CS 208: Automata Theory and Logic
Closure Properties for Regular Languages

Ashutosh Trivedi

∀x(L_a(x) → ∃y.(x < y) ∧ L_b(y))
**Regular Languages: Properties**

**Definition (Regular Languages)**

A *language* is called *regular* if it is accepted by a finite state automaton.

<table>
<thead>
<tr>
<th>Operation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Union: (A \cup B)</td>
<td>({w : w \in A \text{ or } w \in B})</td>
</tr>
<tr>
<td>Intersection: (A \cap B)</td>
<td>({w : w \in A \text{ and } w \in B})</td>
</tr>
<tr>
<td>Complementation: (A^c)</td>
<td>({w : w \not\in A})</td>
</tr>
<tr>
<td>Concatenation: (AB)</td>
<td>({wv : w \in A \text{ and } v \in B})</td>
</tr>
<tr>
<td>Kleene Closure: (A^*)</td>
<td>({w_1 w_2 \ldots w_k : k \geq 0 \text{ and } w_i \in A})</td>
</tr>
</tbody>
</table>

Define the notion of a set being closed under an operation (say, \(N\) and \(\times\)).

**Theorem**

The class of regular languages is closed under union, intersection, complementation, concatenation, and Kleene closure.
A language is called regular if it is accepted by a finite state automaton.

Let $A$ and $B$ be languages (remember they are sets). We define the following operations on them:

1. **Union**: $A \cup B = \{w : w \in A \text{ or } w \in B\}$
2. **Intersection**: $A \cap B = \{w : w \in A \text{ and } w \in B\}$

The class of regular languages is closed under union, intersection, complementation, concatenation, and Kleene closure.
**Regular Languages: Properties**

**Definition (Regular Languages)**

A language is called **regular** if it is accepted by a finite state automaton.

Let $A$ and $B$ be languages (remember they are sets). We define the following operations on them:

1. **Union**: $A \cup B = \{w : w \in A \text{ or } w \in B\}$
2. **Intersection**: $A \cap B = \{w : w \in A \text{ and } w \in B\}$
3. **Complementation**: $\overline{A} = \{w : w \notin A\}$

Define the notion of a set being closed under an operation (say, $\cup$ and $\times$).

**Theorem**

The class of regular languages is closed under union, intersection, complementation, concatenation, and Kleene closure.
Regular Languages: Properties

Definition (Regular Languages)

A language is called regular if it is accepted by a finite state automaton.

Let $A$ and $B$ be languages (remember they are sets). We define the following operations on them:

1. **Union:** $A \cup B = \{ w : w \in A \text{ or } w \in B \}$
2. **Intersection:** $A \cap B = \{ w : w \in A \text{ and } w \in B \}$
3. **Complementation:** $\overline{A} = \{ w : w \notin A \}$
4. **Concatenation:** $AB = \{ wv : w \in A \text{ and } v \in B \}$
Definition (Regular Languages)

A language is called regular if it is accepted by a finite state automaton.

Let $A$ and $B$ be languages (remember they are sets). We define the following operations on them:

1. **Union**: $A \cup B = \{ w : w \in A \text{ or } w \in B \}$
2. **Intersection**: $A \cap B = \{ w : w \in A \text{ and } w \in B \}$
3. **Complementation**: $\overline{A} = \{ w : w \notin A \}$
4. **Concatenation**: $AB = \{ wv : w \in A \text{ and } v \in B \}$
5. **Closure (Kleene Closure, or Star)**:
   
   $A^* = \{ w_1w_2 \ldots w_k : k \geq 0 \text{ and } w_i \in A \}$. In other words:
   
   $$A^* = \bigcup_{i \geq 0} A^i$$

   where $A^0 = \emptyset$, $A^1 = A$, $A^2 = AA$, and so on.
Regular Languages: Properties

Definition (Regular Languages)

A language is called regular if it is accepted by a finite state automaton.

