**CSC301**

**Lecture 7 summary**

1910456

---------------------------------------------------------------------------------------------------------------

---------------------------------------------------------------------------------------------------------------

For an NFA that doesn’t have any - transitions, we follow the steps shown in last lecture.

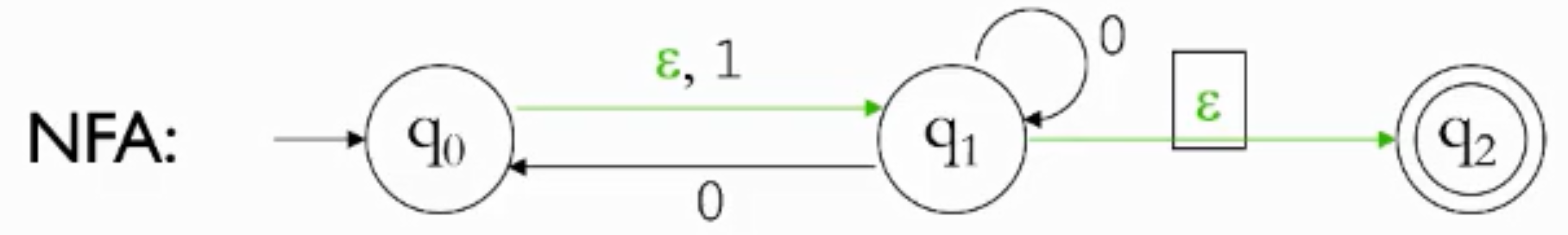
If an NFA has transitions, we must eliminate it first.

----------------------------------

**Removing transitions.**

----------------------------------

[FIG 1: NFA with empty transitions]



To start the process, we first list out the other states we would be in if we were in one specific state (due to the  transitions).

For this Example,

q0 = {q0, q1, q2}

q1 = {q1, q2}

q2 = {q2}

----------------------------------

So, if we are in a specific state, and we receive a string, we have to check what happens for all the other states that we are in according to our initial list.

For this example, if we are in state q0, and we read 0, we see that ,

(q0, 0) but,

(q1, 0) {q0, q1, q2}

(q2, 0)

We checked all the states and ultimately, we can determine that if we read 0 from state q0, we get a self-loop, an arrow to q1, and an arrow to q2.

Now we check what happens if we get 1 while in state q0,

(q0, 1) q1, (q1, 1) , (q2, 1)

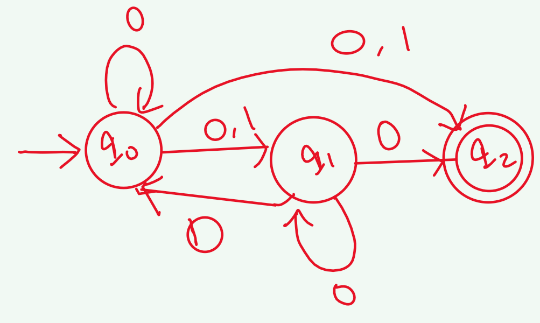
So, we seem to be on q1 only, but being on q1 also means we are at {q1, q2}.

Therefore, from q0, an ‘1’ arrow will point to q1 and q2.

----------------------------------

We do the same thing for all other states and we arrive at this NFA which doesn’t have any transitions.

[FIG 2: NFA with empty transitions removed]



