Programming Languages and Computation

Problem Sheet 2: Deterministic Automata

* 1. For each of the automata in Figure 1, give (i) the trace of the automaton running on input 12212 and (ii) the language recognised by the automaton.

Solution

(a)

$$(q_0, 12212), (q_1, 2212), (q_1, 212), (q_1, 12), (q_2, 2), (q_3, \epsilon)$$

The language is: $\{1w1 \mid w \in \{2\}^*\}$

(b)

$$(q_0, 12212), (q_0, 2212), (q_0, 212), (q_0, 12), (q_0, 2), (q_0, \epsilon)$$

The language is $\{w \mid w \text{ contains 322 as a substring}\}$

(c)

$$(q_0, 12212), (q_1, 2212), (q_2, 212), (q_4, 12), (q_4, 2), (q_4, \epsilon)$$

The language is $\{w \mid w \neq 121v \text{ for any } v\}$

* 2. Give finite automata to recognise each of the following:

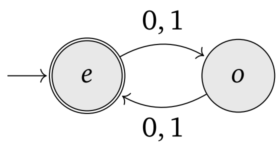
(a) The language of all strings over $\{0, 1\}$ such that each string has even length.

(b) The language of all strings over $\{0, 1\}$ such that each string has a number of 1s that is a multiple of three.

(c) The language of all strings over $\{0, 1\}$ such that each string has even length and a number of 1s that is a multiple of three.

Solution

(a)



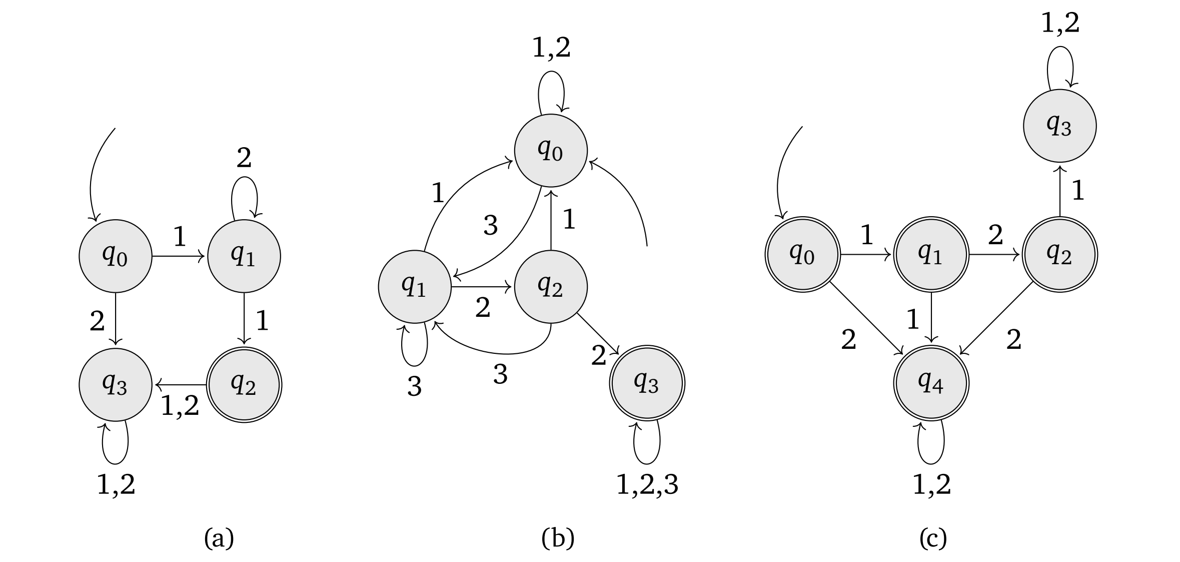
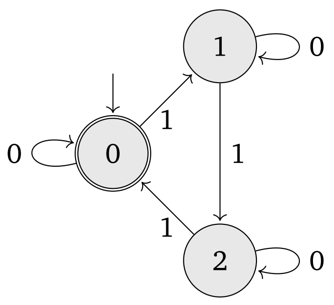
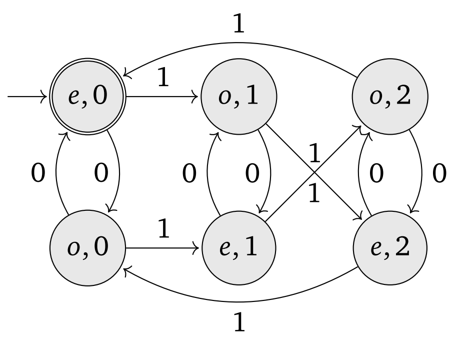


Figure 1: Automata used in Question 1.

(b)



(c) In the following, the idea is that the automaton will be in a state labelled e if it has seen an even number of letters so far, and o otherwise. It will be in a state labelled $i \in \{0, 1, 2\}$ if it has seen a number of 1s whose remainder is i after dividing by 3.