Let $A$ and $B$ be languages (remember they are sets). We define the following operations on them:

1. Union: $A \cup B = \{w : w \in A \text{ or } w \in B\}$
2. Intersection: $A \cap B = \{w : w \in A \text{ and } w \in B\}$
3. Complementation: $\overline{A} = \{w : w \not\in A\}$
4. Concatenation: $AB = \{wv : w \in A \text{ and } v \in B\}$
5. Closure (Kleene Closure, or Star):
   \[A^* = \{w_1w_2\ldots w_k : k \geq 0 \text{ and } w_i \in A\}\] In other words: \[A^* = \cup_{i \geq 0} A^i\]
   where $A^0 = 0, A^1 = A, A^2 = AA$, and so on.

Define the notion of a set being closed under an operation (say, $\mathbb{N}$ and $\times$).
Regular Languages: Properties

Definition (Regular Languages)

A language is called regular if it is accepted by a finite state automaton.

Let $A$ and $B$ be languages (remember they are sets). We define the following operations on them:

1. Union: $A \cup B = \{w : w \in A \text{ or } w \in B\}$
2. Intersection: $A \cap B = \{w : w \in A \text{ and } w \in B\}$
3. Complementation: $\overline{A} = \{w : w \notin A\}$
4. Concatenation: $AB = \{vw : w \in A \text{ and } v \in B\}$
5. Closure (Kleene Closure, or Star):
   
   $A^* = \{w_1w_2 \ldots w_k : k \geq 0 \text{ and } w_i \in A\}$. In other words:
   
   $$A^* = \bigcup_{i \geq 0} A^i$$

   where $A^0 = \emptyset$, $A^1 = A$, $A^2 = AA$, and so on.

Define the notion of a set being closed under an operation (say, $\mathbb{N}$ and $\times$).

Theorem

The class of regular languages is closed under union, intersection, complementation, concatenation, and Kleene closure.
**Lemma**

The class of regular languages is closed under union.

---

**Proof.**

– Prove that for regular languages $L_1$ and $L_2$ that $L_1 \cup L_2$ is regular.

– Let $M_1 = (S_1, \Sigma, \delta_1, s_1, F_1)$ and $M_2 = (S_2, \Sigma, \delta_2, s_2, F_2)$ be DFA for $L_1$ and $L_2$.

– DFA Construction: (the Product Construction)

We claim the DFA $M = (S_1 \times S_2, \Sigma, \delta, (s_1, s_2), F)$ where

– $\delta((s_1, s_2), a) = (\delta_1(s_1, a), \delta_2(s_2, a))$ for all $s_1 \in S_1, s_2 \in S_2, \text{ and } a \in \Sigma$,

– $F = (F_1 \times S_2) \cup (S_1 \times F_2)$.

– $L(M) = L(M_1) \cup L(M_2)$.

– Proof of correctness:

  1. $\hat{\delta}((s_1, s_2), w) = (\hat{\delta}_1(s_1, w), \hat{\delta}_2(s_2, w))$

  2. $\hat{\delta}((s_1, s_2), w) \in F$ iff $\hat{\delta}_1(s_1, w) \in F$ or $\hat{\delta}_2(s_2, w) \in F$.

– $L_1 \cup L_2$ is regular since there is a DFA accepting this language.
## Closure under Union

### Lemma

The class of regular languages is closed under union.

### Proof.

- Prove that for regular languages $L_1$ and $L_2$ that $L_1 \cup L_2$ is regular.
Lemma
The class of regular languages is closed under union.

Proof.
- Prove that for regular languages $L_1$ and $L_2$ that $L_1 \cup L_2$ is regular.
- Let $M_1 = (S_1, \Sigma, \delta_1, s_1, F_1)$ and $M_2 = (S_2, \Sigma, \delta_2, s_2, F_2)$ be DFA for $L_1$ and $L_2$. 

