University of Victoria

CSC 320 - Spring 2023

Foundations of Computer Science

Tutorial 02

Teaching Team

Learning Outcomes:

- Become familiar with DFAs and NFAs.
- Become familiar with the concept of Closure.
- Become familiar with the concept of Kleene Star.
- An introductory level of understanding of Reduction.

Interesting Article:

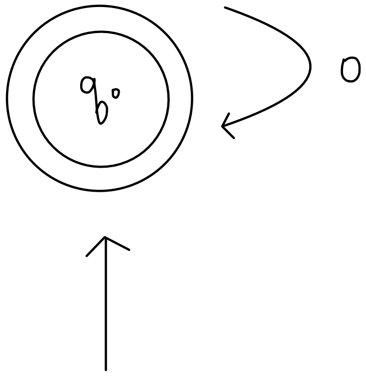
"On Theory of Regular Languages with the Kleene Star Operation" [1]

January 24th, 2023

Question 01

Give the formal specification of a DFA for the following language:

$$L = \{0\}^* \text{ over } \Sigma = \{0\}$$



DFA D

Remember

Formal Definition: $D = (Q, \Sigma, \delta, q_0, F)$

Q : finite set of states

Σ : finite set alphabet

$\delta : Q \times \Sigma \rightarrow Q$

$q_0 \in Q$: start state

$F \subseteq Q$: set of accept (or final) states.

$$\boxed{D = (\{q_0\}, \{0\}, \delta, q_0, \{q_0\}).}$$

Question 02

Give the formal specification of a DFA for the following language:

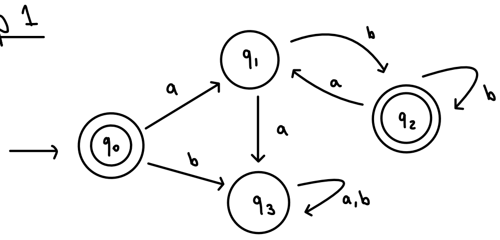
$$L = \{w \in \{a,b\}^* \mid w \text{ is any string } \underline{\text{not in}} \ (ab^+)^*\}$$

We can start by constructing a DFA that recognizes $(ab^+)^*$, and then use the complement $(\overline{L})$ to get strings not in $(ab^+)^*$.

Step 1

Why? Not : a, b, aa, aab, bb, bba, aba, etc.

In : ε, ab, abb, abbb, etc.



DFA D_1

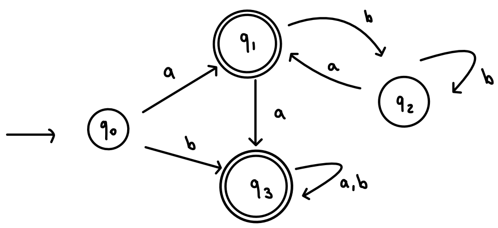
Step 2

Thus, this DFA D_1 recognizes $\overline{L}$.

$D_1 = \{ \{q_0, q_1, q_2, q_3\}, \{a,b\}, \delta, q_0, \{q_0, q_2\})$

Step 3

Now we can complement the previous DFA D_1 by switching accept and non-accept states.



DFA D_2

Step 4

We can now write the formal specification of our DFA D_2 for language L.

Note: This is very important! Why? Words and structure matters! Always be precise.

