Student Information

Full Name : Utku Gungor
Id Number : 2237477

Answer 1

a.

By the definition of A , A_{280} is a finite set which contains string with length at most 280 . We can choose any string from B .

Apart from A and B , C has infinitely many members . {0,1}* is included in a widely used lemma which has too long proof in the theory of computation and this set is countably infinite . But $2^{\{0,1\}*}$ is all subsets of this countably infinite set . Hence C is an infinitely uncountable set , it follows that the set C \ { {x} : x $\in$ ($A_{280} \cup$ B)} is infinitely uncountable . $\diamondsuit$

b.

Given set is composed of intersections . By the definition of A_7 the length of strings of A can be at most 7 . Since it is upper bounded , this set is a finite which is also countable set .

Now time to think of B and C . B and C can be any string but their intersection with A_7 can be strings with the length 7 at most . Since A_7 is a finite set , the set $A_7 \cup$ B* $\cup$ C is finite . $\diamondsuit$

c.

The pushdown automaton that corresponds to the grammar G is :
P = ({p,q} , Σ , V , Δ , p , {p}) , where
Δ = {((p,e,e) , (q,E)),
((q,e,E) , (q,E+T)),
((q,e,E) , (q,T)),
((q,e,T) , (q,TxF)),
((q,e,T) , (q,F)),
((q,e,F) , (q,a)),
((q,e,F) , (q,(E))),
((q,a,a) , (q,e))
((q,+,+) , (q,e))
((q,x,x) , (q,e))
((q,(,() , (q,e)),
((q,),)) , (q,e))}

Let's use L_1 as the name of this language and p as the pumping length . First of all , we assume that L_1 is a context-free language . We can choose the string $a^{p^2}b^p \in L_1$ for pumping.

Now , we will analyze the possible positions of vxy . It is obvious that the string v will be part of a^{p^2} or b^p since if it belongs to both ,the order of a's and b's will be mixed and the string will not be included in L_1 . This condition is valid for the string y , too . We also must satisfy the conditions for pumping lemma : $a^{p^2}b^p$ = uvxyz , |vxy|$\leq$ p and |vy|$\geq$1 . Let's analyze the different cases :

1) v belongs to a^{p^2} and y belongs to b^p :

We have v = a^s and y = b^t and as we have the conditions |vxy|$\leq$ p and |vy|$\geq$1 , there must be the equation 1$\leq$s+t$\leq$p . In contrast , if we pump this string , the equation 1$\leq$s+t$\leq$p will not hold anymore since s+t becomes larger and larger when we compare it with p . And the resulting string will not satisfy the conditions to be included in L_1 .

2) v and y belong to a^{p^2} :

When we try to pump v and y continuously , the string that we will get have the type $a^{p^2+r}b^p$ where r $\geq$ 1 .The resulting string will clearly not be in the language L_1 .

3) v and y belong to b^p :

Similar to the previous case , when we try to pump down v and y continuously , the string that we will get have the type $a^{p^2}b^{p-r}$ where r $\geq$ 1 .The resulting string will clearly not be in the language L_1 .

Since we have reached contradictions in every case , the language L_1 is not a context-free language . $\diamondsuit$

Let's use L_2 as the name of this language and assume another language K = $L_2 \cap$ ab*ab*ab* . If our language L_2 is context-free , this follows that K is context-free , too. Now, our purpose is to show that K is not context-free . Let's consider w = ab^nab^nab^n . According to pumping lemma , we need to have ab^nab^nab^n = uvxyz ,|vxy|$\leq$ p and |vy|$\geq$1 where p is pumping length . Clearly , v and y parts will not include any a's since if they include a's , after pumping the string , there will be more than 3 a's which makes the string not belong to the language K . So v any y will be in one of the three b parts . Let's consider possible cases :

1) If they are included in the same b part (first,second or third), after we pump the string , that b part will have more than b's than other two b parts . This follows that the string does not belong to K .

2) If they are included in two different b parts (first-second or second-third) , similar to the previous case , when we pump the string , two parts will have the same number of b's but the remaining part will have less b's than those two . This follows that the string does not belong to K.