DFA Construction: (the Product Construction)

We claim the DFA $M = (S_1 \times S_2, \Sigma, \delta, (s_1, s_2), F)$ where

- $\delta((s_1, s_2), a) = (\delta_1(s_1, a), \delta_2(s_2, a))$ for all $s_1 \in S_1, s_2 \in S_2$, and $a \in \Sigma$,
- $F = (F_1 \times S_2) \cup (S_1 \times F_2)$.

accepts $L_1 \cup L_2$ i.e. $L(M) = L(M_1) \cup L(M_2)$.

Proof of correctness:
For every string $w$, we have

- $\hat{\delta}((s_1, s_2), w) = (\hat{\delta}_1(s_1, w), \hat{\delta}_2(s_2, w))$
- $\hat{\delta}((s_1, s_2), w) \in F$ iff $\hat{\delta}_1(s_1, w) \in F$ or $\hat{\delta}_2(s_2, w) \in F$.

$L_1 \cup L_2$ is regular since there is a DFA accepting this language.
**Lemma**

The class of regular languages is closed under union.

**Proof.**

- Prove that for regular languages $L_1$ and $L_2$ that $L_1 \cup L_2$ is regular.
- Let $M_1 = (S_1, \Sigma, \delta_1, s_1, F_1)$ and $M_2 = (S_2, \Sigma, \delta_2, s_2, F_2)$ be DFA for $L_1$ and $L_2$.
- DFA Construction: (the Product Construction)
  We claim the DFA $M = (S_1 \times S_2, \Sigma, \delta, (s_1, s_2), F)$ where
  - $\delta((s_1, s_2), a) = (\delta_1(s_1, a), \delta_2(s_2, a))$ for all $s_1 \in S_1$, $s_2 \in S_2$, and $a \in \Sigma$,
  - $F = (F_1 \times S_2) \cup (S_1 \times F_2)$.

accepts $L_1 \cup L_2$ i.e. $L(M) = L(M_1) \cup L(M_2)$. 
Closure under Union

**Lemma**
The class of regular languages is closed under union.

**Proof.**
- Prove that for regular languages $L_1$ and $L_2$ that $L_1 \cup L_2$ is regular.
- Let $M_1 = (S_1, \Sigma, \delta_1, s_1, F_1)$ and $M_2 = (S_2, \Sigma, \delta_2, s_2, F_2)$ be DFA for $L_1$ and $L_2$.
- DFA Construction: *(the Product Construction)*
  We claim the DFA $M = (S_1 \times S_2, \Sigma, \delta, (s_1, s_2), F)$ where
  - $\delta((s_1, s_2), a) = (\delta_1(s_1, a), \delta_2(s_2, a))$ for all $s_1 \in S_1, s_2 \in S_2$, and $a \in \Sigma$,
  - $F = (F_1 \times S_2) \cup (S_1 \times F_2)$.
  accepts $L_1 \cup L_2$ i.e. $L(M) = L(M_1) \cup L(M_2)$.
- **Proof of correctness:**
Closure under Union

Lemma

The class of regular languages is closed under union.

Proof.

- Prove that for regular languages $L_1$ and $L_2$ that $L_1 \cup L_2$ is regular.
- Let $M_1 = (S_1, \Sigma, \delta_1, s_1, F_1)$ and $M_2 = (S_2, \Sigma, \delta_2, s_2, F_2)$ be DFA for $L_1$ and $L_2$.
- DFA Construction: (the Product Construction)
  We claim the DFA $M = (S_1 \times S_2, \Sigma, \delta, (s_1, s_2), F)$ where
  - $\delta((s_1, s_2), a) = (\delta_1(s_1, a), \delta_2(s_2, a))$ for all $s_1 \in S_1, s_2 \in S_2$, and $a \in \Sigma$,
  - $F = (F_1 \times S_2) \cup (S_1 \times F_2)$.

accepts $L_1 \cup L_2$ i.e. $L(M) = L(M_1) \cup L(M_2)$.

