Student Information

Full Name : Yavuz Selim Yesilyurt
Id Number : 2259166

Answer 1

a.

Let us denote every number that the set of rational numbers inside the open interval $(-1, 0)$ contains n as $n = \frac{a}{b}$. We can write the elements of this set as;

$a \setminus b$	1	2	3	4	...
-1	$-\frac{1}{1}$	$-\frac{1}{2}$	$-\frac{1}{3}$	$-\frac{1}{4}$	...
-2	$-\frac{2}{1}$	$-\frac{2}{2}$	$-\frac{2}{3}$	$-\frac{2}{4}$	...
-3	$-\frac{3}{1}$	$-\frac{3}{2}$	$-\frac{3}{3}$	$-\frac{3}{4}$	...
-4	$-\frac{4}{1}$	$-\frac{4}{2}$	$-\frac{4}{3}$	$-\frac{4}{4}$	...
...	...	...	...	...	...

In table, first row, above the horizontal line contains the countably infinite elements of set $\mathbb{N}$ except 0, starting from 1 and the first column, left of vertical line contains the countably infinite elements of set Z^- except 0. The entries in the upper half (triangular) of this table (above the diagonal, diagonal is exclusive) illustrates the numbers $n = \frac{a}{b}$ that is in this set. They can be counted reverse diagonally starting with $-\frac{1}{2}$ (i.e. 1st element: $-\frac{1}{2}$, 2nd element: $-\frac{1}{3}$, 3rd element $-\frac{2}{3}$, 4th element: $-\frac{1}{4}$, 5th element: $-\frac{1}{5}$, 6th element: $-\frac{2}{4}$, 7th element: $-\frac{3}{4}$, 8th element: $-\frac{2}{5}$, 9th element: $-\frac{1}{6}$ and so on). With this way of counting we will eventually get all elements in the set, which we can index them with unique numbers from $\mathbb{N}$ as $0 \rightarrow -\frac{1}{2}$, $1 \rightarrow -\frac{1}{3}$, $2 \rightarrow -\frac{2}{3}$, $3 \rightarrow -\frac{1}{4}$ ...

Therefore we can get the one-to-one correspondence between the set of rational numbers inside the open interval $(-1, 0)$ and $\mathbb{N}$. Thus we can infer that this set is countably infinite.

b.

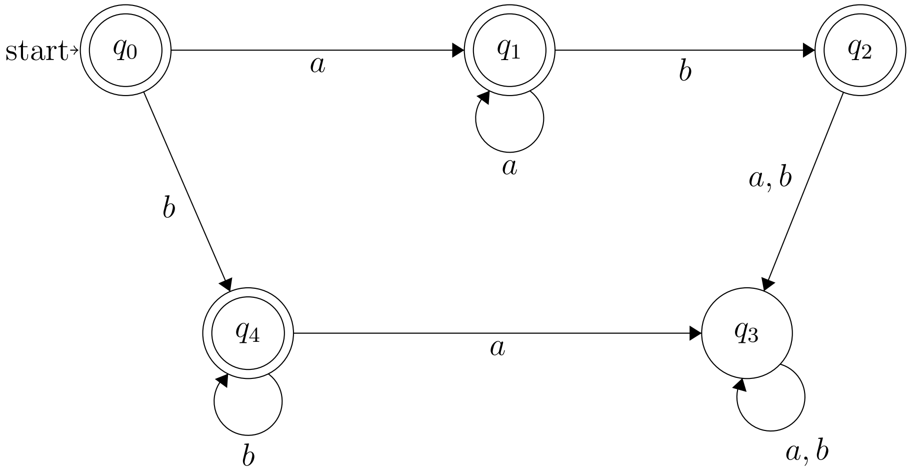
We know that the language L over the unary alphabet $\sum = \{a\}$ is regular with the properties of regular expression since it is finite. We also know that $L^+ = LL^*$. Since the class of regular languages is closed under Kleene Star, L^* is also regular and furthermore since the class of regular languages is also closed under concatenation, $L^+ = LL^*$ is also regular. This makes the set D an empty set since in its definition it is also stated that L^+ is not regular, but this contradicts with the fact that L^+ is regular. Therefore D is countable since it (empty set) is a subset of $\mathbb{N}$.

c.

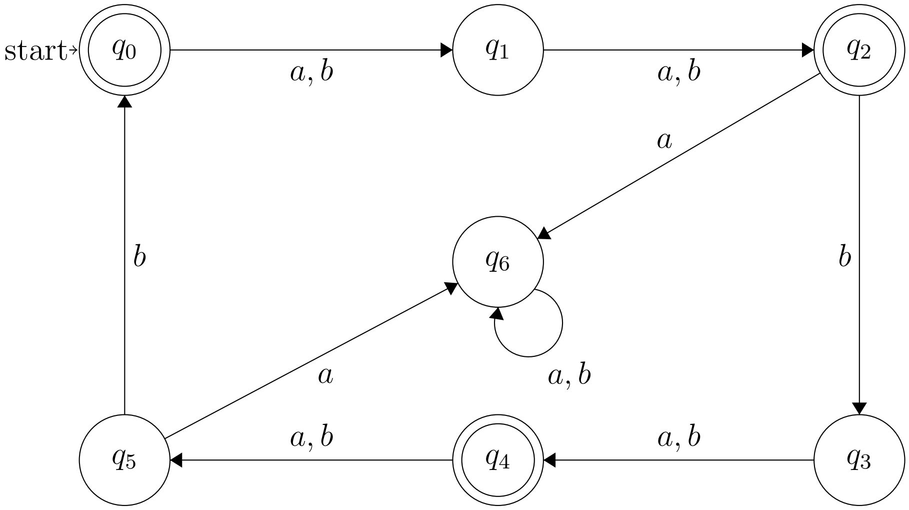
The set of all languages on the binary alphabet $\sum = \{a, b\}$ which cannot be recognized by any Finite Automation is uncountable since any language that cannot be accepted by a Finite Automation is not regular and the set of not regular languages is uncountable.

