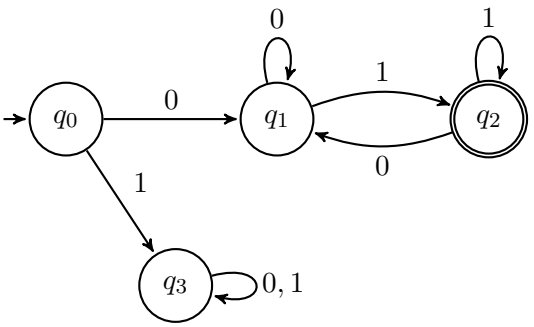
Suggested Solutions to Midterm Problems

1. Give the state diagrams of DFAs, with as few states as possible, recognizing the following languages.

   (a) $\{w \in \{0,1\}^* \mid w \text{ begins with a 0 and ends with a 1}\}$.

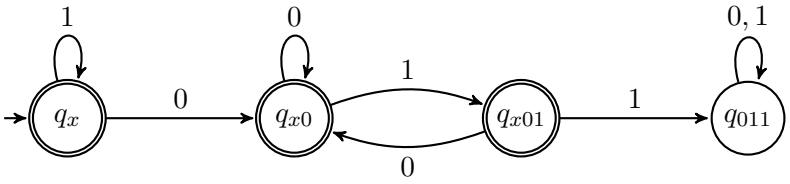
   Solution.



□

   (b) $\{w \in \{0,1\}^* \mid w \text{ doesn't contain the substring 011}\}$.

   Solution.

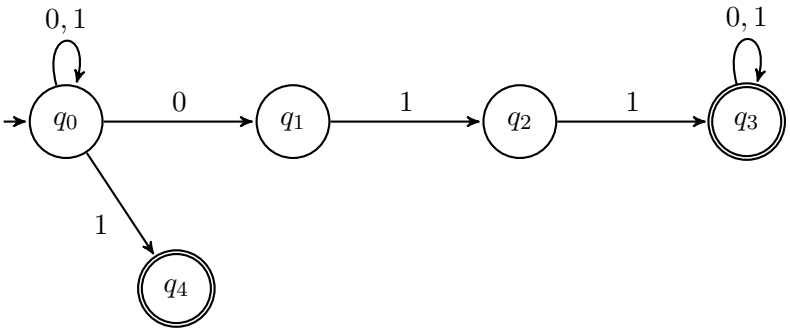


□

2. Let $L = \{w \in \{0,1\}^* \mid w \text{ contains 011 as a substring or ends with a 1}\}$.

   (a) Draw the state diagram of an NFA, with as few states as possible, that recognizes L. The fewer states your NFA has, the more points you will be credited for this problem.

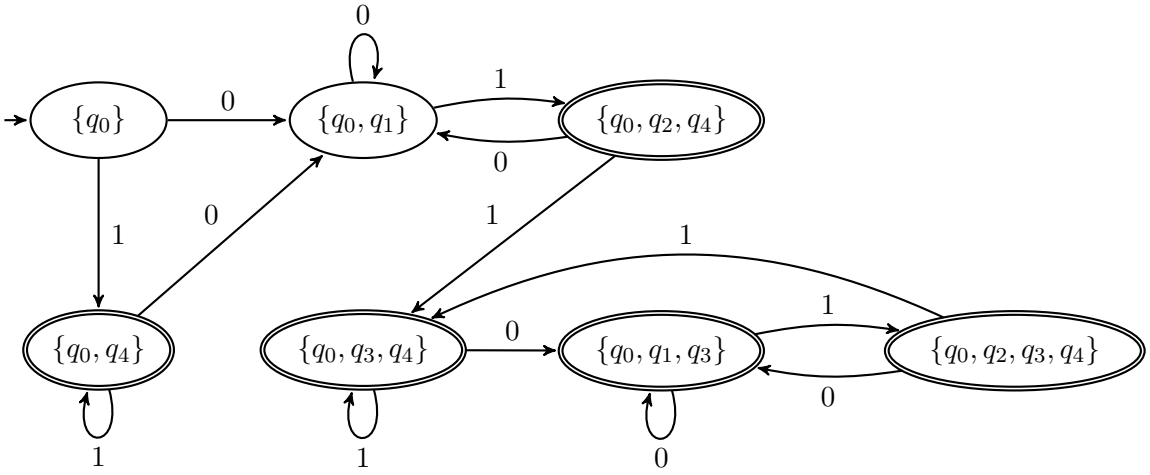
   Solution.



