1. Given an arbitrary regular language L on some alphabet Σ, prove that it is closed under the following operations. In other words, prove the following languages are regular.

   (a) $L^R = \{w^R \mid w \in L\}$

   Solution: Since L is regular, we know that a DFA $M = (Q, \Sigma, \delta, s, A)$ recognizes L. We construct an NFA $M^R = (Q^R, \Sigma, s^R, \delta^R, A^R)$ as follows:

$$
\begin{aligned}
Q^R &= Q \uplus \{s^R\} \quad \text{(Here, } \uplus \text{ represents disjoint union.)}\\
\delta^R(s^R, \varepsilon) &= A\\
\delta^R(s^R, a) &= \varnothing \text{ for all } a \in \Sigma\\
\delta^R(q, \varepsilon) &= \varnothing \text{ for all } q \in Q\\
\delta^R(q, a) &= \{q' \in Q \mid \delta(q', a) = q\} \text{ for all } q \in Q, a \in \Sigma\\
A^R &= \{s\}.
\end{aligned}
$$

   M^R effectively reverses the transitions in M. The sentinel start state s^R with outgoing ε-transitions to all accepting states allows the NFA to effectively start at every accepting state in M. (Note that, by definition, a DFA/NFA can only have one starting state.) Because M^R recognizes L^R, L^R is regular. ■

   (b) $\mathsf{subseq}(L) := \{x \in \Sigma^* | x \text{ is a subsequence of some } y \in L\}$.

   Solution: Construct NFA $M_{\mathsf{subseq}} := (\Sigma, Q, s, A, \delta_{\mathsf{subseq}})$, where

   - $\forall q \in Q, c \in \Sigma$, $\delta_{\mathsf{subseq}}(q, c) := \{\delta(q, c), \varepsilon\}$

   To construct a subsequence out of its original sequence, one may use empty ε to replace any amount of symbols in it, which is demonstrated by the idea of adding ε-transitions to all the existed transitions. ■

2. Let

$$\Sigma_3 = \left\{ \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \ldots, \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \right\}.$$

Σ_3 contains all size 3 columns of 0s and 1s. A string of symbols in Σ_3 gives three rows of 0s and 1s. Consider each row to be a binary number and let

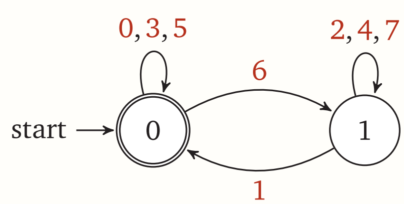
$$B = \{w \in \Sigma_3^* \mid \text{the bottom row of } w \text{ is the sum of the top 2 rows}\}.$$

For example,

$$\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} \in B, \quad \text{but} \quad \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} \notin B.$$

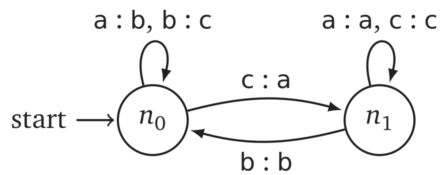
Show that B is regular. (Hint: Working with B^R is easier. Use the result of part (a).)

Solution: One possible solution approach is to simulate long addition, where the carry bits are kept track of via the states in the constructed automaton. Let each symbol in Σ_3 be denoted by their corresponding decimal value as if reading top to bottom were the same as left to right. For example, $\begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$ would be 5. We construct an NFA M, given by the following diagram:



M accepts B^R, which implies that B^R is regular. Because $(B^R)^R = B$, by part (a), B is regular. ■

3. A *finite-state transducer* (FST) is a type of deterministic finite automaton whose output is a string instead of just *accept* or *reject*. The following is the state diagram of finite state transducer FST_0.



Each transition of an FST is labeled at least an input symbol and an output symbol, separated by a colon (:). There can also be multiple input-output pairs for each transitions, separated by a comma (,). For instance, the transition from n_0 to itself can either take a or b as an input, and outputs b or c respectively.

When an FST computes on an input string $s := \overline{s_0 s_1 \ldots s_{n-1}}$ of length n, it takes the input symbols $s_0, s_1, \ldots, s_{n-1}$ one by one, starting from the starting state, and produces corresponding output symbols. For instance, the input string abccba produces the output string bcacbb, while cbaabc produces abbbca.

(a) Assume that FST's have an input alphabet Σ and an output alphabet Γ, give a formal definition of this type of model and its computation. (Hint: An FST is a 5-tuple with no accepting states. Its transition function is of the form $\delta : Q \times \Sigma \to Q \times \Gamma$.)

(b) **Solution:** Formal definition: $\mathsf{FST} := (\Sigma, \Gamma, Q, \delta, s)$, where

- Σ is the input alphabet,
- Γ is the output alphabet,
- Q is the set of all states,
- $s \in Q$ is the start state,
- $\delta : Q \times \Sigma_1 \to Q \times \Gamma_1$ is the transition function. $\forall q_1, q_2 \in Q$, denote the transition between them as $a : b$, where $a \in \Sigma_1, b \in \Gamma_1$, and

$$\delta(q_1, a) = (q_2, b)$$

■

(c) Give a formal description of FST_0.