Answer 2

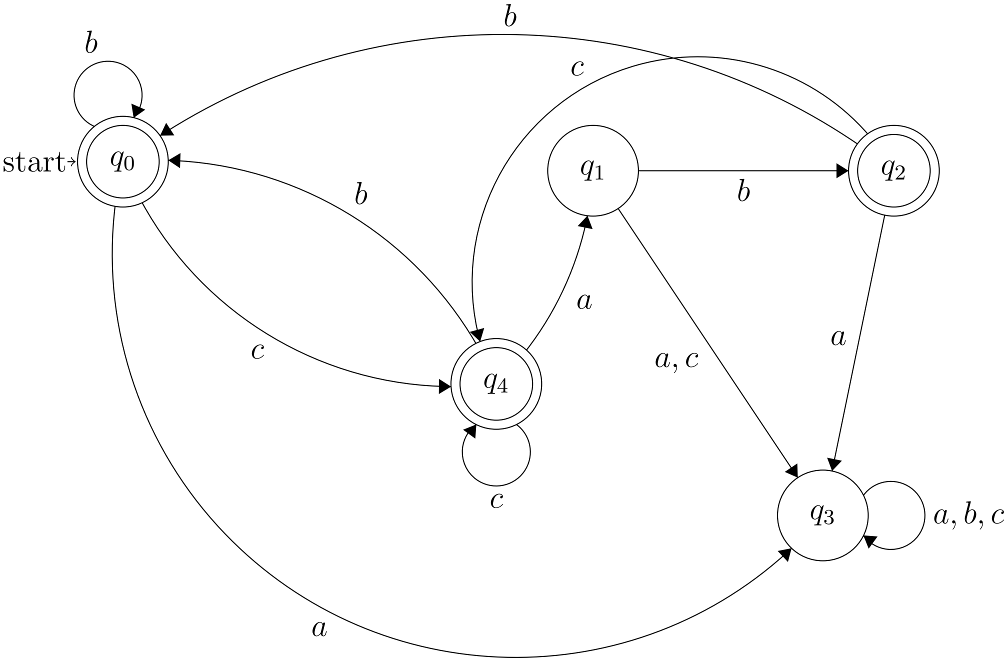
a.



b.


start
q_0
q_1
q_2
q_3
q_4
q_5
q_6
a, b
a, b
a
b
b
a
a, b
a, b
a, b

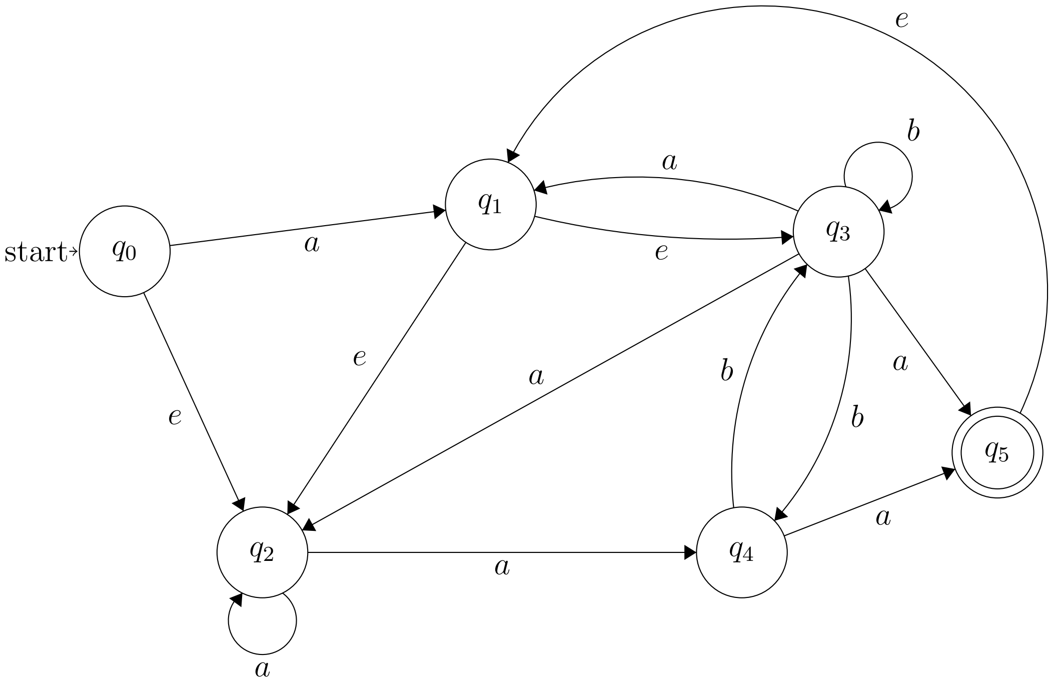
c.


