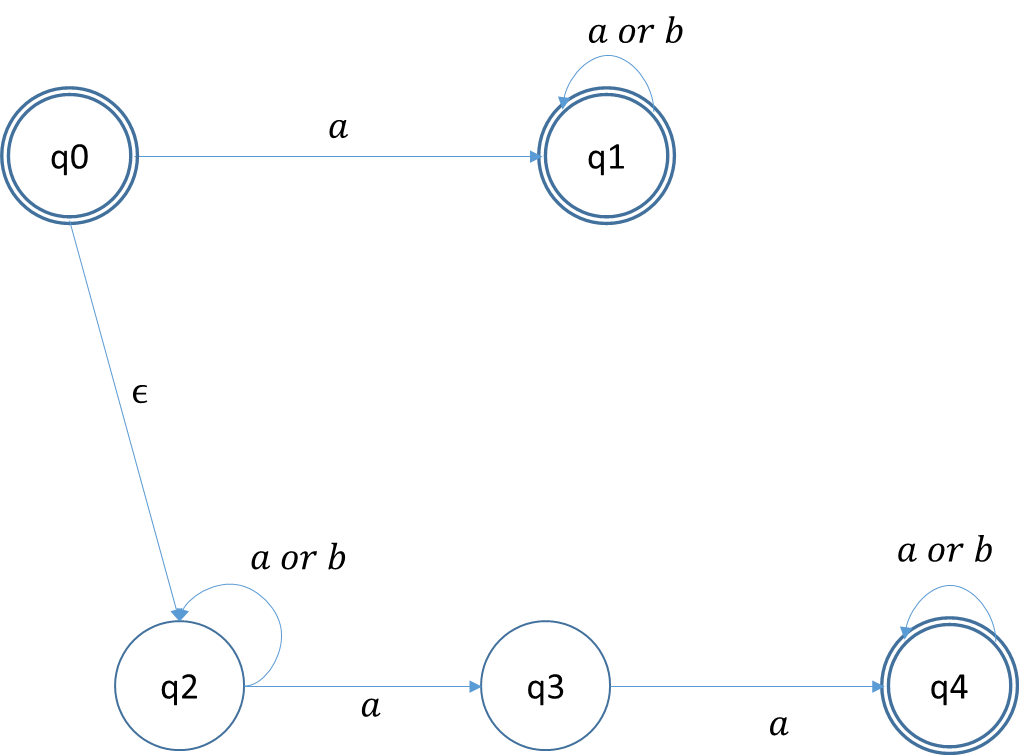