But we are not done. Recalling our initial list, q0 = {q0, q1, q2},

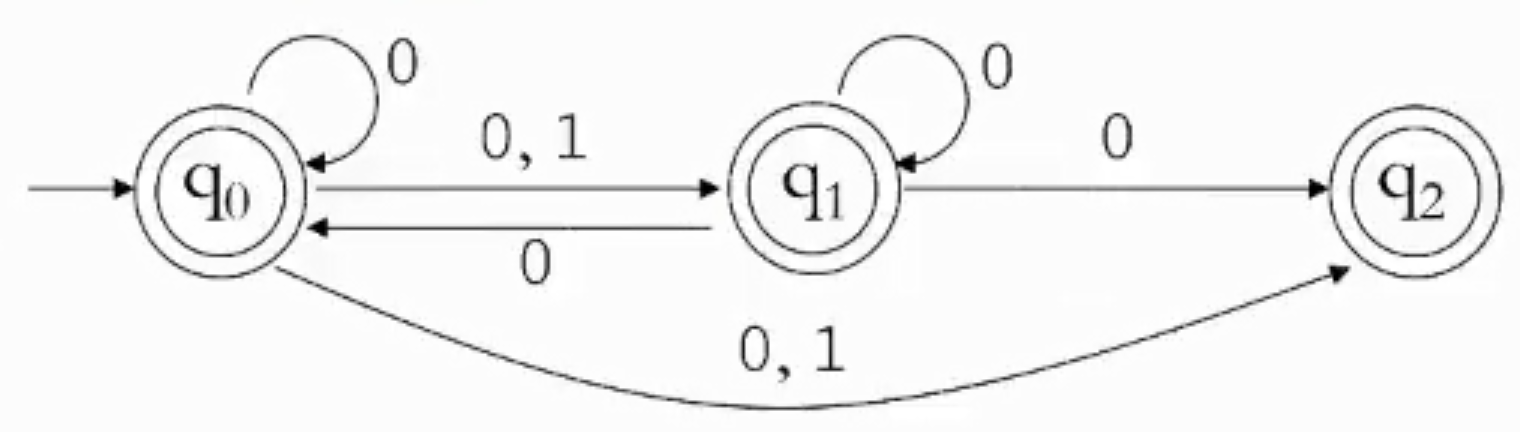
If we are in q0 we are also in q2 which is a **final state**.

we make q0 a final state.

By the same logic, q1 is also a final state as q1 = {q1, q2}.

So, our NFA now looks like this.

[FIG 3: NFA with final states taken into account]



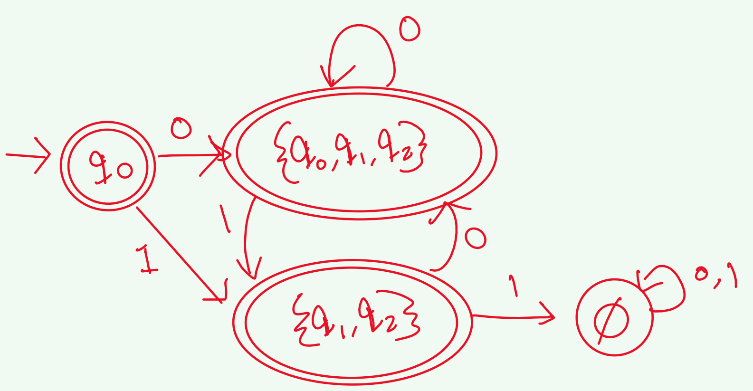
----------------------------------

**Making the DFA**

----------------------------------

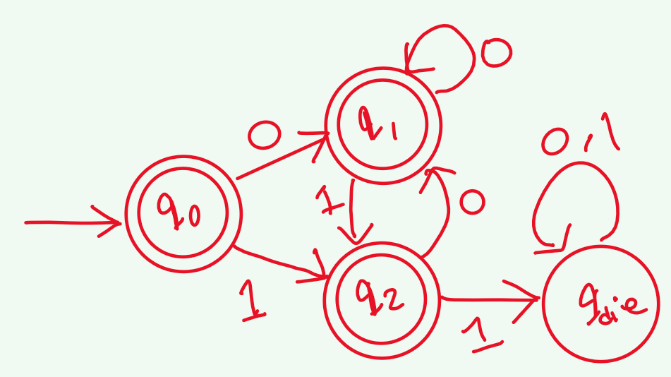
We already know how to make a DFA from an NFA that has no transitions.

[FIG 4: DFA showing grouped states]



So, our final DFA looks like this.

[FIG 5: Final DFA]



---------------------------------------------------------------------------------------------------------------

**NFA to Regular Expressions**

----------------------------------

The first thing that we need to do when attempting these types of problems is making sure that,

There is only **ONE** Final state.

**Initial State** does not have any incoming arrows and,

**Final State** does not have any outgoing arrows.

----------------------------------

If our NFA doesn’t meet the requirements, we have to create new states.

