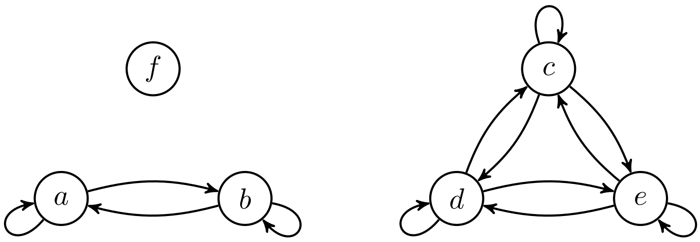
Suggested Solutions to Midterm Problems

1. Let $A = \{a, b, c, d, e, f\}$ and $R = \{(a, b), (d, c), (d, e)\}$, which is a binary relation on A.

   (a) Give a symmetric and transitive but *not* reflexive binary relation on A that includes R. Please present the relation using a directed graph.

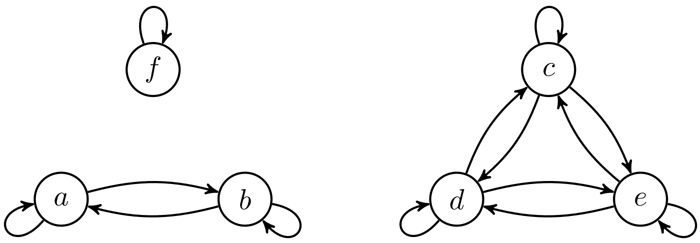
   Solution. (Hung-Wei Hsu)



$\square$

   (b) Find the smallest equivalence relation on A that includes R. Please present the relation using a directed graph.

   Solution. (Hung-Wei Hsu)



$\square$

2. Let L be a language over Σ (i.e., $L \subseteq \Sigma^*$). Two strings x and y in Σ^* are *distinguishable by L* if for some string z in Σ^*, exactly one of xz and yz is in L. When no such z exists, i.e., for every z in Σ^*, either both of xz and yz or neither of them are in L, we say that x and y are *indistinguishable by L*. Is indistinguishability by a language an equivalence relation (over Σ^*)? Please justify your answer.

   Solution.

   Let us refer to the "indistinguishability by a language L" relation as R_L. R_L is an equivalence relation, as it satisfies the following three conditions:

   - Reflexivity (xR_Lx): For every w in Σ^*, xw and xw are identical and either both or neither of them are in L. Hence, xR_Lx.
   - Symmetry (xR_Ly if and only if yR_Lx): If xR_Ly, i.e., for every w in Σ^*, either both of xw and yw or neither of them are in L, then for every w in Σ^*, both of yw and xw or neither of them are in L and hence yR_Lx.

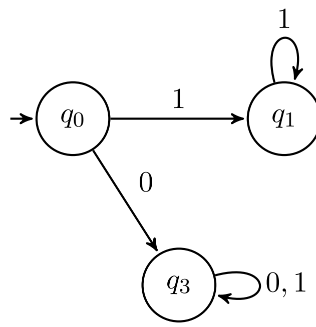
- Transitivity (xR_Ly and yR_Lz implies xR_Lz): Suppose xR_Ly and yR_Lz, i.e., for every w in Σ^*, (a) either both of xw and yw or neither of them are in L and (b) either both of yw and zw or neither of them are in L. If both of xw and yw are in L, then both of yw and zw are also in L and hence both of xw and zw are in L. If neither of xw and yw are in L, then neither of yw and zw are in L and hence neither of xw and zw are in L. So, for every w in Σ^*, either both of xw and zw or neither of them are in L and hence xR_Lz.

□

3. (20 points) Give the state diagrams of DFAs recognizing the following languages. In all parts, the alphabet is $\{0, 1\}$.

   (a) $\{w \mid w$ begins with a 1 and ends with a 0$\}$.

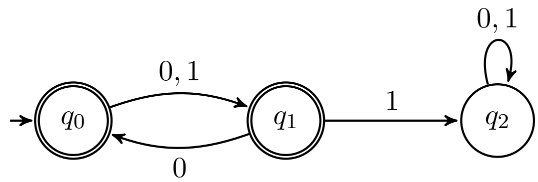
   Solution. (Hung-Wei Hsu)



□

   (b) $\{w \mid$ every even position of w is a 0$\}$ (Note: see w as $w_1w_2\cdots w_n$, where $w_i \in \{0, 1\}$).

