# CS340: Theory of Computation
# Assignment 2 Solutions

Yatharth Goswami
Roll No. 191178

September 11, 2021

# I Problem 1 Solution

In this problem, we were asked to derive a regular expression for the given languages.

(a) First we will construct a GNFA, that accepts the given language and then derive the regular expression for it. I am not writing the edges as $\phi$ as it then becomes difficult to comprehend.

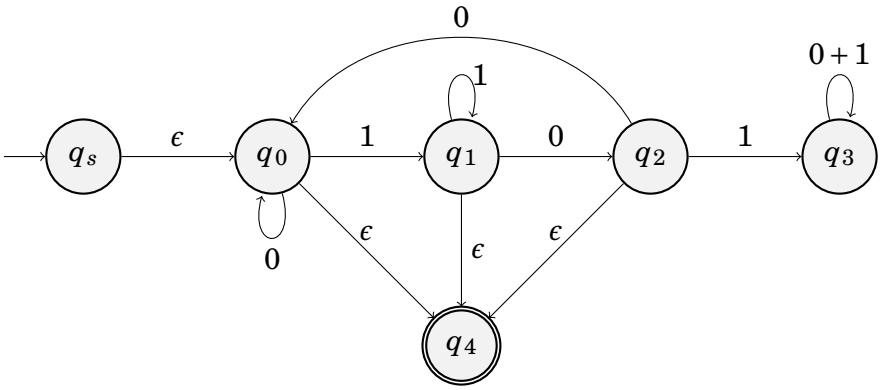


Figure 1: DFA 1.a

First we will remove $q_3$, and none of the edges will be changed because of it, since $q_3$ has edges to none of the other vertices except itself. The new GNFA will look like

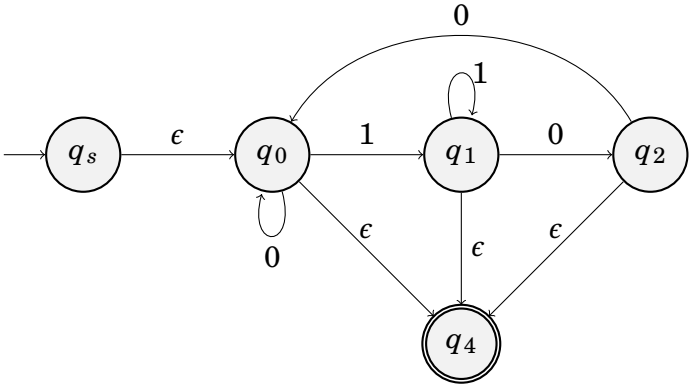


Figure 2: DFA 1.a

Next, let us remove the node $q_2$. Because of this the edge $(q_1, q_0)$ and $(q_1, q_4)$ will get modified. The new GNFA looks like

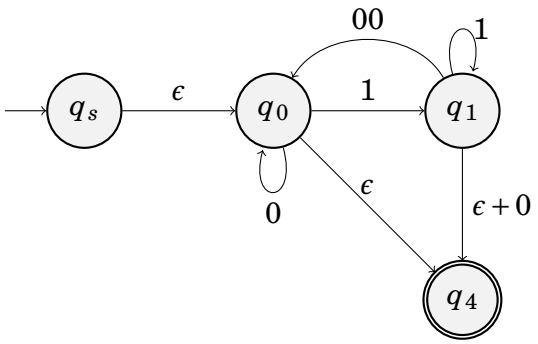


Figure 3: DFA 1.a

Now, we will remove the node $q_1$. Because of this the edges $(q_0, q_0)$ and $(q_0, q_4)$ will change.