□

(b) Convert the preceding NFA systematically into an equivalent DFA (using the procedure discussed in class). Do not attempt to optimize the number of states, though you may omit the unreachable states.

Solution.



□

3. Let $L = \{1^p \mid p \text{ is a prime number less than } 2^{2^{10}}\}$. Is L a regular language? Why or why not?

   Solution. Any finite set of strings is a regular language, as we can easily construct an NFA for each of the strings and take the union of all such NFAs to obtain the final NFA that recognizes the language. L here is finite and hence regular. □

4. For languages A and B, let the *shuffle* of A and B be the language $\{w \mid w = a_1 b_1 \cdots a_k b_k$, where $a_1 \cdots a_k \in A$ and $b_1 \cdots b_k \in B$, each $a_i, b_i \in \Sigma^*\}$. Show that the class of regular languages is closed under shuffle.

   Solution.

   Let $M_A = (Q_A, \Sigma, \delta_A, q_A, F_A)$ and $M_B = (Q_B, \Sigma, \delta_B, q_B, F_B)$ be two DFAs that recognize A and B, respectively. An NFA $M = (Q, \Sigma, \delta, q_0, F)$ that, in each step, simulates either a step of M_A or M_B will recognize the shuffle of A and B. Formally, it is defined as follows:

   - $Q = Q_A \times Q_B$,
   - $\delta((x, y), a) = \{(\delta_A(x, a), y), (x, \delta_B(y, a))\}$ for every $a \in \Sigma, x \in Q_A, y \in Q_B$,
   - $q_0 = (q_A, q_B)$,
   - $F = F_A \times F_B$.

   □

5. A *synchronizing sequence* for a DFA $M = (Q, \Sigma, \delta, q_0, F)$ and some "home" state $h \in Q$ is a string $s \in \Sigma^*$ such that, for every $q \in Q$, $\delta(q, s) = h$. A DFA is said to be *synchronizable* if it has a synchronizing sequence for some state. Prove that, if M is a k-state synchronizable DFA, then it has a synchronizing sequence of length at most k^3. (Note: $\delta(q, s)$ equals the state where M ends up when M starts from state q and reads input s.)

   Solution. Suppose a k-state DFA M is synchronizable. Starting simultaneously from any two different states of M, there must exist some input string that can bring the two states

to the same state; otherwise, we have a contradiction (M would not be synchronizable). How long does that string need to be? There are at most $k(k-1)$ pairs of different states, so the shortest possible input string needs to be at most $k(k-1)$ symbols long.

Now starting simultaneously from the k states of M, an appropriate string of length at most $k(k-1)$ will reduce the number of different states to $k-1$. Repeat this (with possibly different input strings in different stages) $k-2$ more times and the number of states will eventually reduce to one. The concatenation of the input strings from the $k-1$ stages is a synchronizing sequence of length at most $k(k-1) \times (k-1) < k^3$. □

6. Consider the following CFG discussed in class, where for convenience the variables have been renamed with single letters.

$$\begin{array}{lcl} E & \to & E+T \mid T \\ T & \to & T \times F \mid F \\ F & \to & (E) \mid a \end{array}$$

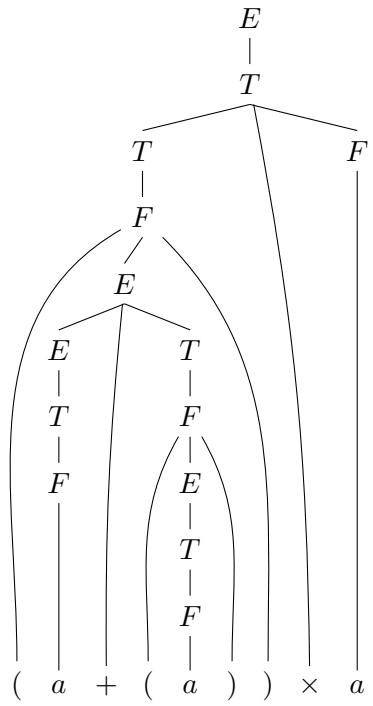
Give the (leftmost) derivation and parse tree for the string $(a+(a)) \times a$.

