

Some worked examples for problem set 4

1. Solution to Chapter 3, Exercise 9, a) - d)

Part (a):

Alphabet: $\{a, b\}$

Set of states: $\{q_0, q_1, q_2, q_3\}$

Initial state: q_0

Set of final states: $\{q_3\}$

Transition relation: The following transitions are allowed: $\langle q_0, a, q_1 \rangle$, $\langle q_0, b, q_2 \rangle$, $\langle q_1, b, q_1 \rangle$, $\langle q_1, a, q_3 \rangle$, $\langle q_2, a, q_2 \rangle$, and $\langle q_2, b, q_3 \rangle$.

Part (b):

A FS_7 computation of $abba$:

Step	State	Scanning Position
0	q_0	$\langle \epsilon, abba \rangle$
1	q_1	$\langle a, bba \rangle$
2	q_1	$\langle ab, ba \rangle$
3	q_1	$\langle abb, a \rangle$
4	q_3	$\langle abba, \epsilon \rangle$

A FS_7 computation of bb :

Step	State	Scanning Position
0	q_0	$\langle \epsilon, bb \rangle$
1	q_2	$\langle b, b \rangle$
2	q_3	$\langle bb, \epsilon \rangle$

Part (c): For each state of FS_7 and symbol of the alphabet, there is at most one arc labeled with the symbol leading from the state. This means that there can be at most one FS_7 computation for each string of the alphabet. The computation for bbb in FS_7 is this.

Step	State	Scanning Position
0	q_0	$\langle \epsilon, bbb \rangle$
1	q_2	$\langle b, bb \rangle$
2	q_3	$\langle bb, b \rangle$

The computation can't go beyond Step 2 because there is no transition for b from q_3 . This computation is not successful because it doesn't reach the end of the string. Since there are no other computations of bbb , this string is not accepted by FS_3 .

Part (d):

$$(\{a\} \circ \{b\}^* \circ \{a\}) \cup (\{b\} \circ \{a\}^* \circ \{b\})$$

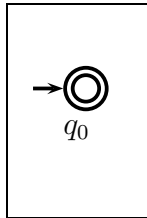
Solutions to Chapter 3, Exercise 10, a) - f) and i):

(a)

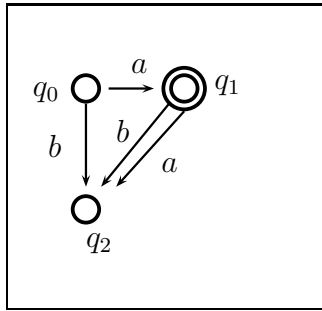
Any machine with no final states will accept exactly the empty set.

(b)

This one is pretty simple: The initial state is a final state, but since there are no transition rules, the only string that can be fully read by this FSA, while terminating in a final state, is ϵ .

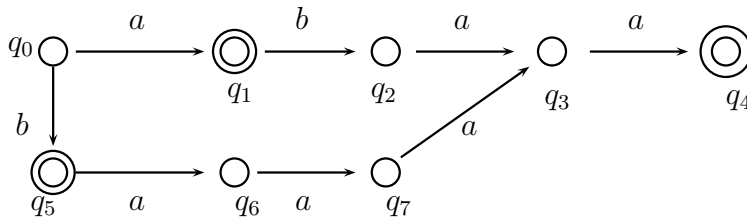


(c)



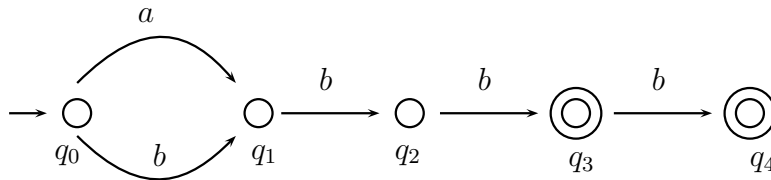
(d)

An FSA that accepts $\{a, b, baaa, abaa\}$:



(e)

An FSA that accepts the language $\{a, b\} \circ \{bb, bbb\}$:



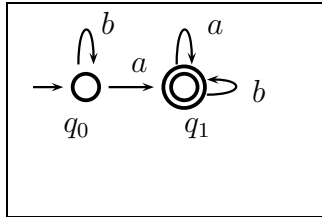
(f)

If you just look at the regular expression describing the language, the problem seems quite complicated, but there is a simple way to look at it. The strings in this language are all and only the strings of the form:

(Some string of a's and b's, including, possibly the empty string)^{*}a^{*}(Some string of a's and b's, including, possibly the empty string)

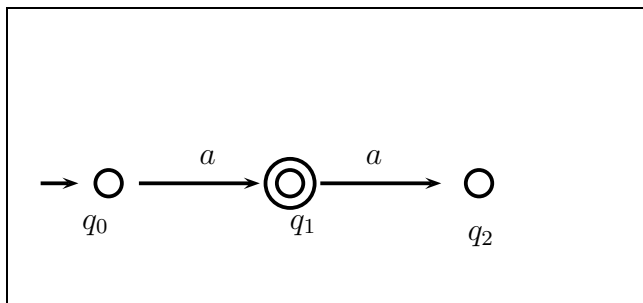
That is, the strings in this language are exactly the strings over the alphabet $\{a, b\}$ that contain at least one a .

So the machine we need will just keep reading letters until it comes across an a , and then it will move into a final state and keep reading the rest of the letters. One machine like that has this diagram:



(i)

Here is an FSA recognizing the set $\{x \mid x \text{ contains exactly 1 occurrence of } a\}$ if the alphabet is just $\{a\}$. If the alphabet is $\{a, b\}$, add an arrow labelled “ b ” from q_0 to itself and from q_1 to itself.



2. Prove that if X is a set of strings recognized by some FSA, then X^* is recognized by an FSA.

The strategy here is similar to the one that will work in the case of \circ on the problem set, except that instead of splicing two machines, you splice the final states of one machine with its own beginning states.

Say that F accepts X . We want to define F^* that will recognize X^* . First, keep all the states and transitions of F in place as states and transitions of F^* .

First we need to ensure that the empty string is recognized. To do this, add an initial state $q_{new\ initial\ state}$ that duplicates the initial state of F, except that it has no arrows coming to it. That is: Say q_0 is the initial state of F, and q' is some state, and there is some transition $\langle q', u, q_0 \rangle$ in F. There will be *no* corresponding transition for $q_{new\ initial\ state}$. However, for any state q' with a transition $\langle q_0, u, q' \rangle$ in F, there will be a transition $\langle q_{new\ initial\ state}, u, q' \rangle$ in F^* . Make $q_{new\ initial\ state}$ a final state of F^* .

Now to make sure that F^* can read the results of multiple strings concatenated together. The idea is just to set things up so that the machine can run over and over again. For every final state \tilde{q} of F, add to F^* every transition $\langle \tilde{q}, u, q' \rangle$ where $\langle q_0, u, q' \rangle$ is a transition of F. That is: Say that F, upon reading a string u , in the initial state q_0 , goes into state q' . Then we will add to F^* the transition that goes directly from final state \tilde{q} to q' . This has the effect of “cycling” the machine over and over again.

(Note: It is a good idea to try this on a few examples. Make a three-state FSA that accepts only $\{aa\}$. What would F^* look like in this case?)

- Design an FSA that accepts this set: $\{b\} \cup \{x/x \text{ consists of an odd number of } a\text{'s}\}$

Here is an FSA that does the job:

