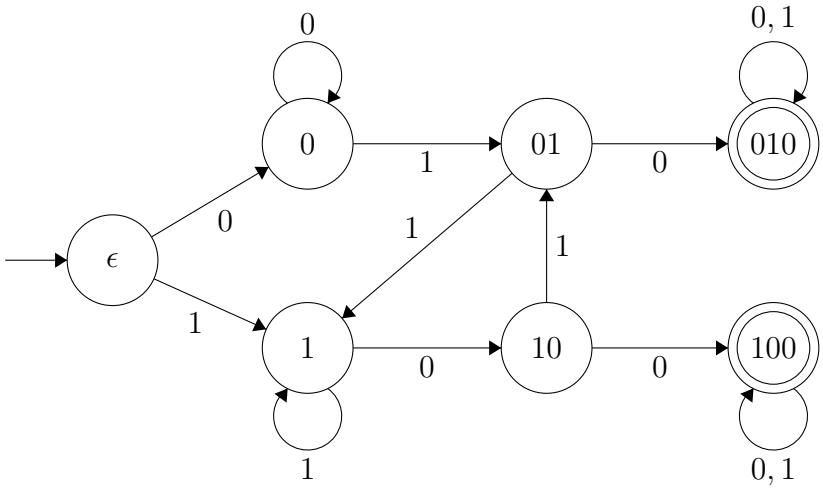
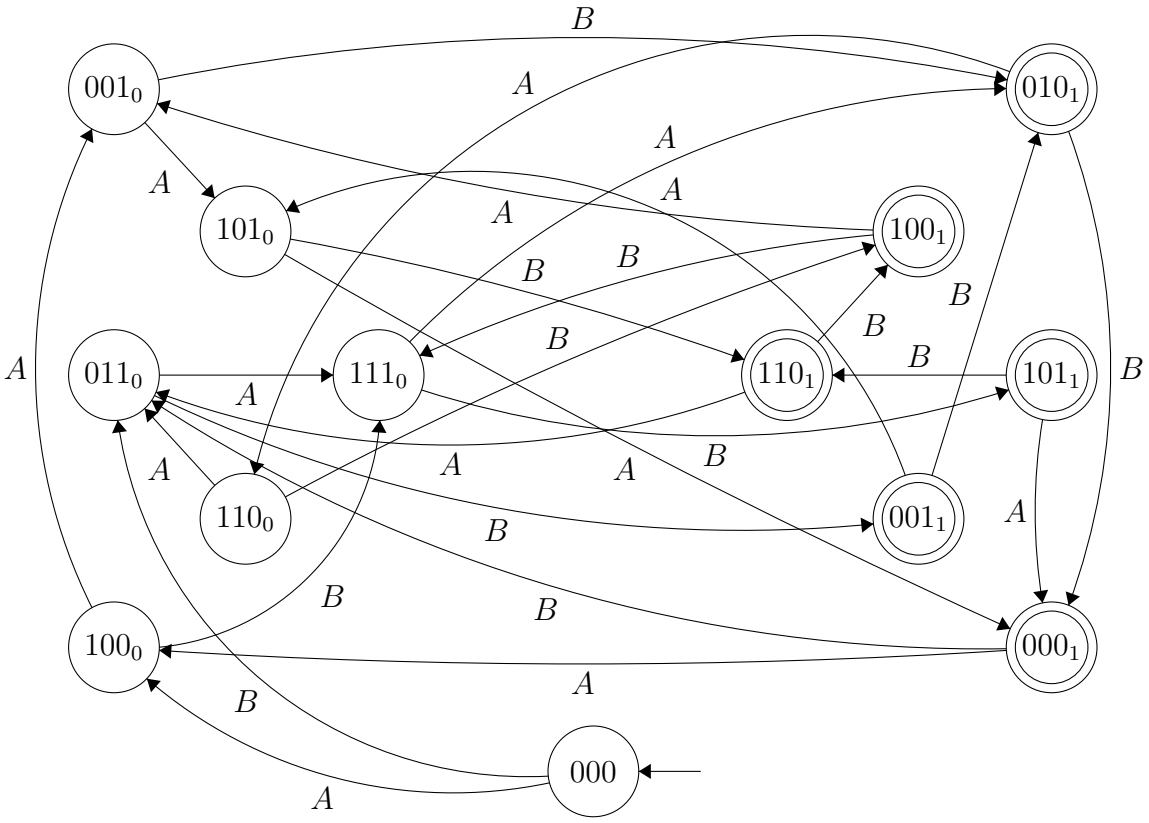
UMC-205 Assignment-1