Since we have shown that the language K is not context-free , we conclude that the language L_2 is not context-free . $\diamondsuit$

d.

By the definition of A and C , $\bigcup$C $\cup$ $\bigcup A_k$ is all the strings that can be composed of 0's and 1's . Hence it is equal to {0,1}* . This is clearly same as the right handside . Therefore, the result of the difference equation is an empty set .

$\bigcup$C $\cup$ $\bigcup A_k$ $\setminus$ {0,1}* = {} $\diamondsuit$

Answer 2

a.

We can have 4 initial states .

We can have $\binom{4}{4} + \binom{4}{3} + \binom{4}{2} + \binom{4}{1} + 1$(no final state) $= 16$ final states .

There are 4 states for a to go and 4 states for b to go . $4 * 4 = 16$. And we have 4 states to be considered . 16^4 .

The answer is $4 * 16 * 16^4 = 2^{22}$ DFA can exist . $\diamondsuit$

b.

We can have 4 initial states .

We can have $\binom{4}{4} + \binom{4}{3} + \binom{4}{2} + \binom{4}{1} + 1$(no final state) $= 16$ final states .

There are $\binom{4}{4} + \binom{4}{3} + \binom{4}{2} + \binom{4}{1} + 1$(not going any state) $= 16$ ways for a . Same number of ways for b and same number of ways for 'empty' . $16 * 16 * 16$. And we have 4 states to be considered again . $(16 * 16 * 16)^4$.

The answer is $4 * 16 * (16 * 16 * 16)^4 = 2^{54}$ NFA can exist . $\diamondsuit$

c.

The first reason of the difference is NFA's can have 'empty' transition while DFA's cannot .

And the second one is for all states of DFA , there must be a and b transitions(one of each) leaving from them . But for NFA's , they do not need to have a property like this .

These property differences lead to different results . $\diamondsuit$

d.

If we have different languages , it brings different conditions . We can have different symbols and states . According to the these conditions and datas , we can find the number of DFA and NFA that can exist . $\diamondsuit$

Answer 3

a.

L_1's members' number of 1's is a multiple of 2 and 4 and this language does not have a string that has two consecutive 1's according to the question . So the regular expression representing the language L_1 is ;

L(α_1) = (00*100*100*100*1)* ∪ (100*100*100*100*)* ∪ 0*

b.

We are given the length of L_2 is an odd number and the members which have even place (w_0,w_2,w_4,...) will be 0 . The regular expression that represents L_2 is ;

L(α_2) = (10 ∪ 00)*.(0 ∪ 1)

c.

There is only one required condition for L_3 : strings of this language have even number of interleaved occurrences of the substring 00. We are taking even number of interleaved occurrences of the substring 00 and other places can change .The regular expression representing L_3 is ;

