THE REGULAR OPERATIONS ON LANGUAGES
The Star Operation

- Suppose there is a language $X$.

- The **star** of language $X$ (written as $X^*$) is a language that is composed of all strings that are formed by concatenating 0 or more strings of $X$. The star operation is also called the **Kleene closure**.

- The star of any language includes the empty string $\varepsilon$ and is always infinite.
• The Union Operation

  – Suppose there are two languages $X$ and $Y$.

  – The **union** of languages $X$ and $Y$ (written as $X \cup Y$) is a language that is composed of all strings $w$ such that $w$ is a string from language $X$ or $w$ is a string from language $Y$.

  – Mathematically,

    $$X \cup Y = \{ w \mid w \in X \text{ or } w \in Y \}$$
The Concatenation Operation

- Suppose there are two languages $X$ and $Y$.

- The *concatenation* of languages $X$ and $Y$ (written as $X \circ Y$) is a language that is composed of all strings $w = xy$ such that $x$ is a string from language $X$ and $y$ is a string from language $Y$.

- Mathematically,

$$X \circ Y = \{xy \mid x \in X \text{ and } y \in Y\}$$
– Examples

• Let language \( X = \{aa, bb\} \) and language \( Y = \{bb, cc, dd\} \).

• The union of \( X \) and \( Y \) is:
  \[ X \cup Y = \{aa, bb, cc, dd\} \]

• The concatenation of \( X \) and \( Y \) is:
  \[ X \circ Y = \{aabb, aacc, aadd, bbbb, bbcc, bbdd\} \]

• The star of \( X \) is:
  \[ X^* = \{\varepsilon, aa, bb, aaaa, aabb, aabbaa, bbbb, bbaabb, \ldots\} \]
Definition of Closure

- A set is said to be **closed** under a certain operation if performing that operation on the elements of the set produces an object that is still a member of that set.

- For example, the set of integers is closed under multiplication since multiplying integers always produces an integer.

- The set of integers is not closed under division since dividing two integers does not always produce another integer.
CLOSURE PROPERTIES OF REGULAR LANGUAGES

- Closure Under The Regular Operations

  - Is the family of regular languages closed under union, concatenation and star operations? Specifically,

  1. Is the union of regular languages also regular?

  2. Is the concatenation of regular languages also regular?

  3. Is the star of a regular language also regular?
Is the union of regular languages also regular?

Assume that \( L_1 \) is a regular language recognized by the NFA \( N_1 \):

\[
N_1 = \{Q_1, \Sigma_1, \delta_1, q_1, F_1\}
\]

Assume that \( L_2 \) is also a regular language recognized by the NFA \( N_2 \):

\[
N_2 = \{Q_2, \Sigma_2, \delta_2, q_2, F_2\}
\]

If the union \( L_1 \cup L_2 \) is regular, then there must be an NFA that recognizes it. Let this NFA be \( N_3 \):

\[
N_3 = \{Q_3, \Sigma_3, \delta_3, q_3, F_3\}
\]
The following are the state diagrams for $N_1$ and $N_2$:

The NFA $N_3$ for the union of $L_1$ and $L_2$ should be able to accept a string if it is a member of either $L_1$ or $L_2$.

In other words, $N_3$ should accept an input string if it is accepted by either $N_1$ or $N_2$. 
For every input string, NFA $N_3$ should be able to start two parallel computations.

One computation will try to see or guess if the string belongs to $L_1$. The other computation will try to see or guess if the string belongs to $L_2$.

$N_3$ will then be a combination of $N_1$ and $N_2$. The state diagram for $N_3$ will be:

![State diagram](image)
Because of the ε-transitions from the start state \( q_0 \) to state \( q_1 \) (the start state of \( N_1 \)) and state \( q_2 \) (the start state of \( N_2 \)), \( N_3 \) automatically starts two computations.

One computation “simulates” \( N_1 \) to determine if the input string is a member of \( L_1 \).

The second computation “simulates” \( N_2 \) to determine if the input string is a member of \( L_2 \).

If either computation ends up in a final state, then \( N_3 \) accepts the input string.
Example:
Let \( L_1 = \{ w \mid \text{w ends with a 00} \} \) and \( L_2 = \{ w \mid \text{w starts with a 1} \} \)

NFA \( N_1 \) for \( L_1 \):

\[
\begin{array}{c}
q_1 \xrightarrow{0, 1} q_2 \xrightarrow{0} q_3
\end{array}
\]

NFA \( N_2 \) for \( L_2 \):

\[
\begin{array}{c}
q_4 \xrightarrow{1} q_5
\end{array}
\]
Therefore, \( L_1 \cup L_2 \) is a regular language.
Is the concatenation of regular languages also regular?

