



## **Assignment 3**

Tim Neutze

5578777

Lorenzo Tecchia

5581906

2023.05.05

# Contents

<b>1</b>	<b>Tasks 4 (e-g-h-i)</b>	<b>4</b>
1.1	e) . . . . .	4
1.2	g) . . . . .	5
1.3	h) . . . . .	5
1.4	i) . . . . .	6
<b>2</b>	<b>Tasks 5 (c-d)</b>	<b>7</b>
<b>3</b>	<b>Task 6</b>	<b>8</b>
<b>4</b>	<b>Task 8 (c)</b>	<b>9</b>

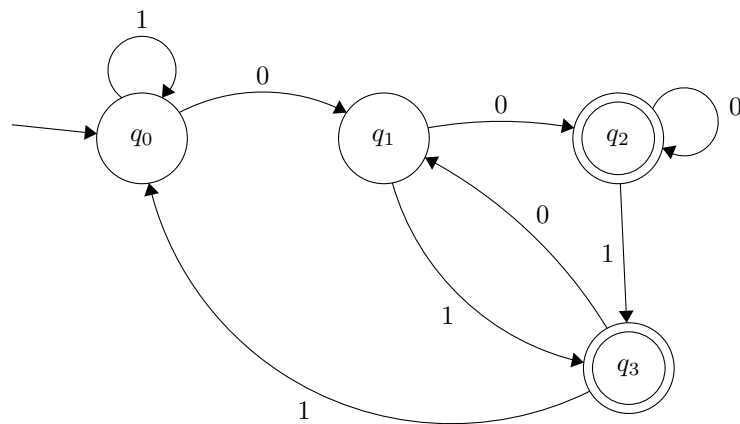
# List of Figures

3.1 Task6 . . . . .	8
---------------------	---

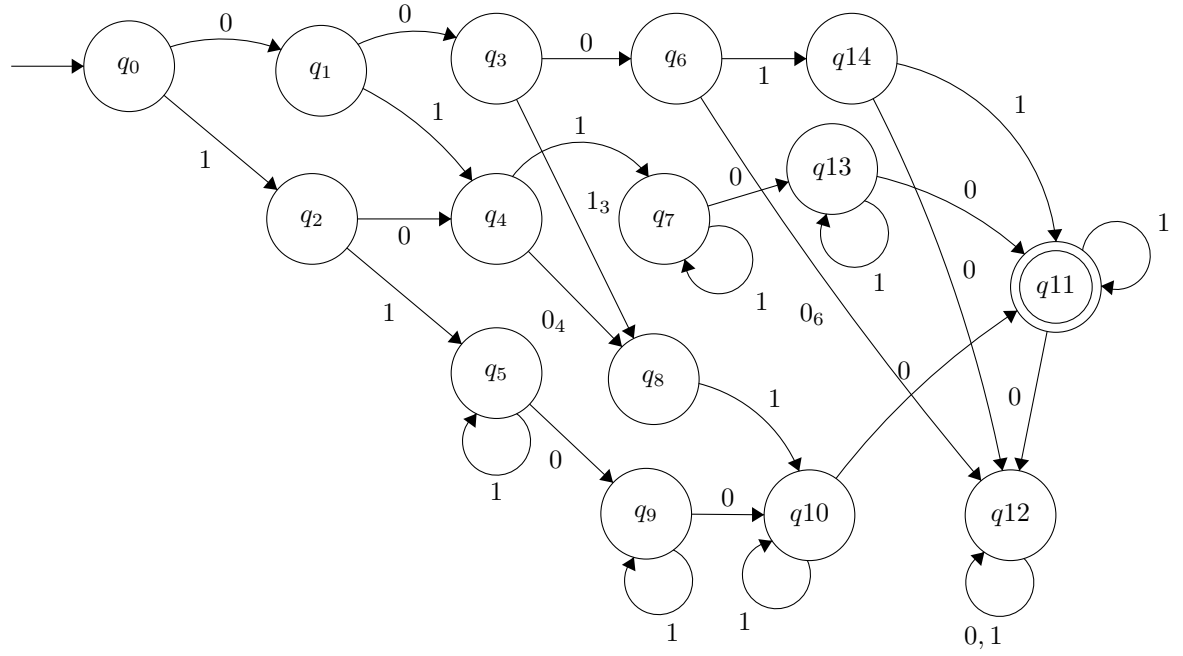
# Chapter 1

## Tasks 4 (e-g-h-i)

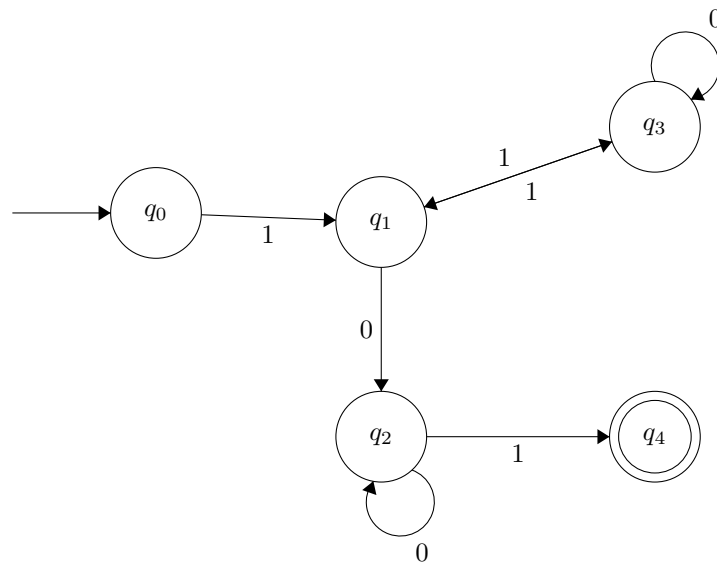
1.1 e)