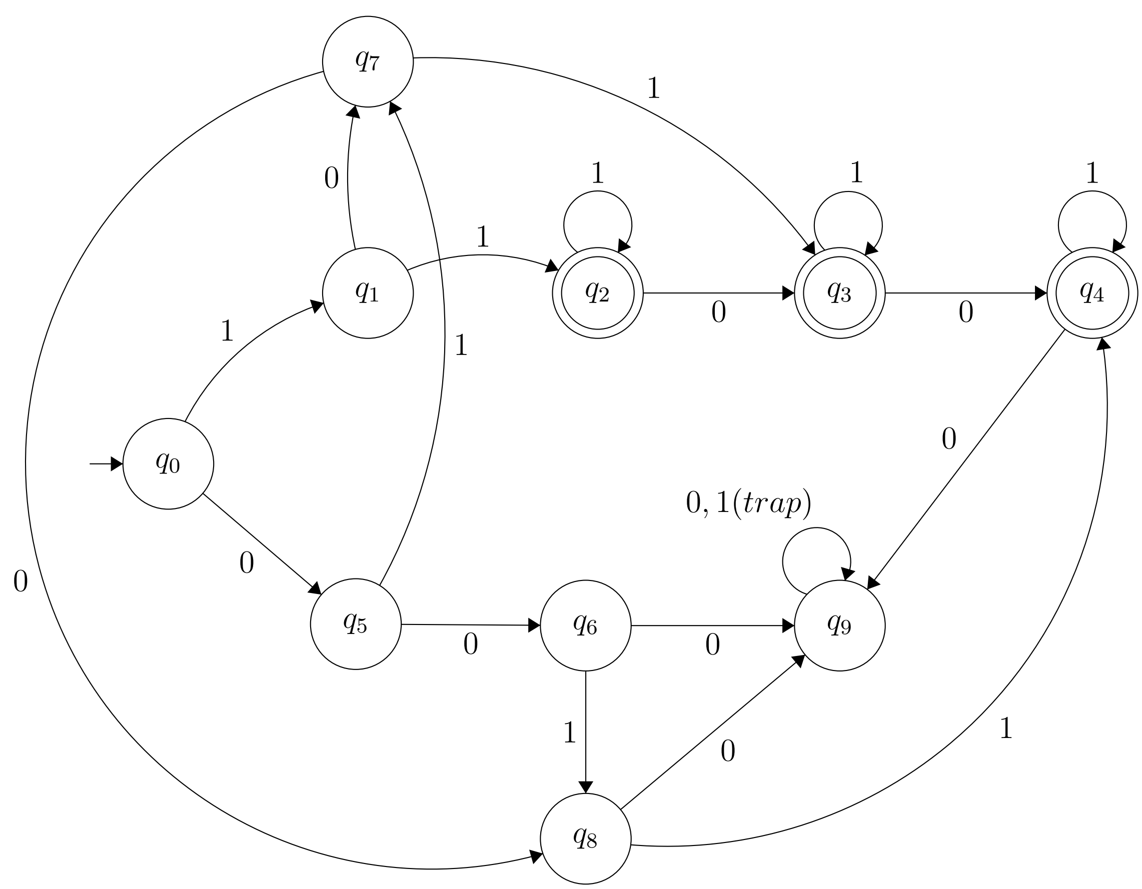
L(α_3) = (1*001*001*)*.(0 ∪ 1*)

Answer 4

a.

The constituents of the DFA I constructed :

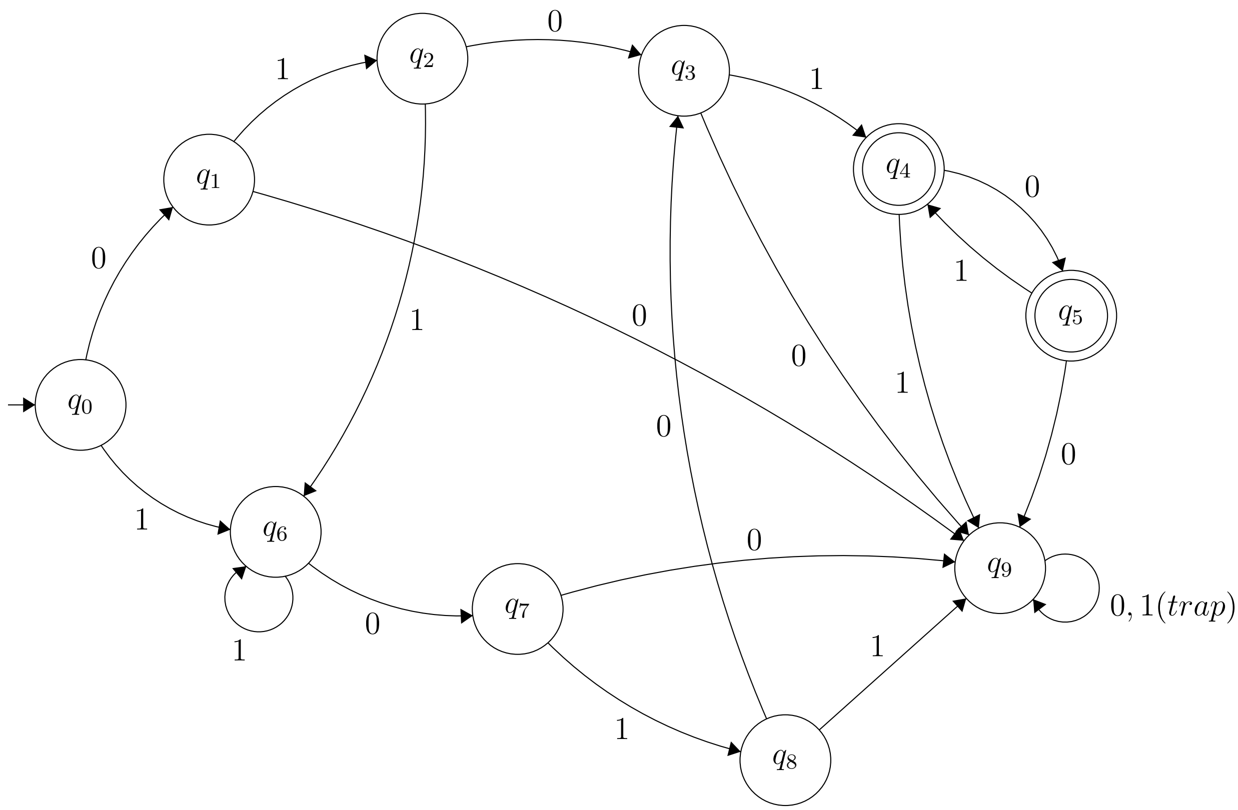
N = (K,{0,1},Δ,q_0,F) in which K = {q_0,q_1,q_2,q_3,q_4,q_5,q_6,q_7,q_8,q_9} and F = {q_2, q_3, q_4}. Note that q_9 is the trap state. The graph of the DFA N is the following :



b.

The constituents of the DFA for this question :

M = (K,{0,1},Δ,q_0,F) in which K = $\{q_0,q_1,q_2,q_3,q_4,q_5,q_6,q_7,q_8,q_9\}$ and F = $\{q_4, q_5\}$. Note that q_9 is the trap state again . The graph of the DFA M is the following :



Answer 5

a.

When the string ω_1 = abaa is given as input , several different sequences of moves may exist . We will check whether there exist one sequence of moves leading to a final state or not . The given string ω_1 may drive K from state q_0 to any of the final states . One of these ways is the following :

$$\begin{aligned}(q_0 \text{ , abaa}) &\vdash_M (q_2 \text{ , abaa}) \\ &\vdash_M (q_2 \text{ , baa}) \\ &\vdash_M (q_2 \text{ , aa}) \\ &\vdash_M (q_4 \text{ , a}) \\ &\vdash_M (q_1 \text{ , e})\end{aligned}$$

Since q_1 is a final state , it follows that $\omega_1 \in$ L(N) . ♢

b.

Since a string is accepted by a NFA if and only if there is at least one sequence of moves leading to a final state , we will check all possible sequence of moves and find out whether at least one of them reaches to the final state or not. Note that ω_2 = babb .

Assume that the string ω_2 is accepted by this NFA . That means after the last b symbol , we will reach one of the final states which is q_1 or q_3 . We can reach them with the b symbol only from q_0 since we just have two transitions that reach q_1 or q_3 . These transitions are (q_0,b,q_1) and (q_0,b,q_3) .

So , if there is a string 'bab' ends at the state q_0 ,our assumption will be true . We will check whether there is a b transition ends at the state q_0 at first . But in the given question , there is no transition leading to q_0 . This shows us the assumption we made at the beginning of the question is wrong . Hence , the string ω_2 is not accepted by this NFA , i.e. $\omega_2 \notin$ L(N) $\diamondsuit$

Answer 6

In the given automaton , we have s' = E(q_0) = $\{q_0, q_2\}$.
(q_0, a, q_0) is onlt transition (q,a,p) for some q $\in$ s'. It follows that

$$\delta\text{'(s',a)} = \text{E}(q_0) = \{q_0, q_2\} \ .$$

Similarly,
(q_0, b, q_1),(q_2, b, q_1),(q_0, b, q_0),(q_2, b, q_3) are all transitions (q,b,p) for some q $\in$ s' , therefore

$$\delta\text{'(s',b)} = \text{E}(q_0) \cup \text{E}(q_1) \cup \text{E}(q_3) = \{q_0, q_1, q_2, q_3\} \ .$$

We'll now repeat this calculation for the new state $\{q_0, q_1, q_2, q_3\}$.

$$\delta\text{'}(\{q_0, q_1, q_2, q_3\},\text{a}) = \{q_0, q_2, q_3, q_4\} \ ,$$
$$\delta\text{'}(\{q_0, q_1, q_2, q_3\},\text{b}) = \{q_0, q_1, q_2, q_3\} \ ,$$

Same operation for the new state $\{q_0, q_2, q_3, q_4\}$;

$$\delta\text{'}(\{q_0, q_2, q_3, q_4\},\text{a}) = \{q_0, q_2, q_3, q_4\} \ ,$$
$$\delta\text{'}(\{q_0, q_2, q_3, q_4\},\text{b}) = \{q_0, q_1, q_2, q_3\} \ ,$$

We have shown all the transitions of our DFA . We have 3 states that we have shown above in the DFA . There is no new state anymore , hence we have found all states of the DFA . In the given NFA , F = $\{q_3, q_4\}$. So in our DFA , every state that include q_3 or q_4 will be final . Therefore , 2 states $\{q_0, q_1, q_2, q_3\}$ and $\{q_0, q_2, q_3, q_4\}$ of M' are final . And the initial state is $\{q_0, q_2\}$ since this set is equal to E(q_0) from the beginning of the solution .

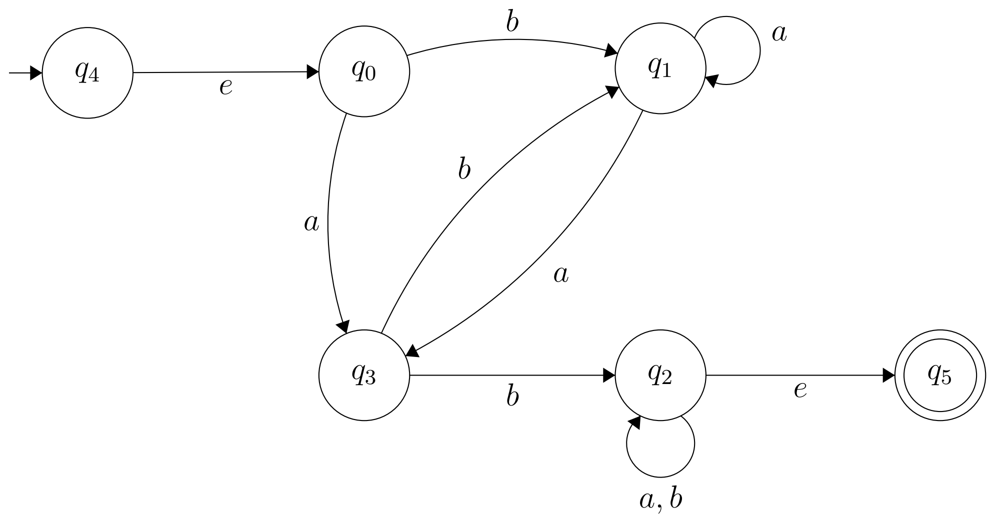
Answer 7

To construct a GFA , we add two more states , say q_4 and q_5 . And we add two more transition from these new states . These two transitions are (q_4,e,q_0) and (q_2,e,q_5) since q_0 is the initial and q_2 is the final state . All other transitions are same in the GFA . So the GFA for N is ;

N' = (K',{a,b},Δ',q_4,F') in which K' = $\{q_0, q_1, q_2, q_3, q_4, q_5\}$,

Δ' = {(q_0,b,q_1) , (q_0,a,q_3) ,
(q_1,a,q_1) , (q_1,a,q_3) ,
(q_2,a,q_2) , (q_2,b,q_2) , (q_2,e,q_5)

$(q_3$,b,$q_1)$, $(q_3$,b,$q_2)$,
$(q_4$,e,$q_0)\}$
and F' = $\{q_5\}$.
Here is the graph of the GFA I constructed :

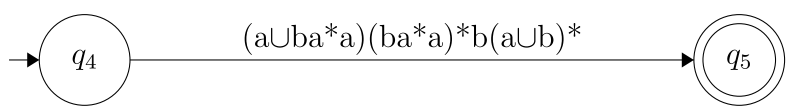


First we compute R(i,j,0)'s . Since q_0 is eliminated , we have new transitions (q_4,b,q_1) and (q_4,a,q_3) .

Now we compute the R(i,j,1)'s . Notice that the q_1 need not to be considered anymore ; all strings have been considered and taken into account in the R(i,j,1)'s . We can say that q_1 is eliminated and we have new transitions that are (q_4,ba*a,q_3) and (q_3,ba*a,q_3) .

Continuing like this , we eliminate q_2 to obtain R(i,j,2)'s . After q_2 is eliminated ,we now have (q_3,b(a∪b)*,q_5) .

At the last step , we will eliminate q_3 and we will have two states which are q_4 and q_5 . And the transition between these two states will be the regular expression L(α) such that L(α) = L(N). When we do this elimination , our GFA has been reduced to a single transition from the initial state to the final state . And the single transition that we have obtained is (q_4,(a∪ba*a)(ba*a)*b(a∪b)*,q_5) .

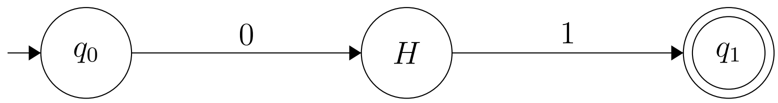


Hence , the regular expression is ;
L(α) = (a∪ba*a)(ba*a)*b(a∪b)* . ♢

Answer 8

Let's define a language D over the alphabet Σ as set of all strings in the form $0\omega 1$. We can easily conclude that D $\in$ L .

Let F be another language such that F = $0(\Sigma^*)1$.
It is obviously seen that F $\cap$ L = D .
We can now define a FA for D as ;



The reason that I constructed an automata like this is we can think H as an inner automata and any word inside H is also accepted by D as long as it starts with a 0 and ends with a 1 .

If L is regular , then D is regular since we have D $\in$ L .

Or we can say that since we can construct an automata for D , D is a regular language . And as the graph shows , H is a part of the Finite Automata graph of D above . It follows that H is a regular language .

Therefore , if L is regular , then H is regular ,too . $\diamondsuit$

Answer 9

a.

Assume that the language L is regular . It follows that it must have a pumping length P .

Let's say S = $1^P 0^P 1^P$ and choose P = 4 since t < 5 . As a result of this choice, we have the string S = 111100001111 . According to the Pumping Lemma Theorem , our string S (|S| >P) may be divided into 3 parts S = xyz such that the following conditions must be true .

1) xy^iz$\in$ L for every i $\geq$ 0 ,
2) |y| > 0 ,
3) |xy| $\leq$ P .

Let's choose i = 2 . We will now check 3 cases for this string :

Case1: y is composed of 1's .

$\underbrace{1}_{x}\ \underbrace{111}_{y}\ \underbrace{00001111}_{z} \quad \Rightarrow \quad xy^2z = 111111100001111$ t = 7 , t < 5 is not satisfied . So xy^2z $\notin$ L .

Case2: y is composed of 0's .

$\underbrace{1111}_{x}\ \underbrace{000}_{y}\ \underbrace{01111}_{z} \quad \Rightarrow \quad xy^2z = 111100000001111$ xy^2z $\in$ L but |xy| $\leq$ P (|xy| = 7 , P = 4) is not satisfied .

Case3: y is composed of both 0's and 1's .

$\underbrace{11110}_{x}\ \underbrace{00011}_{y}\ \underbrace{11}_{z} \quad \Rightarrow \quad xy^2z = 11110000110001111$ since there is the substring 11 at the middle of the 0's , xy^2z $\notin$ L .

We have shown that none of these 3 cases can satisfy all the 3 pumping conditions at the same time which means S cannot be pumped . This follows that we have reached a contradiction with our assumption which says that L is regular . Therefore , the language L is not regular . $\diamondsuit$