b
b
c
start
q_0
q_1
q_2
b
b
a
c
a, c
a
q_4
c
q_3
a, b, c
a

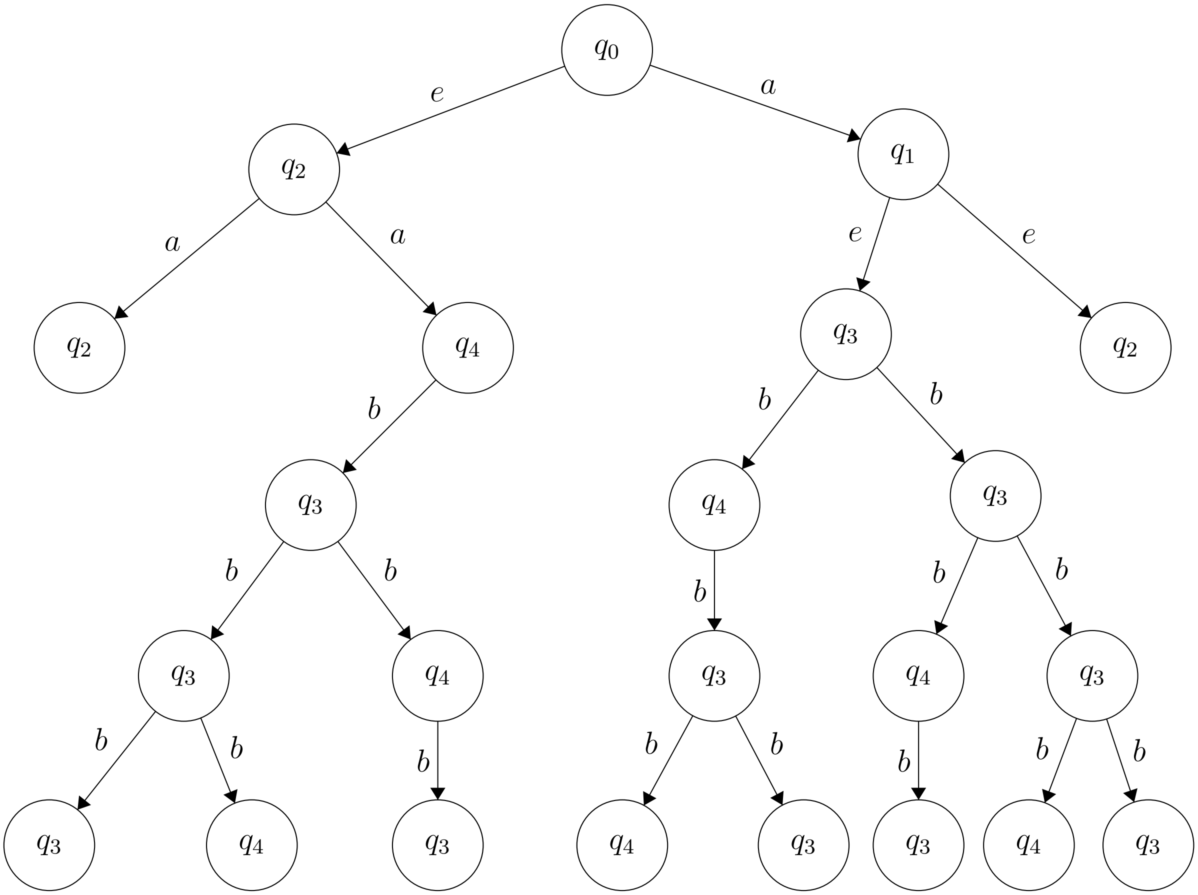
Answer 3

a.

We have an NFA N such that:



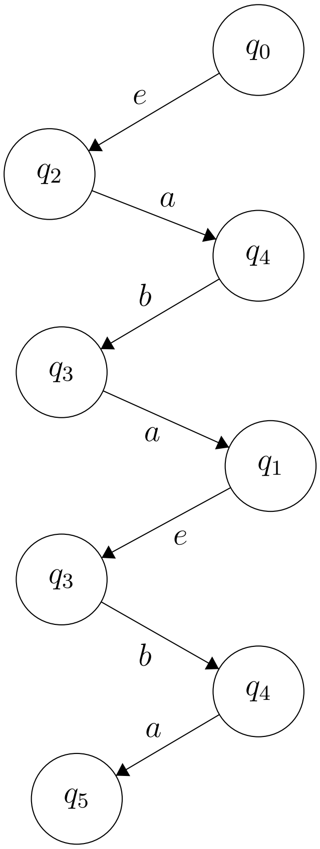
To check if $w_1 = abbb$ is in $L(N)$ or not we need this NFA to read w_1, namely machine should be on its final state (q_5) when the string is read and no letter to be read is left. Any path which leads to leaf which has an accepting state, means the string is in the language of that NFA. Machine must backtrack to its nondeterministic choice when it fails on any of its choices to read the string. We can demonstrate this computation of the machine with a computational tree:



As we can see from the computational tree figure above there is not any leaf in which machine has reached its final state q_5, hence we can infer that the string $w_1 \notin L(N)$.

b.

Same as part a, we will use the same NFA to decide if $w_2 = ababa$ is in $L(N)$ or not. Let us make the computation with the computational tree:

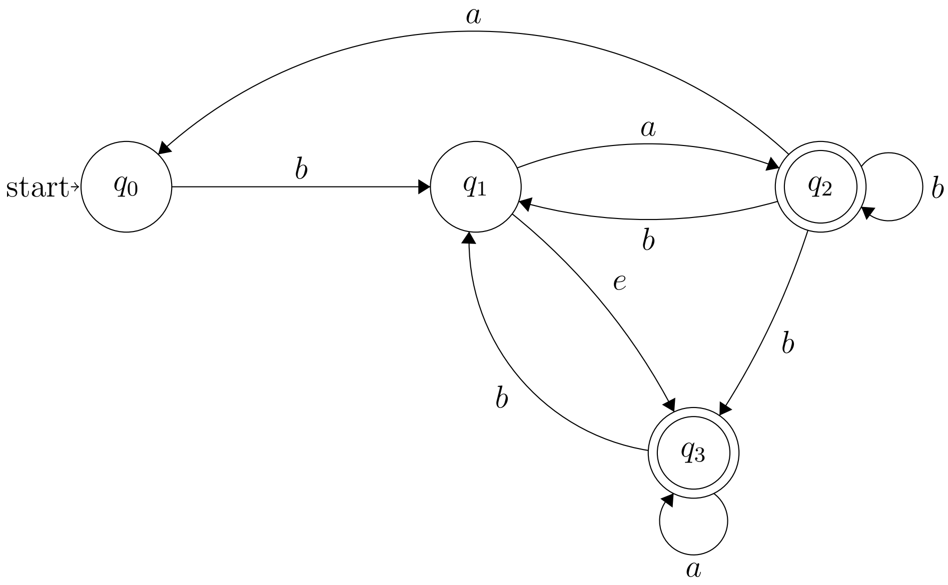


As we can see from the computational tree path figure above there is a configuration of states which leads to final state q_5 on the leaf which allows machine to read the string w_2, hence we can infer that the string $w_2 \in L(N)$.