** 3. Construct a DFA to recognise Haskell floating point literals, e.g. `2.99`, `23.09e+34`, `0.12E−200`, `1.4e1`.

A general description is as follows. A *decimal literal* is a non-empty sequence of digits (0–9). A *floating point literal* is either:

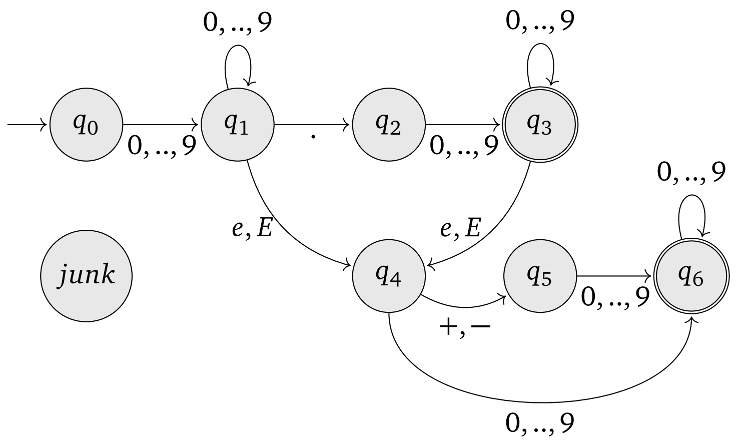
- a decimal literal followed by a decimal point followed by a decimal literal, optionally followed by an exponent
- or, a decimal literal followed by an exponent.

An exponent is either the character e or the character E; optionally followed by the character $+$ or the character $-$; followed in all cases by a decimal literal.

Since the diagram for this automaton will be relatively complex, you may wish to adopt the following convention, which allows you to omit certain transitions:

> If a state in the diagram does not have an outgoing edge for every letter of the alphabet, then it is implied that the missing edges all lead to a distinguished "junk" state, which is not an accepting state.

Solution



** 4. Suppose $M = (Q, \Sigma, \delta, q_0, F)$ is a DFA. Construct a DFA $\overline{M}$ with $L(\overline{M}) = \{w \in \Sigma^* \mid w \notin L(M)\}$. You will not be able to answer this question with a diagram.

Solution

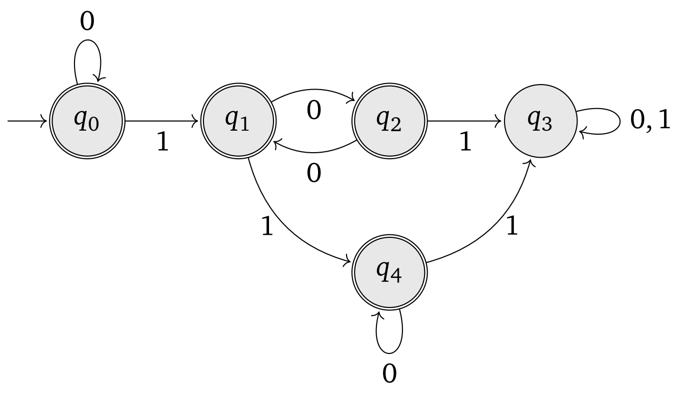
Let $\overline{F} = Q \setminus F$, i.e. $\{q \in Q \mid q \notin F\}$. Then define $\overline{M} = (Q, \Sigma, \delta, q_0, \overline{F})$.

** 5. Construct a DFA to recognise the language of strings over {0, 1} such that any two occurrences of 1 are separated by an even number of letters.

$\boxed{\text{Hint}}$ how many 1s can occur in a word of this language?

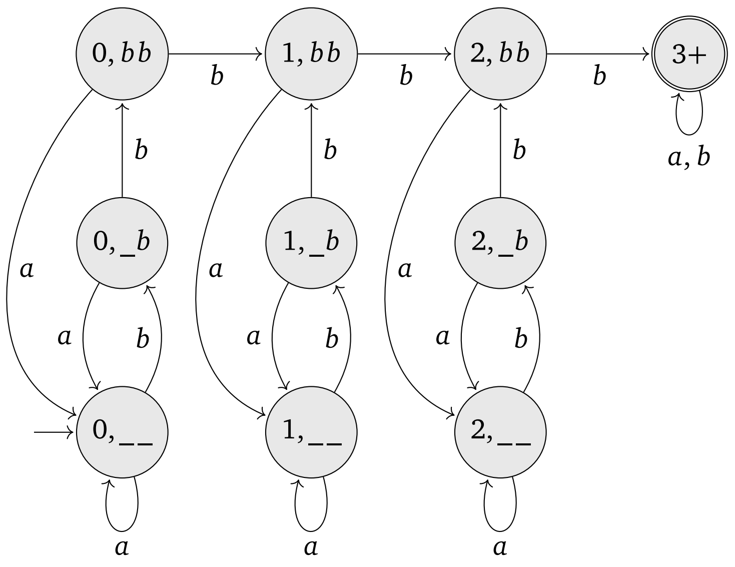
Solution

Some thought about the hint reveals that at most two 1s can occur in a word of this language, so we design around that:



*** 6. Construct a DFA to recognise the language of strings over the alphabet $\{a, b\}$ containing at least three occurrences of three consecutive b, overlapping permitted (e.g. the word $bbbbb$ should be accepted).