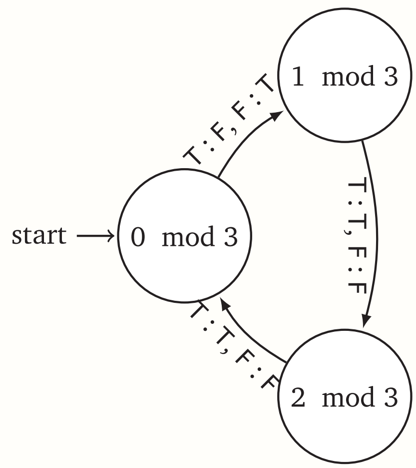
Solution: $\mathsf{FST}_0 := (\Sigma_0, \Gamma_0, Q_0, \delta_0, s_0)$, where

- $\Sigma_0 := \{a, b, c\}$,
- $\Gamma_0 := \{a, b, c\}$,
- $Q_0 := \{n_0, n_1\}$,
- $s_0 := n_0$ is the start state.
- $\delta_0 : Q_0 \times \Sigma_0 \to Q_0 \times \Gamma_0$ is defined as,

$$\begin{aligned} &\delta_0(n_0, a) = (n_0, b), && \delta_0(n_0, b) = (n_0, c), && \delta_0(n_0, c) = (n_1, a), \\ &\delta_0(n_1, a) = (n_1, a), && \delta_0(n_1, b) = (n_0, b), && \delta_0(n_1, c) = (n_1, c). \end{aligned}$$

■

(d) Give a state diagram of an FST with the following behavior. Its input and output alphabets are {T, F}. Its output string is inverted on the positions with indices divisible by 3 and is identical on all the other positions. For instance, on an input TFTTFTFT it should output FFTFFTTT.



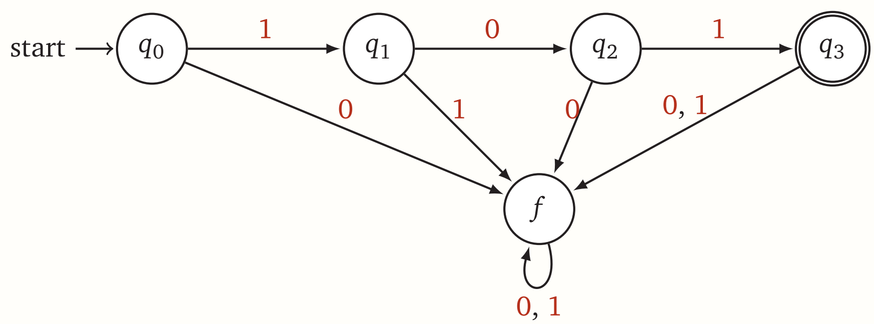
Solution:

■

4. **Another language transformation:** Given an arbitrary regular language L on some alphabet Σ, prove that it is closed under the following operation:

$$\mathsf{cycle}(L) := \{xy | x, y \in \Sigma^*, yx \in L\} \tag{1}$$

Solution: The given language $\mathsf{cycle}(L)$ is a set of strings that can be obtained by spliting a string $w \in L$ into two parts and swapping the order of the parts. As an example, if $L = \{101\}$, then $\mathsf{cycle}(L) = \{101, 011, 110\}$. To get the idea, consider the following DFA $M = (\Sigma, Q, s, A, \delta)$ for the langauge L.



Suppose we start from the state q_2 instead of q_0, traverse through the DFA to reach q_3, take an ϵ-transition to q_0, then continue traversal until reaching back to q_2. This traversal would represent the string 110, which is in $\mathsf{cycle}(L)$. Therefore, if we could start from an arbitrary state $q \in Q$ and traverse the DFA in a similar way as presented above, the traversals would represent the language $\mathsf{cycle}(L)$.

At a high-level, we construct an NFA with $|Q|$ different copies of a pair of M(therefore, it would be the total of $2|Q|$ copies of M). Each pair would correspond to a certain starting state, among all states in Q. For each pair, one copy of M corresponds to pre-cycle, and the other corresponds to post-cycle. We also add a pseudo start state s' that can ϵ-transition to one of the copies. Then, we modify the transition function so it allows the traversal explained above.

Formally, we construct NFA $M' := (\Sigma, Q', s', A', \delta')$, where

- $Q' := (Q \times Q \times \{pre, post\}) \cup \{s'\}$
- $A' := \{(q, q, post) \mid q \in Q\}$
- The transition function δ' is defined as follows,

$$\delta'(s', \epsilon) = \{(q, q, pre) \mid q \in Q\}$$

$$\delta'((q_i, q_j, pre), x) = \begin{cases} (q_i, s, post) & \text{if } q_j \in A, x = \epsilon \\ (q_i, \delta(q_j, x), pre) & otherwise \end{cases}$$

$$\delta'((q_i, q_j, post), x) = (q_i, \delta(q_j, x), post)$$

A state $q' = (q_i, q_j, pre)$, for an example, represents that the traversal started from q_i, so far the input string led to q_j, and we haven't cycled yet. Once we reach one of the original accepting states within a pre-cycle copy, we can take an ϵ-transition to the original starting state s of the corresponding post-cycle copy, and then continue traversal. We accept when we reach the state from which we started the traversal within the post-cycle copy. ■