Aditya Gupta
SR No: 22205

Question 1



Question 2



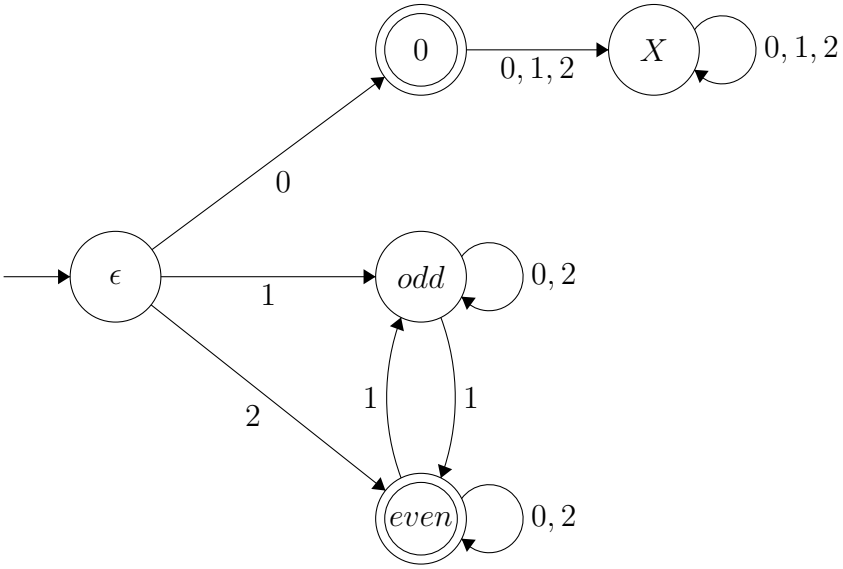
In the above DFA, the state name represents the orientation of the x,y and z lever respectively. '0' refers to the lever in ↙ position while '1' refers to the lever in ↘ position. Using this notation, we describe the states of the DFA.

The subscript '0' at the end of the state refers to the last ball falling in bin C while '1' refers to the last ball falling in bin D. The state '000' represents the initial state of the DFA. We need to accept the strings which end with the last ball falling in bin D. Hence, the finals states are the ones which end with '1' subscript.

If we take all possible permutations of the states, we should get 16 states. However, here we only have 12 states and 1 starting state. The remaining states are unreachable from the starting state. Hence, we can remove them from the DFA.

Question 3

The set of strings in $\{0, 1, 2\}^*$ which are base 3 representations of even numbers is a regular language since we can construct a DFA for it. The DFA is as follows:



When a new digit is added, the entire previous number gets multiplied by 3 and then the new number is added to it. Multiplying by 3 does not affect divisibility by 2. Also, adding 0 or 2 does not affect divisibility by 2. However, adding a 1 changes it. Hence we obtain this DFA where the state can be changed only when we encounter a 1 in the string.

We are also considering the case of the number '0', which is an even number. However, any number with leading 0's will be considered invalid (represented by X state in the DFA). So we make a separate branch in the DFA to account for this case.

Since we want to accept strings which are divisible by 2, the final states are '0' and 'even'.

Question 4

We have to show that for the DFA $\mathcal{A} = (Q, s, \delta, F)$ over the alphabet A, any string $x, y \in A^*$ satisfies:

$$\hat{\delta}(q, xy) = \hat{\delta}(\hat{\delta}(q, x), y)$$

We show this by induction on the length of the string y. Note that the function $\hat{\delta}$ is defined as follows:

$$\hat{\delta}(q, \epsilon) = q \tag{1}$$
$$\hat{\delta}(q, xa) = \delta(\hat{\delta}(q, x), a) \tag{2}$$

Let $P(n)$ be the statement that for any string $x, y \in A^*$ with $|y| = n$, the required property holds.

Base Case: $P(0)$ refers to $|y| = 0 \implies y = \epsilon$.

$$\begin{aligned} \therefore\ \hat{\delta}(\hat{\delta}(q, x), y) &= \hat{\delta}(\hat{\delta}(q, x), \epsilon) \\ &= \hat{\delta}(q, x) && \text{(from 1)} \\ &= \hat{\delta}(q, xy) \end{aligned}$$

Hence $P(0)$ is true.

Inductive Hypothesis: Assume that $P(n)$ is true for some $n > 0$. That is, for any string $x, y \in A^*$ with $|y| = n$, the required property holds.

Consider any string $x, y' \in A^*$ with $|y'| = n + 1$. We can write $y' = ya$ where $y \in A^*$ and $a \in A$. Then we have:

$$\begin{aligned} \hat{\delta}(\hat{\delta}(q, x), y') &= \hat{\delta}(\hat{\delta}(q, x), ya) \\ &= \delta(\hat{\delta}(\hat{\delta}(q, x), y), a) && \text{(from 2)} \\ &= \delta(\hat{\delta}(q, xy), a) && \text{(by inductive hypothesis)} \\ &= \hat{\delta}(q, xya) && \text{(from 2)} \\ &= \hat{\delta}(q, xy') \end{aligned}$$

Hence $P(n + 1)$ is true.