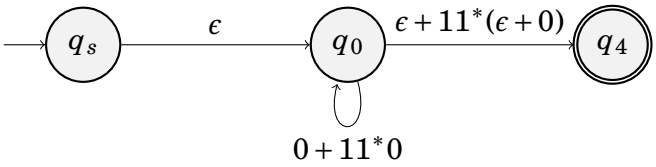


Figure 4: DFA 1.a

From the final GNFA, we get that the regular expression for this language is:

$$\boxed{(0 + 11^*00)^*(\epsilon + 11^*(\epsilon + 0))}$$

(b) In this part we were asked to provide a regular expression for language containing of strings with atmost 2 0's and 3 1's. For solving this, we first observe that no string of length $\geq 6$ will belong to the language. Further, we observe that the strings of length $\leq 5$ are somewhat a subset of those of length 5. Hence, we will try to form the strings that satisfy the condition of length 5 and simultaneously generate smaller strings as well. The final regular expression will look like

$$\begin{aligned}
&(0+\epsilon)(0+\epsilon)(1+\epsilon)(1+\epsilon)(1+\epsilon) + (0+\epsilon)(1+\epsilon)(0+\epsilon)(1+\epsilon)(1+\epsilon) + (0+\epsilon)(1+\epsilon)(1+\epsilon)(0+\epsilon)(1+\epsilon) \\
&+ (0+\epsilon)(1+\epsilon)(1+\epsilon)(1+\epsilon)(0+\epsilon) + (1+\epsilon)(0+\epsilon)(0+\epsilon)(1+\epsilon)(1+\epsilon) + (1+\epsilon)(0+\epsilon)(1+\epsilon)(0+\epsilon)(1+\epsilon) \\
&+ (1+\epsilon)(0+\epsilon)(1+\epsilon)(1+\epsilon)(0+\epsilon) + (1+\epsilon)(1+\epsilon)(0+\epsilon)(0+\epsilon)(1+\epsilon) + (1+\epsilon)(1+\epsilon)(0+\epsilon)(1+\epsilon)(0+\epsilon) \\
&+ (1+\epsilon)(1+\epsilon)(1+\epsilon)(0+\epsilon)(0+\epsilon)
\end{aligned}$$

# II Problem 2 solution

This problem required us to find the regular expression for the DFA. First we will construct a GNFA from the provided DFA, so that we can convert it to regular expression by eliminating nodes one by one. I am not showing the tranisitions which are labelled as $\phi$ for reducing cluttering. The initial GNFA will be

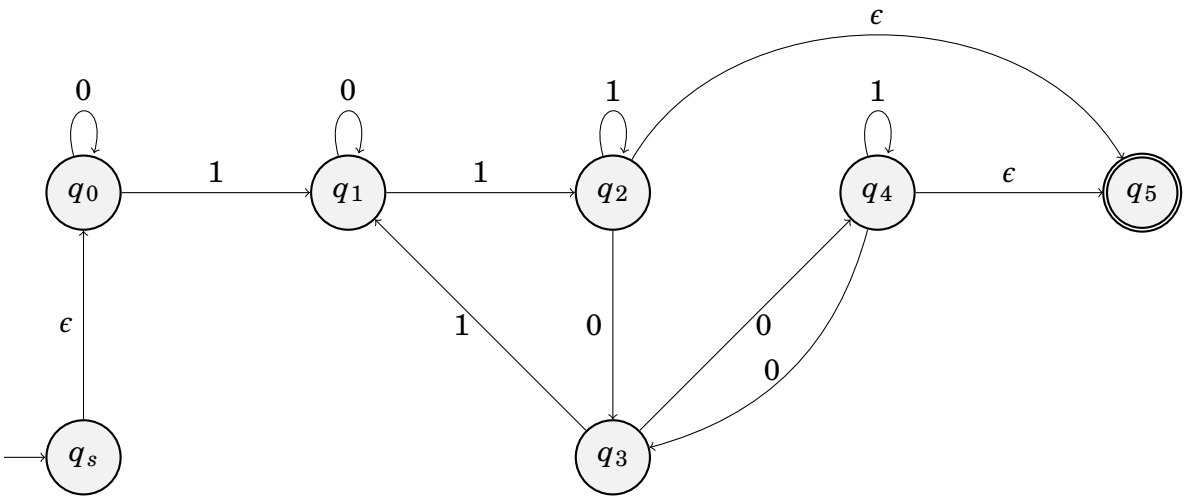


Figure 5: DFA 2:a

First let's remove the node $q_0$. This will lead to update of the edge $(q_s, q_1)$. The GNFA will then look like

