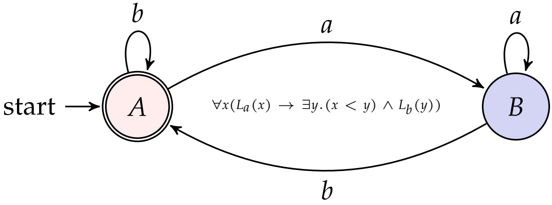
CS 208: Automata Theory and Logic

Closure Properties for Regular Languages

Ashutosh Trivedi



Department of Computer Science and Engineering,
Indian Institute of Technology Bombay.

Regular Languages: Properties

Definition (Regular Languages)

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

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: $A \cap B = \{w \,:\, w \in A \text{ and } w \in B\}$

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: $\overline{A} = \{w \ : \ w \notin A\}$

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: $AB = \{wv \,:\, w \in A \text{ and } v \in B\}$

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: $AB = \{wv : w \in A \text{ and } v \in B\}$
5. Closure (Kleene Closure, or Star):
   $A^* = \{w_1 w_2 \ldots w_k : k \geq 0 \text{ and } w_i \in A\}$. In other words:

   $$A^* = \cup_{i \geq 0} A^i$$

   where $A^0 = \emptyset$, $A^1 = A$, $A^2 = AA$, 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: $\overline{A} = \{w : w \notin A\}$
4. Concatenation: $AB = \{wv : w \in A \text{ and } v \in B\}$
5. Closure (Kleene Closure, or Star):
   $A^* = \{w_1 w_2 \dots w_k : k \geq 0 \text{ and } w_i \in A\}$. In other words:

$$A^* = \cup_{i \geq 0} A^i$$

   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 $\times$).

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: $AB = \{wv \; : \; w \in A \text{ and } v \in B\}$
5. Closure (Kleene Closure, or Star):
   $A^* = \{w_1 w_2 \dots w_k \; : \; k \geq 0 \text{ and } w_i \in A\}$. In other words:
   $$A^* = \cup_{i \geq 0} A^i$$
   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.

Closure under Union

Lemma

The class of regular languages is closed under union.

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.

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.

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. $L(M) = L(M_1) \cup L(M_2)$.

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.

□

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.

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. $L(E_1 + E_2) = L(E_1) \cup L(E_2)$.

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.

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.
- $L_1 \cup L_2$ is regular since there is a REGEX $E_1 + E_2$ accepting this language.

□

Closure under Complementation

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.

Closure under Complementation

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

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

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

Closure under Complementation

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

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

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

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

  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 $\hat{\delta}(s_0, w) \in F'$.
- $\overline{L}$ is regular since there is a DFA accepting this language.

□

Closure under Intersection

Lemma

The class of regular languages is closed under intersection.

Closure under Intersection

Lemma

The class of regular languages is closed under intersection.

Proof.

– DFA based via product construction,

Closure under Intersection

Lemma

The class of regular languages is closed under intersection.

Proof.

- DFA based via product construction,
- Using De Morgan's laws.

□

Closure under Concatenation

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 $L_1.L_2$ is a regular languages.
- Let E_1 and E_2 be REGEX accepting L_1 and L_2.

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

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 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 $L_1.L_2$, 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 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 $L_1.L_2$, 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.

□

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.

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. $L(E^*) = (L(E))^*$.

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.

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 $h : \Sigma \to \Gamma^*$.

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:

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

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

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

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

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

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.

Closure under 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.

Closure under 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.
– REGEX Construction: We claim the REGEX E_h defined inductively as

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

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

$$
\begin{aligned}
LHS &= L(E_h) = L(h(\varepsilon)) = L(\varepsilon) = \{\varepsilon\} \\
RHS &= h(L(E)) = h(L(\varepsilon)) = h(\{\varepsilon\}) = \{\varepsilon\}.
\end{aligned}
$$

  - Similarly for $E = \emptyset$.

Closure under Homomorphism

- Proof of correctness: Prove that $L(E_h) = h(L(E))$.
  - if $E = \varepsilon$, then

$$\begin{aligned} LHS &= L(E_h) = L(h(\varepsilon)) = L(\varepsilon) = \{\varepsilon\} \\ RHS &= h(L(E)) = h(L(\varepsilon)) = h(\{\varepsilon\}) = \{\varepsilon\}. \end{aligned}$$

  - Similarly for $E = \emptyset$.
  - if $E = a$, then

$$\begin{aligned} LHS &= L(E_h) = L(h(a)) = \{h(a)\} \\ RHS &= h(L(E)) = h(L(a)) = h(\{a\}) = \{h(a)\}. \end{aligned}$$

  - if $E = F + G$, then

$$\begin{aligned} L(h(E)) &= L(h(F+G)) = L(h(F) + h(G)) = L(h(F)) \cup L(h(G)) \\ h(L(E)) &= h(L(F+G)) = h(L(F)) \cup h(L(G)). \end{aligned}$$

    From inductive hypothesis both of these expression are equal.
  - Other inductive cases are similar, and hence omitted.

Closure under Inverse-Homomorphism

Lemma

The class of regular languages is closed under homomorphism.

Proof.

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

Practice Questions

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

$$\begin{aligned} L/a &= \{w \,:\, wa \in L\}. \\ a/L &= \{w \,:\, aw \in L\}. \\ L.a &= \{wa \,:\, w \in L\}. \\ a.L &= \{aw \,:\, w \in L\}. \end{aligned}$$

2. $\min(L)$ is the set of strings w such that $w \in L$ and no proper prefix of w is in L.
3. $\max(L)$ 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 $wx \in L$.