By the principle of mathematical induction, $P(n)$ is true for all $n \geq 0$. Hence, the required property holds for all strings $x, y \in A^*$.

Question 5

The language mentioned in the question is not a regular language. To show this, we show that the pumping lemma does not hold for this language.

To demonstrate this, we use the demon-human game for the pumping lemma. Let the demon provide an integer $k > 0$ to the human. The human would then provide him the string:

$$t = c(/*)^k c(*/)^k c$$

The human would decompose the above string as:

$$x = c$$
$$y = (/*)^k$$
$$z = c(*/)^k c$$

Now the demon has to break y into uvw such that v is not an empty string. Now, we check two cases here:

1. **len(v) is odd:** In this case, v has either an odd number of '/' or an odd number of '*'. In either case, the string xuv^2wz would have an odd number of either '/' or '*'. Hence it would not be in the language. Thus, the demon loses.
2. **len(v) is even:** In this case, v has an even number of both '/' and '*'. Now, the string xuv^2wz would have more (/*) pairs on the left of the middle c compared to the number of (*/) pairs on the right. Hence, we would have some instance of (/*) which is not closed by (*/). So, the string would not be in the language. Thus, the demon loses.

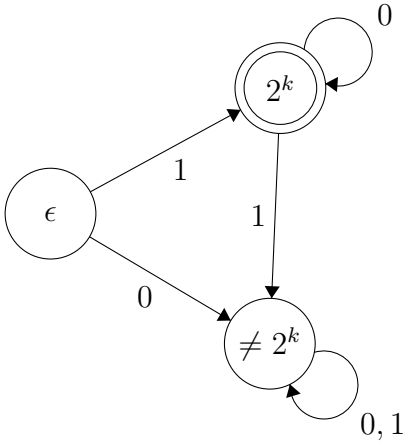
In both cases, the human has a winning strategy by simply choosing i = 2. Hence, the language is not regular.

Question 6

Propostion A: For all X, if binary(X) is regular then so is unary(X)

This statement is false. We can demonstrate this using a counter-example.

Let $X = \{2^k \mid k \geq 0\}$. Then binary(X) = $\{1a^k \mid k \geq 0\}$. This is a regular language since we can construct a DFA for it. The DFA is as follows:



However, unary(X) = $\{1^{2^k} \mid k \geq 0\}$, which is not a regular language. This can be proved by showing that the language does not satisfy the pumping lemma for regular languages (already discussed in class).

Hence, the proposition is false.

Propostion B: For all X, if unary(X) is regular then so is binary(X)

This statement is true. To show this, we use the fact that regular languages satisfy the ultimately periodic criteria.

The value of a number is the number of times '1' would appear in its unary representation. For example, the value of 111 is 3. Thus, the set X is:

$$X = \{len(a) \mid a \in \text{unary}(X)\}$$

However since unary(X) is a regular language, the set of lengths of elements of unary(X) is ultimately periodic, ie it is the union of a finite set and arithmetic progressions with the same difference. But the set X is precisely the set consisting of lengths of elements of unary(X). Hence, the set X is also a union of a finite set and arithmetic progressions with the same period.

Now we construct the set binary(X). To do this, we just write down the elements of X in their binary representation. By doing this, the set binary(X) is also a union of a finite set and arithmetic progressions with the same period since the difference between the elements in binary(X) is the same as the difference between them in X.

Now we wish to construct a DFA for the language binary(X). Let the period of an arithmetic progression $a = (a_1, a_1 + p, a_1 + 2p, \ldots)$ be p. We construct a DFA which keeps track of the remainder of the number when divided by p. The DFA can be represented as $\mathcal{A} = (Q, s, \delta, F)$ where:

$$\begin{aligned}
Q &= \{\epsilon, 0, 1, 2, \ldots, p-1\} \\
s &= \{\epsilon\} \\
F &= \{a_1 \bmod p\} \\
\delta(\epsilon, 0) &= 0 \\
\delta(\epsilon, 1) &= 1 \\
\delta(q, a) &= 2q \bmod p \text{ if } a = 0 \\
&= (2q+1) \bmod p \text{ if } a = 1
\end{aligned}$$

This is a well defined DFA since the remainder of a number when divided by p is always between 0 and $p-1$. Also, the DFA is deterministic since for each state and input, there is only one possible transition. This gives us a DFA for a.

We can do this for every arithmetic progression in binary(X) and construct their respective DFA's. We can also easily construct a DFA to accept the elements in the finite set. Since regular languages are closed under union, we take the union of all these DFA's to get a DFA for binary(X). Hence, binary(X) is a regular language.

Note: Since all the arithmetic progressions have the same period, we can construct a single DFA for all of them. We use the same Q, s, δ with $F = \{a_1 \bmod p, b_1 \bmod p, \ldots\}$ where $\{a_1, b_1, \ldots\}$ are the first terms of the arithmetic progressions.