Reading:

Thursday’s lecture will cover the NFA-to-DFA theorem and give a way of solving it that is more expeditious than the text’s when ε-arcs are present (IMHO). For next week, please read the rest of section 1.3 on “generalized NFAs” and the conversion to regular expressions. The lecture on Thursday 2/25 may in its second half get into the subject of the Myhill-Nerode theorem, which will substitute for the “Pumping Lemma” in the text’s section 1.4. My suggestion is to read my own handout https://cse.buffalo.edu/~regan/cse396/CSE396MNT.pdf first, hear the lecture, and then compare-contrast it with the examples in the text’s section 1.4, before the following Tuesday’s lecture (Mar. 2) which will lay out the procedure for doing Myhill-Nerode non-regularity proofs.

Homework—part online (TopHat), part written, and all individual work:

(1) Using TopHat, the “Worksheet” titled S21 HW2 Online Part. There are 10 questions, each worth 2 points, for 20 total.

The other two problems are to be submitted as PDFs using the CSE Autograder system. Scans of handwritten sheets are fine provided they are easily legible and do not have excessive file-size. Or you may type your answers.

(2) For each of the following languages $A$, write a regular expression $r$ such that $L(r) = A$, and then give an NFA $N_r$ such that $L(N_r) = A$. Well, if you give a DFA, that counts as an NFA, but in one or two cases you may find the NFA easier to build especially once you have $r$. For part (b), note that a string can be broken uniquely into maximal “blocks” of consecutive letters. For instance, in “Tennessee” the blocks are $T$, $e$, $nn$, $e$ again, $ss$, and $ee$.

(a) The language of strings over $\{a, b\}$ in which every $b$ is followed immediately by at least two $a$’s.

(b) The language of strings over $\{a, b\}$ in which every $a$ belongs to a “block” of at least 2 $a$’s and every $b$ belongs to a block of at least 3 $b$’s.

(c) The language of strings over $\{a, b\}$ with at least 3 characters, such that the last character equals the third-from-last character. (6 + 6 + 12 = 24 pts.)

(3) (a) Again over $\Sigma = \{a, b\}$, design a DFA $M$ such that $L(M)$ equals the language of strings that begin with $bbaa$. Note that if you delete the dead state and the edges involving it, you get what is technically an NFA with only 5 instructions.

(b) Now design an NFA $N$ with only 5 instructions such that $L(N)$ equals the language of strings that end in $aab$. (As in part (a), a single edge or loop labeled with two chars counts as two instructions.)

(c) Then show the conversion of $N$ into an equivalent DFA, following the method in class. Compare the number of instructions and states that you get between the two. (6+6+12 = 24 pts., for 68 total on the set)