Solution.

The leftmost derivation

$$\begin{array}{rcl} E & \Rightarrow & T \\ & \Rightarrow & T \times F \\ & \Rightarrow & F \times F \\ & \Rightarrow & (E) \times F \\ & \Rightarrow & (E+T) \times F \\ & \Rightarrow & (T+T) \times F \\ & \Rightarrow & (F+T) \times F \\ & \Rightarrow & (a+T) \times F \\ & \Rightarrow & (a+F) \times F \\ & \Rightarrow & (a+(E)) \times F \\ & \Rightarrow & (a+(T)) \times F \\ & \Rightarrow & (a+(F)) \times F \\ & \Rightarrow & (a+(a)) \times F \\ & \Rightarrow & (a+(a)) \times a \end{array}$$

The parse tree

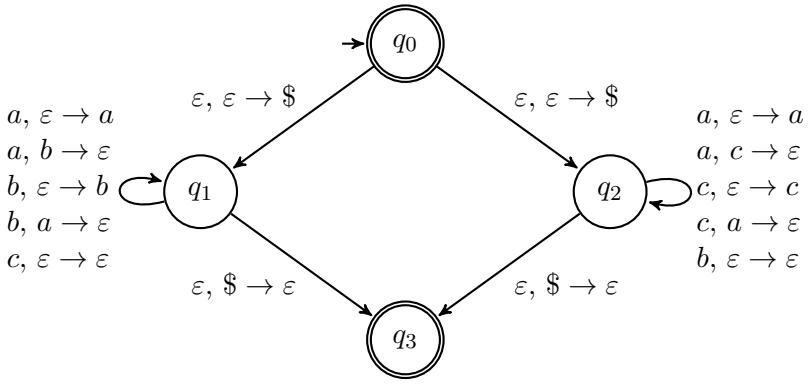


□

7. Draw the state diagram of a pushdown automaton (PDA) that recognizes the following language: $\{w \in \{a,b,c\}^* \mid$ the number of a's in w equals that of b's or c's$\}$ (no restriction is imposed on the order in which the symbols may appear). Please make the PDA as simple as possible and explain the intuition behind the PDA.

Solution. A PDA that recognizes the language is shown below. In the intial state, the PDA nondeterministically chooses to check whether the number of a's equals to that of

b's (q_1) or c's (q_2). It accepts the input if one of the two checks passes. Take state q_1 for example. State q_1 reacts only to characters a and b. As the input symbols come in no specific order, the number of a's may exceed that of b's at any point and vice versa. In the first case, it pushes an a onto the stack if the next symbol is an a and pops an a out of the stack if the next symbol is a b; analogously in the second case.



□

8. For two given languages A and B, define $A \diamond B = \{xy \mid x \in A \text{ and } y \in B \text{ and } |x| = |y|\}$. Prove that, if A and B are regular, then $A \diamond B$ is context-free. (Hint: construct a PDA where the stack is used to ensure that x and y are of equal length.)

   Solution. Given NFAs N_A and N_B respectively for A and B, the basic idea of constructing a PDA for recognizing $A \diamond B$ is to first simulate N_A and then nondeterministically switch to simulate N_B. To ensure that $|x| = |y|$, the PDA counts the number of symbols while simulating N_A by pushing a marker onto the stack whenever it reads an input symbol and it later cancels out the markers with the input symbols while simulating N_B.

