CS 172 — Spring 2000
Computability and Complexity
Prelim #1 Solutions

Problem 1. (90 points) Let $M_1 = (Q_1, \Sigma, \delta_1, s_1, F_1)$ and $M_2 = (Q_2, \Sigma, \delta_2, s_2, F_2)$ be two finite automata over a common alphabet.

a. Construct a finite automaton $M = (Q, \Sigma, \delta, q_0, F)$ such that $L(M)$ contains precisely the words in $L(M_1)$ which have a suffix in $L(M_2)$.

b. Apply your construction from part (a) to the following two automata.

c. Is the result from part (b) deterministic? If not, use the subset construction to determinize it.

d. Is the result from part (c) minimal? If not, use the minimization algorithm to minimize it.

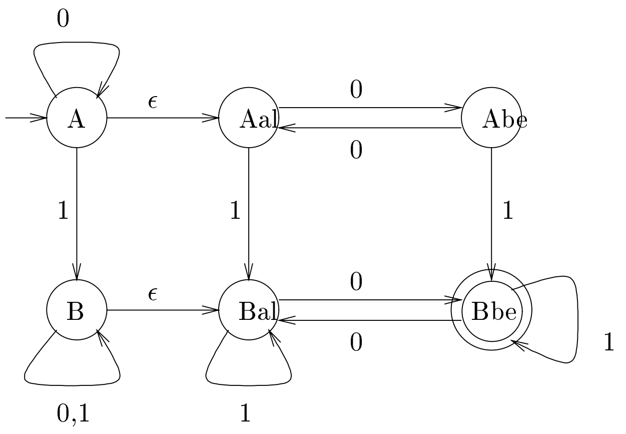
e. Let A be the language of the automata from parts (b)–(c). What is the index of A? Describe each $\equiv_A$-equivalence class by a regular expression.

You can double check your result by thinking about which languages are accepted by the automata of part (b), and which language results from applying the operation of part (a) to these languages.
Solution 1.

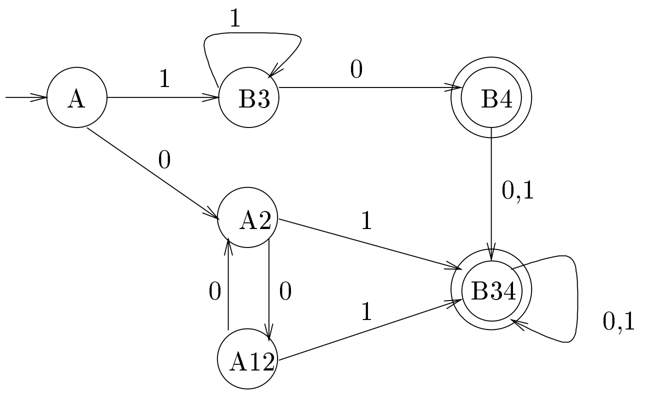
a We construct a nondeterministic machine that has two parts: a copy of machine M_1, and a copy of the product machine $M_1 \times M_2$. It starts out by simulating the machine M_1, and at each state, guesses if the "correct" suffix starts there (and switches to simulating the product machine). The machine accepts if finally both machines in the product accepts. The formal construction of M is:

$$\begin{array}{lcl} Q & = & Q_1 \uplus Q_1 \times Q_2 \\ \delta(q,a) & = & \{\delta_1(q,a)\} \\ \delta(q,\epsilon) & = & \{(q,s_2)\} \\ \delta((q,q'),a) & = & \{(\delta_1(q,a),\delta_2(q',a))\} \\ q_0 & = & s_1 \\ F & = & \{(q_1,q_2) \mid q_1 \in F_1 \text{ and } q_2 \in F_2\} \end{array}$$

b When we do the construction in part (a), we get



c On determinizing the automaton from part (b) we get:

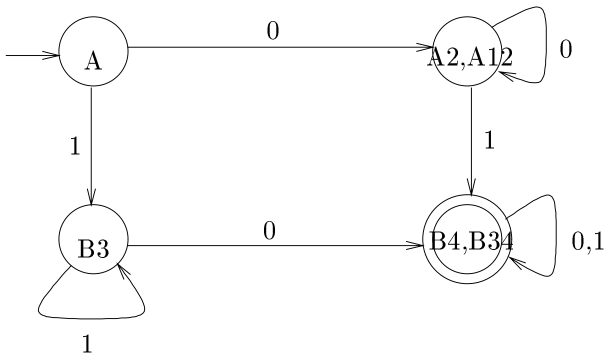


where we write 1 for the state $A\alpha$, 2 for the state $A\beta$, 3 for the state $B\alpha$ and 4 for the state $B\beta$.

d Run the minimization algorithm on the machine from part (c).

	A	A2	B3	A12	B4	B34
A						
A2	1					
B3	1	1				
A12	1		1			
B4	0	0	0	0		
B34	0	0	0	0		

The states $A2$ and $A12$ are equivalent, so are $B4$ and $B34$. The minimal automaton accepting the same language is



e The index of the language is 4 (size of the minimal automaton). The $\equiv_A$ classes are:

$$\begin{array}{lcl} \equiv_A & : & \epsilon \\ \equiv_{A2,A12} & : & \mathtt{00}^* \\ \equiv_{B3} & : & \mathtt{11}^* \\ \equiv_{B4,B34} & : & (\mathtt{11}^*\mathtt{0} \cup \mathtt{00}^*\mathtt{1})(\mathtt{0} \cup \mathtt{1})^* \end{array}$$

Problem 2. (60 points) For the two languages B_1 and B_2 defined below do the following.

