COMP 330 Autumn 2018
Assignment 1
Solutions

Prakash Panangaden

Question 1[20 points] Fix a finite alphabet Σ and let $\emptyset \neq L \subseteq \Sigma^*$. We define the following relation R on words from Σ^*:

$$\forall x, y \in \Sigma^*, xRy \text{ if } \forall z \in \Sigma^*, xz \in L \text{ iff } yz \in L.$$

Prove that this is an equivalence relation.

Solution. We must check the three properties:

reflexivity: We want to show $\forall x \in \Sigma^*, xRx$. This means that $\forall z \in \Sigma^*, xz \in L$ iff $xz \in L$; this is clearly true since the two sides of the "iff" are exactly the same.

symmetry: Here we want to show that $\forall x, y \in \Sigma^*, xRy$ implies yRx.

$$\forall z \in \Sigma^*, xz \in L \text{ iff } yz \in L$$

implies that

$$\forall z \in \Sigma^*, yz \in L \text{ iff } xz \in L.$$

But "iff" is reversible so clearly this holds.

transitivity: Here we must show that $\forall x, y, z \in \Sigma^*$ if we have (first assumption)

$$\forall w \in \Sigma^*, xw \in L \text{ iff } yw \in L$$

and (second assumption)

$$\forall w \in \Sigma^*, yw \in L \text{ iff } zw \in L$$

then (conclusion)

$$\forall w \in \Sigma^*, xw \in L \text{ iff } zw \in L.$$

Suppose w is some word in Σ^* and suppose that $xw \in L$, then the first assumption tells us that $yw \in L$. Using this fact with the second assumption tells us that $zw \in L$. Thus $xw \in L$ implies that zw is in L. Proceeding in the same way, if $zw \in L$ we must have $yw \in L$ and thence $xw \in L$. Thus we have established the conclusion.

This is a very pedantically written proof, far too detailed for most tastes. I just want you to have the sense of what goes into such a proof. In future you can state obvious things as "obvious" but don't try and pass off things that are not at all obvious as obvious.

Question 2[20 points] Consider, pairs of natural numbers $\langle m, n \rangle$ where $m, n \in \mathbf{N}$. We order them by the relation $\langle m, n \rangle \sqsubseteq \langle m', n' \rangle$ if $m < m'$ or $(m = m') \wedge n \leq n'$, where $\leq$ is the usual numerical order. Prove that the relation $\sqsubseteq$ is a partial order. [10 points]

Solution From the definition of a partial order we have to verify that $\sqsubseteq$ is (i) reflexive, (ii) antisymmetric and (iii) transitive. We tackle each property in turn. (i) If we compare $\langle m, n \rangle$ with itself we see that $m = m$ so we are in the second case, and here we have $n \leq n$ so we have $\langle m, n \rangle \sqsubseteq \langle m, n \rangle$ [You may be tempted to just say this is obvious; yes it is and I would accept that, in this case.] (ii) Suppose that $\langle m_1, n_1 \rangle \sqsubseteq \langle m_2, n_2 \rangle$ and $\langle m_2, n_2 \rangle \sqsubseteq \langle m_1, n_1 \rangle$. Now suppose that $m_1 < m_2$ then it is impossible for $\langle m_2, n_2 \rangle \sqsubseteq \langle m_1, n_1 \rangle$ to hold. So if both hold we must have $m_1 = m_2$. Then we know that $n_1 \leq n_2$ and $n_2 \leq n_1$. But the ordinary $\leq$ relation is known to be antisymmetric so $n_1 = n_2$. Thus the two pairs are equal as pairs. (iii) Since there are two possible cases for each instance of the $\sqsubseteq$ relation we have 4 combinations to consider. Suppose $\langle m_1, n_1 \rangle \sqsubseteq \langle m_2, n_2 \rangle$ and $\langle m_2, n_2 \rangle \sqsubseteq \langle m_3, n_3 \rangle$. We must show that $\langle m_1, n_1 \rangle \sqsubseteq \langle m_3, n_3 \rangle$.

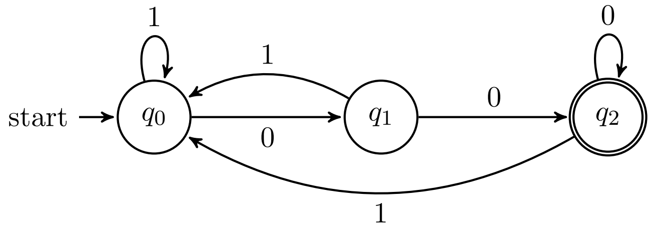
1. Suppose that $m_1 = m_2$, $n_1 \leq n_2$, $m_2 = m_3$ and $n_2 \leq n_3$. Then clearly we have $m_1 = m_3$ (equality is transitive) and $n_1 \leq n_3$ (the $\leq$ relation is transitive) so we have $\langle m_1, n_1 \rangle \sqsubseteq \langle m_3, n_3 \rangle$.
2. Suppose $m_1 < m_2$ and $m_2 = m_3$ then $m_1 < m_3$ and we have $\langle m_1, n_1 \rangle \sqsubseteq \langle m_3, n_3 \rangle$.
3. Suppose that $m_1 = m_2$ and $m_2 = m_3$, again we have $m_1 < m_3$ so $\langle m_1, n_1 \rangle \sqsubseteq \langle m_3, n_3 \rangle$.
4. Suppose $m_1 < m_2$ and $m_2 < m_3$ then $m_1 < m_3$ and we have $\langle m_1, n_1 \rangle \sqsubseteq \langle m_3, n_3 \rangle$.