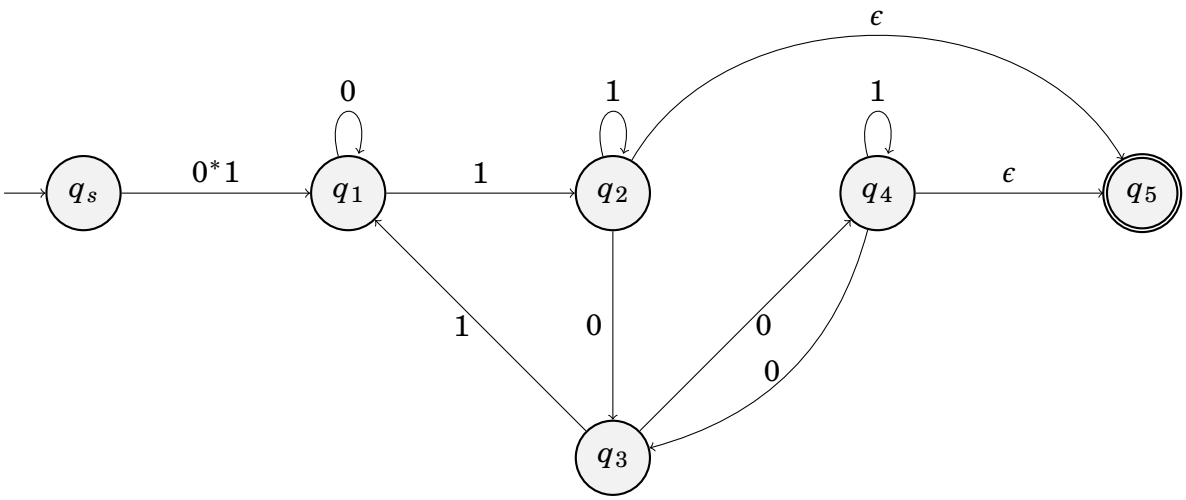


Figure 6: DFA 2:b

Let us now remove the vertex $q_1$. Two edges $(q_s, q_2)$ and $(q_3, q_2)$ will be updated.

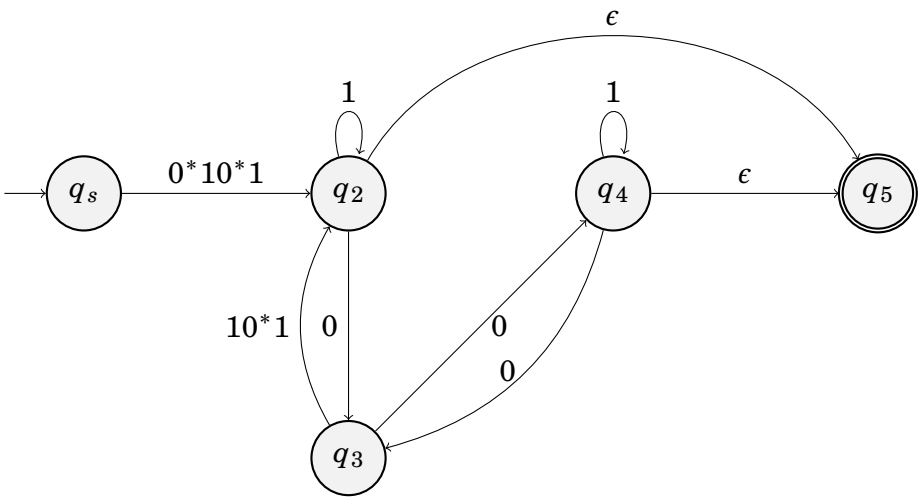


Figure 7: DFA 2:c

Now, we will remove $q_4$. Two edges $(q_3, q_5)$ and $(q_3, q_3)$ will be updated.