- Proof of correctness: For every string $w$, we have
  1. $\hat{\delta}((s_1, s_2), w) = (\hat{\delta}_1(s_1, w), \hat{\delta}_2(s_2, w))$
  2. $\hat{\delta}((s_1, s_2), w) \in F$ iff $\hat{\delta}_1(s_1, w) \in F$ or $\hat{\delta}_2(s_2, w) \in F$. 
Closure under Union

Lemma
The class of regular languages is closed under union.

Proof.

- Prove that for regular languages $L_1$ and $L_2$ that $L_1 \cup L_2$ is regular.
- Let $M_1 = (S_1, \Sigma, \delta_1, s_1, F_1)$ and $M_2 = (S_2, \Sigma, \delta_2, s_2, F_2)$ be DFA for $L_1$ and $L_2$.
- DFA Construction: (the Product Construction)
  We claim the DFA $M = (S_1 \times S_2, \Sigma, \delta, (s_1, s_2), F)$ where
  - $\delta((s_1, s_2), a) = (\delta_1(s_1, a), \delta_2(s_2, a))$ for all $s_1 \in S_1, s_2 \in S_2$, and $a \in \Sigma$,
  - $F = (F_1 \times S_2) \cup (S_1 \times F_2)$.
  accepts $L_1 \cup L_2$ i.e. $L(M) = L(M_1) \cup L(M_2)$.
- Proof of correctness: For every string $w$, we have
  1. $\hat{\delta}((s_1, s_2), w) = (\hat{\delta}_1(s_1, w), \hat{\delta}_2(s_2, w))$
  2. $\hat{\delta}((s_1, s_2), w) \in F$ iff $\hat{\delta}_1(s_1, w) \in F$ or $\hat{\delta}_2(s_2, w) \in F$.
- $L_1 \cup L_2$ is regular since there is a DFA accepting this language.
Lemma

The class of regular languages is closed under union.

Proof.

- Prove for arbitrary regular languages $L_1$ and $L_2$ that $L_1 \cup L_2$ is a regular languages.
- Let $E_1$ and $E_2$ be REGEX accepting $L_1$ and $L_2$. 
Lemma
The class of regular languages is closed under union.

Proof.
- Prove for arbitrary regular languages $L_1$ and $L_2$ that $L_1 \cup L_2$ is a regular languages.
- Let $E_1$ and $E_2$ be REGEX accepting $L_1$ and $L_2$.
- **REGEX Construction:**
  We claim the REGEX
  
  \[ E = E_1 + E_2 \]

  accepts $L_1 \cup L_2$, i.e. $L(E_1 + E_2) = L(E_1) \cup L(E_2)$. 

# Closure under Union via RegEx

## Lemma

The class of regular languages is closed under union.

## Proof.

- Prove for arbitrary regular languages $L_1$ and $L_2$ that $L_1 \cup L_2$ is a regular languages.
- Let $E_1$ and $E_2$ be REGEX accepting $L_1$ and $L_2$.
- **REGEX Construction:**
  
  We claim the REGEX
  
  \[ E = E_1 + E_2 \]
  
  accepts $L_1 \cup L_2$, i.e. $L(E_1 + E_2) = L(E_1) \cup L(E_2)$.
- **Proof of correctness:** trivial by definition of regular expressions.
Lemma

The class of regular languages is closed under union.

Proof.

- Prove for arbitrary regular languages $L_1$ and $L_2$ that $L_1 \cup L_2$ is a regular languages.
- Let $E_1$ and $E_2$ be REGEX accepting $L_1$ and $L_2$.
- **REGEX Construction:**
  We claim the REGEX
  $$ E = E_1 + E_2 $$
  accepts $L_1 \cup L_2$, i.e. $L(E_1 + E_2) = L(E_1) \cup L(E_2)$.
- **Proof of correctness:** trivial by definition of regular expressions.
- $L_1 \cup L_2$ is regular since there is a REGEX $E_1 + E_2$ accepting this language.
Lemma

The class of regular languages is closed under complementation.