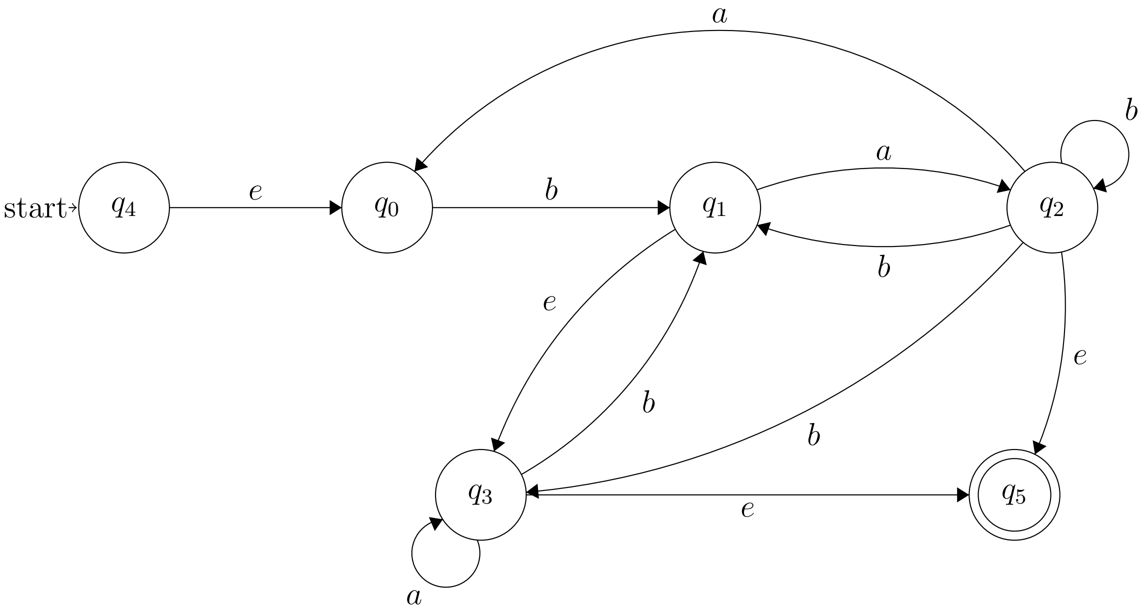
Answer 4

a.

We have an NFA (N) such that:

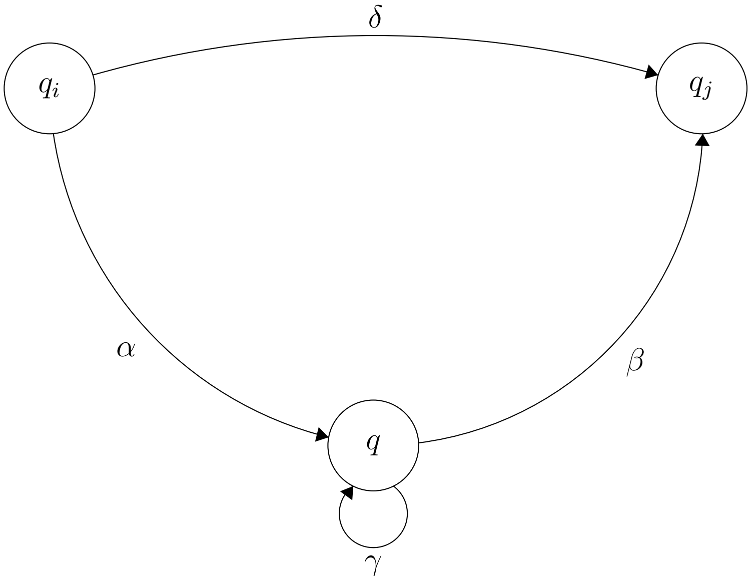


To be able to specify N as a GFA and to apply the state elimination on N, it must have only 1 final state with no outgoing transitions and its initial state must not have any incoming transitions. Therefore we add 2 more steps with e transitions to/from the old initial/final states as the new initial and final step respectively. The GFA for N:

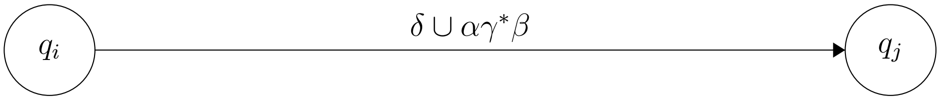


b.