   Solution. (Hung-Wei Hsu)

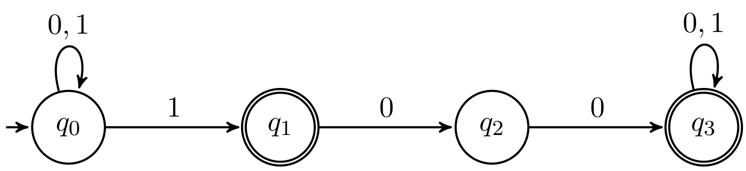


□

4. Let $L = \{w \in \{0, 1\}^* \mid w$ contains 100 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.

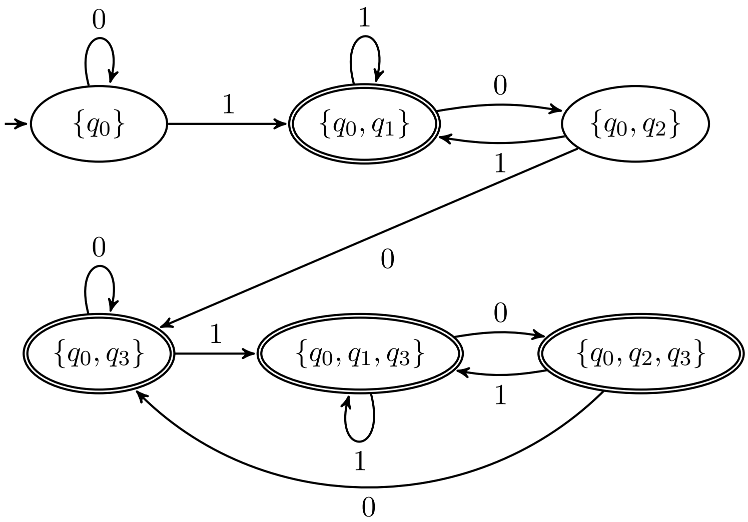
   Solution. (Hung-Wei Hsu)



□

(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. (Hung-Wei Hsu)



□

5. 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. (Hung-Wei Hsu)

   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. We can define an NFA $M = (Q, \Sigma, \delta, q_0, F)$ that recognizes the shuffle of A and B as follows:

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

□

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

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

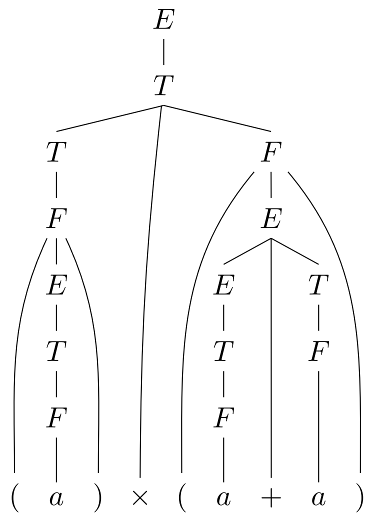
   Give the (leftmost) derivation and parse tree for the string $(a) \times (a + a)$.

   Solution. (Hung-Wei Hsu)

The leftmost derivation.

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

The parse tree.

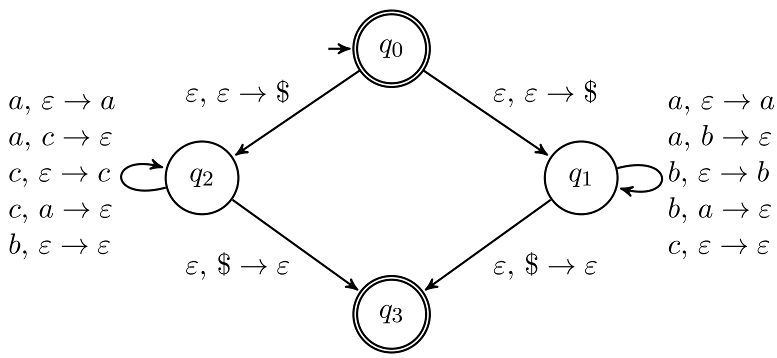


□

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 input symbols may appear). Please make the PDA as simple as possible and explain the intuition behind the PDA.

