Student Information

Full Name : Furkan Göksel
Id Number : 2237436

Answer 1

Before the start let me classify which of them countable or uncountable.
-For A_k, it is finite since for each $k \in N$ A_k has a upper limit such that $|w| \leq k$. For instance for $k = 2$ it is $\{e, 0, 1, 00, 01, 10, 11\}$
-For B, it is a regular expression. Observe that the set $\Sigma \cup \{(,), \emptyset, \cup, *\}$ is nonempty and finite, and by definition $(\Sigma \cup \{(,), \emptyset, \cup, *\})^*$ is infinitely countable (can be enumerated in lexicographic order) and clearly $B \subseteq (\Sigma \cup \{(,), \emptyset, \cup, *\})^*$. Since by theorem, subsets of infinitely countable set are countable, B is also countably infinite.
-For C $= 2^{\{0,1\}^*}$, firstly we can say that $\{0, 1\}^*$ is countable and we can count it in lexicographic order, like $\{e, 0, 1, 00, 01, 10, 11, ..\}$, and since it is not finite, $\{0, 1\}^*$ is infinitely countable, and we can say that since there is a bijection with N it is equinumerous to N. And it's cardinality is equal to $|N|$. But since $2^{|N|}$ is infinitely uncountable by diagonalization princible, C is also infinitely uncountable.

a.

C is the power set of all strings generated from $\{0, 1\}$ and it is infinitely uncountable, and the right hand side of the given equation is infinitely countable, And the result is infitinely uncountable proof is following. (Proof of infinitely uncountable / infinitely countable = infinitely uncountable) Assume that A is infinitely uncountable and B is infinitely countable and A/B is infinitely countable. Then from the theorem we know that union of two countable set is also countable so $(A/B) \cup B$ is countable. Then we can say that $A \subseteq (A/B) \cup B$, however A is uncountably infinite and there is a contradiction.Hence our assumption is wrong and A/B is infinitely uncountable.

b.

Observe that $A_7 \cap B^* \cap C \subseteq A_7$ Since A_7 is finite, this equation is also finite and countable. Moreover it is $\emptyset$ since there is no intersection between C and others, since while C's members are sets of strings, B^* and A_7's members are strings. As a result cardinality is 0.

c.

When we apply $\cup C$, it means that all sets $(\forall S \in C)$ in C will be united, and it looks like $\{a, b, aa, bb, ...\}$ and it is countably infinite set (C is collection of countable sets, so $\cup C$ is countable), we can enumerate them in lexicographic order. Also, the other side of the equation is a cartesian product and 2-tuple which is infinitely countable such that we can enumerate them (first element of finite set A_2,first element from B in lexicographical order),(second element of finite set A_2,first element from B in lexicographical order),... (counting them according to elements of the finite set) and with this way we can enumerate all of them so countably infinite when we extract them, nothing happens and still we get $\cup C$, and so it is infinitely countable.

d.

C is collection of countable sets, so $\cup C$ is countable, same theorem is valid for A_k, hence it is also infinitely countable, and union of two countably infinite set is also countable. Therefore, Left hand side of the equation means that set of all strings generated from $\{0, 1\}$, also again $\{0, 1\}^*$ means set of all strings generated from $\{0, 1\}$ so whole expression means $\emptyset$, and cardinality is 0.

Answer 2

a.

In deterministic finite automata, we have $M = (K, \Sigma, \delta, s, F)$, and with some specification: K can't be empty ($K \neq \emptyset$) because s is an element from K, δ is a **function**, and F can be empty ($F = \emptyset$). From these specifications and our assumption (two FA are not different if and only if all of the associated five tuples are equal to each other) , we can say that s can't be empty and from K it can be any stage we choose and it is **4** different case, also F is subset of K ($F \subseteq K$) and can be any subset (including $\emptyset$), so number of all subsets of K is $2^{|K|}$ which is 2^4 different case, and lastly δ is a function: $K \text{x} \Sigma \rightarrow K$. And the number of different function is $|K|^{|K|*|\Sigma|}$ which is 4^8. As a result total number of different deterministic FA is $4 * 4^8 * 2^4 = 2^{22}$

b.

In nondeterministic finite automata, we have $M = (K, \Sigma, \Delta, s, F)$, and with some specification: K can't be empty ($K \neq \emptyset$) because s is an element from K, Δ is a **subset** which is $\Delta \subseteq K\text{x}(\Sigma \cup \{e\})\text{x}K$, and F can be empty ($F = \emptyset$). From these specifications and our assumption (two FA are not different if and only if all of the associated five tuples are equal to each other) , we can say that s can't be empty and from K it can be any stage we choose and it is **4** different case, also F is subset of K ($F \subseteq K$) and can be any subset (including $\emptyset$), so number of all subsets of K is $2^{|K|}$ which is 2^4 different case, and finally Δ can be any subset of $K\text{x}(\Sigma \cup \{e\})\text{x}K$ and total number of different subsets is $2^{|K|*|\Sigma \cup \{e\}|*|K|}$ which is 2^{48}. As a result, total number of different nondeterministic FA is $4 * 2^4 * 2^{48} = 2^{54}$.

c.

Difference occurs because of Δ and δ. While Δ is a relation set and may not be a function, δ must be a function and because of this difference, total numbers of deterministic FA and nondeterministic FA differ.

d.

According to our assumption, two FA are not different if and only if all of the associated five tuples are equal to each other, and we know that if two machines are equivalent ($M_1 \approx M_2$), their languages are equal ($L(M_1) = L(M_2)$). So if our five tuples are different then our machine is different, hence in (a) and (b) we calculated total number of different five tuple hence these numbers are equal to the number of different languages corresponding FA M recognize.

Answer 3

a.

$0^* \cup 0^*100^*100^*100^*1(\emptyset^* \cup (00^*100^*100^*100^*10^*)^*)$

b.

$(1 \cup 0)(00 \cup 01)^*$ (Assumption 1 and 0 are in language since $|w| = 1$ and odd).

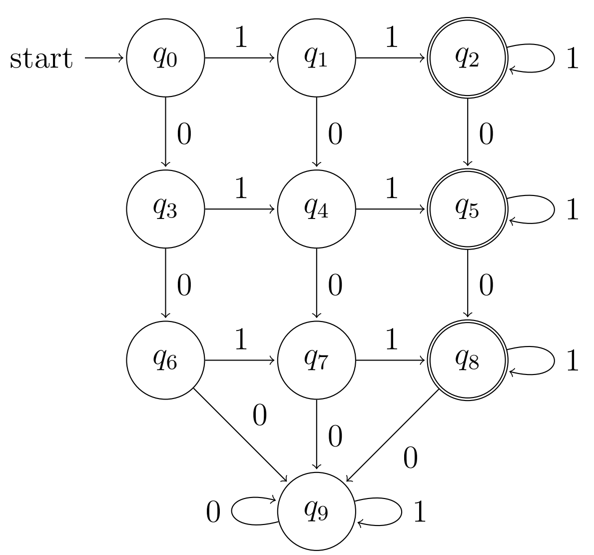
c.

$1^* \cup ((101)^*(1^*00001^*)(1^*001^*)^*)^* \cup (101)^*(1^*001^*)(101)^*$ (Assumption 0 occurence is even number)

Answer 4

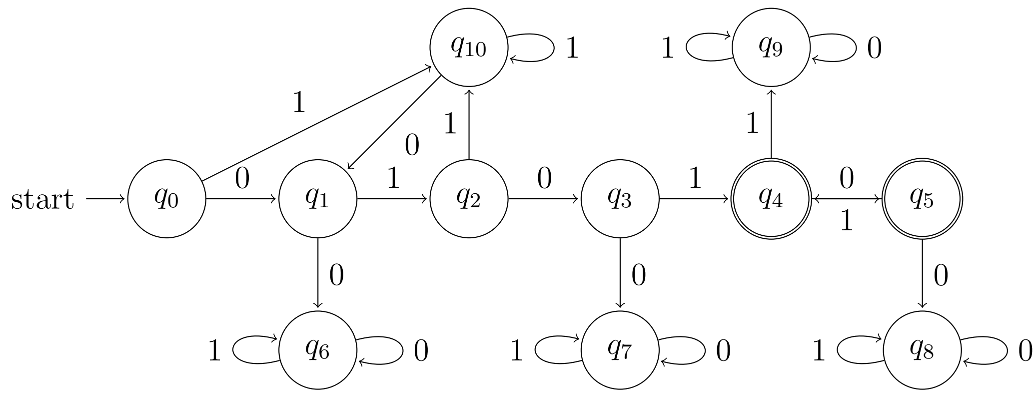
a.

We can say that L_1 means that a word from L_1 contains at least two 1, and at most two 0. Hence, M=($\{q_0, q_1, q_2, q_3, q_4, q_5, q_6, q_7, q_8, q_9\}, \{0, 1\}, \delta, q_0, \{q_2, q_5, q_8\}$)



b.

M=$(\{q_0, q_1, q_2, q_3, q_4, q_5, q_6, q_7, q_8, q_9, q_{10}\}, \{0, 1\}, \delta, q_0, \{q_4, q_5\})$



(I hope you can infer that in q_4 reading 0 takes to q_5 and reading 1 in q_5 leads to q_4 I couldn't fix it)

Answer 5

A string is accepted by a nondeterministic finite automaton if and only if there is at least one sequence of moves leading to a final state. (from book pg.67), but every branch must reject for the overall NFA to reject.

a.

$(q_0, abaa) \vdash_M (q_2, abaa) \vdash_M (q_2, baa) \vdash_M (q_2, aa) \vdash_M (q_4, a) \vdash_M (q_1, e)$
So we find a computation which ends at a final state. So, $(q_0, abaa) \vdash_{M^*} (q_1, e)$ and $abaa \in L(N)$

b.

We must try all computation possibilities.

1. $(q_0, babb) \vdash_M (q_3, abb) \vdash_M (q_1, bb)$ Machine stuck
2. $(q_0, babb) \vdash_M (q_1, abb) \vdash_M (q_1, bb)$ Machine stuck
3. $(q_0, babb) \vdash_M (q_2, babb) \vdash_M (q_2, abb) \vdash_M (q_2, bb) \vdash_M (q_2, b) \vdash_M (q_2, e)$ $q_2 \notin F$
4. $(q_0, babb) \vdash_M (q_2, babb) \vdash_M (q_2, abb) \vdash_M (q_4, bb) \vdash_M (q_4, b) \vdash_M (q_4, e)$ $q_4 \notin F$
5. $(q_0, babb) \vdash_M (q_3, abb) \vdash_M (q_4, abb) \vdash_M (q_1, bb)$ Machine stuck
6. $(q_0, babb) \vdash_M (q_3, abb) \vdash_M (q_4, abb) \vdash_M (q_2, bb) \vdash_M (q_2, b) \vdash_M (q_2, e)$ $q_2 \notin F$

We have tried all possible computations, so $babb \notin L(N)$

Answer 6

Step 1: Finding all E(q)(means all stages reachable from q without reading input) sets

1. $E(q_0) = \{q_0, q_2\}$
2. $E(q_1) = \{q_1\}$
3. $E(q_2) = \{q_2\}$
4. $E(q_3) = \{q_3\}$
5. $E(q_4) = \{q_3, q_4\}$

So $s' = \{q_0, q_2\}$, $\Sigma = \{a, b\}$

Step 2:

$(\{q_0, q_2\}, a) = E(q_0) = \{q_0, q_2\}$
$(\{q_0, q_2\}, b) = E(q_0) \cup E(q_1) \cup E(q_3) = \{q_0, q_1, q_2, q_3\}$
We have already calculated $\{q_0, q_2\}$.

Step 3:

$(\{q_0, q_1, q_2, q_3\}, a) = E(q_0) \cup E(q_4) = \{q_0, q_2, q_3, q_4\}$
$(\{q_0, q_1, q_2, q_3\}, b) = E(q_0) \cup E(q_1) \cup E(q_3) = \{q_0, q_1, q_2, q_3\}$
We have already calculated $\{q_0, q_1, q_2, q_3\}$.

Step 4:

$(\{q_0, q_2, q_3, q_4\}, a) = E(q_0) \cup E(q_4) = \{q_0, q_2, q_3, q_4\}$
$(\{q_0, q_2, q_3, q_4\}, b) = E(q_0) \cup E(q_1) \cup E(q_3) = \{q_0, q_1, q_2, q_3\}$
We have already calculated both so algorithm stops. Hence new machine:

$M' = (K', \Sigma, \delta', s', F')$ where

K' = $\{\{q_0, q_2\}, \{q_0, q_1, q_2, q_3\}, \{q_0, q_2, q_3 q_4\}\}$

$\Sigma = \{a, b\}$

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

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

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

s' = $\{q_0, q_2\}$

F' = $\{\{q_0, q_1, q_2, q_3\}, \{q_0, q_2, q_3, q_4\}\}$

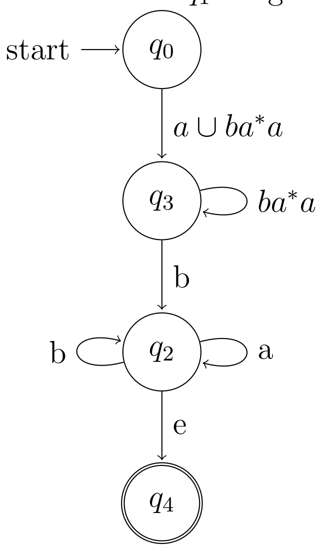
Answer 7

Firstly we should turn the specialized form:
-The automaton has a single final state, $F = \{f\}$
-The inital state does not have an incoming transition
-The final state does not have an outgoing transition.

So GFA is $N' = (K \cup \{q_4\}, \{a, b\}, \Delta \cup \{(q_2, e, q_4)\}, q_0, F)$ where $F = \{q_4\}$, and to obtain RE we should calculate $R(i, j, n)$s.

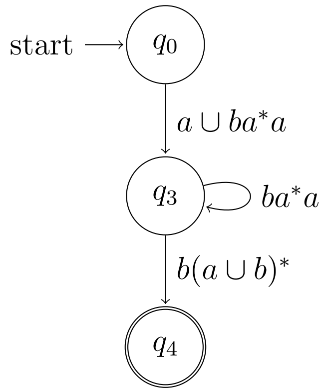
First Step

Eliminate q_1 we get something like this $R(3, 3, 1) = ba^*a$ and $R(0, 3, 1) \cup R(0, 3, 0) = ba^*a \cup a$



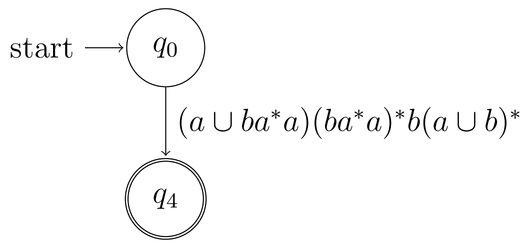
Second Step

Eliminate q_2 we get something like this $R(3, 4, 1) = ba^*e \cup bb^*e$



Last Step

Eliminate q_3 and we done $R(0,4,1) = (a \cup ba^*a)(ba^*a)^*(ba^* \cup bb^*)$



Our RE is $R(0,4,1) = (a \cup ba^*a)(ba^*a)^*b(a \cup b)^*$

Answer 8

Assume that L is regular. Then we can say that there is a DFA $M_L = (K, \Sigma, \delta, s, F)$ such that it recognizes the language L. Then we can construct a DFA M_H such that it can recognize H. Since H generates their strings from the L ($0w1 \in L$ and $w \in H$) we can construct M_L such that

$$M_H = (K', \Sigma', \delta', s', F') \tag{1}$$

Such that $K' = K$, $\Sigma' = \Sigma$, $\delta' = \delta$, s' $= \delta(s, 0)$, $F' = \{q : \ \sigma(q,1) = z \ \ q \in K \ and \ z \in F\}$ for any given DFA M_L.

In order to prove $L(M_H) = H$ we have to show that
1) H $\subseteq L(M_H)$

Assume that there there exist a word w from H ($w \in H$) such that $0w1 \in L$ and by definition $(s, 0w1) \vdash_M^* (z, e) \quad z \in F_L$ so $0w1 \in L(M_L)$. And then from closure of yield-one-step relation we can say that $(s, 0w1) \vdash_M (s', w1) \vdash_M^* (q, 1) \vdash_M (z, e) \quad q \in F_H, z \in F_L$. From that $(s', w1) \vdash_M^* (q, 1) \quad q \in F_H$ is valid operation, and from the property of yield-one-step relation $(s', w) \vdash_M^* (q, e) \ \ q \in F_H$, so $w \in L(M_H)$ and from that proof we can say that $H \subseteq L(M_H)$.

2) $L(M_H) \subseteq H$

Assume that there exists a computation in $L(M_L)$ such that $L(M_L) = \{w \in \Sigma^* \quad (s, 0w1) \vdash_M (s', w1) \vdash_M^* (q, 1) \vdash_M (z, e) \; some \; z \in F_L\}$ and $0w1 \in L(M_L)$ then in $L(M_H) = \{(s', w) \vdash_M^* (q, 1) \; \; for \; some \; q \in F_M\}$ hence $w \in L(M_H)$ and $L(M_H) \subseteq H$.

Answer 9

a.

Assume that L is regular and p be the pumping length. Let's say our string is $w = 0^p1^{2^{p+p!}}$ since t should be less than 5 and it is a natural number we can easily select that $t = 0$ and construct such a string, and $p \leq |w|$, and finally this w is in the language L because $p \neq p + p!$. And let's split this string such that $0^a0^b0^c1^{2^{p+p!}}$ and $a + b + c = p$ now we can assume that this word can be rewritten as $w = xyz$ such that $x = 0^a$, $y = 0^b$, $z = 0^c1^{2^{p+p!}}$, and assume that $p \geq b > 0$. Such a split is valid because $xy \leq p$ and $y \neq e$ and this split holds these rules. By pumping lemma for all i such that $i \geq 1$, $xy^iz \in L$, however when we are taking $i = \frac{p!}{b} + 1$ (we can take i such a form because p = a+b+c and p! includes b inside it, then the result will be again natural number). we can see that $w = 0^a0^{b+p!}0^c1^{2^{p!+p}}$ and this is equal to $w = 0^{p+p!}1^{2^{p+p!}}$ and in this situation $n = m$ and hence this string is not in the L. So it is a contradiction, hence our first assumption is wrong, and L is not regular.