CmSc 365 Theory of Computation

Deterministic Finite State Automata (FSA)

Problems:

Problem 1:

Let $M$ be a deterministic FSA. Under exactly what circumstances is $e \in L(M)$?

Answer:

$e \in L(M)$ iff the starting state is a final state, i.e. $s \in F$.

Problem 2:

Describe informally the language accepted by the following FSAs:

(initial states are represented as:

```
  ▶
```

Final states are represented as:

```
    ●
```

(a.

The language is represented by the regular expression $a(ba)^*$.)
b. 

The language is represented by the regular expression \( a^*b \)

**Problem 3:**
Construct a deterministic FSA accepting each of the following languages:
(an FSA per language)

a. \( \{ w \in \{a,b\}^*: \text{each } a \text{ in } w \text{ is immediately preceded by } b \} \)
b. \( \{ w \in \{a,b\}^*: w \text{ has } abab \text{ as a substring} \} \)
c. \( \{ w \in \{a,b\}^*: w \text{ has neither } aa \text{ nor } bb \text{ as a substring} \} \)
d. \( \{ w \in \{a,b\}^*: w \text{ has an odd number of } a \text{'s and an even number of } b \text{'s } \} \)
e. \( \{ w \in \{a,b\}^*: w \text{ has both } ab \text{ and } ba \text{ as substrings} \} \)

**The basic technique to construct an FSA:**
Think of the minimal instance of a string satisfying the conditions of the problem, and build incompletely defined FSA that accepts that string. Then make the FSA complete by adding the necessary transitions.
**Solutions:**

**Problem 3,a:** \( \{ w \in \{a, b\}^* : \text{each } a \text{ in } w \text{ is immediately preceded by } b \} \)

The alphabet would be: \{a,b\}

Let \( q_0 \) be the initial state. We want each \( a \) to be preceded by \( b \), so the first symbol to be accepted should be \( b \) (i.e. the strings should start with \( b \))

There may be only one \( b \), so \( q_1 \) should be a final state.

Before the first \( a \) to come there may be any number of \( b \)s, accepted while the FSA is in state \( q_1 \):

We are ready to accept \( a \) now. This may be the last symbol, so from \( q_1 \) on \( a \) the FSA may go to a final state, or may continue to accept \( b \)'s

Thus we make a link from \( q_1 \) to \( q_0 \), and adding \( q_0 \) to the set of final states:
The FSA built so far accepts strings where each \( a \) is preceded by \( b \)

However, the FSA is not completely defined.
So far we have:

\[
\begin{align*}
\delta(q_0, a) &= ? & \delta(q_0, b) &= q_1 \\
\delta(q_1, a) &= q_0 & \delta(q_1, b) &= q_1 
\end{align*}
\]

If at \( q_0 \) the input is \( a \), the string should not be accepted. The technique here is:
Add another state, \( q_2 \), not final
Add
\[
\begin{align*}
\delta(q_0, a) &= q_2 \\
\delta(q_2, a) &= q_2 \\
\delta(q_2, b) &= q_2 
\end{align*}
\]

Thus if the string contains two or more consecutive \( a \)'s
the FSA will wind up in \( q_2 \) and will stay there till the end of the string. Since \( q_2 \) is not a final state, the string would not be accepted.

The final FSA is:
Formally, the FSA is represented by:

\[ K = \{ q_0, q_1, q_2 \} \]
\[ \Sigma = \{ a, b \} \]
\[ s = q_0 \]
\[ F = \{ q_1 \} \]

The transition function is given by the following table:

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>q0</td>
<td>q2</td>
<td>q1</td>
</tr>
<tr>
<td>q1</td>
<td>q0</td>
<td>q1</td>
</tr>
<tr>
<td>q2</td>
<td>q2</td>
<td>q2</td>
</tr>
</tbody>
</table>

**Problem 3.b:** \( \{ w \in \{ a, b \}^*: w \text{ has } abab \text{ as a substring} \} \)

We first build an incomplete FSA that accepts \( abab \), and then we'll make it complete.

The initial state will be \( q_0 \), the final state - \( q_4 \)

Now we have to completely define the transition function.

1. What happens if at \( q_0 \) we get \( b \)'s? We just stay there waiting for an \( a \).
2. What happens if at \( q_1 \) we get one or more \( a \)'s? We stay there waiting for \( b \) to come.
3. What happens if at \( q_2 \) we get \( b \)? The FSA has read \( ab \). Another \( b \) will spoil the substring \( abab \), so we have to start where we wait for an \( a \), i.e. at \( q_0 \)
4. What happens if at \( q_3 \) we get \( a \)? We have read so far \( aba \), and wait for \( b \). An \( a \) will spoil the substring, but it can be considered as the first \( a \) of another substring "abab", so we have to go where the first \( b \) is waited: at \( q_1 \).
5. At \( q_4 \) we are sure that the string contains \( abab \), so any sequence of \( a \)'s and \( b \)'s is allowed.
Thus the FSA will be:

Formally, the FSA is represented by:

K = \{q0, q1, q2, q3, q4\}
\Sigma = \{a, b\}
s = q0
F = \{q4\}
The transition function is given by the following table:

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>q0</td>
<td>q1</td>
</tr>
<tr>
<td>q1</td>
<td>q1</td>
</tr>
<tr>
<td>q2</td>
<td>q3</td>
</tr>
<tr>
<td>q3</td>
<td>q4</td>
</tr>
<tr>
<td>q4</td>
<td>q4</td>
</tr>
</tbody>
</table>
Problem 3,c
\{w \in \{a,b\}^* : w \text{ has neither } aa \text{ nor } bb \text{ as a substring} \} \\

Here we don't want \text{aa} or \text{bb}, so we'll have one path for \text{aa} and one path for \text{bb} winding up in a state that is not final and staying there till the end of the string.

