Closure properties of regular languages

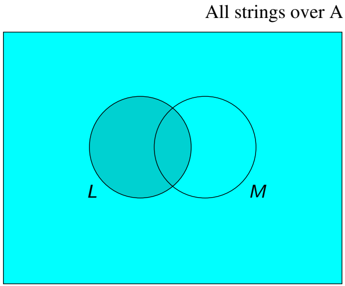
Meenakshi D'Souza

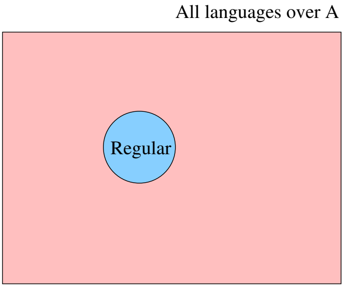
IIIT-Bangalore.

Term II 2022-'23

Closure properties

- Class of Regular languages is closed under
  - Complement, intersection, and union.
  - Concatenation, Kleene iteration.
- Non-deterministic Finite-state Automata (NFA) $=$ DFA.





Closure under complementation

- Idea: Flip final states.
- Formal construction:
  - Let $\mathcal{A} = (Q, s, \delta, F)$ be a DFA over alphabet Σ.
  - Define $\mathcal{B} = (Q, s, \delta, Q \setminus F)$.
  - Claim: $L(\mathcal{B}) = \Sigma^* \setminus L(\mathcal{A})$.

Proof of claim

- $L(\mathcal{B}) \subseteq \Sigma^* \setminus L(\mathcal{A})$.

$$\begin{aligned} w \in L(\mathcal{B}) &\implies \widehat{\delta}(s, w) \in (Q \setminus F). \\ &\implies \widehat{\delta}(s, w) \notin F \\ &\implies w \notin L(\mathcal{A}) \\ &\implies w \in \Sigma^* \setminus L(\mathcal{A}). \end{aligned}$$

- $L(\mathcal{B}) \supseteq \Sigma^* - L(\mathcal{A})$.

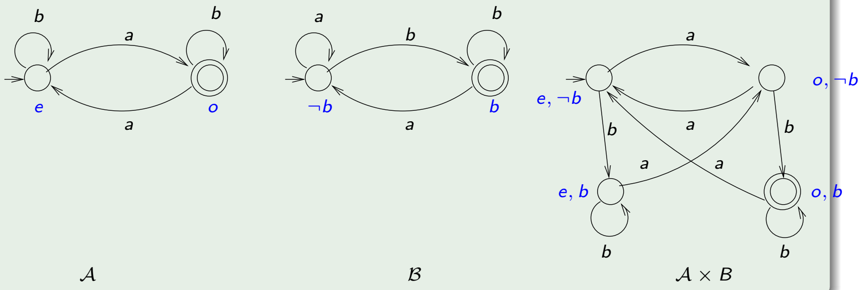
Closure under intersection

Product construction. Given DFA's $\mathcal{A} = (Q, s, \delta, F)$, $\mathcal{B} = (Q', s', \delta', F')$, define product $\mathcal{C}$ of $\mathcal{A}$ and $\mathcal{B}$:

$$\mathcal{C} = (Q \times Q', (s, s'), \delta'', F \times F'),$$

where $\delta''((p, p'), a) = (\delta(p, a), \delta'(p', a))$.

Product construction example



Correctness of product construction

Claim: $L(\mathcal{C}) = L(\mathcal{A}) \cap L(\mathcal{B})$.

Proof of claim $L(\mathcal{C}) = L(\mathcal{A}) \cap L(\mathcal{B})$.

- $L(\mathcal{C}) \subseteq L(\mathcal{A}) \cap L(\mathcal{B})$.

$$\begin{aligned}
w \in L(\mathcal{C}) &\implies \widehat{\delta}''((s,s'),w) \in F \times F'. \\
&\implies (\widehat{\delta}(s,w), \widehat{\delta}'(s',w)) \in F \times F' \text{ (by subclaim)} \\
&\implies \widehat{\delta}(s,w) \in F \text{ and } \widehat{\delta}'(s',w) \in F' \\
&\implies w \in L(\mathcal{A}) \text{ and } w \in L(\mathcal{B}) \\
&\implies w \in L(\mathcal{A}) \cap L(\mathcal{B}).
\end{aligned}$$

- $L(\mathcal{C}) \supseteq L(\mathcal{A}) \cap L(\mathcal{B})$.

Subclaim: $\widehat{\delta}''((s,s'),w) = (\widehat{\delta}(s,w), \widehat{\delta}'(s',w))$.

