CSC301

Lecture 7 summary

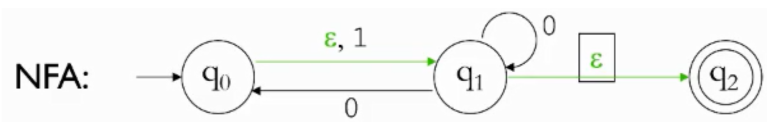
1910456

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

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

Removing ε 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 ε transitions).

For this Example,

q_0 = {q_0, q_1, q_2}

q_1 = {q_1, q_2}

q_2 = {q_2}

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 q_0, and we read 0, we see that ,

(q_0, 0)→ $\varnothing$ but,

(q_1, 0)→ {q_0, q_1, q_2}

(q_2, 0)→ $\varnothing$

∴ We checked all the states and ultimately, we can determine that if we read 0 from state q_0, we get a self-loop, an arrow to q_1, and an arrow to q_2.

Now we check what happens if we get 1 while in state q_0,

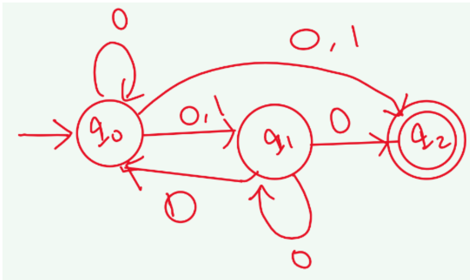
(q_0, 1)→ q_1, (q_1, 1)→ $\varnothing$, (q_2, 1)→ $\varnothing$

So, we seem to be on q_1 only, but being on q_1 also means we are at {q_1, q_2}.

Therefore, from q_0, an '1' arrow will point to q_1 and q_2.

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

[FIG 2: NFA with empty transitions removed]



But we are not done. Recalling our initial list, $q_0 = \{q_0, q_1, q_2\}$,

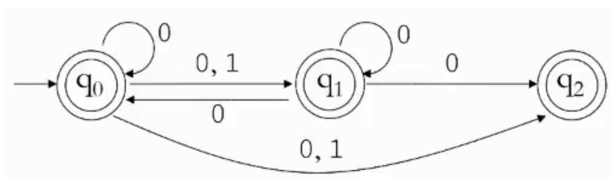
If we are in q_0 we are also in q_2 which is a **final state**.

$\therefore$ we make q_0 a final state.

By the same logic, q_1 is also a final state as $q_1 = \{q_1, q_2\}$.

So, our NFA now looks like this.

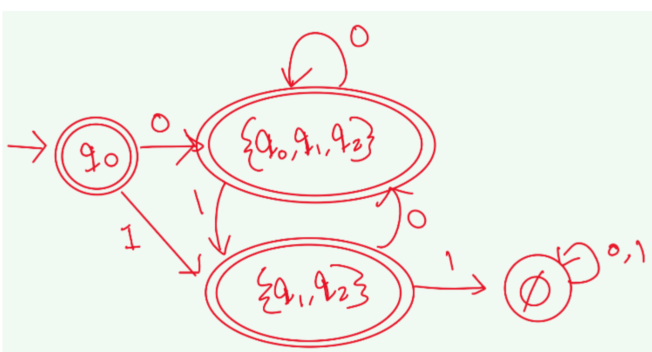
[FIG 3: NFA with final states taken into account]



Making the DFA

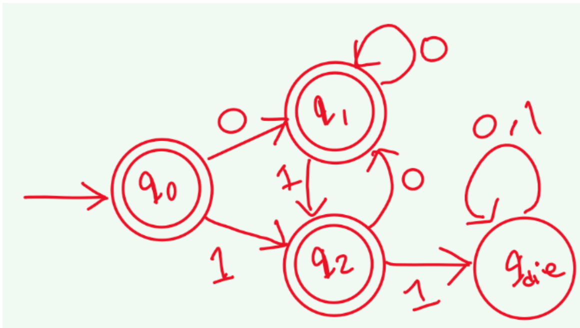
We already know how to make a DFA from an NFA that has no ε transitions.

[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.

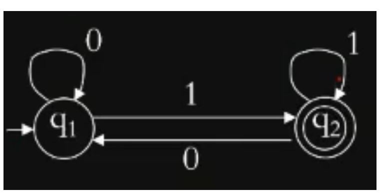
If our NFA doesn't meet the requirements, we have to create new states.

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.

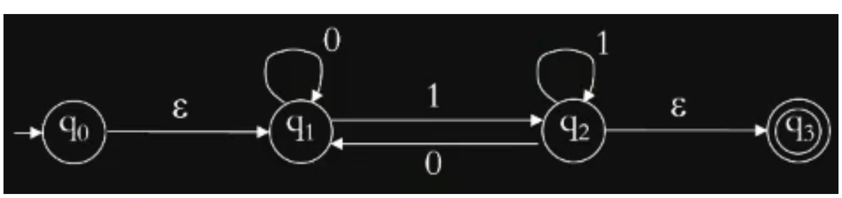
For Example, if we are given this NFA,

[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 q_1,

Which means we are trying to move directly from q_0 to q_2.

So, we have to figure out what steps to take from q_0 to reach q_2.

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

So ε0*1 or 0*1.

But after we remove q_1, we notice that there is no "U turn" path from $q_2 \rightarrow q_1 \rightarrow q_2$.

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

From q_2, firstly we need a '0' to go to q_1 and as many '0's as we want and then a '1' again to make the loop and return to q_2.

So, we add a self-loop of 00*1 at q_2.

Since there was already a self-loop in q_2, we merge the two in to one. i.e 00*1+1.

[FIG 8: NFA with q_1 removed]

After removing q_1, **we now remove q_2**,

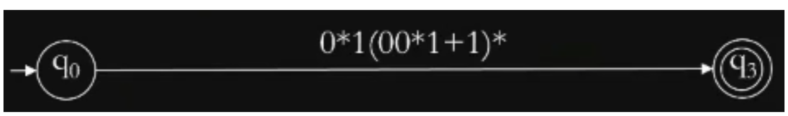
So, we are looking to go directly from q_0 to q_3.

The actions we take are,

A "0*1" to go to q_2 and then as many "00*1+1"s as we want, then a ε to go to q_3.

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

[FIG 9: NFA with q_2 removed]

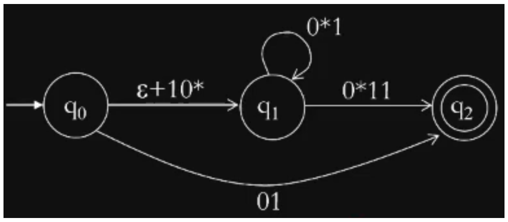


Then we check if the strings generated by our obtained regex lead us to the final state in the original NFA that we had.

Generalized NFAs

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

[FIG 10: GNFA Example]



[Other examples discussed in class]