Solution



The key to this automaton is to recognise that you have to remember *two* things at any point in time:

- The number of sequences of 3 consecutive letter b.
- For those states in which we have not yet seen at least 3 consecutive subsequences of b, we should additionally remember whether the two letters immediately preceding were both b, or (if not) was the immediately preceding letter a b, or is it neither of the above.

We write the former as a number 0, 1, 2 or 3+; and the latter as bb, _b and _ _ respectively. The important thing is to know how many of the immediately preceding letters were b, so the initial state is associated with _ _ as there have been no letters at all yet.

Hopefully the need to remember the first of these is quite clear, the second helps to track progress towards completing a subsequence. When we have overlapping subsequences, as in $bbbb$ - after reading the first three b we can note that we have read one sequence of three consecutive b and, since the preceding two letters (i.e. the second and third letters of the word) were also b, reading in one more b will give another complete subsequence.

**** 7. (Optional) Let Σ be the following set of binary vectors:

$$\begin{pmatrix}0\\0\end{pmatrix}, \begin{pmatrix}0\\1\end{pmatrix}, \begin{pmatrix}1\\0\end{pmatrix}, \begin{pmatrix}1\\1\end{pmatrix}$$

We use each word w over alphabet Σ to encode a pair of natural numbers, written $[\![w]\!]$, which is defined by the following recursive function:

$$\begin{array}{rcl} [\![\epsilon]\!] & = & (0,0) \\ [\![\begin{pmatrix}b_1\\b_2\end{pmatrix} \cdot w]\!] & = & (2*m+b_1,\, 2*n+b_2) \\ & & \quad \textit{where}\ (m,\, n) = [\![w]\!] \end{array}$$

For example:

$$[\![\begin{pmatrix}0\\1\end{pmatrix}\begin{pmatrix}0\\0\end{pmatrix}\begin{pmatrix}1\\1\end{pmatrix}\begin{pmatrix}0\\1\end{pmatrix}]\!] = (4,\, 13)$$

$$[\![\begin{pmatrix}0\\1\end{pmatrix}\begin{pmatrix}1\\1\end{pmatrix}\begin{pmatrix}0\\0\end{pmatrix}\begin{pmatrix}1\\0\end{pmatrix}\begin{pmatrix}1\\0\end{pmatrix}]\!] = (26,\, 3)$$

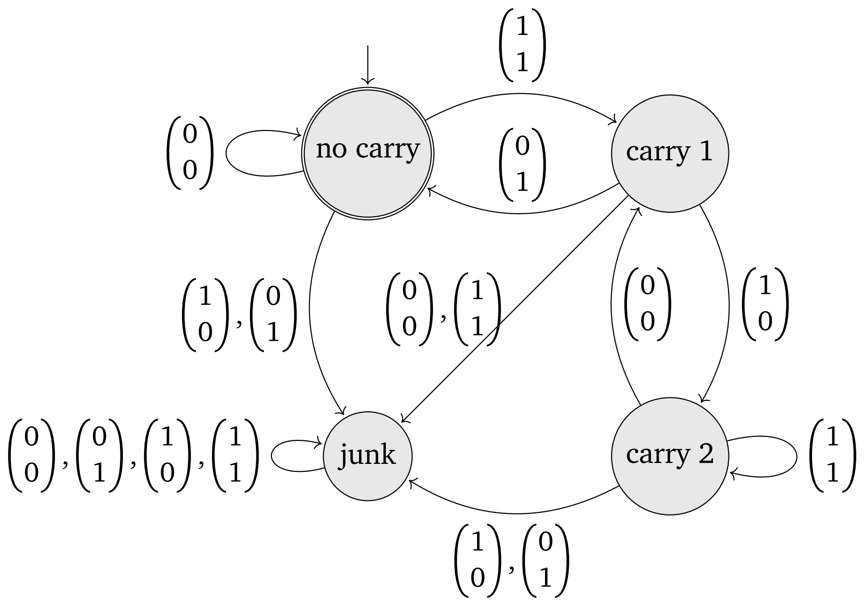
$$[\![\begin{pmatrix}1\\1\end{pmatrix}\begin{pmatrix}0\\1\end{pmatrix}\begin{pmatrix}0\\1\end{pmatrix}\begin{pmatrix}0\\0\end{pmatrix}\begin{pmatrix}0\\1\end{pmatrix}]\!] = (1,\, 23)$$

Construct a DFA to recognise the language of "multiplication by 3":

$$\{w \in \Sigma^* \mid \exists m \in \mathbb{N}.\ [\![w]\!] = (m,\, 3*m)\}$$

Solution

To answer this question, it is best to think of multiplying by three as doing three binary additions of the given number in one go. This will require an extended notion of "carry bit".



no carry
carry 1
junk
carry 2
$\binom{0}{0}$
$\binom{1}{1}$
$\binom{0}{1}$
$\binom{1}{0}, \binom{0}{1}$
$\binom{0}{0}, \binom{1}{1}$
$\binom{0}{0}$
$\binom{1}{0}$
$\binom{0}{0}, \binom{0}{1}, \binom{1}{0}, \binom{1}{1}$
$\binom{1}{1}$
$\binom{1}{0}, \binom{0}{1}$