CS Theory (Fall '25) September 24, 2025

Regular languages are closed under $\circ$ and $*$

Instructors: *William Pires and Toniann Pitassi*

1 Closure under $\circ$

Fix some alphabet Σ. Recall that given two languages L_1, L_2 over Σ we define

$$L_1 \circ L_2 = \{w \mid w = xy, x \in L_1, y \in L_2\}.$$

We want to prove the following:

Theorem 1. *If L_1, L_2 are regular languages over Σ then $L_1 \circ L_2$ is regular.*

Proof. Let L_1, L_2 be regular languages. Let $D_1 = (\Sigma, R = \{r_0, \ldots, r_k\}, r_0, F_1, \delta_1)$ and $D_1 = (\Sigma, S = \{s_0, \ldots, s_\ell\}, s_0, F_2, \delta_2)$ be DFAs for L_1, L_2 respectively. We build an NFA N for $L_1 \circ L_2$ as follows:

- Alphabet: Σ
- Set of states: $R \cup S$.
- Start state: r_0
- Accept states: F_2
- transition function δ' where:

$$\delta'(q, c) = \begin{cases} \{\delta_1(q, c)\} & \text{if } q \in R \text{ and } c \in \Sigma \\ \{\delta_2(q, c)\} & \text{if } q \in S \text{ and } c \in \Sigma \\ \{s_0\} & \text{if } q \in F_1 \text{ and } c = \epsilon \end{cases}$$

[1]

What the above means, is: In the NFA N are states are the ones of D_1 and D_2. We add ϵ-transitions from the former accept states of D_1 to the start state of D_2. The only accept states are the ones in D_2.

See figures Figures 1 and 2 bellow. [2]

[1] Be careful, the transition function δ' is for an NFA so you should use set notation.

[2] The illustration is only here to help you understand the construction. It's not enough to draw a picture when doing this kind of proofs. You need to give a 5-tuple as we did above.

Figure 1: The two DFAs D_1 and D_2

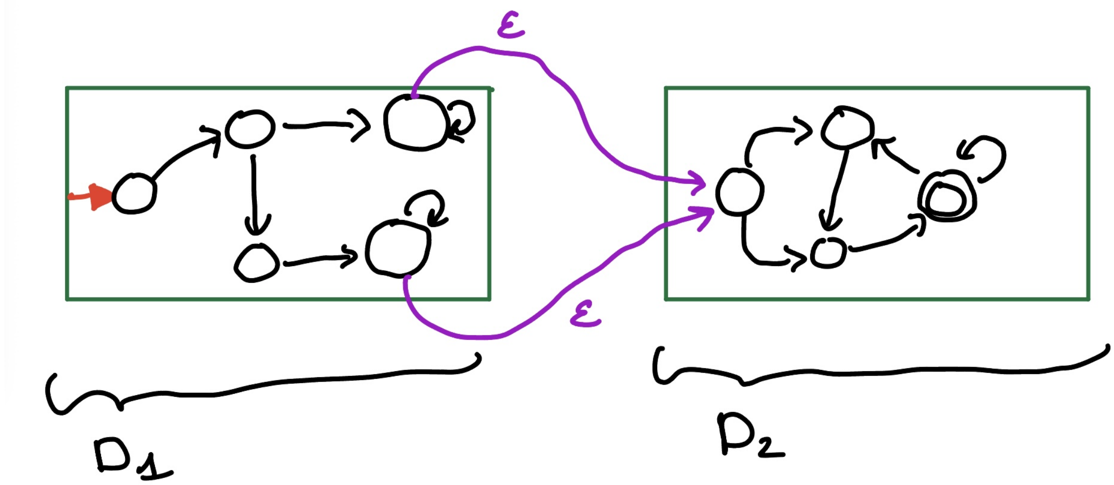


Figure 2: The NFA obtained following the construction

We now need to prove this construction is correct. To do this, we need to include two directions: if $w \in L_1 \circ L_2$ then N accepts w. If N accepts w then $w \in L_1 \circ L_2$.

If $w \in L_1 \circ L_2$ then N accepts w: $L_1 \circ L_2 \subseteq L(N)$

Let $w \in L_1 \circ L_2$. We have that $w = xy$ where $x \in L_1$ and $y \in L_2$. Since $x \in L_1$, D_1 accepts x. So when running D_1 on x, we take a path starting in r_0 and end in an accept state in $r^* \in F_1$. Call this path p_1. Since $y \in L_2$, D_2 accepts y. So when running D_2 on y, we take a path starting in s_0 and end in an accept state in $s^* \in F_2$. Call this path p_2.

Now, in N, on input $w = xy$, we start in r_0. We can first read the x part of w, take the path p_1 and end in r^*. Since $r^* \in F_1$, we take the ϵ transition from r^* to s_0. Now that we're in s_0, we can read the y part of w, take the path p_2 and end in s^*. Since $s^* \in F_2$, s^* is an accept state of N and the NFA accepts w.