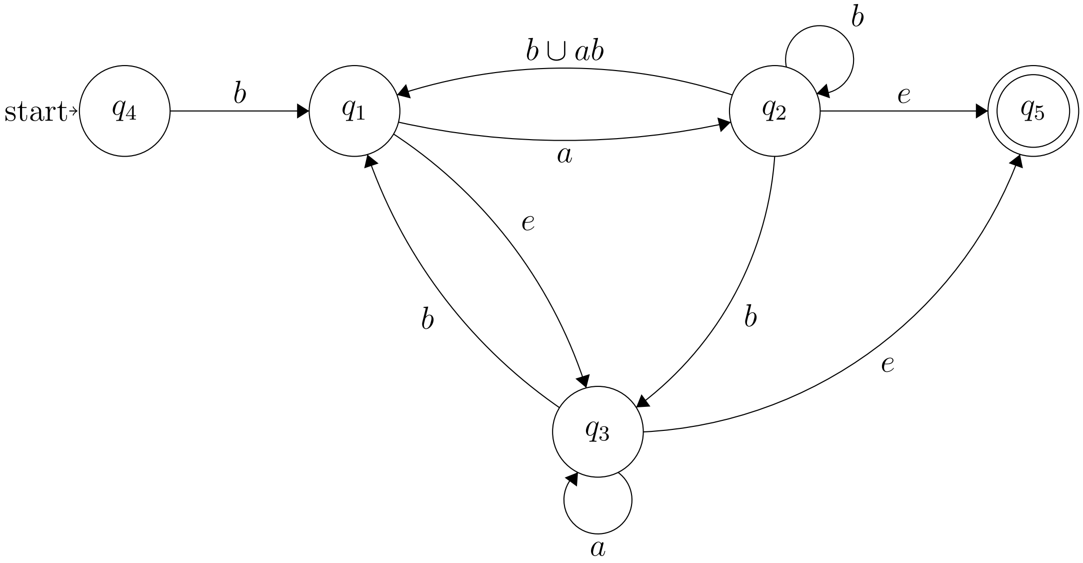
While eliminating states to convert GFA into a regular expression we will consider the following triangular (rule):



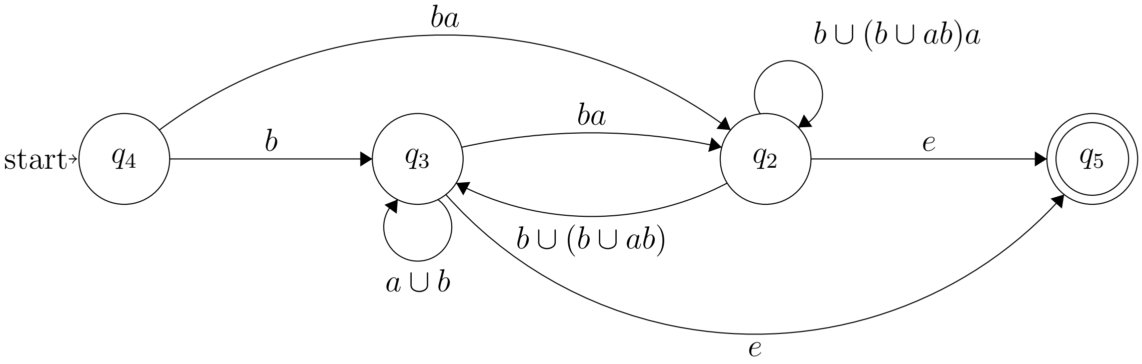
Eliminate q:



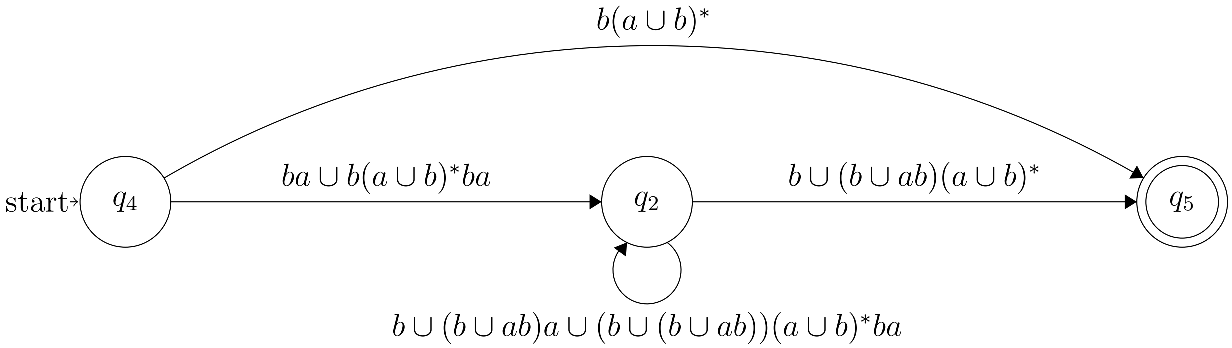
So let us start state elimination with eliminating q_0:



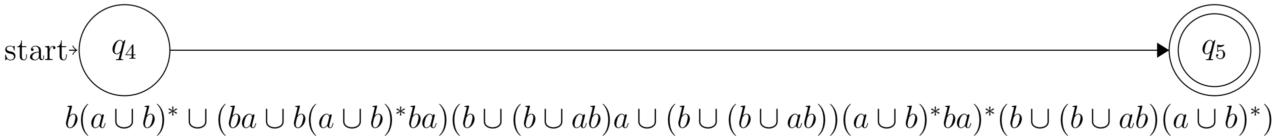
Now eliminate q_1:



Now eliminate q_3:


$b(a \cup b)^*$
start
q_4
$ba \cup b(a \cup b)^* ba$
q_2
$b \cup (b \cup ab)(a \cup b)^*$
q_5
$b \cup (b \cup ab)a \cup (b \cup (b \cup ab))(a \cup b)^* ba$

Now finally eliminate q_2:


start
q_4
q_5
$b(a \cup b)^* \cup (ba \cup b(a \cup b)^* ba)(b \cup (b \cup ab)a \cup (b \cup (b \cup ab))(a \cup b)^* ba)^*(b \cup (b \cup ab)(a \cup b)^*)$