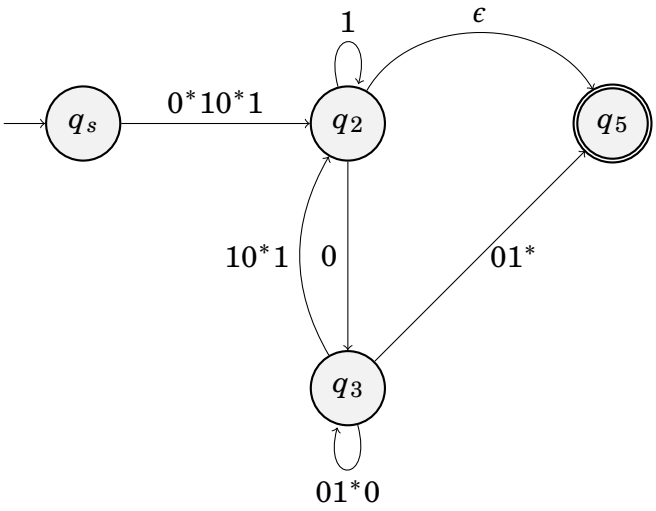


Figure 8: DFA 2:d

Next, we will remove the node $q_3$. Because of this, the edges $(q_2, q_5)$ and $(q_2, q_2)$ will be modified.

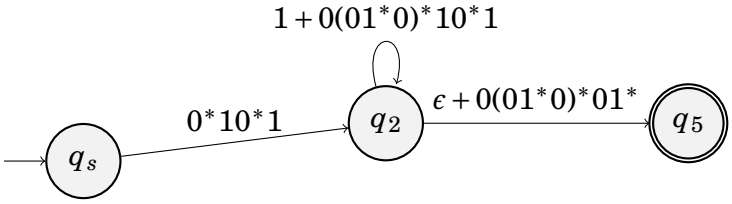


Figure 9: DFA 2:e

Finally, on removing $q_2$, we obtain the final regular expression as

$$\boxed{(0^*10^*1)(1+0(01^*0)^*10^*1)^*(\epsilon+0(01^*0)^*01^*)}$$

# III Problem 3 Solution

In this problem, we were asked to minimise the provided DFA. We will follow the steps of the DFA minimisation algorithm for achieving this. Let us proceed step by step.

1. The first step involves removing the unreachable states which can't be reached from the start state. In this case, the state $q_4$ is an unreachable state. Hence, it gets removed in this step.

2. The next step is to remove any dead states. The states from which no accept state can be reached. The accept states are $q_1$ and $q_3$ which are reachable from all the present states. Hence, no state gets removed in this step.

3. This step merges the non-distinguishable states in the DFA. Let us run the algorithm and find pairs which are same.

   (a) First we will mark all pairs of states in which one of the state is accepting and other is non-accepting. Therefore the following pairs will get marked.

   | | | | |
   |---|---|---|---|
   | $q_0$ | | | |
   | X | $q_1$ | | |
   | - | X | $q_2$ | |
   | X | - | X | $q_3$ |

   (b) Now, in the first iteration we will try to remove pairs of states that on some transition lead to a marked pair. After this the following pairs will get marked.

   | | | | |
   |---|---|---|---|
   | $q_0$ | | | |
   | X | $q_1$ | | |
   | X | X | $q_2$ | |
   | X | - | X | $q_3$ |

   (c) In the second iteration, we will try to mark the only unmarked pair $(q_1, q_3)$. On 0, the states go to the pair $(q_2, q_2)$ which is unmarked by definition and on 1 they remain in the same state. Hence this pair will still remain unmarked and the algorithm terminates. The final matrix will look like

   | | | | |
   |---|---|---|---|
   | $q_0$ | | | |
   | X | $q_1$ | | |
   | X | X | $q_2$ | |
   | X | - | X | $q_3$ |

   We get that the states $q_1$ and $q_3$ are non-distinguishable and hence can be merged into one.

