Prep Work 4 - Regex Conversions

CS 234

Daniel Lee

1. Conversions

1. Look at figure 6.2. In your own words, how does it make an automaton for the union (+) of the languages of NFA1 and NFA2?

We create a new start state and connect it to the NFA1's and NFA2's start state via λ-transitions.

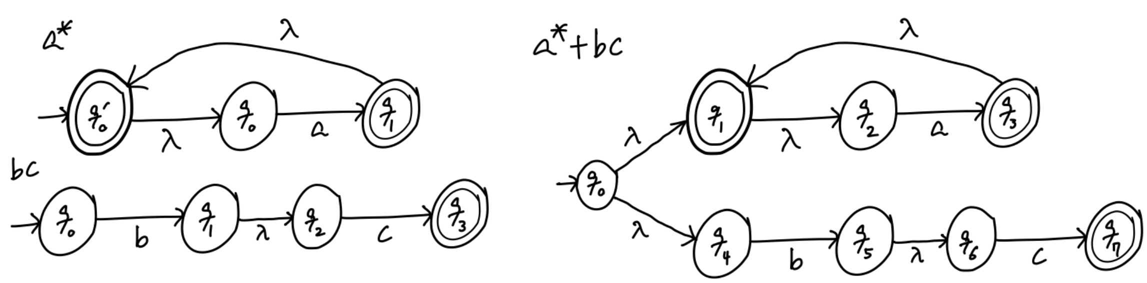
2. Look at figure 6.3. In your own words, how does it make an automaton for the concatenation (·) of the languages of NFA1 and NFA2?

We add λ-transitions from each accepting states from the first λ-NFA to the start state of the second λ-NFA and then convert those accepting states to non-accepting states.

3. Look at figure 6.4. In your own words, how does it make an automaton for the Kleene star (∗) of the language of NFA1?

First, we create a new start state that is also an accepting state. Second, we add a λ-transition from this new start state to the start state of the original λ-NFA. Third, we add λ-transitions from the respective accepting states of the original λ-NFA to the new start state that we have initially created.

4. Try converting a* + bc to an εNFA using the method described in section 6.2.



1. In your own words, how does the transition labelled RS*T added in step 2.a.i.B preserve the set of paths from A to C?

The regex RS*T preserves the set of all possible paths reaching from state A to state C by taking into account the fact that the state B in the automaton from step 2.a.i.A has a self-loop and receives input R from state A and outputs T to state B.

2. In your own words, why is R* the associate regex for the automaton in step 2.b.i?

It is because the automaton in step 2.b.i has only one state and it is a starting-accepting with a self-loop, which means that the final and start state are the same.

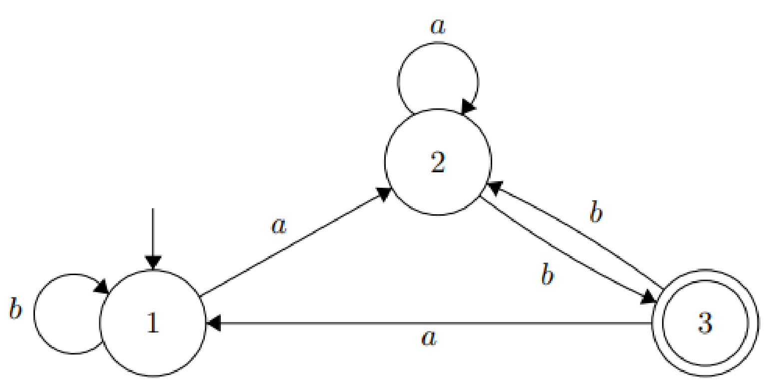
3. In your own words, why is (R + ST*U)*ST* the associate regex for the automaton in step 2.b.ii?

This is because the automaton in step 2.b.ii can have the transitions that end at the start state including a self-loop at the start state (R) or reaching to the end state and returning back to the start state (ST*U). All combinations of these options could be expressed as (R + ST*U)*. In the end, we should reach the end state from the start state, and that case could be expressed as ST*. So (R + ST*U)*ST* is the associate regex for the automaton in step 2.b.ii.

4. In your own words, why do we sum together the regexes obtained for each final state in step 3?

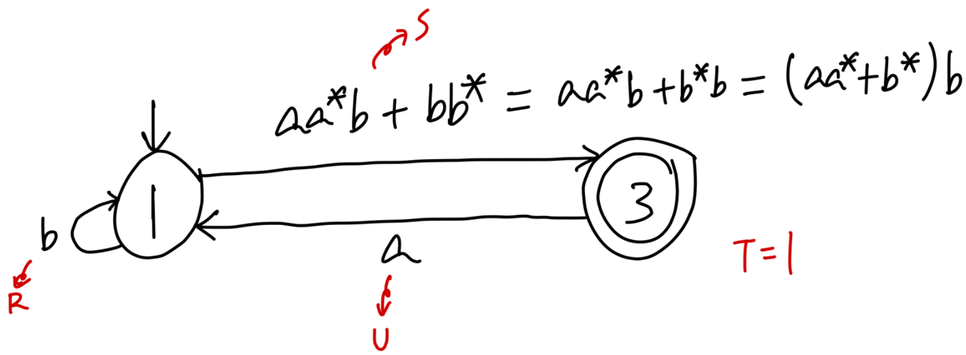
It's because there could be multiple accepting states and the algorithm is based on obtaining a regex for each final state. So in order to obtain the regex for the language of the original automaton, we should get a union of all obtained regexes before finishing the algorithm.

5. If you were to run the algorithm by hand on the following automaton, you would only need to eliminate state 2. What are all the pairs of incoming-outgoing transitions to consider for state 2?



{(a, b), (b, b)}

6. Run the algorithm by hand on the above automaton to find a regex for its language.



$$(R+ST^*U)^*ST^* = (b+(aa^*+b^*)b1^*a)^*(aa^*+b^*)b1^*$$