If B_i is regular: Give a finite automaton that recognizes B_i.

If B_i is not regular: Give both a proof that B_i is not regular (use the pumping lemma or the fact that $\{\mathtt{0}^n\mathtt{1}^n : n \geq 0\}$ is not regular) and a pushdown automaton that recognizes B_i.

a. B_1 is the set of all words in $\{\mathtt{0},\mathtt{1}\}^*$ that contain an equal number of occurrences of `00` and `11`. (Count overlapping occurrences; for example, `110001111` contains 2 occurrences of `00` and 4 occurrences of `11`.)

b. B_2 is the set of all words in $\{\mathtt{0},\mathtt{1}\}^*$ that contain an equal number of occurrences of `01` and `10`.

Before you begin to answer the questions, think about which words are in B_1 and B_2, and which are not, by listing a few examples.
Solution 2.

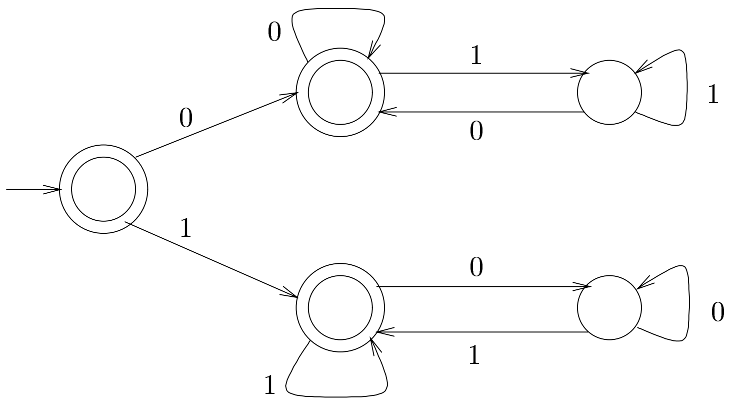
a B_1 is not regular. Consider the words 0^n1^n. These are in B_1 (there are $(n-1)$ occurrences each of 00 and 11). Suppose we take the language $L = B_1 \cap \mathtt{0}^*\mathtt{1}^*$. Then $L = \{\mathtt{0}^n\mathtt{1}^n \mid n \geq 0\} \cup \{\mathtt{0},\mathtt{1}\}$. We are almost there! We know that the language $\{\mathtt{0},\mathtt{1}\}$ is regular, hence so is the language $L' = \{\mathtt{0},\mathtt{1}\}^* \setminus \{\mathtt{0},\mathtt{1}\}$ (closure under complementation). Thus, we take $L'' = L \cap L'$. Equivalently, $L'' = B_1 \cap (\mathtt{0}^*\mathtt{1}^* \cap (\{\mathtt{0},\mathtt{1}\}^* \setminus \{\mathtt{0},\mathtt{1}\}))$. But L'' is the language $\{\mathtt{0}^n\mathtt{1}^n \mid n \geq 0\}$. If B_1 were regular, then so would be L'' because L'' is the intersection of B_1 with a regular language. Hence, B_1 must be nonregular.

However, B_1 is context free. We define a pushdown automaton that accepts B_1. The pushdown automaton accepting B_1 will have two stack symbols: Z and O (standing for `00` and `11` respectively). If the automaton reads a `00`, and the top of the stack is a O, the stack is popped (the O on the top is matched with this `00`), else a new Z is pushed. Similarly, if the automaton reads a `11`, and the top of the stack is a Z, the stack is popped (the Z on the top is matched with this `11`), else a new O is pushed. Finally, the word is accepted if at the end the automaton is in a final state and the stack does not contain a Z or a O at the top. The formal construction follows.

$$\begin{array}{lcl}
Q & = & \{s_s, s_0, s_1, s_f, s_r\} \\
\Sigma & = & \{\mathtt{0}, \mathtt{1}\} \\
\Gamma & = & \{Z, O, BOT\} \\
\\
\delta(s_s, 0, BOT) & = & (s_0, BOT) \\
\delta(s_s, 0, BOT) & = & (s_1, BOT) \\
\delta(s_0, 0, Z) & = & (s_0, ZZ) \\
\delta(s_0, 0, O) & = & (s_0, \epsilon) \\
\delta(s_0, 1, Z) & = & (s_1, Z) \\
\delta(s_0, 1, O) & = & (s_1, O) \\
\delta(s_1, 0, Z) & = & (s_1, \epsilon) \\
\delta(s_1, 1, O) & = & (s_1, OO) \\
\delta(s_1, 0, Z) & = & (s_0, Z) \\
\delta(s_1, 0, O) & = & (s_0, O) \\
\delta(s_0, \epsilon, BOT) & = & (s_f, BOT) \\
\delta(s_1, \epsilon, BOT) & = & (s_f, BOT) \\
\delta(s_f, 0, BOT) & = & (s_r, BOT) \\
\delta(s_f, 1, BOT) & = & (s_r, BOT) \\
\\
q_0 & = & s_s \\
Z_0 & = & BOT \\
F = & = & \{s_f\}
\end{array}$$

The start state is s_s, the state s_0 (respectively, s_1 corresponds to a state where the last symbol seen is a `0` (respectively, a `1`). The state s_f is a final state, and s_r is a reject state.

b Note that the `01`'s and the `10`'s occur in pairs: we cannot have two `01`'s without having a `10` in the middle. Hence, we can recognize this language with only a finite amount of memory, and this language is regular. Here is a finite automaton for B_2:


0
1
1
0
0
1
0
0
1
1