Proof.

- Prove for arbitrary regular language $L$ that $\overline{L}$ is a regular language.
- Let $M = (S, \Sigma, \delta, s_0, F)$ be a DFA accepting $L$. 
Lemma
The class of regular languages is closed under complementation.

Proof.
- Prove for arbitrary regular language $L$ that $\overline{L}$ is a regular languages.
- Let $M = (S, \Sigma, \delta, s_0, F)$ be a DFA accepting $L$.
- DFA Construction:
  We claim the DFA
  
  $$M' = (S, \Sigma, \delta, s_0, F')$$
  
  where $F' = Q \setminus F$

  accepts $\overline{L}$, i.e. $L(M') = \{w : w \notin L(M)\}$. 

Lemma

The class of regular languages is closed under complementation.

Proof.

- Prove for arbitrary regular language \( L \) that \( \overline{L} \) is a regular languages.
- Let \( M = (S, \Sigma, \delta, s_0, F) \) be a DFA accepting \( L \).
- DFA Construction: We claim the DFA

\[
M' = (S, \Sigma, \delta, s_0, F') \quad \text{where} \quad F' = Q \setminus F
\]

accepts \( \overline{L} \), i.e. \( L(M') = \{w : w \notin L(M)\} \).
- Proof of correctness: For every string \( w \), we have
  1. \( \hat{\delta}(s_0, w) \notin F \) iff \( \hat{\delta}(s_0, w) \in F' \).
Closure under Complementation

**Lemma**

The class of regular languages is closed under complementation.

**Proof.**

- Prove for arbitrary regular language $L$ that $\bar{L}$ is a regular language.
- Let $M = (S, \Sigma, \delta, s_0, F)$ be a DFA accepting $L$.
- **DFA Construction:**
  We claim the DFA $M' = (S, \Sigma, \delta, s_0, F')$ where $F' = Q \setminus F$ accepts $\bar{L}$, i.e. $L(M') = \{w : w \notin L(M)\}$.
- **Proof of correctness:** For every string $w$, we have
  1. $\hat{\delta}(s_0, w) \notin F$ iff $\hat{\delta}(s_0, w) \in F'$.
- $\bar{L}$ is regular since there is a DFA accepting this language.
Lemma

The class of regular languages is closed under intersection.
Lemma

The class of regular languages is closed under intersection.

Proof.

- DFA based via product construction,
Lemma

*The class of regular languages is closed under intersection.*

Proof.

- DFA based via **product construction**,  
- Using De Morgan’s laws.
Lemma
The class of regular languages is closed under concatenation.
Closure under Concatenation

Lemma

*The class of regular languages is closed under concatenation.*

Proof.

- Prove for arbitrary regular languages $L_1$ and $L_2$ that $L_1 \cdot L_2$ is a regular language.
- Let $E_1$ and $E_2$ be REGEX accepting $L_1$ and $L_2$. 

Proof of correctness: trivial by definition of regular expressions.

$L_1 \cdot L_2$ is regular since there is a REGEX $E_1 \cdot E_2$ accepting this language.
Lemma

The class of regular languages is closed under concatenation.

Proof.

- Prove for arbitrary regular languages $L_1$ and $L_2$ that $L_1.L_2$ is a regular language.
- Let $E_1$ and $E_2$ be REGEX accepting $L_1$ and $L_2$.
- **REGEX Construction:**
  We claim the REGEX
   
   $$ E = E_1.E_2 $$

   accepts $L_1.L_2$, i.e. $L(E_1.E_2) = L(E_1).L(E_2)$. 
Lemma

The class of regular languages is closed under concatenation.

Proof.