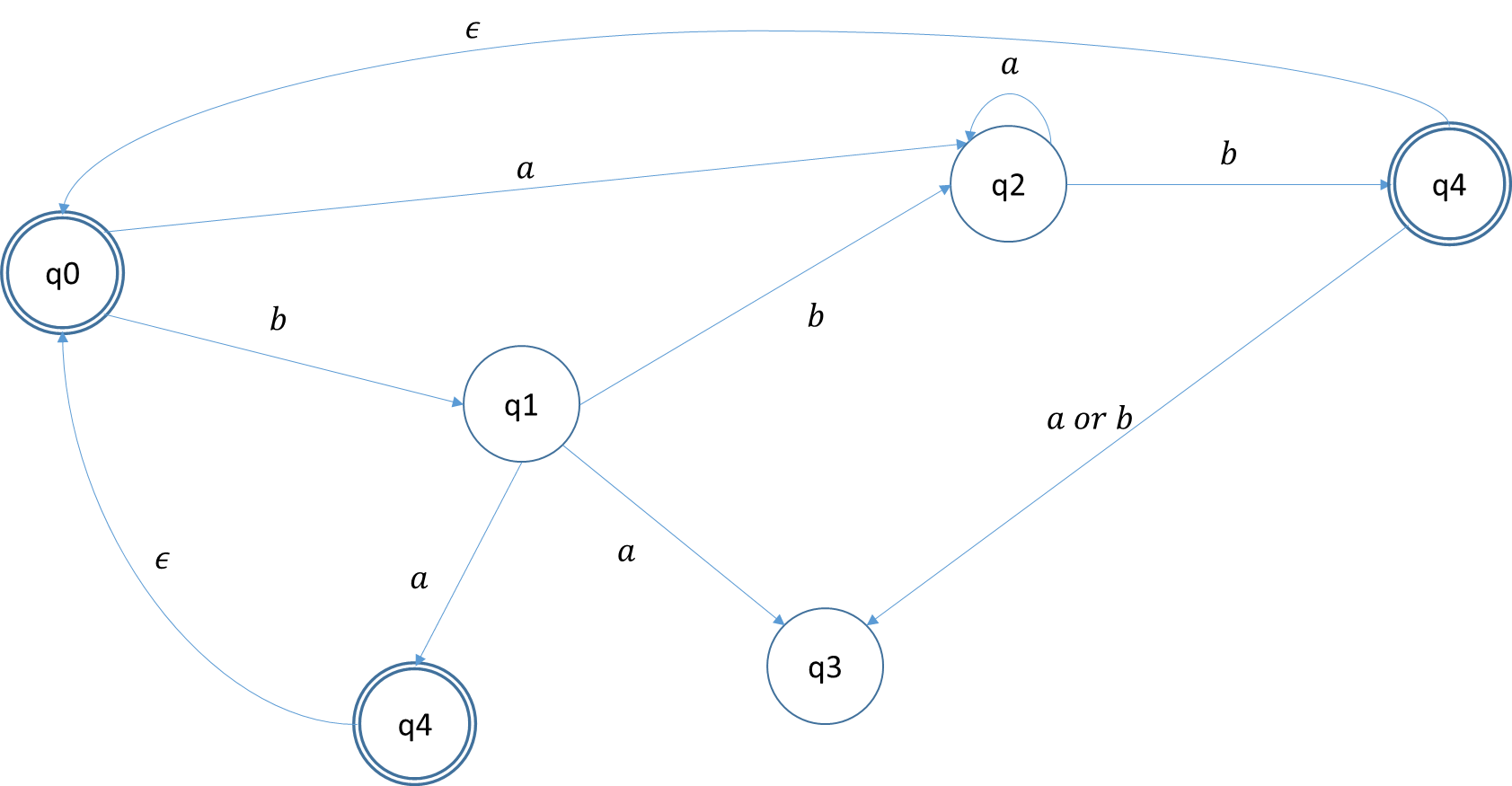
# Question 1