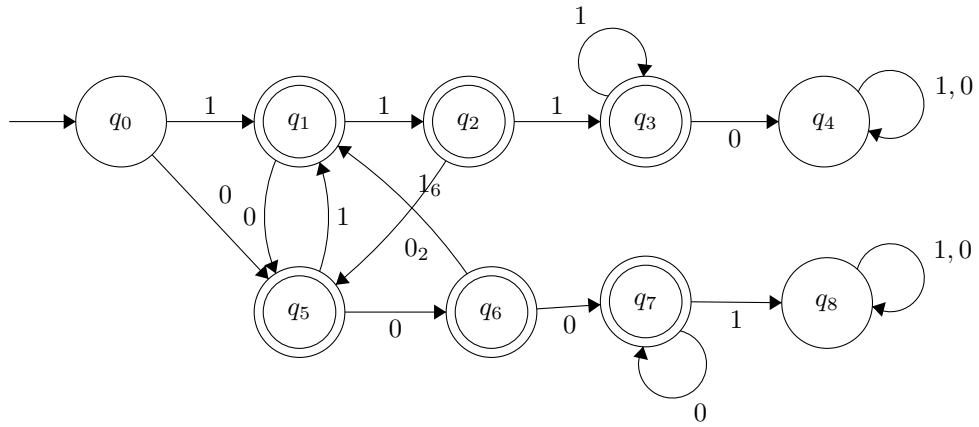
1.2 g)



1.3 h)



1.4 i)



## Chapter 2

### Tasks 5 (c-d)

- Let's first prove the closure under concatenation of regular languages. Let  $L_1$  and  $L_2$  be arbitrary regular languages. Because they are regular languages, we know there are minimal DFAs for  $L_1$  and  $L_2$ ; let's call these M1 and M2, respectively.

To see that the concatenation of these languages must be regular, construct a machine  $M^*$  as follows:

- the states of  $M^*$  are the states of M1 and M2 put together
- the alphabet of  $M^*$  is the union of the alphabets of M1 and M2
- initial state of  $M^*$  is the initial state of M1
- accepting states of  $M^*$  are the accepting states of M2
- $M^*$  has all the same transitions as M1 and M2 put together, plus empty/epsilon/lambda transitions from all the accepting states in M1 to the initial state of M2

This defines an NFA-lambda (NFA with empty/lambda/epsilon transitions). We know those are equivalent to DFAs and all DFAs can be minimized; let us call the equivalent minimal DFA  $M^{**}$ .

Because there is a minimal DFA for the concatenation of  $L_1$  and  $L_2$ , the concatenation must be regular.

Having proved that, we could see doubling each letter in each word of the language  $L_1$ , to form the language  $L_2$  as the concatenation of  $L_1$  on itself; thus proving that  $L_2$  is also a regular language.

- we have already proved the closeness of concatenation in the previous point.[1]

## Chapter 3

### Task 6

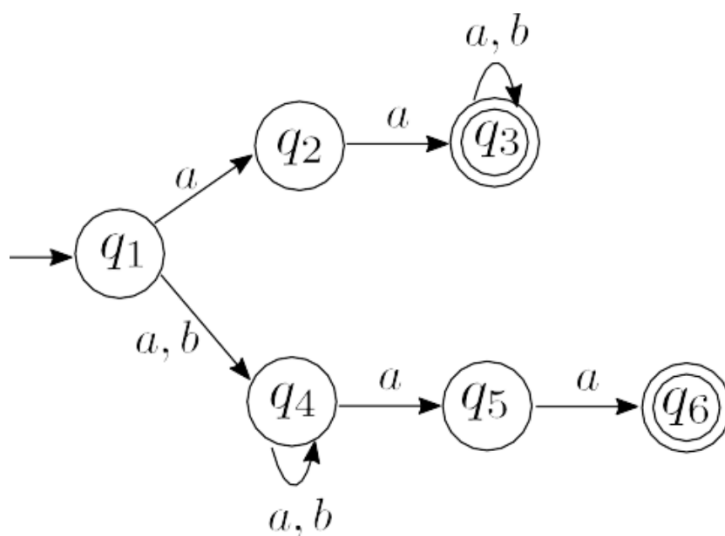


Figure 3.1: Task6

The Figure shown above, represent a NFA that accepts the language composed by the alphabet  $a, b, \epsilon$  where all strings accepted are the ones that contain the substring: "aa".

The reason being the fact that NFA when given an input that would be processed by multiple state, like in this case the input  $a$  would be accepted simultaneously by state  $q_2$  and  $q_4$ ; the machine would be split up in two different machine and if even one of the multiple machine in which an NFA would split accept the given input then, the input is accepted.



## Chapter 4

### Task 8 (c)

If the positive integers and decimal would be represented by the formal alphabet of  $\Sigma = 0, 1$  with 1 being the positive integers and 0 being the decimals, then the sum would be described by:

$$1^*0^* \cup 1^*0^*$$

# Bibliography

Sipser, Michael. *Introduction to the theory of computation*. 2nd ed. Boston: Thomson Course Technology, 2006. ISBN: 0534950973. URL: <http://www.loc.gov/catdir/enhancements/fy1103/2006271030-b.html>.