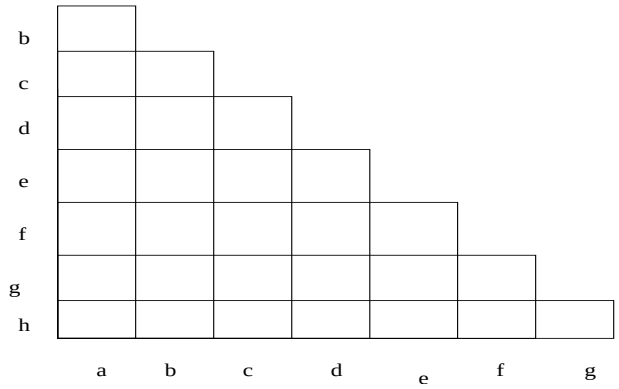
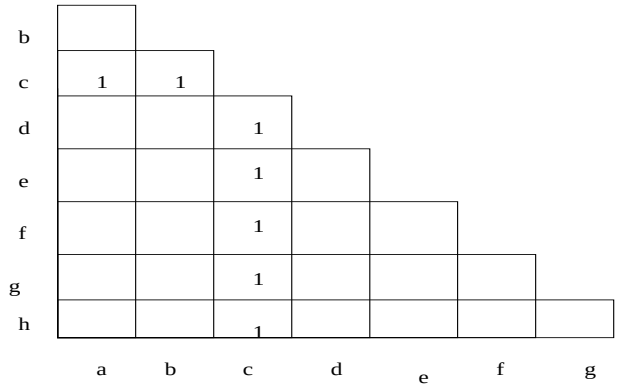
Solutions to Homework #6

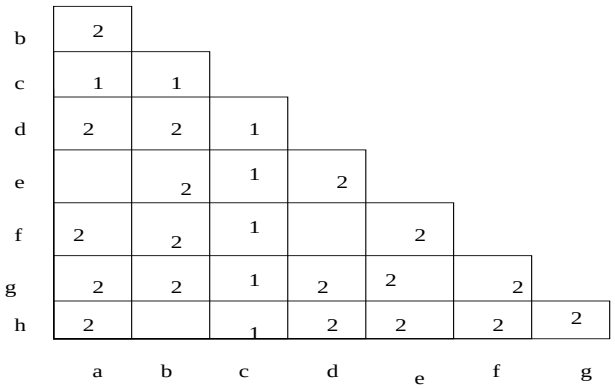
1. List all the state pairs (unordered) as below:



Pass 1: Mark all the state pairs with exactly one state being a final state.

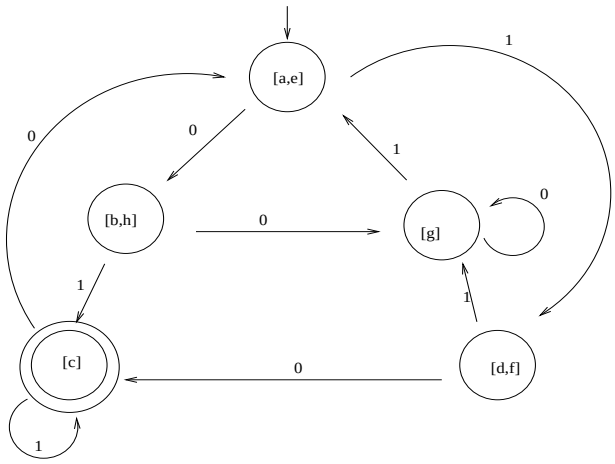


Pass 2: Mark each unmarked state pair (r, s) satisfying: there are (p, q) and a such that $\delta(r, a) = p$ and $\delta(s, a) = q$ and (p, q) was marked before.



Pass 3: no new pairs can be marked.

Now, we have $a \equiv e, b \equiv h$ and $d \equiv f$. The minimal automaton can be drawn as below:



2. Notice that the problem is equivalent to check the emptiness of $(L(M_1) - L(M_2)) \cup (L(M_2) - L(M_1))$. Since regular languges are closed under Boolean operations, there for we have an algorithm to construct a FA M from M_1 and M_2 to accept $(L(M_1) - L(M_2)) \cup (L(M_2) - L(M_1))$. Recall we also have an algorithm to check whether $L(M)$ is empty.