The final minimised DFA will look like

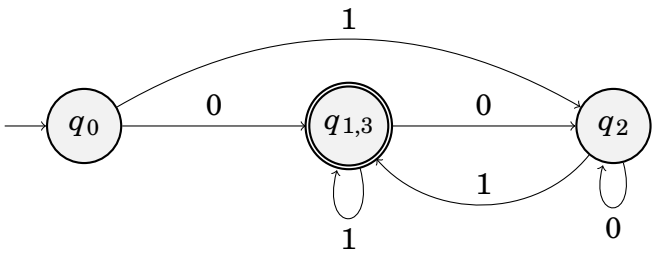


Figure 10: Minimised DFA for 3

# IV Problem 4 Solution

In this problem, we were asked to provide the Context-free-grammar that will accept the provided language.

(a) $L_1 = \{a^m b^n c^n d^{2m} \mid n, m \geq 0\}$

Let us define a grammar $G = (V, \Sigma, P, S)$ in this way

- $V = \{S, T\}$
- $\Sigma = \{a, b, c, d\}$
- Set of Production rules $P$:
  S $\rightarrow aSdd \mid T$
  T $\rightarrow bTc \mid \epsilon$
- Start State = $S$

The above provided grammar accepts the language $L_1$.

(b) $L_2 = \{a^n b^m \mid n \neq m\}$ Let us define a grammar $G = (V, \Sigma, P, S)$ in this way

- $V = \{S, T, R\}$
- $\Sigma = \{a, b\}$
- Set of Production rules $P$:
  $S \rightarrow aSb \mid T \mid R$
  $T \rightarrow aT \mid a$
  $R \rightarrow bR \mid b$
- Start State = $S$

The above provided grammar accepts the language $L_2$.

# V Problem 5 solution

In this problem, we were asked to show that the language $L = \{a^n b^m | gcd(n,m) = 1 \text{ and } n,m \geq 0\}$ is not regular. I will use the pumping lemma for proving this.

1. Let's say that the opponent provides us with a $p \geq 0$.

2. We will first look for two primes $m$ and $n$ with $m \neq n$, both of which are greater than $p$ and then choose the string $w = a^m b^n$. Note that, since both $m$ and $n$ are different primes, they have to be co-prime to each other as well. Hence $w$ belongs to $L$ and $|w| \geq p$.

3. For all partitions of $w = xyz$ with $|xy| \leq p$, we get that $y$ will contain one or more $a$'s. Let the opponent provides us with $y = a^t, t > 0$ and $t < min(n,m)$.

4. Now, we have to find an $i \geq 0$ such that $xy^i z$ does not belong to $L$. We can see that $xy^i z = a^{m+(i-1)t} b^n$. So, it will not belong to $L$, if we can find an $i$ such that $gcd(m+(i-1)t, n) > 1$. In short, the problem boils down to finding if there exists an $i \geq 0$ such that $m + (i-1)t \equiv 0 \mod n$. From this, the existence of an $i$ is only guaranteed if there exists and inverse of $t$ modulo $n$. Note that this is guaranteed to exist since the inverse is uniquely defined in the field of a prime for all numbers less than that prime. Say, $tt_1 \equiv 1 \mod n$. So, we want

$$\begin{aligned} m + (i-1)t &\equiv 0 \mod n \\ (i-1)t &\equiv -m \mod n \\ i - 1 &\equiv -mt^{-1} \mod n \\ i - 1 &\equiv -mt_1 \mod n \\ i &\equiv 1 - mt_1 \mod n \end{aligned}$$

   Hence, we can find such an $i \geq 0$ and $i = kn + 1 - mt_1, k > 0$ for which $xy^i z$ is not in $L$ and therefore using pumping lemma, the language is not regular.