Assume that $L_1$ is a regular language recognized by the NFA $N_1$:

$$N_1 = \{Q_1, \Sigma_1, \delta_1, q_1, F_1\}$$

Assume that $L_2$ is also a regular language recognized by the NFA $N_2$:

$$N_2 = \{Q_2, \Sigma_2, \delta_2, q_2, F_2\}$$

If the concatenation $L_1 \circ L_2$ is regular, then there must be an NFA that recognizes it. Let this NFA be $N_3$ defined by

$$N_3 = \{Q_3, \Sigma_3, \delta_3, q_3, F_3\}$$
The following are the state diagrams for $N_1$ and $N_2$:

The NFA $N_3$ for the concatenation of $L_1$ and $L_2$ should be able to accept a string if it is of the form $xy$ where $x \in L_1$ and $y \in L_2$. In other words, $N_3$ should accept an input string if it can be divided into two parts where the first part is accepted by $N_1$ and the second part is accepted by $N_2$. 
NFA $N_3$ will first try to see if the first part of the input string is accepted by $N_1$.

Once $N_1$ is in a final state, $N_3$ tries to see or guess if that is the point where the first part stops and the second part begins. So $N_2$ performs its computation.
The start state of $N_3$ is the start state of $N_1$.

Upon arrival of the first symbol of the input string, $N_3$ starts "simulating" $N_1$ to determine if the first part of the string is a member of $L_1$. The state diagram for NFA $N_3$ will be:
Every time the computation reaches a final state of $N_1$, $N_3$ assumes or guesses that this is the point where the first part ends and the second begins. Hence, $N_3$ starts simulating $N_2$ to determine if the second part of the string is a member of $L_2$.

The set of final states of $N_3$ is the set of final states of $N_2$. 

The state diagram for NFA $N_3$ will be:
Example:
Let $L_1 = \{w \mid w \text{ ends with a } 00\}$ and $L_2 = \{w \mid w \text{ starts with a } 1\}$

NFA $N_1$ for $L_1$:

NFA $N_2$ for $L_2$: 
The state diagram of NFA $N_3$ for $L_1 \circ L_2$

Therefore, $L_1 \circ L_2$ is a regular language.
Is the star of regular languages also regular?

Assume that $L_1$ is a regular language recognized by the NFA $N_1$.

$$N_1 = \{Q_1, \Sigma_1, \delta_1, q_1, F_1\}$$

If $L_1^* \text{ (the star of} \ L_1)$ is regular, then there must be an NFA that recognizes it. Let this NFA be $N_2$.

$$N_2 = \{Q_2, \Sigma_2, \delta_2, q_2, F_2\}$$

The following is the state diagram for $N_1$: 

![State Diagram for NFA N1](image)
NFA $N_2$ should be able to accept a string if it is of the form $x_1x_2x_3\ldots$ where $x_i \in L_1$. In other words, $N_2$ should accept an input string if it can be divided into several parts where each part is accepted by $N_1$.

NFA $N_2$ will first try to see if the first part of the input string is accepted by $N_1$.

Once $N_1$ is in a final state, $N_2$ tries to see or guess if the second part is also accepted by $N_1$. So $N_2$ goes back to the start and begins computing again.

By definition of the star operation, $N_2$ should also be able to accept empty strings.
The state diagram for NFA $N_2$ will be

The start state of $N_2$ is a new state $q_0$ which is also a final state. Adding this state ensures that the empty string is also accepted by $N_2$.

Upon arrival of the first symbol of the input string, $N_2$ starts “simulating” $N_1$ to determine if the first part of the string is a member of $L_1$. 
Every time the computation reaches a final state of $N_1$, $N_2$ assumes or guesses that this is the point where the first part ends and the second begins. Hence, $N_2$ goes back to the start and tries to see if the next part is also accepted by $N_1$.

The set of final states of $N_2$ is the set of final states of $N_1$ plus state $q_0$. 
Example:

Let $L_1 = \{w \mid w \text{ ends with a } 00\}$

NFA $N_1$ for $L_1$: 

![NFA Diagram]
CLOSURE PROPERTIES OF REGULAR LANGUAGES

The state diagram of NFA $N_2$ for $L_1^*$:

Therefore, $L_1^*$ is a regular language.
CLOSURE PROPERTIES OF REGULAR LANGUAGES

- **Theorems on Closure**
  - **Theorem 2**
    The family of regular languages is closed under the union operation.
  - **Theorem 3**
    The family of regular languages is closed under the concatenation operation.
  - **Theorem 4**
    The family of regular languages is closed under the star operation.