$D_2 = (Q, \Sigma, \delta, q_0, F)$, where

$Q : \{q_0, q_1, q_2, q_3\}$

$\Sigma : \{a, b\}$

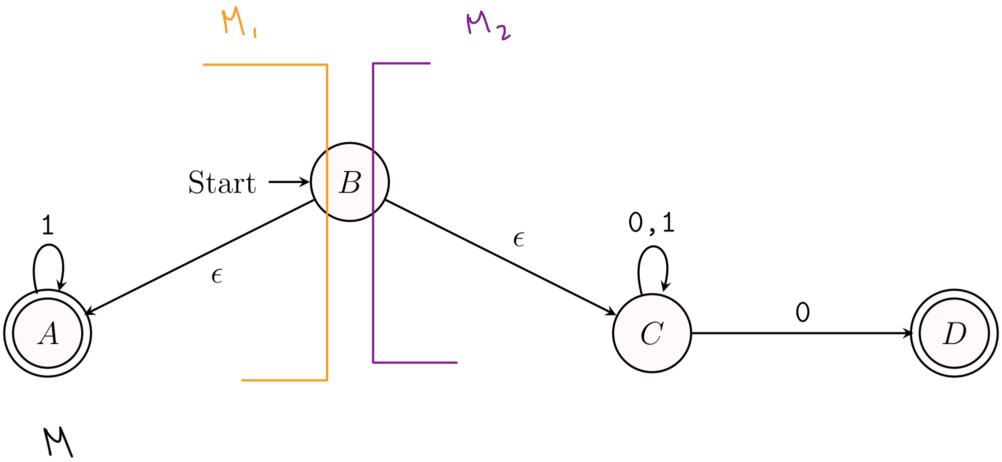
δ : Defined by our state diagram.

$q_0 : q_0$

$F : \{q_1, q_3\}$

Question 03

Consider the following state diagram:



(a) Is the string 0011 accepted by this state machine? How about 1100?

0011

B — A — cannot read string
B — C — read 0011 with C loop transition
B — C — D — cannot read string after th first 0

No, it is not accepted.

1100

B – A — cannot read string after th first and second 1
B – C – read 110 with C loop transition — D

Yes, it is accepted.

(b) What is the language of this machine?

Let our machine be M.
We can view our machine M as two machines, let us say M_1 and M_2.

① M_1 accepts any string containing any number of only 1's, thus, $\{1\}^*$. 1, 11, 111, etc.

② M_2, accepts any binary string ending in 0, thus, $\{0,1\}^* 0$.
1100, 10, 0, etc.

Therefore, we can use union to create L(M).

$$L(M) = \{1\}^* \cup \{w0 \mid w \in \{0,1\}^*\}$$

Note: It is important to include w such that x in our statement.

Question 04

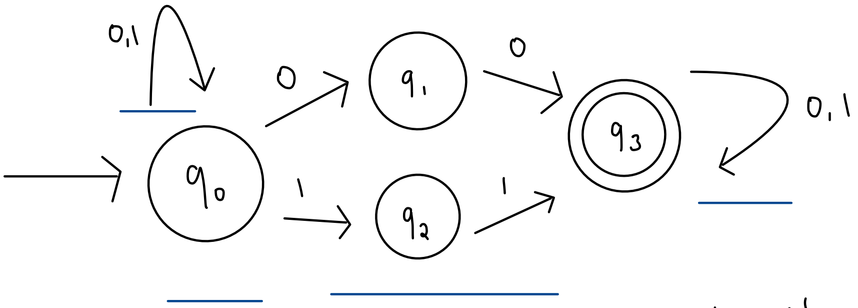
Design an NFA state diagram for the following language:

$$\{w \in \{0,1\}^* \mid w \text{ contains } 00 \text{ or } 11 \text{ as a substring}\}$$

We can begin by defining what is and isn't include (i.e., build our intuition).

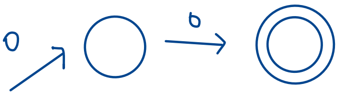
Not: 0, 1, ε, 01, 10, etc.

In: 00, 11, 0011, 000, 111, 1111, 110, etc.



NFA N

- ☐ Start state cannot be an accept state
- ☐ Next, we remember that an NFA does not require all transition to include th the entire alphabet and allow multiple state outputs for a single element.
  - ☐ So, we want to require 00 or 11 be present.



- ☐ We note that we now want to allow any combination of 0,1 before and after our require substring.

i.e., not allow 0, 01, etc. but allow 000, 0100.
So we need to have a self loop of (0,1) and an exit for 0 and 1.