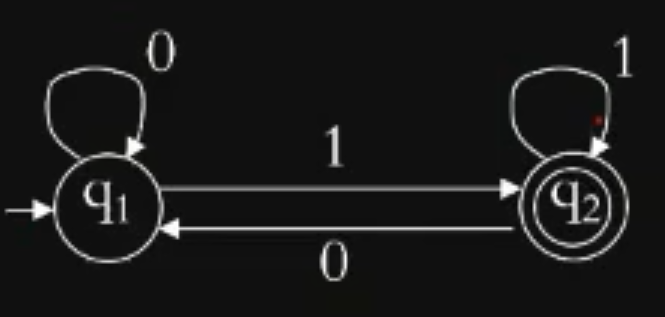
The new initial state will have empty transitions going to the original NFAs, initial state.

The new Final State will have empty transition(s) from original Final State(s) coming into the state.

----------------------------------

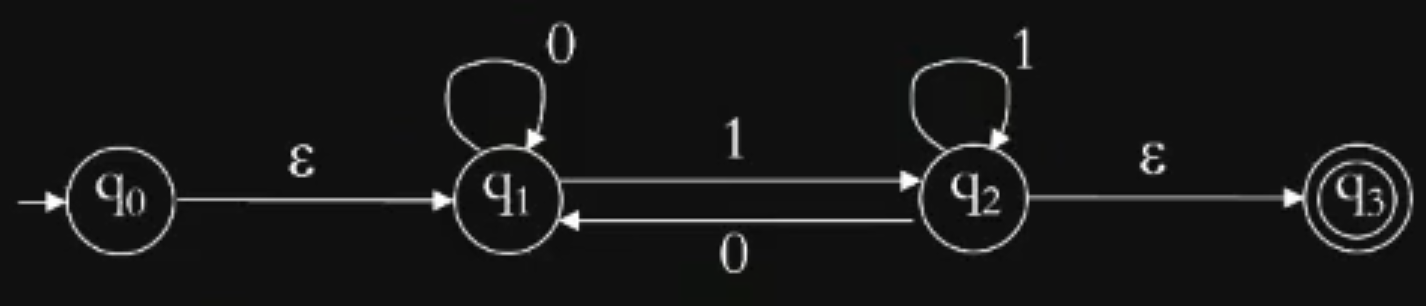
For Example, if we are given this NFA,

[FIG 6: Example NFA]



We firstly, have to change it to:

[FIG 7: NFA with added final and initial state]



----------------------------------

**Now we start removing states**

----------------------------------

**We start with removing q1,**

Which means we are trying to move directly from q0 to q2.

So, we have to figure out what steps to take from q0 to reach q2.

From the NFA, we see that we need a , then as many ‘0’s as we want, then a ‘1’.

So 0\*1 or 0\*1.

But after we remove q1, we notice that there is no “U turn” path from q2  q1 q2.

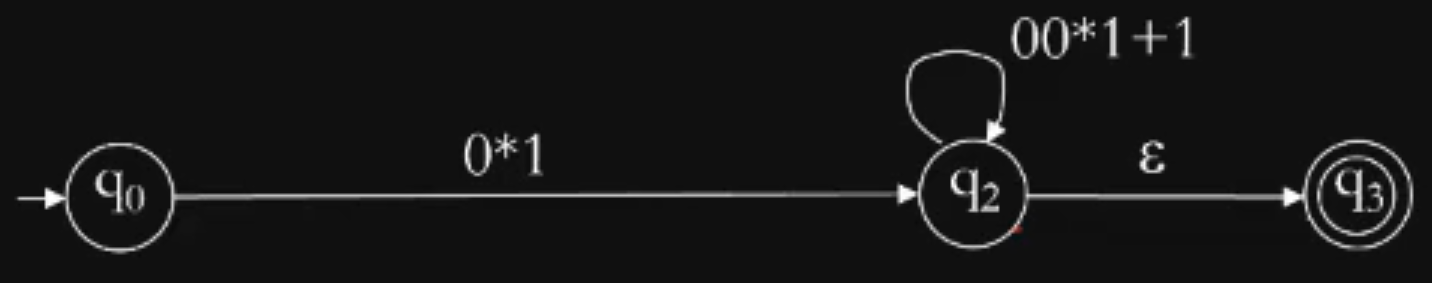
We can rectify this by seeing what steps are needed for the loop to occur.

From q2, firstly we need a ‘0’ to go to q1 and as many ‘0’s as we want and then a ‘1’ again to make the loop and return to q2.

So, we add a self-loop of 00\*1 at q2.

Since there was already a self-loop in q2, we merge the two in to one. i.e 00\*1+1.

[FIG 8: NFA with q1 removed]



----------------------------------

After removing q1, **we now remove q2**,

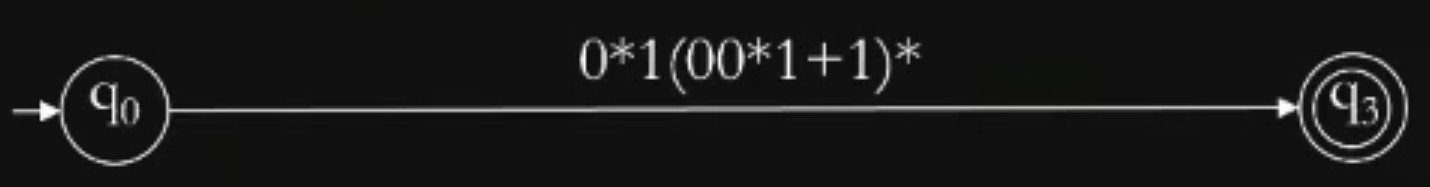
So, we are looking to go directly from q0 to q3.

The actions we take are,

A “0\*1” to go to q2 and then as many “00\*1+1”s as we want, then a to go to q3.

So we can write it as (0\*1) (00\*1+1)\*.

[FIG 9: NFA with q2 removed]



----------------------------------

**Then we check if the strings generated by our obtained regex lead us to the final state in the original NFA that we had.**

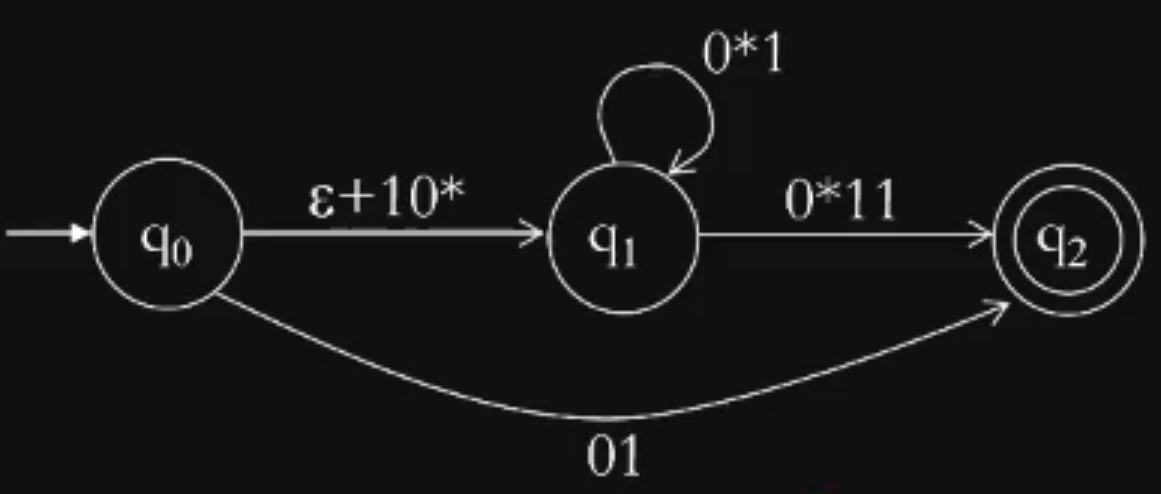
---------------------------------------------------------------------------------------------------------------

**Generalized NFAs**

----------------------------------

These are simply NFAs, whose transitions are labeled by regular expressions.

[FIG 10: GNFA Example]



[Other examples discussed in class]