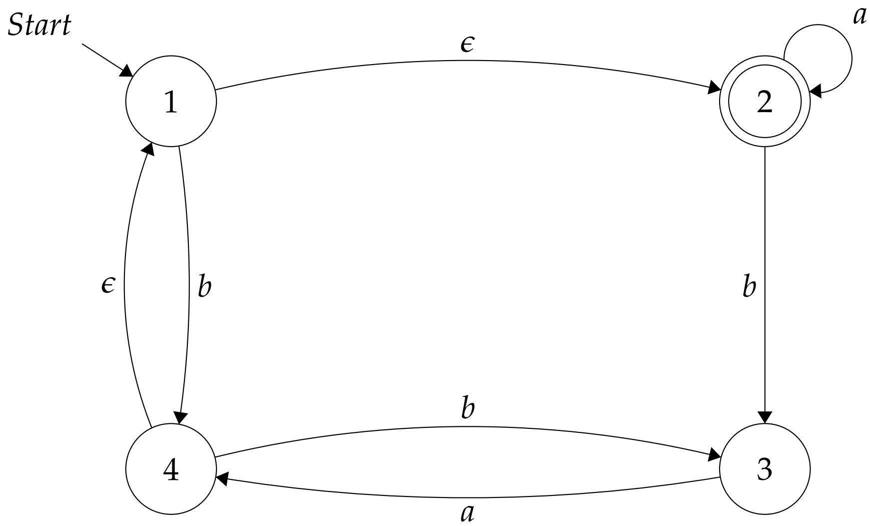
(2) Design both nondeterministic finite automata $N_a, N_b, N_c$ and regular expressions $r_a, r_b, r_c$ that denote the following three languages described in prose. (It is OK for your NFAs to have $\epsilon$-arcs, and it is fine if one or more are DFAs since a DFA "Is-A" NFA.) All use the alphabet $\Sigma = \{0, 1\}$. $(3 \times (6 + 6) = 36$ points total)

(a) $L_a$ = the set of binary strings ending in 011 or 110.

(b) $L_b$ = the set of binary strings that have both a 010 and a 101 in them somewhere.

(c) $L_c$ = the set of binary strings having no occurrence of the substring 11 and at most one occurrence of 00.

(3) Convert the folowing NFA $N$ into an equivalent DFA $M$:




Also answer (for 18 + 12 = 30 pts. giving 90 total on the set):

(a) Is there a string $u$ that $N$ can process from its start state to each and every one of its four states? Viewing your DFA $M$, give a shortest $u$ if so.

(b) Is there a string $v$ that $N$ cannot process from start to any state? Again give a shortest such $v$.

(c) Is there a string $w$ such that no matter what string $y$ follows it, the string $wy$ is accepted? Again use your $M$.

(d) Stronger than (b) and counter to (c), is there a string $z$ that $N$ cannot process at all, not from any of its states $p$ to any state $q$? Again use your $M$ to explain your answer.