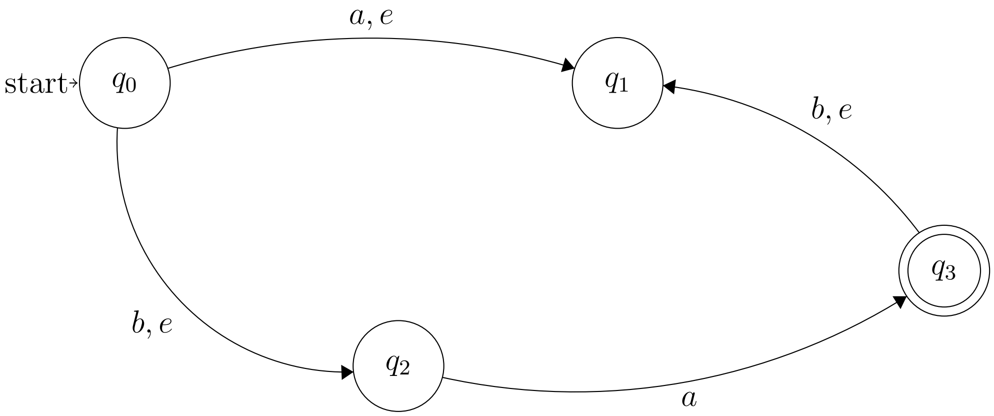
Therefore the converted form of GFA into regular expression is:

$$b(a \cup b)^* \cup (ba \cup b(a \cup b)^* ba)(b \cup (b \cup ab)a \cup (b \cup (b \cup ab))(a \cup b)^* ba)^*(b \cup (b \cup ab)(a \cup b)^*)$$

Answer 5

a.

We have an NFA N such that:



To find an equivalent DFA M such that $L(M) = L(N)$ we will use subset construction algorithm. Firstly, Since N has 5 states, M will have $2^5 = 32$ states. However, only a few of these states will be relevant to the operation of M-namely those states that can be reached from state s' by reading some input string. Obviously, any state in K' that is not reachable from s' is irrelevant to the operation of M and to the language accepted by it. We shall build the reachable part of N by starting from s' and introducing a new state only when it is needed as the value of $\delta(q, x)$ for some state $q \in K'$ already introduced and some $x \in \Sigma$.
So let's define every $E(q)$ for each state q of N:

$$E(q_0) = \{q_0, q_1, q_2\}$$
$$E(q_1) = \{q_1\}$$
$$E(q_2) = \{q_2\}$$
$$E(q_3) = \{q_1, q_3\}$$

Since $s' = E(q_0) = \{q_0, q_1, q_2\}$, $(q_0, a, q_1), (q_2, a, q_3)$ are all transitions (q, a, p) for some $q \in s'$. It follows that

$$\delta'(s', a) = E(q_1) \cup E(q_3) = \{q_1, q_3\}.$$

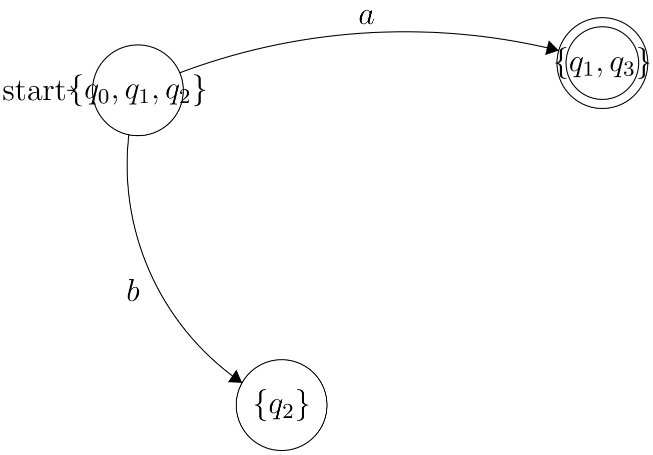
Similarly,

$$(q_0, b, q_2)$$

is the transition from (q, b, p) for some $q \in s'$, so

$$\delta'(s', b) = E(q_2) = \{q_2\}.$$

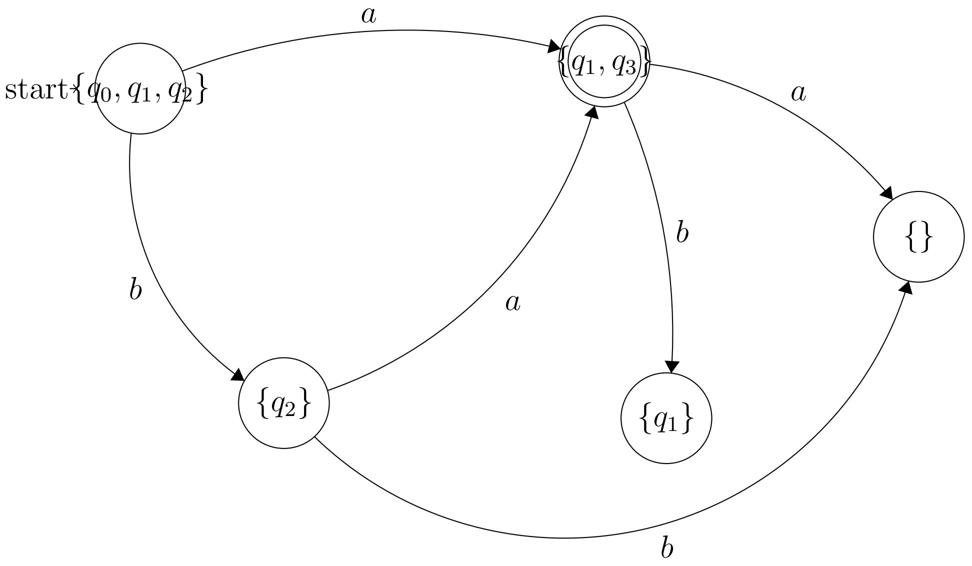
After this operations we have:



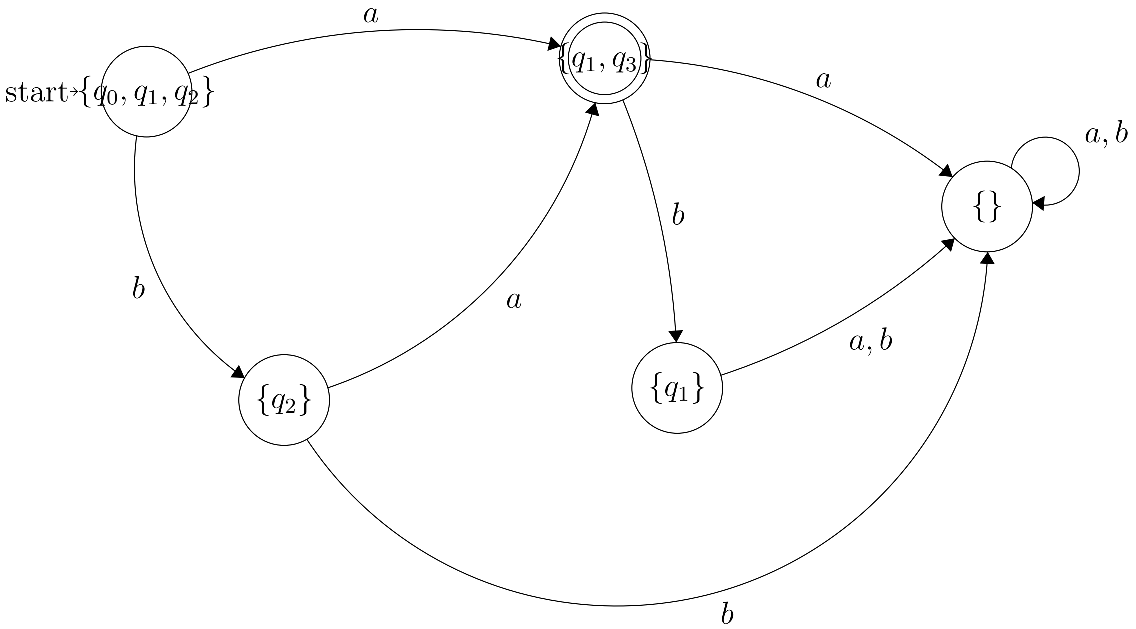
Repeating this calculation for the newly introduced states, we have the following;

$$\begin{aligned}
\delta'(\{q_1, q_3\}, a) &= \emptyset = \{\} \\
\delta'(\{q_1, q_3\}, b) &= E(q_1) = \{q_1\} \\
\delta'(\{q_2\}, a) &= E(q_3) = \{q_1, q_3\} \\
\delta'(\{q_2\}, b) &= \emptyset = \{\} \\
\delta'(\{q_1\}, a) &= \emptyset = \{\} \\
\delta'(\{q_1\}, b) &= \emptyset = \{\}
\end{aligned}$$

The relevant part of M will be our new DFA. F', the set of final states, contains each set of states of which q_3 is a member, since q_3 is the sole member of F; so all states which contains q_3 will be the final states. Step by step our new DFA:

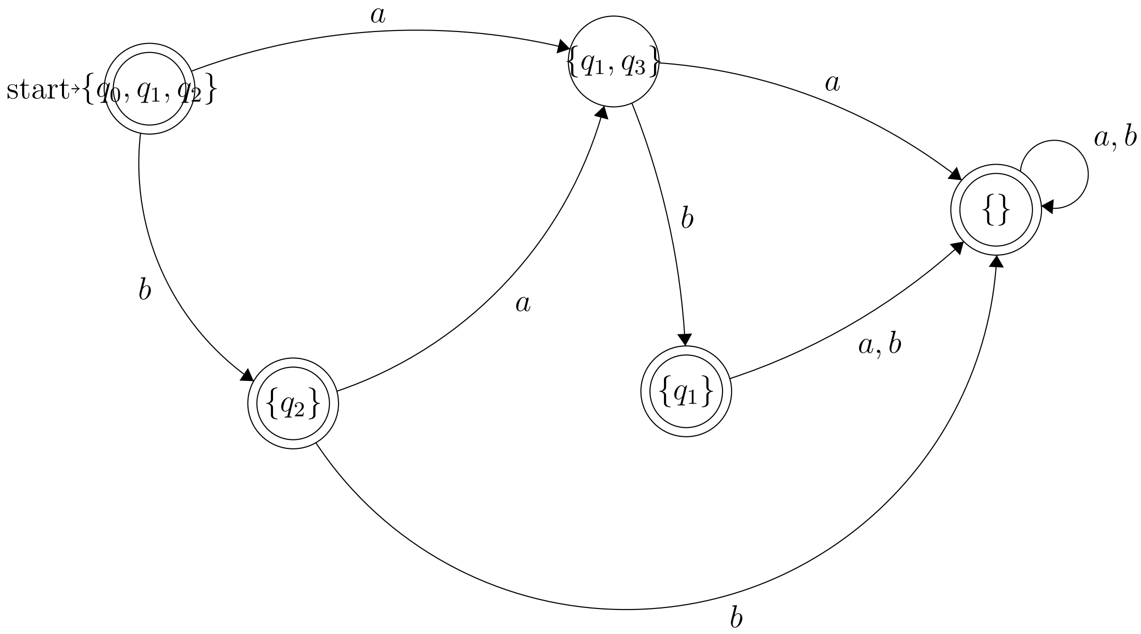


And finally:



b.