# Question 2







# Question 3

In fact the only condition for it to work is to satisfies this expression which correspond to having at least an

Let take a string that satisfies the expression

Let split this string in two pieces with n in and in and

If then we are done the string is in the set

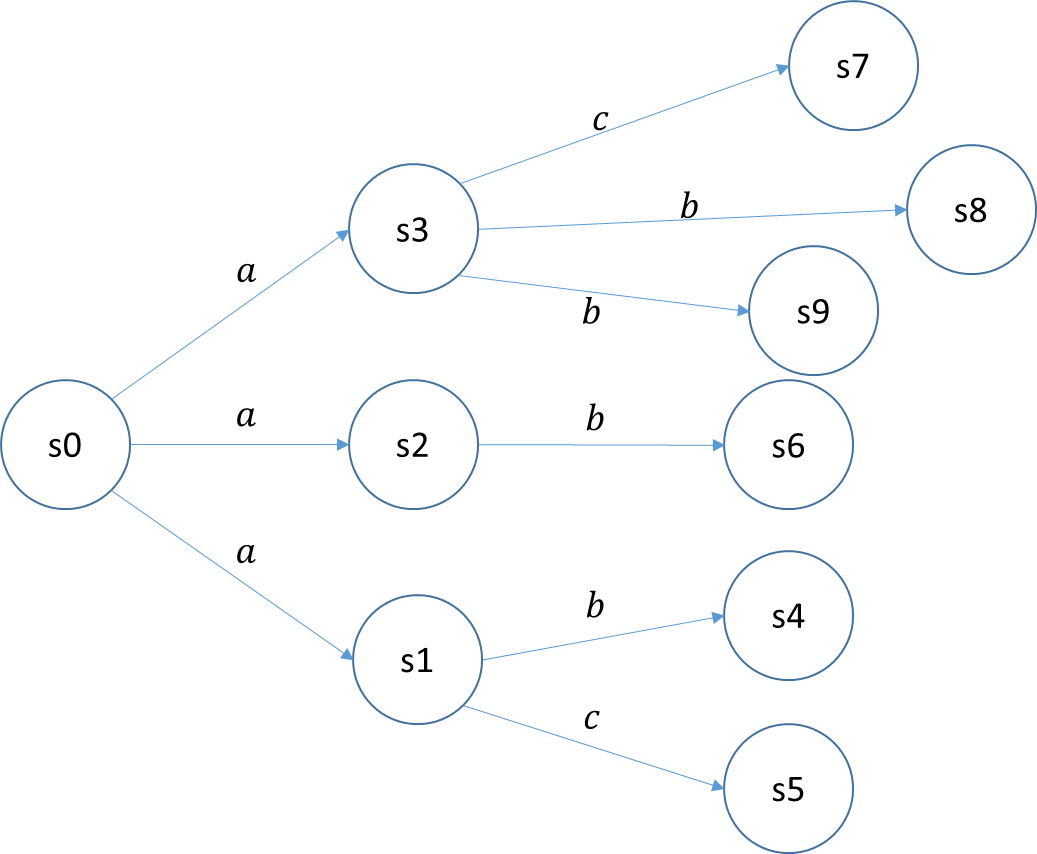
Let suppose

Then we can move the split of the string one letter further. If the letter is then we have and still , if it’s a b then we have still and Then we check if the amount are the same now, if not then we can repeat this step until they match

Similarly with we move the split left

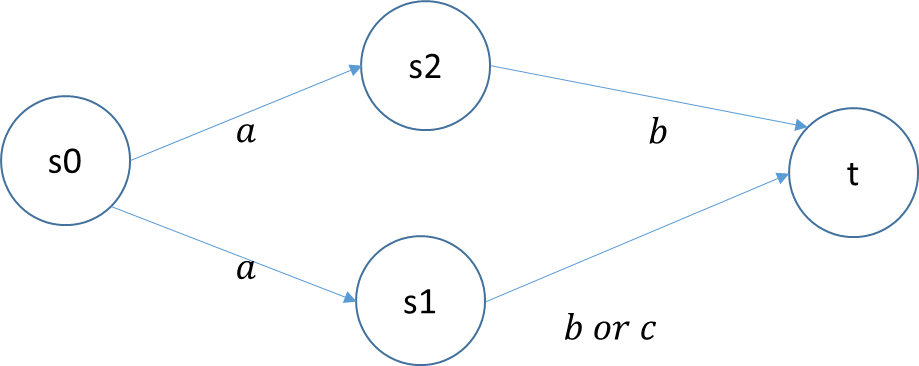
Then this expression define this language and thus it’s a regular language.

# Question 4



We see that and are bisimilar as we have the same outgoing transition (b and c)

We also have to bisimilar as they don’t have any outgoing transitions



# Question 5

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
|  | Eq |  |  |  |  |
|  |  | Eq |  |  |  |
|  |  |  | Eq |  |  |
|  |  |  |  | Eq |  |
|  |  |  | Eqq |  | Eq |

Nothing starts with b that is in . All equivalent classes on are equivalent to themselves. is equivalent to for any .

The only equivalent classes in are on

Because there are infinite equivalence classes as there is an equivalent classes for each where , is distinct from itself