The states q1 and q2 are guards: if at q1 a second a comes, the FSA gets into q3, if at q2 a second b comes, the FSA again gets into q3, and once at q3, there is no way out - q3 is designed not to be a final state.

Now we have constructed an FSA that rejects any string starting with \text{aa}, or \text{bb}. We still have not defined the set of final states, and the transition function for (q1,b) and(q2,a).

If the string is empty, \text{aa} and \text{bb} would not be there, so the empty string is acceptable, hence q0 has to be a final state.

If the string has only one a, or only one b, it is acceptable, so q1 and q2 have to be final states too.

Thus \(F = \{q0,q1,q2\}\)

If at q1 the input is b, the FSA has to enter q2 - this is the state that guards the b's.

If at q2 we have an a, the FSA has to enter q1 - this is the state that guards the a's.

As long as the input is \text{ababababa}... or \text{bababababa}... the FSA moves between q1 and q2 and can stop at either of them - the string is acceptable. If however at q1 we get an a, it would be preceded by another a, and then the string is not acceptable - that is why we end up in q3 and stay there no matter what comes next. Same is true if at q2 we get a b.
Thus we have:

\[ K = \{q_0, q_1, q_2, q_3\} \]
\[ \Sigma = \{a, b\} \]
\[ s = q_0 \]
\[ F = \{q_0, q_1, q_2\} \]

The transition function is given in the following table.

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>q0</td>
<td>q1</td>
</tr>
<tr>
<td>q1</td>
<td>q3</td>
</tr>
<tr>
<td>q2</td>
<td>q1</td>
</tr>
<tr>
<td>q3</td>
<td>q3</td>
</tr>
</tbody>
</table>
Problem 3,d

\{ w \in \{a,b\}^*: w \text{ has an odd number of } a\text{'s and an even number of } b\text{'s } \}

The state diagram of this FSA is as follows:

The initial state is q0, and the final state is q1. Consider the paths from q0 to q1 without cycles.

If you go to the right, (q0,q1) this path has odd number of a's (exactly one a) and even number of b's (zero b's).

If you go to the left (q0,q2,q3,q1) this path has odd number of a's (q2,q3) and even number of b's ((q0,q2) and (q3,q1)).

If you go back and forth, you will make cycles, and each cycle in this diagram gives even number of a's and even number of b's. You know that when an even number is added to an odd number the result is an odd number. Also, when an even number is added to an even number, the result is an even number.

Hence, the FSA recognizes strings that have odd number of a's and even number of b's.

Thus we have:
K = \{q0,q1,q2,q3\}
Sigma = \{a,b\}
s = q0
F = \{q1\}

The transition function is given in the following table.

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>q0</td>
<td>q1</td>
<td>q2</td>
</tr>
<tr>
<td>q1</td>
<td>q0</td>
<td>q3</td>
</tr>
<tr>
<td></td>
<td>a</td>
<td>b</td>
</tr>
<tr>
<td>------</td>
<td>----</td>
<td>----</td>
</tr>
<tr>
<td>q2</td>
<td>q3</td>
<td>q0</td>
</tr>
<tr>
<td>q3</td>
<td>q2</td>
<td>q1</td>
</tr>
</tbody>
</table>
Problem 3.e \( \{ w \in \{a, b\}^* : w \) has both \( ab \) and \( ba \) as substrings\} 

The minimal strings that contain \( ab \) and \( ba \) as substrings are: \( aba \) and \( bab \).

So we need one path of three links, corresponding to \( aba \), and one path of three links corresponding to \( bab \). Both will start at the initial state, and will end at some final state, probably the same for the two paths.

The FSA that accepts \( aba \) and \( bab \) is the following:

This FSA is still not completed. We have to see what would be the transition at q1 and q5 on \( a \), at q2 and q4 on \( b \), and at q3 on \( a \) and \( b \).

If we are at q1 or q5 and another \( a \) comes, we just stay there waiting for \( b \) to come. In any case \( b \) will be preceded by an \( a \), so we will not miss the substring \( ab \).

If we are at q2 or q4 and a \( b \) comes, we stay there waiting for the \( a \) to come. In any case \( a \) will be preceded by a \( b \), so the substring \( ba \) will be there.

At q3 we have already the necessary substrings, so we accept anything that follows.

Thus we arrive at the following diagram:
Thus we have:

\[ K = \{q0,q1,q2,q3,q4,q5\} \]
\[ \Sigma = \{a,b\} \]
\[ s = q0 \]
\[ F = \{q3\} \]

The transition function is given in the following table.

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>q0</td>
<td>q1</td>
<td>q4</td>
</tr>
<tr>
<td>q1</td>
<td>q1</td>
<td>q2</td>
</tr>
<tr>
<td>q2</td>
<td>q3</td>
<td>q2</td>
</tr>
<tr>
<td>q3</td>
<td>q3</td>
<td>q3</td>
</tr>
<tr>
<td>q4</td>
<td>q5</td>
<td>q4</td>
</tr>
<tr>
<td>q5</td>
<td>q5</td>
<td>q3</td>
</tr>
</tbody>
</table>