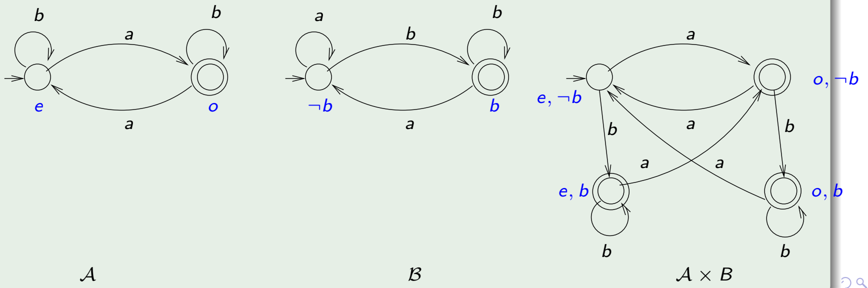
Closure under union

- Follows from closure under complement and intersection since $L_1 \cup L_2 = \overline{\overline{L_1} \cap \overline{L_2}}$.

Closure under union

- Follows from closure under complement and intersection since $L_1 \cup L_2 = \overline{\overline{L_1} \cap \overline{L_2}}$.
- Can also do directly by product construction: Given DFA's $\mathcal{A} = (Q, s, \delta, F)$, $\mathcal{B} = (Q', s', \delta', F')$, define $\mathcal{C}$: $\mathcal{C} = (Q \times Q', (s, s'), \delta'', (F \times Q') \cup (Q \times F'))$, where $\delta''((p, p'), a) = (\delta(p, a), \delta(p', a))$.

Union construction



Proof of subclaim

Exercise: Prove the Subclaim:

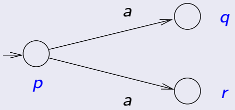
$$\widehat{\delta}''((s,s'),w) = (\widehat{\delta}(s,w), \widehat{\delta}'(s',w)).$$

using induction.

Nondeterministic Finite-state Automata (NFA)

- Allows multiple start states.
- Allows more than one transition from a state on a given letter.

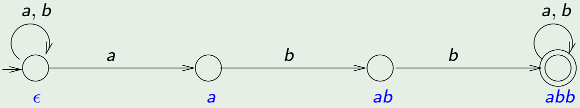
Non-deterministic transitions



- A word is accepted if there is some path on it from a start to a final state.

Example NFA's

NFA for "contains *abb* as a subword"



NFA definition

- Mathematical representation of NFA
  - $\mathcal{A} = (Q, S, \Delta, F)$, where $S \subseteq Q$, and $\Delta : Q \times \Sigma \to 2^Q$.
  - Define relation $p \xrightarrow{w} q$ which says there is a path from state p to state q labelled w.
    - $p \xrightarrow{\epsilon} p$
    - $p \xrightarrow{ua} q$ iff there exists $r \in Q$ such that $p \xrightarrow{u} r$ and $q \in \Delta(r, a)$.
  - Define $L(\mathcal{A}) = \{w \in \Sigma^* \mid \exists s \in S, f \in F : \; s \xrightarrow{w} f\}$.
- NFA $\to$ DFA: Subset construction
  - Example: determinize NFA for "contains *abb*."
  - Formal construction
  - Correctness

Closure under concatenation and Kleene iteration

- Concatenation of languages:

$$L \cdot M = \{u \cdot v \mid u \in L,\ v \in M\}.$$

- Kleene iteration of a language:

$$L^* = \{\epsilon\} \cup L \cup L^2 \cup L^3 \cup \cdots ,$$

where

$$\begin{aligned} L^n &= L \cdot L \cdots L \ (n \text{ times}). \\ &= \{w_1 \cdots w_n \mid \text{each } w_i \in L\}. \end{aligned}$$

Examples of Regular Expressions

Expressions built from a, b, ϵ, using operators $+$, $\cdot$, and $*$.

- $(a^* + b^*) \cdot c$
  "Strings of only a's or only b's, followed by a c."
- $(a+b)^* abb (a+b)^*$
  "contains abb as a subword."
- $(a+b)^* b (a+b)(a+b)$
- $(b^* a b^* a)^* b^*$

Examples of Regular Expressions

Expressions built from a, b, ϵ, using operators $+$, $\cdot$, and $*$.

- $(a^* + b^*) \cdot c$
  "Strings of only a's or only b's, followed by a c."
- $(a+b)^* abb (a+b)^*$
  "contains abb as a subword."
- $(a+b)^* b (a+b)(a+b)$
  "3rd last letter is a b."
- $(b^* a b^* a)^* b^*$

Examples of Regular Expressions

Expressions built from a, b, ϵ, using operators $+$, $\cdot$, and $*$.