Thus, we obtain our NFA N, that accepts our language. ■

DFA Union Closure

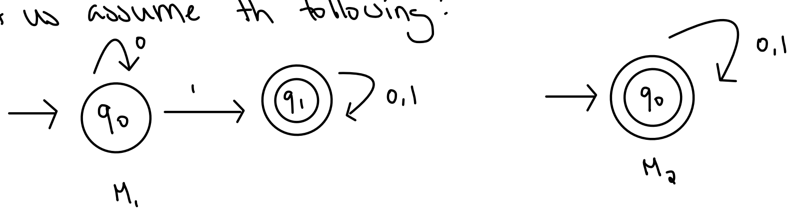
Regular languages are closed under union.

What does "closed" mean?

> A set S is closed under operation O if $O(S) \in S$.
>
> Let $S = \{a, b, c\}$. Define O as such: $O(a) = b$, $O(b) = c$, and $O(c) = a$.
>
> Notice that applying O yields elements that are all in set S. So S is closed under O. If O were defined the same but $O(c) = z$, then S is no longer closed under O.

Example

Let us assume th following:



So, a language L is regular if there exists a DFA that recognizes L.

Thus, $L(M_1)$ and $L(M_2)$ are regular by definition.

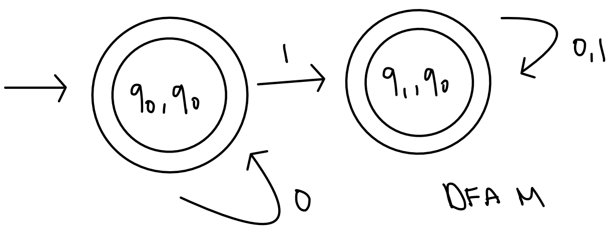
Thought

Is $L(M_1) \cup L(M_2)$ regular? How would we know and prove?

Idea

Since we know the definition, we can prove by building a DFA for our union language.

We can begin with "gluing" the states together.



M_1

δ	0	1
q_0	q_0	q_1
q_1	q_1	q_1

M_2

δ	0	1
q_0	q_0	q_0

We include ALL accept states in either M_1 OR M_2.

So, they become accept states in "glued" states.

Therefore, we obtain a DFA M that accepts $L(M) = L(M_1) \cup L(M_2)$.

And we can see that it is regular by definition. ■

Kleene Star Proof

Prove that regular languages are closed under Kleene star.

Review

We remember that Kleene star is the concatenation of a set $\emptyset$ or more times with itself.

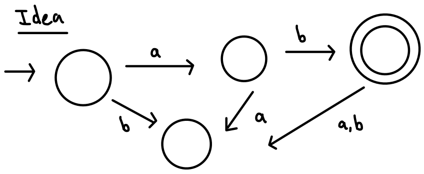
e.g., $L^* = \{\epsilon\} \cup L \cup LL \cup LLL \cup \ldots$

In class it was shown that regular languages are closed under concatenation.

— L_1L_2 is regular if L_1 and L_2 are regular.

For Kleene star, we can take $L_2 = L_1$ so we have L_1L_1 since L_1 is regular than L_1L_1 is regular.

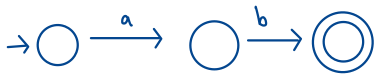
Idea



Given DFA D_L

$D_L = \{Q_L, \Sigma_L, \delta_L, q_0^L, F_L\}$

Remember there is an equivalent NFA N_L



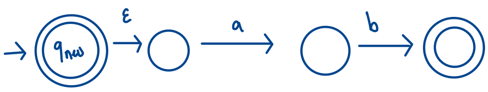
We can use general proof from class. Thus, it could stay the same.

