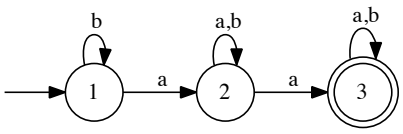
Informatics 1 - Computation & Logic: Tutorial 6 Solutions

Computation: Non-deterministic FSMs and FSM composition

Week 8: 4-8 November 2013

1. Consider the Finite State Machine in the diagram:



   (a) Draw its transition table:

	a	b
1	$\{2\}$	$\{1\}$
2	$\{2,3\}$	$\{2\}$
3	$\{3\}$	$\{3\}$

   (b) Is it a deterministic or non-deterministic FSM (N-FSM)?

   It is a non-deterministic FSM because there are two possible transitions from node 2 given the a input symbol.

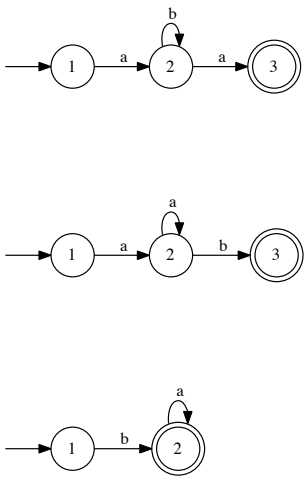
2. Sometimes it is useful to combine simple FSMs to obtain one that can accept more complex languages. Take the following language:

$$(ab^*a|aa^*b)ba^*$$

   (a) What is the meaning in words of the formula?

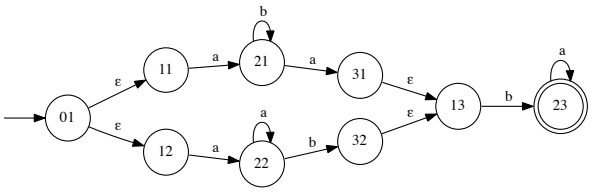
   The sequence ab^*a or the sequence aa^*b, followed by the sequence ba^*, where the $*$ symbol means that the input letter can be repeated any number of times, including 0 times.

(b) Draw the simple FSMs that accept the subparts of the formula:



Where the first accepts ab^*a, the second aa^*b, and the third ba^*.

(c) Compose them to obtain an N-FSM able to accept the language in the formula:



The two N-FSMs for ab^*a and aa^*b are in parallel with four empty transitions ϵ, followed by FSM for ba^*.

3. Draw a finite state acceptor which accepts an infinite set of strings each of which consists of an *odd number of* 0*s* and an *even number of* 1*s greater than* 0. To be more precise, if string s is accepted by the machine, then s must:

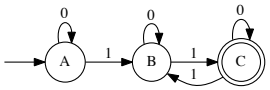
   (a) consists of an odd number of symbols;

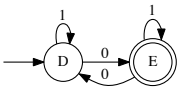
   (b) contain only 0s and 1s;

   (c) contain an odd number of 0s; and,

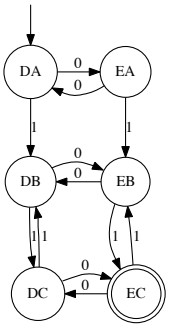
   (d) contain an even number of 1s greater than 0.

   To create such acceptor, first create an acceptor for the even number of 1s, then another acceptor for the odd numbers of 0s, and then put the two acceptors together using the correct operator. Motivate your choice of the operator.





The first accepts an even number of 1s greater than 0, the second an odd number of 0s.



Intersection: a string x will be accepted by the intersection of the two machines only if it is accepted by both machines:

- 1100 is accepted by the first (even 1s), but not by the second (odd 0s)
- 1000 is accepted by the second, not by the first
- 11000 is accepted by both, and it is accepted by the intersection

4. (a) Do D-FSMs and N-FSMs recognise the same languages?

Yes, they recognise the same class of languages, the class of *regular languages*.

(b) What advantages do N-FSMs have over D-FSMs, and vice-versa?

The determinism of D-FSMs is important when we are implementing a program for recognising a regular language, whereas the reverse is true for N-FSMs - their nondeterminism makes them difficult to implement. However, N-FSMs have the advantage that they are easier to build, especially when modelling real systems (with inherent nondeterminism). As we have seen, we can build build complex N-FSMs in a simple piecewise fashion — fitting smaller FSMs together.

This tutorial exercise sheet was written by Paolo Besana, and extended by Thomas French and Areti Manataki. Send comments to A.Manataki@ed.ac.uk