   Suppose $N_A = (Q_A, \Sigma, \delta_A, q_A, F_A)$ and $N_B = (Q_B, \Sigma, \delta_B, q_B, F_B)$, assuming A and B have the same alphabet. The PDA $M = (Q, \Sigma, \Gamma, \delta, q_{\text{start}}, \{q_{\text{accept}}\})$ for $A \diamond B$ is formally defined as follows:

   - $Q = \{q_{\text{start}}, q_{\text{accept}}\} \cup Q_A \cup Q_B$, where $q_{\text{start}}, q_{\text{accept}} \notin Q_A \cup Q_B$.
   - $\Gamma = \{x, \$\}$.
   - δ is defined as follows.

$$\begin{cases} \delta(q_{\text{start}}, \varepsilon, \varepsilon) = \{(q_A, \$)\} & \\ \delta(q, a, \varepsilon) = \{(q', x) \mid q' \in \delta_A(q, a)\} & q \in Q_A \text{ and } a \neq \varepsilon \\ \delta(q, \varepsilon, \varepsilon) = \{(q', \varepsilon) \mid q' \in \delta_A(q, \varepsilon)\} & q \in Q_A \\ \delta(q, \varepsilon, \varepsilon) = \{(q_B, \varepsilon)\} & q \in F_A \\ \delta(q, a, x) = \{(q', \varepsilon) \mid q' \in \delta_B(q, a)\} & q \in Q_B \text{ and } a \neq \varepsilon \\ \delta(q, \varepsilon, \varepsilon) = \{(q', \varepsilon) \mid q' \in \delta_B(q, \varepsilon)\} & q \in Q_B \\ \delta(q, \varepsilon, \$) = \{(q_{\text{accept}}, \varepsilon)\} & q \in F_B \\ \delta(q, a, t) = \emptyset & \text{otherwise} \end{cases}$$

   It should be clear that $L(M) = A \diamond B$; we omit the detailed proof. □

9. Prove that the class of context-free languages is not closed under *complement*.

   Solution. Recall that the class of context-free languages is not closed under intersection. Let $A = \{a^n b^n c^m \mid n, m \geq 0\}$ and $B = \{a^m b^n c^n \mid n, m \geq 0\}$, which are context free. $A \cap B = \{a^n b^n c^n \mid n \geq 0\}$ is not context free.

Intersection may be expressed in terms of complement and union: $A_1 \cap A_2 = \overline{\overline{A_1} \cup \overline{A_2}}$. We know that the class of context-free languages is closed under the union operation. If the class of context-free languages were closed under the complement operation, then it would be closed under intersection, contradicting the preceding result. □

10. Prove, using the pumping lemma, that $\{a^m b^n c^{m \times n} \mid m, n \geq 1\}$ is not context free.

    Solution. Assume toward contradiction that p is the pumping length for $\{a^m b^n c^{m \times n} \mid m, n \geq 1\}$, referred to as language A below. Consider a string $s = a^p b^p c^{p^2}$ in the language. The string s may be divided as $uvxyz$ such that $|vy| \geq 0$ and $|vxy| \leq p$ in several different ways. We argue below, for each division case, $uv^i xy^i z \notin A$ for some $i \geq 0$ and conclude that s cannot be pumped, leading to a contradiction.

    - Case 1: v and y contain only a's. In this case, when i either goes up or down, $uv^i xy^i z$ will have a mismatch between the number of c's (which remains p^2) and the product of the number of a's (which is less or more than p) and that of b's (which remains p).
    - Case 2: v and y contain only b's. This is similar to Case 1.
    - Case 3: v and y contain only c's. In this case, when i either goes up or down, $uv^i xy^i z$ will have more or less than p^2 occurrences of c's, while the product of the number of a's and that of b's remains p^2.
    - Other cases: either v contains some a's and some b's or y contains some b's and some c's. In these cases, when i goes up, $uv^i xy^i z$ will not even be in the form of $a^* b^* c^*$.

    □

Appendix

- (Pumping Lemma for Context-Free Languages)

  If A is a context-free language, then there is a number p such that, if s is a string in A and $|s| \geq p$, then s may be divided into five pieces, $s = uvxyz$, satisfying the conditions:

  1. for each $i \geq 0$, $uv^i xy^i z \in A$,
  2. $|vy| > 0$, and
  3. $|vxy| \leq p$.