- Prove for arbitrary regular languages \( L_1 \) and \( L_2 \) that \( L_1.L_2 \) is a regular language.
- Let \( E_1 \) and \( E_2 \) be REGEX accepting \( L_1 \) and \( L_2 \).
- **REGEX Construction:**
  We claim the REGEX
  \[
  E = E_1.E_2
  \]
  accepts \( L_1.L_2 \), i.e. \( L(E_1.E_2) = L(E_1).L(E_2) \).
- **Proof of correctness:** trivial by definition of regular expressions.
Closure under Concatenation

**Lemma**
The class of regular languages is closed under concatenation.

**Proof.**
- Prove for arbitrary regular languages $L_1$ and $L_2$ that $L_1.L_2$ is a regular language.
- Let $E_1$ and $E_2$ be REGEX accepting $L_1$ and $L_2$.
- **REGEX Construction:**
  We claim the REGEX $E = E_1.E_2$ accepts $L_1.L_2$, i.e. $L(E_1.E_2) = L(E_1).L(E_2)$.
- **Proof of correctness:** trivial by definition of regular expressions.
- $L_1.L_2$ is regular since there is a REGEX $E_1.E_2$ accepting this language.
Lemma

The class of regular languages is closed under Kleene star operation.

Proof.

– Prove for arbitrary regular language $L$ that $L^*$ is a regular languages.
– Let $E$ be REGEX accepting $L$. 
Lemma

The class of regular languages is closed under Kleene star operation.

Proof.

– Prove for arbitrary regular language $L$ that $L^*$ is a regular languages.
– Let $E$ be REGEX accepting $L$.
– **REGEX Construction:**
  We claim the REGEX

\[ E' = E^* \]

accepts $L$, i.e. $L(E^*) = (L(E))^*$.
Lemma

The class of regular languages is closed under Kleene star operation.

Proof.

- Prove for arbitrary regular language $L$ that $L^*$ is a regular languages.
- Let $E$ be REGEX accepting $L$.
- **REGEX Construction:**
  We claim the REGEX $E' = E^*$ accepts $L$, i.e. $L(E^*) = (L(E))^*$.
- **Proof of correctness:** trivial by definition of regular expressions.
Closure under Kleene Star Operation

Lemma
The class of regular languages is closed under Kleene star operation.

Proof.
- Prove for arbitrary regular language $L$ that $L^*$ is a regular languages.
- Let $E$ be REGEX accepting $L$.
- **REGEX Construction:**
  We claim the REGEX
  
  $$E' = E^*$$

  accepts $L$, i.e. $L(E^*) = (L(E))^*$.
- **Proof of correctness:** trivial by definition of regular expressions.
- $L^*$ is regular since there is a REGEX $E^*$ accepting this language.
Closure under Homomorphism

- A **homomorphism** is just substitution of strings for letters.
- Formally a homomorphism is a function $h : \Sigma \rightarrow \Gamma^*$. 
Closure under Homomorphism

- A **homomorphism** is just substitution of strings for letters.
- Formally a homomorphism is a function $h : \Sigma \rightarrow \Gamma^*$.
- Homomorphism can be extended from **letters** to strings $\hat{h} : \Sigma^* \rightarrow \Gamma^*$ in a straightforward manner:

$$\hat{h}(w) = \begin{cases} \varepsilon & \text{if } w = \varepsilon \\ \hat{h}(w) \cdot h(a) & \text{if } w = xa \end{cases}$$

- We can apply homomorphism to languages as well, for a homomorphism $h$ and a language $L \subseteq \Sigma^*$ we define $h(L) \subseteq \Gamma^*$ as

$$h(L) = \{ \hat{h}(w) \in \Gamma^* : w \in L \subseteq \Sigma^* \}.$$

- We define inverse-homomorphism of a language $L \in \Gamma^*$ as

$$h^{-1}(L) = \{ w \in \Sigma^* : \hat{h}(w) \in L \subseteq \Gamma^* \}.$$
Closure under Homomorphism