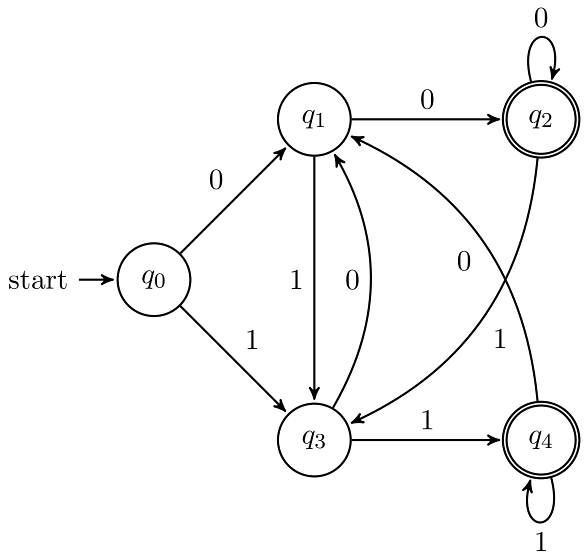
We have checked all possible cases so we conclude that $\sqsubseteq$ is transitive and have thus completed the proof that it is a partial order.

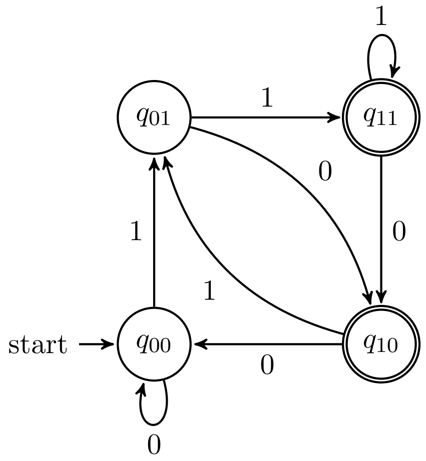
Question 3[20 points] Give deterministic finite automata accepting the following languages over the alphabet $\{0, 1\}$.

1. The set of all words ending in 00.
2. The set of all words ending in 00 *or* 11.
3. The set of all words such that the *second* last element is a 1. By "second last" I mean the second element counting backwards from the end. Thus, 0001101 is not accepted and 10101010 is accepted.

Solutions: The automata are shown in the following pictures:







Question 4[20 points] Suppose that L is a language accepted by a DFA (i.e. a regular language) show that the following language is also regular:

$$\mathsf{lefthalf}(L) := \{w_1 | \exists w_2 \in \ \Sigma^* \text{ such that } w_1 w_2 \in L \text{ and } |w_1| = |w_2|\}.$$

[Hint: nondeterminism.]
Solution Suppose L is defined by $(Q, \Sigma, \delta, q_0, F)$. We will make a new NFA $(Q', \Sigma, \delta', Q_0, F')$.

$$\begin{aligned} & Q' = Q \times Q \\ & Q_0 = \{(q_0, f) | f \in F\} \\ & F' = \{(s, s) | s \in Q\} \\ & \delta'((s,t), a) = \{(\delta(s,a), t') | t = \delta(t', b) \text{ for some } b \in \Sigma\} \end{aligned}$$

The states in the new machine are pairs of states from the old machine. Given a string w, the first coordinate just keeps track of how the old machine would act on w. The second coordinate acts a bit like the machine for $rev(L)$. It starts in an accepting state from the old machine, and works backward, keeping track of all the possible paths $rev(L)$ could take, given any input of the same length as the part of w that the machine has processed at that step. If the machine reaches the end of w and one path has ended up in a state (s, s), then the machine for L would be in state s upon reading w. Furthermore since the second coordinate started out in an accepting state and went backward to s in $|w|$ steps, there is some forward path in the old machine of length $|w|$ starting in s and ending up in an accepting state. This path defines a string w' such that $|w| = |w'|$, and if we input ww' into the old machine, it reads w and reaches s, then starting from s and reading w', it ends up in an accepting state. So ww' is in L, and w is accepted by the new machine if and only if it is in $\mathsf{lefthalf}(L)$. Therefore $\mathsf{lefthalf}(L)$ is a regular language for any regular language L.

Here is another solution: Suppose L is defined by $(Q, \Sigma, \delta, q_0, F)$. We will make a new NFA $(Q', \Sigma, \delta', Q_0, F')$.

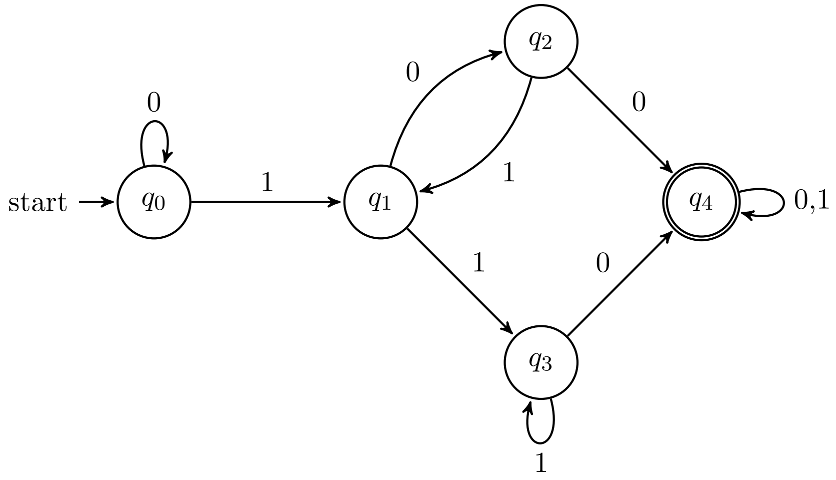
$$\begin{aligned}
&Q' = Q \times Q \times Q \\
&Q_0 = \{(q_0, s_1, s_1) | s_1 \in Q\} \\
&F' = \{(s_1, s_1, s_2) | s_1 \in Q \text{ and } s_2 \in F\} \\
&\delta'((s_1, s_2, s_3), a) = \{(\delta(s_1, a), s_2, s_4) | s_4 = \delta(s_3, b) \text{ for some } b \in \Sigma\}
\end{aligned}$$

Each state of this machine is a triple of states from the old machine. Given a word w, the first coordinate keeps track of how the old machine would act on w, the third coordinate simultaneously keeps track of how the old machine would act on any possible input of the same length as what it has read so far of w starting at any state, and the second coordinate keeps track of whether the start state of the third coordinate is the same as the end state of the first coordinate. So, if a word w_1 is in lefthalf(L), there is a word w_2 with $|w_1| = |w_2|$ and $w_1w_2 \in L$. Our machine starts out in all possible states (q_0, q, q). Then at step $n + 1$, it goes to a state where the first coordinate is the state that our old machine would be in if it had read the first n characters of w_1. The second coordinate never changes, and the third coordinate is any state the old machine could be in after reading n characters of any word w_2, starting in q, the state represented by the second coordinate. So, when the machine has read all of w_1, if the first coordinate is q, then the old machine would have arrived in state q after having read w_1. And if the last coordinate is also an accepting state, then there is some word w_2 with $|w_1| = |w_2|$ so that the old machine would reach an accepting state if it started in q and read w_2. Thus, the old machine would read w_1, get to q, then read w_2 and finally arrive in an accepting state. Therefore, the new machine accepts exactly $\mathsf{lefthalf}(L)$.

Question 5[20 points]

1. Give a deterministic finite automaton accepting the following language over the alphabet $\{0, 1\}$: The set of all words containing 100 or 110. [5 points]
2. Show that *any* DFA for recognizing this language must have at least 5 states. [15 points]

Solution We need to remember the last two characters seen so far and we need to know that we have never seen a 1, so intuitively one expects to have 5 states. Of course, this is not a proof that you really *need* 5 states. Here is the automaton:



To prove that 5 states are really necessary we should find 5 strings and show that they all must end up in different states. If you look at the automaton you can see 5 strings that take you to each one of the states. These are $w_0 = \varepsilon$, $w_1 = 1$, $w_2 = 10$, $w_3 = 11$ and $w_4 = 100$. We have numbered them so that w_i takes you to state q_i. Now let us consider any putative recognizer for the language. Let the states that are reached from the start state by string w_0 be called A, the state reached by w_1 be called B and so on to give states A, B, C, D, E. In this machine we do not know that these are all distinct states; we have to prove that. We will analyze all the possible cases.

- We can see right away that all the states A, B, C, D are different from E since w_4 is accepted and all the others are rejected.
- Suppose $A = B$ so w_0 and w_1 end up in the same state. Now consider the strings $w_0 \cdot 00 = \varepsilon \cdot 00 = 00$ and $w_1 \cdot 00 = 1 \cdot 00 = 100$. The first string should be rejected and the second one should be accepted so A and B cannot be the same state.
- Suppose $A = C$. This is not possible since $w_0 \cdot 0 = 0$ should be rejected and $w_2 \cdot 0 = 100$ should be accepted.
- Suppose $A = D$. This time we choose to extend the strings by 0 and we get a contradiction.
- Suppose $B = C$. We choose to extend the strings w_1 and w_2 by 0 and get a contradiction.
- Suppose $B = D$. We choose to extend the strings w_2 and w_3 by 0 and get a contradiction.
- Suppose $C = D$. We choose to extend the strings w_3 and w_4 by 10 and get a contradiction.