The language $\bar{L}$ where $L = L(N)$ (complementation algorithm) requires N to be an equivalent DFA, therefore we will use the equivalent DFA M which we found on part a by subset construction algorithm. First let us demonstrate $\bar{L}$ by swapping all final states and non-final states:



In order to find a regular expression for the language $\bar{L}$ where $L = L(N)$ (we will use the DFA equivalent of N which is M), we need to generate a GFA for the automaton and then step by step convert it to a regular expression as we did in Question 4. By doing this, at the end there will only be two nodes and one transition, where the transition's expression will be our regular expression, which is:

$$(b \cup (ab \cup bab)) \cup ((aa \cup b(b \cup aa)) \cup (((ab \cup bab)(a \cup b))(a \cup b)^*)$$

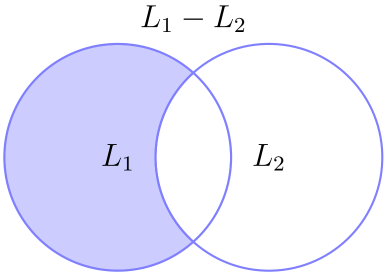
Since we took the complement of language L to construct the language $\bar{L}$, the complementation property of the language is using set notation and we write it as:

$$\bar{L} = \{w \in \{a, b\}^* : w \in U \ and \ w \notin L\}$$

where U is the set of all languages that can be composed by any number of the symbols over the alphabet $\Sigma = \{a, b\}$

Answer 6

Let L_1 and L_2 two arbitrary regular languages and let $M_1 = (K_1, \Sigma, \Delta_1, s_1, F_1)$ and $M_2 = (K_2, \Sigma, \Delta_2, s_2, F_2)$ NFA's of the languages L_1 and L_2 respectively. To prove that class of regular languages is closed under set difference operation we will construct an NFA for the language $L_1 - L_2$. This language will have an automata that accepts strings when L_1 accepts and L_2 does not. This operation corresponds to $(\bar{L_1} \cup L_2)'$ and that is equivalent to $L_1 \cap \bar{L_2}$. Since the class of regular languages are closed under complementation, union and intersection, the set difference operation is also closed and reachable from these basic set of operations. We can easily observe that $L_1 - L_2 = L_1 \cap \bar{L_2} = (\bar{L_1} \cup L_2)'$ by the Venn Diagram of the language $L_1 - L_2$ below:



To construct the NFA for the language we can use the algorithms defined for complementation and union of languages since we require them to construct the difference of two language. First start with taking complement of L_1. To get complement of a FSA we need to have a DFA, therefore we actually start with converting M_1 to its equivalent DFA by subset construction algorithm (as we did in Question 5.a to find an equivalent DFA of given NFA), and get $M_1' = (K_1', \Sigma, \delta_1, s_1', F_1')$. Now swap final and non-final states: $\bar{M_1'} = (K_1', \Sigma, \delta_1, s_1', K_1' - F_1')$.

Secondly we have to make a union operation with $L(\bar{M_1'})$ and $L(M_2)$ to get $\bar{L_1} \cup L_2$. Use the union construcion algorithm to achieve the union between these two languages. Which corresponds to creating a new initial state q_x, binding it to $\bar{M_1'}$'s and M_2's initial state with e transitions and getting $N = (K_N, \Sigma, \Delta_N, q_x, F_N)$ where N is the NFA of the language $\bar{L_1} \cup L_2$.

Now convert this NFA to its equivalent DFA with again subset construction algorithm and get $N' = (K_N', \Sigma, \delta_N', s_N', F_N')$. Finally take the complement of it by swapping the final and non-final states and get $\bar{N'} = (K_N', \Sigma, \delta_N', s_N', K_N' - F_N')$.

The language that is accepted by this DFA (trivially NFA) $\bar{N'}$ is $L_1 \cap \bar{L_2}$ and that language is equivalent to $L_1 - L_2$. Since $\bar{N'}$ is constructed above with the algorithms defined for union and complementation operations (in which class of regular languages are closed) the language it represents $(L_1 - L_2)$ is also regular which shows that regular languages are closed under the set difference operation.

Answer 7

We are given the language $L = \{w \in \{a,b\}^* : f(a,w) = n^2 \text{ for some } n \in \mathbb{N}\}$ in which f is a function defined as $\Sigma \times \Sigma^* \to \mathbb{N}$ where Σ is a finite alphabet, returning the number of occurrences of its first parameter within its second parameter.

We know for sure that a will occur n^2 times in w for some $n \in \mathbb{N}$, but we do not have any information about its order in w and same (order issue) is present for symbol b as well, we do not have any information regarding its order in w, furthermore for b, we also do not know its number of occurence in w (we can simply represent this with Kleene Star *). Therefore we don't know the form of w in L, so we will just assume that its in the form $a^{n^2}b^p$ for some $p \in \mathbb{N}$.

Assume that L is regular. Then, by **Pumping Lemma**, there exists an integer m such that, for any string $w \in L$ with $|w| \geq m$ there exists a split such that, $w = xyz$ with $y \neq e$, $|xy| \leq m$ and $xy^iz \in L$ for each $i \geq 0$.

Let $w = a^{m^2}b^p$, observe that $|w| \geq m$. Notice also that, since $|xy| \leq m$, $|xy|$ must consist of only a's but no b's and since $y \neq e$, y has to be one or more a's, so suppose $|y| = k$ where $0 < k \leq m$, hence y is in the form a^k.

Then by **Pumping Lemma** we try to pump another y to the string, $xyyz$, and that string must be in this language, namely $xyyz \in L$, which is in the form $a^m a^k a^{m^2-k} b^p$, which equals to $a^{m^2+k}b^p$. From here we can clearly see that $a^{m^2+k}b^p \notin L$ since $k > 0$ and $m^2 + k \neq m^2$