   Solution. (Hung-Wei Hsu)

   We construct a PDA that recognizes the language as shown below. 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. Prove *by induction* that, if G is a CFG in Chomsky normal form, then for any string $w \in L(G)$ of length $n \geq 1$, exactly $2n - 1$ steps are required for any derivation of w.

   Solution. The proposition still holds even if we include all other strings not in $L(G)$ that can be derived from non-start symbols. We will prove this stronger variant by induction on n, the length of an arbitrary nonempty string w. The strengthening in fact will make the inductive proof easier, as we will have a stronger induction hypothesis for the inductive step.

Base case ($|w| = 1$): The only way to produce a string of length 1 is by applying at the beginning a rule of the form $A \to a$, which constitutes a one-step derivation.

Inductive step ($|w| = n > 1$): To produce a string of length larger than one, one must first apply a rule of the form $A \to BC$, where B and C are non-start symbols. Suppose the B part eventually produces a string x of length l and the C part a string y of length m such that $xy = w$ and $l + m = n$. From the induction hypothesis, these two parts of derivation take $2l - 1$ and $2m - 1$ steps, respectively. So, the derivation of a string of length n requires $1 + (2l - 1) + (2m - 1) = 2(l + m) - 1 = 2n - 1$ steps. □

9. Prove, using the pumping lemma, that $\{x\#wxy \mid w, x, y \in \{a, b\}^*\}$ is not context-free.

   Solution. We take s to be $a^pb^p\#a^pb^p$, where p is the pumping length, and show that s cannot be pumped. There are basically three ways to divide s into $uvxyz$ such that $|vy| > 0$ and $|vxy| \leq p$:

   Case 1: vxy falls within the first occurrence of a^pb^p (before $\#$). No matter how we divide s, when we pump *up*, the substring before $\#$ will become longer than the one after $\#$ and the whole string cannot belong to the language.

   Case 2: vxy falls within the substring $b^p\#a^p$. Neither v nor y may contain $\#$, otherwise we will get more than one $\#$'s when we pump up the string. So, s must be divided as $uvxyz = (a^pb^{p-j-k})(b^j)(b^k\#a^l)(a^m)(a^{p-l-m}b^p)$, where $j, k, l, m \geq 0$ and j and m can not both be 0. If $j > 0$, we pump *up* to get $uv^2xy^2z = (a^pb^{p-j-k})(b^{2j})(b^k\#a^l)(a^{2m})(a^{p-l-m}b^p)$. The substring before $\#$ will have more b's than the one after $\#$ and hence the whole string cannot belong to the language. If $m > 0$, we pump *down* to get $uv^0xy^0z = (a^pb^{p-j-k})(\varepsilon)(b^k\#a^l)(\varepsilon)(a^{p-l-m}b^p)$. The substring before $\#$ will have more a's than the one after $\#$ and hence the whole string cannot belong to the language.

   Case 3: vxy falls within the second occurrence of a^pb^p (after $\#$). No matter how we divide s, when we pump *down*, the substring after $\#$ will become shorter than the one before $\#$ and the whole string cannot belong to the language.

   □

Appendix

- Common properties of a binary relation R on A:
  - R is *reflexive* if for every $x \in A$, xRx.
  - R is *symmetric* if for every $x, y \in A$, xRy if and only if yRx.
  - R is *transitive* if for every $x, y, z \in A$, xRy and yRz implies xRz.

- A context-free grammar is in **Chomsky normal form** if every rule is of the form

  $$\begin{array}{lcl} A & \to & BC \quad \text{or} \\ A & \to & a \end{array}$$

  where a is any terminal and A, B, and C are any variables—except that B and C may not be the start variable. In addition,

  $$S \to \varepsilon$$

  is permitted if S is the start variable.

- (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 $i \geq 0$, $uv^ixy^iz \in A$, (2) $|vy| > 0$, and (3) $|vxy| \leq p$.