Section Handout 4

Problem One: Constructing DFAs

For each of the following alphabets and languages over those alphabets, construct a DFA that recognizes that language.

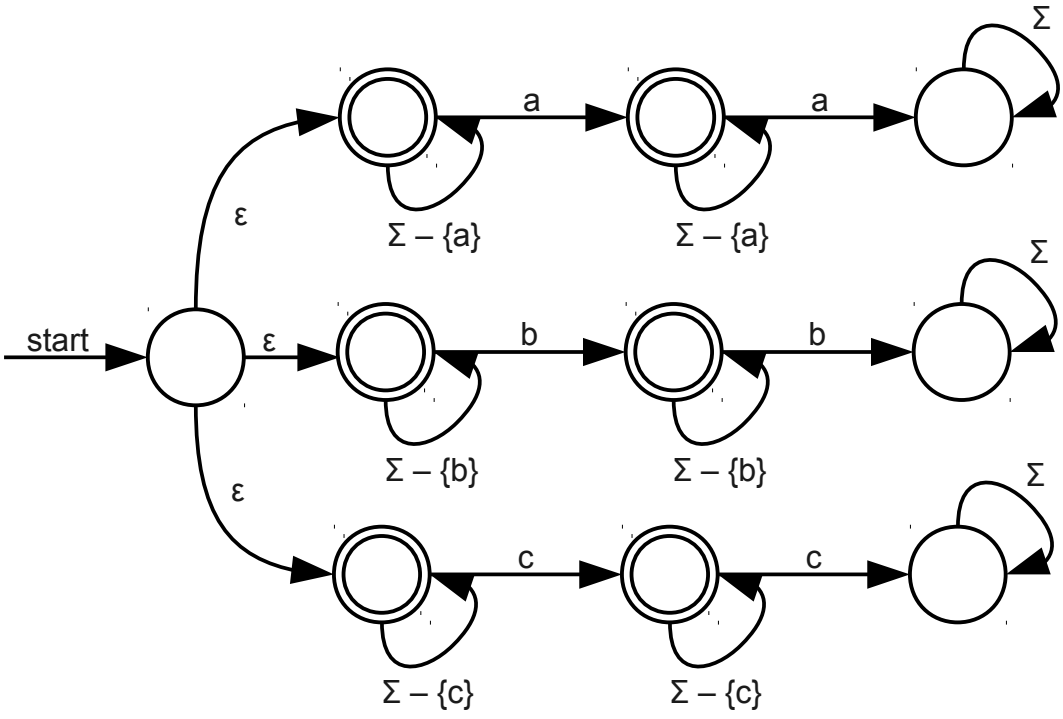
i. For the alphabet $\Sigma = \{0, 1\}$, construct a DFA for the language $\{ w \mid w$ begins with a 0 and ends with a 1 $\}$

ii. For the alphabet $\Sigma = \{0, 1\}$, construct a DFA for the language $\{ w \mid$ the number of 1s in w is a multiple of three $\}$

iii. For the alphabet $\Sigma = \{0, 1, 2, \ldots, 9\}$, construct a DFA for the language $\{ w \mid w$ encodes a base-10 number that is divisible by three, and $|w| \geq 1 \}$

Problem Two: Finding Flaws in NFAs

Consider the alphabet $\Sigma = \{a, b, c\}$ and the language $L = \{ w \mid w$ contains at most one copy of each letter in $\Sigma \}$. Suppose that someone claims that the following NFA accepts the language L:

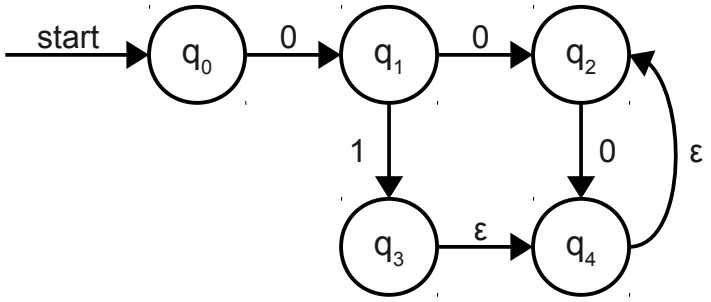


The intuition is as follows: the start state contains ε-transitions to three states, each of which represents not having seen some particular character. Each of those states lead to two other states – one where we have seen some particular character exactly once, and one where we have seen that character exactly twice. The intuition is (or at least, claims to be) that when the NFA first starts processing a string, it guesses which character, if any, is duplicated. It then ε-transitions into the three-state chain for that character. If it reads that character twice, it ends in a rejecting state. Otherwise, it ends in an accepting state.

However, this NFA does **not** correctly accept the language L. Find an example of a string that is incorrectly accepted or incorrectly rejected by the above NFA and explain why. Then, provide a description of the language that this NFA actually does accept.

Problem Three: The Subset Construction

Using the subset construction, convert the following NFA into an equivalent DFA.



Problem Four: Writing Regular Expressions

For each of the following alphabets and languages over those alphabets, write a regular expression that describes the given language.

i. For the alphabet Σ = {0, 1}, write a regular expression for the language
L = { w | w does not contain both 0 and 1. }

ii. For the alphabet Σ = {0, 1, 2}, write a regular expression for the language
L = { w | w contains exactly two 2s. }

iii. For the alphabet Σ = {a, b, c, …, z}, write a regular expression for the language
L = { w | w contains the word "cocoa" as a substring. } To make this more feasible, you may use the shorthand [a-z] to represent (a|b|c|...|z)

iv. For the alphabet Σ = {a, b, c, …, z}, write a regular expression for the language
L = { w | w contains the word "pirate" or the word "spire" as a substring. } To make this more feasible, you may use the shorthand [a-z] to represent (a|b|c|...|z)

v. Suppose that you are playing the game of Nim with two piles, each of which has 2 stones in it at the beginning of the game. Let the alphabet $\Sigma = \{1_a, 2_a, 1_b, 2_b\}$, where 1_a represents the move "remove one stone from pile a," 1_b represents the move "remove one stone from pile b," etc. We can represent a game of Nim as a string of moves, where it is implied that the first symbol corresponds to the first player's first move, the second symbol corresponds to the second player's first move, the third symbol corresponds to the first player's second move, etc. For example, in the game $1_a\ 2_b\ 1_a$, the first player removes one stone from pile A, the second player removes two stones from pile B, and the first player then wins the game by removing one stone from pile A. Write a regular expression for the language L = { w | w is a string describing a legal game of Nim starting with two stones in each pile }.

Problem Five: Infinite Automata

The automata we have seen in lecture have all been finite automata, meaning that their set of states is finite. However, it is also possible to construct *infinite* automata, computing machines with an infinite number of states. Formally, a **deterministic infinite automaton** (or DIA) is a five-tuple $(Q, \Sigma, \delta, q_0, F)$ that follows the definition of a DFA, except that Q is an infinite set of states rather than a finite set of states.

Prove that for any language L, there is a DIA D such that $L(D) = L$. *(Hint: What if you have one state for every possible string?)*