3. Suppose L can be accepted by a DFA M. Now we construct a Λ-NFA M^R that "reverses" M as below. M^R has the same state space as M. But for

each transition $\delta(p, a) = q$ in M, we have $p \in \delta'(q, a)$ (That is, the arrow on each arc is reversed). The initial state of M is the final state of M^R, and the final states in A of M are the initial states of M^R. But since we do not allow more than one initial state in a FA, we have to add a new state q_d into M^R such that the new state is connected to each state in A by a Λ-transition. Here is the explicit construction:

Suppose DFA $M = \langle Q, \Sigma, q_0, A, \delta \rangle$. Let $M^R = \langle Q \cup \{q_d\}, q_d, \{q_0\}, \delta' \rangle$ be a Λ-NFA satisfying:

$p \in \delta'(q, a)$ iff $\delta(p, a) = q$, for all $p, q \in Q$ and $a \in \Sigma$,
and $\delta'(q_d, \Lambda) = A$.

Clearly, (you need to figure it out), $L(M^R) = L^R$. From Kleene's Theorem, L^R is regular.

4. Suppose L can be accepted by a DFA M. Now we construct a Λ-NFA M^P that accepts $Prefix(L)$ and therefore $Prefix(L)$ is regular. The ideas are as follows. x is accepted by M^P iff x can be extended to another string xy such that xy can accepted by M. M^P works as follows. On the given input word x, M^P simulates M. Suppose after consuming x, M falls into state q. At this time, M^P starts to simulate M again from state q – but notice that x is all the input word we have, the simulation can not consume any more input. Here comes the tricky part. Instead of simulating M, M^P simulates a Λ-version of M (i.e., change any transition in M into a Λ-transition) from state q. If the Λ-version of M (from state q) falls into an accepting state – this means there is a y with xy in $L(M)$.

In order to construct M^P, we need two copies of M: one is the original M, the other is the Λ-version of M. That is, suppose DFA $M = \langle Q, \Sigma, q_0, A, \delta \rangle$. Let $M^R = \langle Q \cup Q', \Sigma, q_0, A', \delta' \rangle$ be a Λ-NFA satisfying:

(1). $Q' = \{q' : q \in Q\}$ with each state $q \in Q$ renamed as q'. That is, the state space of M^P is the union of Q and Q'.

(2). $A' = \{q' : q \in A\}$.

(3). δ' (the transitions of M^P) are:

(3.1). Those transitions in M, i.e., $p \in \delta'(q, a)$ iff $\delta(q, a) = p$, for all $p, q \in Q$ and $a \in \Sigma$,

(3.2). Those transitions in Λ-version of M, i.e., $p' \in \delta'(q', \Lambda)$ iff there exists $a \in \Sigma$ such that $\delta(q, a) = p$, for all $p, q \in Q$.

(3.3). Λ-transitions connecting the two copies: $q' \in \delta'(q, \Lambda)$ for all $q \in Q$.

(4). Each δ' can only be constructed from above.

5. Suppose L can be accepted by a DFA M. Now we construct a Λ-NFA M^H that accepts $Half(L)$ and therefore $Half(L)$ is regular. The ideas are as follows. x is accepted by M^H iff x can be extended to another string xy such that xy can accepted by M and $|x| = |y|$. If we make two copies of M just as in Problem 5, we will face a problem on how to make sure that $|x| = |y|$ – that is, to make sure that M^H executes the same number of transitions when simulates M as when simulates the Λ-version of M. A trick is to make the two copies running synchronously: whenever M fires a transition, the Λ-version of M also fires a (Λ-) transition. But how do we know which state the Λ-version of M starts from? M^H guesses a starting state for the Λ-version and at the end simulating M, M^H checks the state that M falls into is exactly the starting state guessed for the Λ-version. M^H accepts the input x if the Λ-version falls into an accepting state.

Here is the construction: suppose DFA $M = \langle Q, \Sigma, q_0, A, \delta \rangle$. Let $M^H = \langle \{q_d\} \cup (Q \times Q \times Q), \Sigma, q_d, A', \delta' \rangle$ be a Λ-NFA satisfying:

(1). The state space of M^H is all the state triples of Q, plus the newly added state q_d,

(2). the newly added state q_d is the initial state of M^H,

(3). the accepting states of M^H is $A' = \{(p, p, q) : p \in Q, q \in A\}$.

(4). Transitions in M^H are:

(4.1). $(p, q_0, p) \in \delta'(q_d, \Lambda)$, for all $p \in Q$. (This part corresponds to "guessing a starting state for the Λ-version")

(4.2). $(p, q, r) \in \delta'((p, s, t), a)$ iff $\delta(s, a) = q$ and there exists $b \in \Sigma$ such that $\delta(t, b) = r$, for all $p, q, r, s, t \in Q$. (This corresponds to there is a transition in M (the first copy) from state s to state q on reading input symbol a (this a is from the input word x), and synchronously, there is a Λ-transition in the Λ-version from state t to state r. Notice that p is used to remember the guessed starting state for the Λ-version).

(4.3). All the transitions of M^H are constructed from above.