If N accepts w, then $w \in L_1 \circ L_2$: $L(N) \subseteq L_1 \circ L_2$ Assume w is accepted by N. Then there is some path p in N, starting in r_0 and ending in an accept state $s^* \in F_2$, such that w can take the path p in N.

In particular, the path must at some reach a state $r^* \in F_1$, take the ϵ transition from r^* to s_0 and then eventually reach s^* (I.e we go from the D_1 part of N to the D_2 part). By construction, this is only time the path can take an ϵ transition. Hence, we can split p into two parts, the first part p_1 between r_0 and r^*, and the second part p_2 between s_0 and s^*. In particular, observe that the path p_1 is made of states only in R (states of D_1) while the path p_2 is made of states only in S (states of D_2).

Now, let x be the sequence of symbols read from while following the path p_1. And let y be the sequence of symbols read while following the path p_2. We have $w = x \circ \epsilon \circ y = xy$, where the ϵ comes from the ϵ-transition taken between r^* and s^*.

Now observe that x must be accepted by D_1. Indeed, when running D_1 on x, we start in r_0, follow the path p_1 and end in $r^* \in F_1$. So $x \in L_1$. Similarly, when running D_2 on y, we start in s_0, follow the path p_2 and end in $s^* \in F_2$. So $y \in L_2$ since D_2 accepts y.

Hence we have that $w = xy$ where $x \in L_1$ and $y \in L_2$.

Conclusion: We have shown $L_1 \circ L_2 \subseteq L(N)$ and $L(N) \subseteq L_1 \circ L_2$. Hence we have that $L(N) = L_1 \circ L_2$, so N is an NFA for $L_1 \circ L_2$. This shows $L_1 \circ L_2$ is a regular language. □

2 Closure under *

Fix some alphabet Σ. Recall that given a language L over Σ we define L^* to be the set of all strings $w \in \Sigma^*$ such that there exists some $k \geq 0$ such that w can be written as $w = u_1 u_2 \dots u_k$ where for every $i \leq k$, $u_i \in L$.

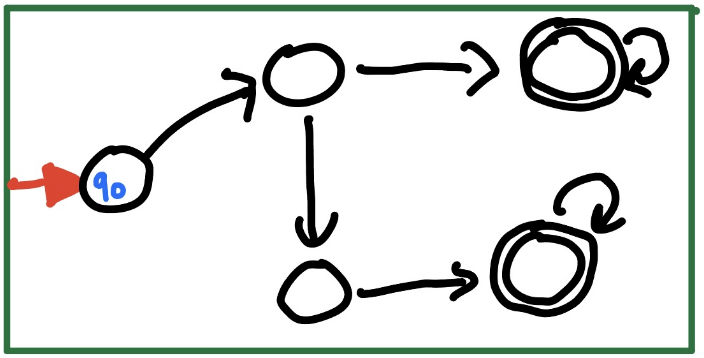
Theorem 2. *If L is a regular language over Σ then L^* is regular.*

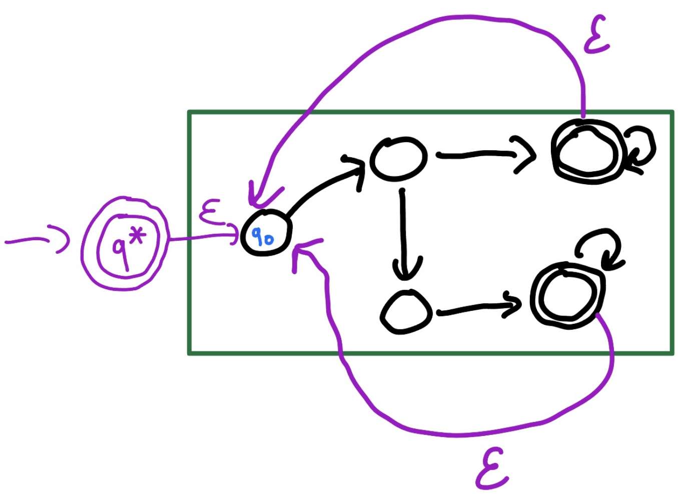
Proof. Let L be a regular language. Let $M = (\Sigma, R = \{r_0, \dots, r_k\}, r_0, F_1, \delta_1)$ be a DFA for L. We will build an NFA, N, for L^* as follows:

- Alphabet: Σ
- Set of states: $R \cup \{q_0\}$.
- Start state: q_0
- Accept states: $F_1 \cup \{q_0\}$
- transition function δ where:

$$\delta(q, c) = \begin{cases} \{\delta_1(q, c)\} & \text{if } q \in R \text{ and } c \in \Sigma \\ \{r_0\} & \text{if } q \in F_1 \text{ and } c = \epsilon \\ \{r_0\} & \text{if } q = q_0 \text{ and } c = \epsilon \end{cases}$$

What the above means is: in the NFA N, the states are all states of M plus a new start state, q_0. The transition function for N keeps all transitions of M and additionally we add: (i) ϵ-transitions from each accept state of M to r_0 (the original start state), and (ii) an ϵ transition from the new start state q_0 to r_0. See the figures below for an example of this construction.