- A **homomorphism** is just substitution of strings for letters.
- Formally a homomorphism is a function $h : \Sigma \rightarrow \Gamma^*$.
- Homomorphism can be extended from *letters* to strings $\hat{h} : \Sigma^* \rightarrow \Gamma^*$ in a straightforward manner:

$$\hat{h}(w) = \begin{cases} 
\varepsilon & \text{if } w = \varepsilon \\
\hat{h}(w).h(a) & \text{if } w = xa
\end{cases}$$

- We can apply homomorphism to languages as well, for a homomorphism $h$ and a language $L \subseteq \Sigma^*$ we define $h(L) \subseteq \Gamma^*$ as

$$h(L) = \{ \hat{h}(w) \in \Gamma^* : w \in L \subseteq \Sigma^* \}.$$
Closure under Homomorphism

- A **homomorphism** is just substitution of strings for letters.
- Formally a homomorphism is a function \( h : \Sigma \rightarrow \Gamma^* \).
- Homomorphism can be extended from **letters** to strings \( \hat{h} : \Sigma^* \rightarrow \Gamma^* \) in a straightforward manner:

\[
\hat{h}(w) = \begin{cases} 
\varepsilon & \text{if } w = \varepsilon \\
\hat{h}(w)h(a) & \text{if } w = xa
\end{cases}
\]

- We can apply homomorphism to languages as well, for a homomorphism \( h \) and a language \( L \subseteq \Sigma^* \) we define \( h(L) \subseteq \Gamma^* \) as

\[
h(L) = \{ \hat{h}(w) \in \Gamma^* : w \in L \subseteq \Sigma^* \}.
\]

- We define **inverse-homomorphism** of a language \( L \in \Gamma^* \) as

\[
h^{-1}(L) = \{ w \in \Sigma^* : \hat{h}(w) \in L \subseteq \Gamma^* \}.
\]
Closure under Homomorphism

- A **homomorphism** is just substitution of strings for letters.
- Formally a homomorphism is a function $h : \Sigma \to \Gamma^*$.
- Homomorphism can be extended from **letters** to strings $\hat{h} : \Sigma^* \to \Gamma^*$ in a straightforward manner:

$$
\hat{h}(w) = \begin{cases} 
\varepsilon & \text{if } w = \varepsilon \\
\hat{h}(w).h(a) & \text{if } w = xa
\end{cases}
$$

- We can apply homomorphism to languages as well, for a homomorphism $h$ and a language $L \subseteq \Sigma^*$ we define $h(L) \subseteq \Gamma^*$ as

$$
h(L) = \{ \hat{h}(w) \in \Gamma^* : w \in L \subseteq \Sigma^* \}.
$$

- We define **inverse-homomorphism** of a language $L \in \Gamma^*$ as

$$
h^{-1}(L) = \{ w \in \Sigma^* : \hat{h}(w) \in L \subseteq \Gamma^* \}.
$$

**Theorem**

The class of regular languages is closed under **homomorphism**, and **inverse-homomorphism**.
**Lemma**

The class of regular languages is closed under homomorphism.

**Proof.**

- Prove for arbitrary regular language $L$ and homomorphism $h$ that $h(L)$ is a regular languages. Let $E$ be REGEX accepting $L$. 

\[
E_h =
\begin{cases}
\varepsilon & \text{if } E = \varepsilon \\
\emptyset & \text{if } E = \emptyset \\
h(a) & \text{if } E = a \\
F_h + G_h & \text{if } E = F + G \\
F_h G_h & \text{if } E = FG \\
(F_h)^* & \text{if } E = F^*
\end{cases}
\]

This REGEX $E_h$ accepts $h(L)$, i.e., $L(E_h) = h(L(E))$. 

Lemma

The class of regular languages is closed under homomorphism.

Proof.

- Prove for arbitrary regular language $L$ and homomorphism $h$ that $h(L)$ is a regular languages. Let $E$ be REGEX accepting $L$.
- **REGEX Construction**: We claim the REGEX $E_h$ defined inductively as

$$
E_h = \varepsilon \quad \text{if } E = \varepsilon \\
E_h = \emptyset \quad \text{if } E = \emptyset \\
E_h = h(a) \quad \text{if } E = a \\
E_h = F_h + G_h \quad \text{if } E = F + G \\
E_h = F_h.G_h \quad \text{if } E = F.G \\
E_h = (F_h)^* \quad \text{if } E = F^*
$$

accepts $h(L)$, i.e. $L(E_h) = h(L(E))$. 