- $(a^* + b^*) \cdot c$
  "Strings of only a's or only b's, followed by a c."
- $(a + b)^* abb(a + b)^*$
  "contains abb as a subword."
- $(a + b)^* b(a + b)(a + b)$
  "3rd last letter is a b."
- $(b^* ab^* a)^* b^*$
  "Even number of a's."

Examples of Regular Expressions

Expressions built from a, b, ϵ, using operators $+$, $\cdot$, and $*$.

- $(a^* + b^*) \cdot c$
  "Strings of only a's or only b's, followed by a c."
- $(a+b)^* abb(a+b)^*$
  "contains abb as a subword."
- $(a+b)^* b(a+b)(a+b)$
  "3rd last letter is a b."
- $(b^* ab^* a)^* b^*$
  "Even number of a's."
- "Every 4-bit block of the form $w[4i, 4i+1, 4i+2, 4i+3]$ has even parity."

Examples of Regular Expressions

Expressions built from a, b, ϵ, using operators $+$, $\cdot$, and $*$.

- $(a^* + b^*) \cdot c$
  "Strings of only a's or only b's, followed by a c."
- $(a+b)^* abb(a+b)^*$
  "contains abb as a subword."
- $(a+b)^* b(a+b)(a+b)$
  "3rd last letter is a b."
- $(b^* ab^* a)^* b^*$
  "Even number of a's."
- "Every 4-bit block of the form $w[4i, 4i+1, 4i+2, 4i+3]$ has even parity."
  $(0000 + 0011 + \cdots + 1111)^*(\epsilon + 0 + 1 + \cdots + 111)$

Formal definitions

- Syntax of regular expresions over an alphabet A:

$$r ::= \emptyset \mid a \mid r + r \mid r \cdot r \mid r^*$$

  where $a \in A$.

- Semantics: associate a language $L(r) \subseteq A^*$ with regexp r.

$$\begin{array}{lcl} L(\emptyset) & = & \{\} \\ L(a) & = & \{a\} \\ L(r + r') & = & L(r) \cup L(r') \\ L(r \cdot r') & = & L(r) \cdot L(r') \\ L(r^*) & = & L(r)^*. \end{array}$$

Formal definitions

- Syntax of regular expresions over an alphabet A:

$$r ::= \emptyset \mid a \mid r + r \mid r \cdot r \mid r^*$$

  where $a \in A$.

- Semantics: associate a language $L(r) \subseteq A^*$ with regexp r.

$$\begin{array}{lcl} L(\emptyset) & = & \{\} \\ L(a) & = & \{a\} \\ L(r + r') & = & L(r) \cup L(r') \\ L(r \cdot r') & = & L(r) \cdot L(r') \\ L(r^*) & = & L(r)^*. \end{array}$$

- Question: Do we need ϵ in syntax?

Formal definitions

- Syntax of regular expresions over an alphabet A:

$$r ::= \emptyset \mid a \mid r + r \mid r \cdot r \mid r^*$$

  where $a \in A$.

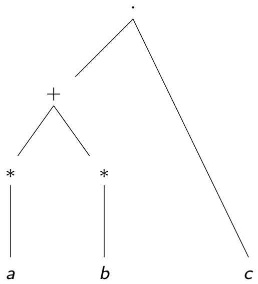
- Semantics: associate a language $L(r) \subseteq A^*$ with regexp r.

$$\begin{array}{lcl} L(\emptyset) & = & \{\} \\ L(a) & = & \{a\} \\ L(r + r') & = & L(r) \cup L(r') \\ L(r \cdot r') & = & L(r) \cdot L(r') \\ L(r^*) & = & L(r)^*. \end{array}$$

- Question: Do we need ϵ in syntax?
  No. $\epsilon \equiv \emptyset^*$.