We now can create an NFA N to recognize L^+

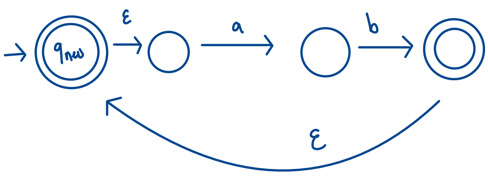
$N = \{Q, \Sigma, \delta, q_0, F\}$

Add new start state $q_0 = q_{new}$, q_{new} will also be an accept state (i.e., $q_{new} \in F$).

So, q_{new} will transition to start state of D_L via ε (i.e., $\delta(q_{new}, \varepsilon) = q_0^L$).



The final states of D_L will transition to newly added start state q_{new} via ε (i.e., $\delta(a, \varepsilon) = q_{new}, a \in F_L$).



Proof

If L is a regular language, there exists a DFA D_L which recognizes L.

$D_L = \{Q_L, \Sigma_L, \delta_L, q_0^L, F_L\}$.

We construct an NFA N that recognizes L^*.

$N = \{Q, \Sigma_L, \delta, q_0, F\}$.

Where,

$Q = Q_L \cup \{q_{new}\}$

$q_0 = q_{new}$

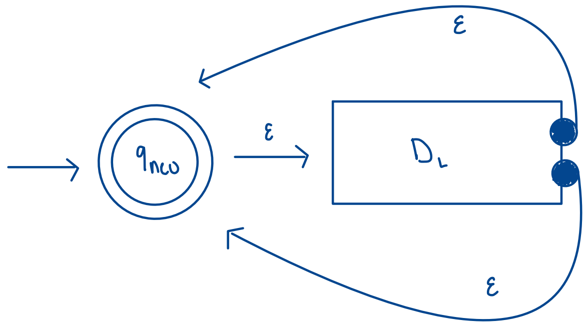
$F = F_L \cup \{q_{new}\}$

We define δ so that for any state $q \in Q$ and $a \in \Sigma_L$.

$$\delta(q, a) = \begin{cases} \{\delta_L(q, a)\}, & q \neq q_{new} \text{ and } a \in \Sigma. \\ \{q_{new}\}, & q_0 \in F_L,\ a = \varepsilon. \\ \{q_0^L\}, & q = q_{new},\ a = \varepsilon. \end{cases}$$

Correctness

General Idea



We want to show that

$L^* \subseteq L(N)$

Let $w \in L^*$, $w = w_1, w_2, \ldots, w_k$ where $w_i \in L$ or $w = \varepsilon$.

- Since $q_0 \in F$, ε is accepted by N and...
- Since $\delta(a, \varepsilon) = q_0$, $a \in F_L$ and $\delta(q_{new}, \varepsilon) = q_0^L$.

N will loop around to the start state of D_L and follow the states and transitions to only accept strings $w_i \in L$ giving $w \in L^*$.

Therefore, $L^* \subseteq L(N)$.

Now we want to show that

$L(N) \subseteq L^*$

Let $w \in L(N)$. When computing w we start a q_{new} followed by q_0^L, then the computations follows D_L, and finally the computation ends in a final state of D_L, $F_i \in F_L$.

- This computation can end immediately in q_{new}, which would accept ε.
- Otherwise, the computation ends in F_i, accepting $s \in L$ or any number of concatenations of s when the computation loops from F_i to q_{new}.

Therefore, $L(N) \subseteq L^*$. $\blacksquare$

$$\overbrace{q_{new}, q_0^L, \ldots, F_1}^{L}, \overbrace{q_{new}, q_0^L, \ldots, F_2}^{L}, q_{new}, \ldots$$

Reduction Discussion

Reduction...

Problem A: Will Ammar Brush His Hair?

Problem B: Is Angela Happy?

Reduction:

$$A \longrightarrow B \text{ “A reduces to B”}$$

The outcome of A relies on the outcome of B.

Resources

[1] B. Karlov, "On Theory of Regular Languages with the Kleene Star Operation," English, *Lobachevskii journal of mathematics*, vol. 41, no. 9, pp. 1660–1665, 2020, ISSN: 1995-0802. DOI: `https://doi.org/10.1134/S1995080220090164`.