Closure under Homomorphism

- **Proof of correctness:** Prove that $L(E_h) = h(L(E))$.
  - if $E = \varepsilon$, then
    
    \[
    \begin{align*}
    \text{LHS} & = L(E_h) = L(h(\varepsilon)) = L(\varepsilon) = \{\varepsilon\} \\
    \text{RHS} & = h(L(E)) = h(L(\varepsilon)) = h(\{\varepsilon\}) = \{\varepsilon\}.
    \end{align*}
    \]

  - Similarly for $E = \emptyset$.

From inductive hypothesis both of these expressions are equal.

- Other inductive cases are similar, and hence omitted.
Closure under Homomorphism

- **Proof of correctness**: Prove that \( L(E_h) = h(L(E)) \).
  - if \( E = \varepsilon \), then
    \[
    \begin{align*}
    \text{LHS} & = L(E_h) = L(h(\varepsilon)) = L(\varepsilon) = \{\varepsilon\} \\
    \text{RHS} & = h(L(E)) = h(L(\varepsilon)) = h(\{\varepsilon\}) = \{\varepsilon\}.
    \end{align*}
    \]
  - Similarly for \( E = \emptyset \).
  - if \( E = a \), then
    \[
    \begin{align*}
    \text{LHS} & = L(E_h) = L(h(a)) = \{h(a)\} \\
    \text{RHS} & = h(L(E)) = h(L(a)) = h(\{a\}) = \{h(a)\}.
    \end{align*}
    \]
  - if \( E = F + G \), then
    \[
    \begin{align*}
    L(h(E)) & = L(h(F + G)) = L(h(F) + h(G)) = L(h(F)) \cup L(h(G)) \\
    h(L(E)) & = h(L(F + G)) = h(L(F)) \cup h(L(G)).
    \end{align*}
    \]
    From inductive hypothesis both of these expression are equal.
  - Other inductive cases are similar, and hence omitted.
Lemma

The class of regular languages is closed under homomorphism.

Proof.

Let $A = (S, \Gamma, \delta, s_0, F)$ be a DFA accepting $L$ and $h: \Sigma \rightarrow \Gamma^*$ be an arbitrary homomorphism. We show that the DFA $h^{-1}(A) = (S', \Sigma, \delta', s'_0, F')$ defined below accepts $h^{-1}(L)$.

- $S' = S, s'_0 = s_0, F' = F$
- $\delta'(s, a) = \hat{\delta}(s, h(a))$

It is an easy induction over $w$ that $\hat{\delta'}(s, w) = \hat{\delta}(s, h(w))$. Now, since accepting states of $A$ and $h^{-1}(A)$ are the same, $h^{-1}(A)$ accepts $w$ iff $A$ accepts $h(w)$. 

Practice Questions

1. **Quotient Language.** For $a \in \Sigma$ and $L \subseteq \Sigma^*$ we define

$$L/a = \{w : wa \in L\}.$$  
$$a/L = \{w : aw \in L\}.$$  
$$L.a = \{wa : w \in L\}.$$  
$$a.L = \{aw : w \in L\}.$$ 

2. $\text{min}(L)$ is the set of strings $w$ such that $w \in L$ and no proper prefix of $w$ is in $L$. 

3. $\text{max}(L)$ is the set of strings such that $w \in L$ and no proper extension $wx \in L$. 

4. $\text{INIT}(L)$ is the set of strings $w$ such that for some $x$ we have that $wx \in L$. 

5. $\text{HALF}(L)$ is the set of strings $w$ such that for some string $x$ of same size as $w$ we have that $wx \in L$. 