We now need to prove this construction is correct. To do this, we need to prove both directions: if $w \in L^*$ then N accepts w, and if N accepts w then $w \in L^*$.

If $w \in L^*$ then N accepts w: $L^* \subseteq L(N)$

Let $w \in L^*$. Then there exists $k \geq 0$ such that we can write $w = u_1 u_2 \ldots u_k$ such that for every $i \leq k$, $u_i \in L$. If $k = 0$, then $w = \epsilon$, and since q_0 is an accept state of N, ϵ is accepted.

The second case is when $k \geq 1$. As in the above argument for proving closure under concatenation, since every u_i is in L, when running M on u_i, there is an accepting computation path starting in r_0 and ending in an accept state of M. Let's call p_i the accepting path of M on input u_i. Now we will describe the existence of an accepting computation path when we run N on input $w = u_1 u_2 \ldots u_k$. Starting in the start state q_0, we take the ϵ-transition to r_0. Then for $i = 1, \ldots, k$: starting in r_0, we run N on u_i following the path p_i. Since this path ends in an accept state there is an ϵ-transition from this accept state back to r_0 so we follow this ϵ-transition. After processing all u_i's we are guaranteed to be in an accept state since the path p_k ends in an accept state. Therefore N accepts w.

If N accepts w, then $w \in L^*$: $L(N) \subseteq L^*$

Assume w is accepted by N. Then there is some accepting computation path p when we run N on input w, where p starts in q_0 and ends in an accept state of N. The first case is where the path p has length 0. This can only happen when $w = \epsilon$ and then $w \in L^*$ as desired.

The second more complicated case is when the path p has length at least 1, and therefore the first transition taken in p is the ϵ-transition from q_0 to r_0. Now we will partition the path p into parts, based on where the ϵ transitions occur in p: we will write $p = \epsilon p_1 \epsilon p_2 \dots \epsilon p_k$, where each p_i consists of a subpath of p_i (of length at least 1) that contains no ϵ-transitions, and one ϵ-transition is taken between subsequent p_i's. Since there are no ϵ-transitions in any p_i, each p_i must consist of a sequence of edges that correspond to a substring, u_i, of w. Thus the partition of p induces a partition of w as well: w can be written as $w = u_1 \dots, u_k$, where each u_i corresponds to the edge labels on the path p_i.

We want to argue that for each i, u_i is in L. Let's first consider the case where $k = 1$ (the base case). Then $p = \epsilon p_1$, so $w = u_1$, and $w = u_1$ corresponds to the edge labels on the path p_1. In other words, on input $w = u_1$, the path p starts with an ϵ-transition from q_0 to r_0 and then follows the unique path p_1 in the original DFA M (for L) from r_0 to an accept state. Therefore $w = u_1$ is accepted by N as desired.

Now consider the case when $k \geq 2$. In this case $w = u_1 \dots u_k$, $k \geq 2$ and $p = \epsilon p_1 \dots \epsilon p_k$, and $w = u_1 \dots u_k$. Therefore the path $p = \epsilon p_1 \epsilon p_2 \dots \epsilon p_k$ can be broken into phases: In the first phase we process ϵp_1 as follows: starting in q_0 we take the ϵ-transition to r_0, and then we process u_1 following path p_1. In the next phase we process ϵp_2. Now since we take an ϵ-transition immediately after p_1, it must be that p_1 ends in an accept state (since ϵ-transitions can only be taken from an accept state), and therefore u_1 must be in L. Furthermore since all ϵ-transitions go to r_0, the next subpath p_2 must begin in r_0. So now we process u_2 following path p_2 starting in r_0 to finish phase 2. If we are not finished (i.e., if $k \geq 3$), then by the same argument, since there is an ϵ-transition following p_2, path p_2 must end in an accept state, and therefore u_2 is also in L. We continue arguing in the same way to show that $u_1, \dots, u_{k-1}$ must all be in L, and in the last phase, we process the final substring u_k following path p_k starting in r_0. Now since p ends in an accept state, and p ends with p_k, p_k must also end in an accept state, and therefore the last string u_k is also in L.

To summarize: for every $i = 1, \dots, k$, p_i is labelled by a substring u_i of w where $w = u_1 \dots, u_k$. For each i, p_i is the unique path taken in M on input u_i , which as we have argued above must end in an accept state. Therefore each substring $u_1, \dots, u_k$ is in L, and since $w = u_1 \dots u_k$, we can conclude that $w \in L^*$.

□