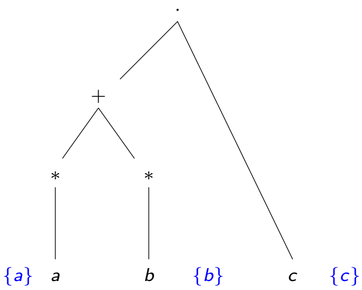
Example: Semantics of regexp

$(a^* + b^*) \cdot c$



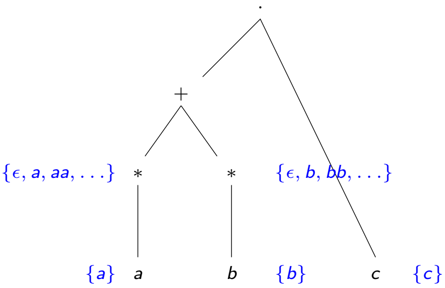
Example: Semantics of regexp

$(a^* + b^*) \cdot c$



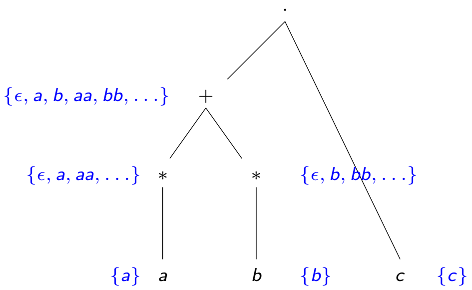
Example: Semantics of regexp

$(a^* + b^*) \cdot c$



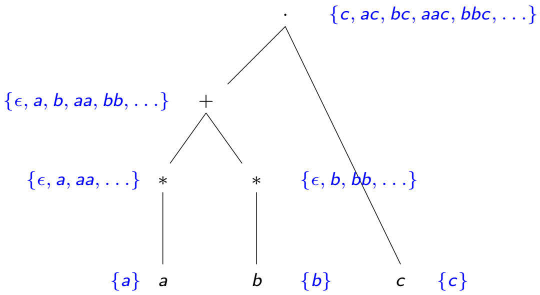
Example: Semantics of regexp

$(a^* + b^*) \cdot c$



Example: Semantics of regexp

$(a^* + b^*) \cdot c$



Kleene's Theorem: RE = DFA

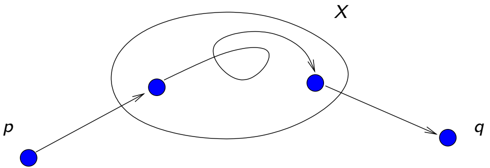
Class of languages defined by regular expressions coincides with regular languages.

Proof

- RE $\rightarrow$ DFA: Use closure properties of regular languages.
- DFA $\rightarrow$ RE:

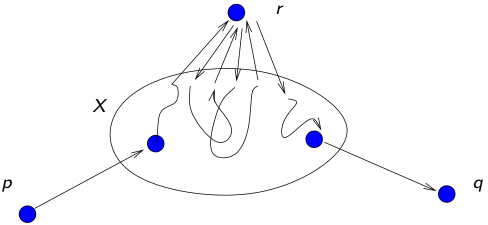
DFA $\rightarrow$ RE: Kleene's construction

- Let $\mathcal{A} = (Q, s, \delta, F)$ be given DFA.
- Define $L_{pq} = \{w \in \Sigma^* \mid \widehat{\delta}(p, w) = q\}$.
- Then $L(\mathcal{A}) = \bigcup_{f \in F} L_{sf}$.
- For $X \subseteq Q$, define $L^X_{pq} = \{w \in A^* \mid \widehat{\delta}(p, w) = q$ via a path that stays in X except for first and last states$\}$



- Then $L(\mathcal{A}) = \bigcup_{f \in F} L^Q_{sf}$.

DFA $\rightarrow$ RE: Kleene's construction



Advantage:

$$L_{pq}^{X \cup \{r\}} = L_{pq}^{X} + L_{pr}^{X} \cdot (L_{rr}^{X})^* \cdot L_{rq}^{X}.$$

DFA → RE: Kleene's construction (2)

Method:

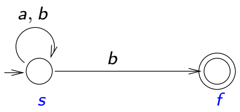
- Begin with L_{sf}^{Q} for each $f \in F$.
- Simplify by using terms with strictly smaller X's:

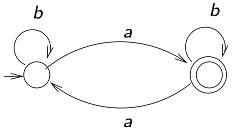
$$L_{pq}^{X \cup \{r\}} = L_{pq}^{X} + L_{pr}^{X} \cdot (L_{rr}^{X})^{*} \cdot L_{rq}^{X}.$$

- For base terms, observe that

$$L_{pq}^{\{\}} = \begin{cases} \{a \mid \delta(p,a) = q\} & \text{if} \quad p \neq q \\ \{a \mid \delta(p,a) = q\} \cup \{\epsilon\} & \text{if} \quad p = q. \end{cases}$$

- Exercise: convert NFA/DFA's below to RE's:





DFA → RE: Kleene's construction (2)

Method:

- Begin with L_{sf}^{Q} for each $f \in F$.
- Simplify by using terms with strictly smaller X's:

$$L_{pq}^{X \cup \{r\}} = L_{pq}^{X} + L_{pr}^{X} \cdot (L_{rr}^{X})^{*} \cdot L_{rq}^{X}.$$

- For base terms, observe that

$$L_{pq}^{\{\}} = \begin{cases} \{a \mid \delta(p,a) = q\} & \text{if} \quad p \neq q \\ \{a \mid \delta(p,a) = q\} \cup \{\epsilon\} & \text{if} \quad p = q. \end{cases}$$

- Exercise: convert NFA/DFA's below to RE's: