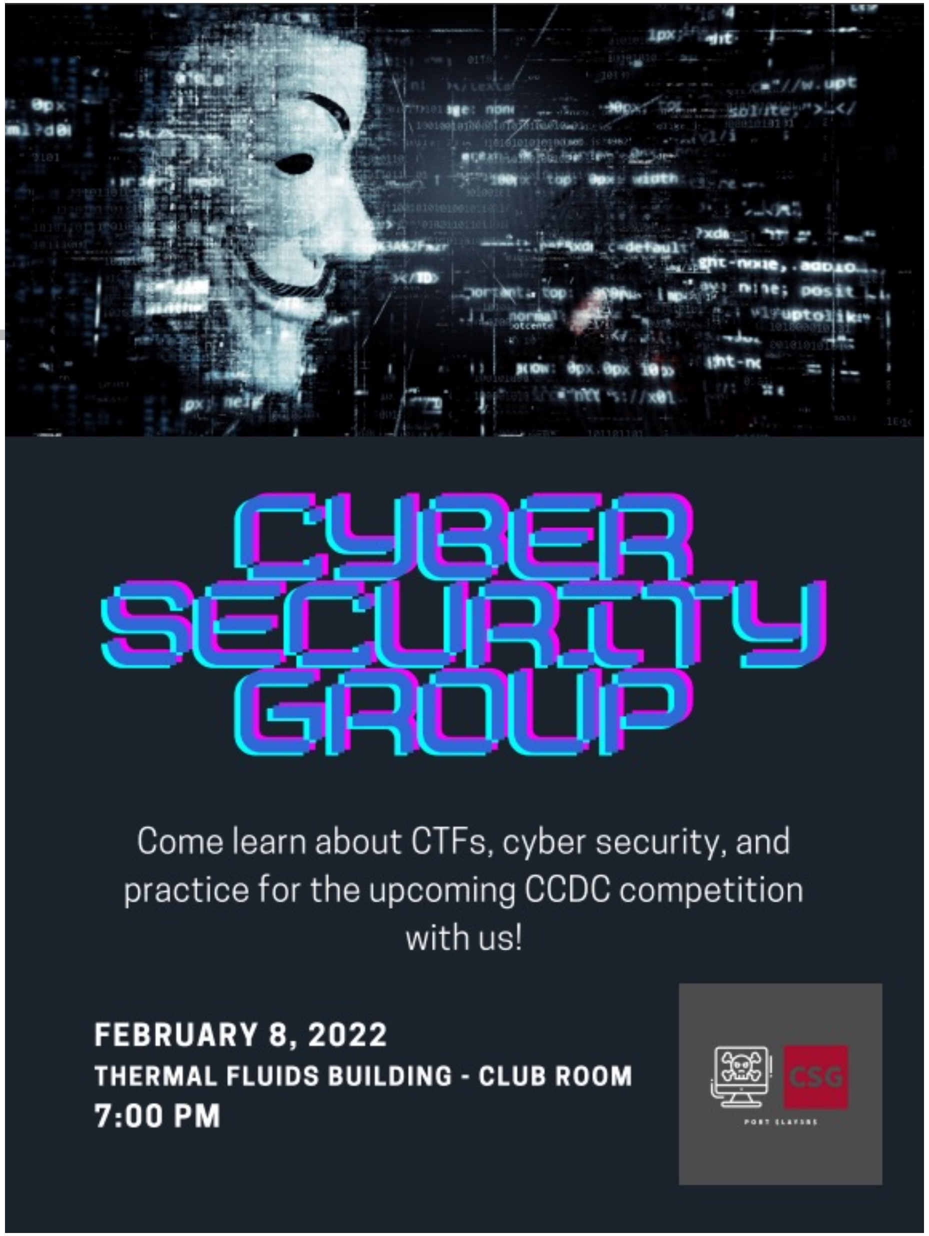
Regular Expressions and NFAs

Featured Student Club



Formal definition of a regular expression

DEFINITION 1.52

Say that R is a ***regular expression*** if R is

1. a for some a in the alphabet Σ,
2. ε,
3. $\emptyset$,
4. $(R_1 \cup R_2)$, where R_1 and R_2 are regular expressions,
5. $(R_1 \circ R_2)$, where R_1 and R_2 are regular expressions, or
6. (R_1^*), where R_1 is a regular expression.

In items 1 and 2, the regular expressions a and ε represent the languages $\{a\}$ and $\{\varepsilon\}$, respectively. In item 3, the regular expression $\emptyset$ represents the empty language. In items 4, 5, and 6, the expressions represent the languages obtained by taking the union or concatenation of the languages R_1 and R_2, or the star of the language R_1, respectively.

Examples

In all cases below, alphabet Σ is $\{0,1\}$

1. $0^*10^* =$
2. $1^*(01^+)^* =$
3. $\Sigma^*001\Sigma^* =$
4. $\Sigma^*1\Sigma^* =$
5. $(\Sigma\Sigma)^* =$
6. $(\Sigma\Sigma\Sigma)^* =$
7. $0\Sigma^*0 \cup 1\Sigma^*1 \cup 0 \cup 1 =$
8. $1^*\emptyset =$
9. $\emptyset^* =$
10. $(0 \cup \varepsilon)(1 \cup \varepsilon) =$

Examples

In all cases below, alphabet Σ is $\{0,1\}$

1. 0^*10^* = {w| w contains a single 1}
2. $1^*(01^+)^*$ =
3. $\Sigma^*001\Sigma^*$ =
4. $\Sigma^*1\Sigma^*$ =
5. $(\Sigma\Sigma)^*$ =
6. $(\Sigma\Sigma\Sigma)^*$ =
7. $0\Sigma^*0 \cup 1\Sigma^*1 \cup 0 \cup 1$ =
8. $1^*\emptyset$ =
9. $\emptyset^*$ =
10. $(0 \cup \varepsilon)\ (1 \cup \varepsilon)$ =

Examples

In all cases below, alphabet Σ is $\{0,1\}$

1. 0^*10^* = {w| w contains a single 1}
2. $1^*(01^+)^*$ = {w| every 0 in w is followed by at least one 1}
3. $\Sigma^*001\Sigma^*$ =
4. $\Sigma^*1\Sigma^*$ =
5. $(\Sigma\Sigma)^*$ =
6. $(\Sigma\Sigma\Sigma)^*$ =
7. $0\Sigma^*0 \cup 1\Sigma^*1 \cup 0 \cup 1$ =
8. $1^*\emptyset$ =
9. $\emptyset^*$ =
10. $(0 \cup \varepsilon)(1 \cup \varepsilon)$ =

Examples

In all cases below, alphabet Σ is $\{0,1\}$

1. 0^*10^* = {w| w contains a single 1}
2. $1^*(01^+)^*$ = {w| every 0 in w is followed by at least one 1}
3. $\Sigma^*001\Sigma^*$ = {w| w contains 001 as a substring}
4. $\Sigma^*1\Sigma^*$ =
5. $(\Sigma\Sigma)^*$ =
6. $(\Sigma\Sigma\Sigma)^*$ =
7. $0\Sigma^*0 \cup 1\Sigma^*1 \cup 0 \cup 1$ =
8. $1^*\emptyset$ =
9. $\emptyset^*$ =
10. $(0 \cup \varepsilon)(1 \cup \varepsilon)$ =

Examples

In all cases below, alphabet Σ is $\{0,1\}$

1. 0^*10^* = {w| w contains a single 1}
2. $1^*(01^+)^*$ = {w| every 0 in w is followed by at least one 1}
3. $\Sigma^*001\Sigma^*$ = {w| w contains 001 as a substring}
4. $\Sigma^*1\Sigma^*$ = {w| w contains at least one 1}
5. $(\Sigma\Sigma)^*$ =
6. $(\Sigma\Sigma\Sigma)^*$ =
7. $0\Sigma^*0 \cup 1\Sigma^*1 \cup 0 \cup 1$ =
8. $1^*\emptyset$ =
9. $\emptyset^*$ =
10. $(0 \cup \varepsilon)(1 \cup \varepsilon)$ =

Examples

In all cases below, alphabet Σ is $\{0,1\}$

1. $0^*10^* = \{w \mid w \text{ contains a single } 1\}$
2. $1^*(01^+)^* = \{w \mid \text{every } 0 \text{ in } w \text{ is followed by at least one } 1\}$
3. $\Sigma^*001\Sigma^* = \{w \mid w \text{ contains } 001 \text{ as a substring}\}$
4. $\Sigma^*1\Sigma^* = \{w \mid w \text{ contains at least one } 1\}$
5. $(\Sigma\Sigma)^* = \{w \mid w \text{ is a string of even length}\}$
6. $(\Sigma\Sigma\Sigma)^* = \{w \mid \text{the length of } w \text{ is a multiple of } 3\}$
7. $0\Sigma^*0 \cup 1\Sigma^*1 \cup 0 \cup 1 =$
8. $1^*\emptyset =$
9. $\emptyset^* =$
10. $(0 \cup \varepsilon)\,(1 \cup \varepsilon) =$

Examples

In all cases below, alphabet Σ is $\{0,1\}$

1. 0^*10^* = {w| w contains a single 1}
2. $1^*(01^+)^*$ = {w| every 0 in w is followed by at least one 1}
3. $\Sigma^*001\Sigma^*$ = {w| w contains 001 as a substring}
4. $\Sigma^*1\Sigma^*$ = {w| w contains at least one 1}
5. $(\Sigma\Sigma)^*$ = {w| w is a string of even length}
6. $(\Sigma\Sigma\Sigma)^*$ = {w| the length of w is a multiple of 3}
7. $0\Sigma^*0 \cup 1\Sigma^*1 \cup 0 \cup 1$ = {w| w starts and ends with the same symbol}
8. $1^*\emptyset$ =
9. $\emptyset^*$ =
10. $(0 \cup \varepsilon)(1 \cup \varepsilon)$ =

Examples

In all cases below, alphabet Σ is $\{0,1\}$

1. $0^*10^* = \{w \mid w \text{ contains a single } 1\}$
2. $1^*(01^+)^* = \{w \mid \text{every } 0 \text{ in } w \text{ is followed by at least one } 1\}$
3. $\Sigma^*001\Sigma^* = \{w \mid w \text{ contains } 001 \text{ as a substring}\}$
4. $\Sigma^*1\Sigma^* = \{w \mid w \text{ contains at least one } 1\}$
5. $(\Sigma\Sigma)^* = \{w \mid w \text{ is a string of even length}\}$
6. $(\Sigma\Sigma\Sigma)^* = \{w \mid \text{the length of } w \text{ is a multiple of } 3\}$
7. $0\Sigma^*0 \cup 1\Sigma^*1 \cup 0 \cup 1 = \{w \mid w \text{ starts and ends with the same symbol}\}$
8. $1^*\emptyset = \emptyset$
9. $\emptyset^* = \{\varepsilon\}$
10. $(0 \cup \varepsilon)(1 \cup \varepsilon) = \{\varepsilon, 0, 1, 01\}$

Identities

- Let R be any regular expression,
  - $R \cup \emptyset = R$
  - $R \circ \varepsilon = R$

- However,
  - $R \cup \varepsilon$ may not equal R
    - E.g. if $R = 0$, then $L(R) = \{0\}$, but $L(R \cup \varepsilon) = \{0, \varepsilon\}$
  - $R \circ \emptyset$ may not equal R
    - E.g. if $R = 0$, then $L(R) = \{0\}$, but $L(R \circ \emptyset) = \emptyset$

Regular expressions in compilers

- Regular expressions are useful tools in the design of compilers for programming language
- **Tokens** such as variable names and constants may be expressed with regular expressions
  - For example, a **numerical constant that may include a fractional part and/or sign** maybe described as a member of the language

$$(+ \cup - \cup \varepsilon)\ (D^{+} \cup D^{+}.D^{*} \cup D^{*}.D^{+})$$

where $D = \{0,1,2,3,4,5,6,7,8,9\}$ is the alphabet of decimal digits.

- Once the syntax of a programming language has been described with a regular expression in terms of its token, automatic systems can generate the **lexical analyzer**, the part of a compiler that initially processes the input program.

Equivalence with Finite Automata

- Regular expressions and finite automata are equivalent in their descriptive power
- This is surprising because FA and regular expressions outwardly are different
- However, any regular expression can be converted into an automata that recognizes the languages it describes, and vice versa

Theorem:

A language is regular if and only if some regular expression describes it.

One direction of the theorem

Lemma:

If a language is described by a regular expression, then it is regular.

Proof Idea:

Say that we have a regular expression R describing some language A. We show how to convert R into an NFA recognizing A. We know if an NFA recognizes A then A is regular.

Recall definition of a regular expression

DEFINITION 1.52

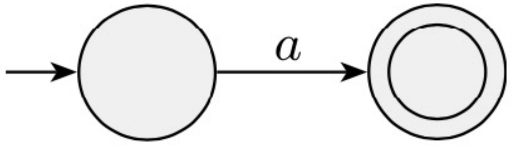
Say that R is a ***regular expression*** if R is

1. a for some a in the alphabet Σ,
2. ε,
3. $\emptyset$,
4. $(R_1 \cup R_2)$, where R_1 and R_2 are regular expressions,
5. $(R_1 \circ R_2)$, where R_1 and R_2 are regular expressions, or
6. (R_1^*), where R_1 is a regular expression.

In items 1 and 2, the regular expressions a and ε represent the languages $\{a\}$ and $\{\varepsilon\}$, respectively. In item 3, the regular expression $\emptyset$ represents the empty language. In items 4, 5, and 6, the expressions represent the languages obtained by taking the union or concatenation of the languages R_1 and R_2, or the star of the language R_1, respectively.

We consider the six cases turn by turn

1. $R = a$ for some $a \in \Sigma$. Then $L(R) = \{a\}$, and the following NFA recognizes $L(R)$.

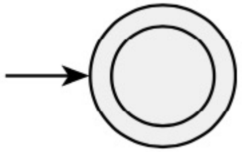


Note that this machine fits the definition of an NFA but not that of a DFA because it has some states with no exiting arrow for each possible input symbol. Of course, we could have presented an equivalent DFA here; but an NFA is all we need for now, and it is easier to describe.

Formally, $N = (\{q_1, q_2\}, \Sigma, \delta, q_1, \{q_2\})$, where we describe δ by saying that $\delta(q_1, a) = \{q_2\}$ and that $\delta(r, b) = \emptyset$ for $r \neq q_1$ or $b \neq a$.

We consider the six cases turn by turn

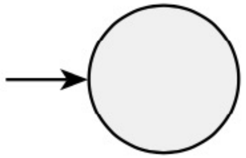
2. $R = \varepsilon$. Then $L(R) = \{\varepsilon\}$, and the following NFA recognizes $L(R)$.



Formally, $N = \big(\{q_1\}, \Sigma, \delta, q_1, \{q_1\}\big)$, where $\delta(r, b) = \emptyset$ for any r and b.

We consider the six cases turn by turn

3. $R = \emptyset$. Then $L(R) = \emptyset$, and the following NFA recognizes $L(R)$.



Formally, $N = (\{q\}, \Sigma, \delta, q, \emptyset)$, where $\delta(r, b) = \emptyset$ for any r and b.

We consider the six cases turn by turn

4. $R = R_1 \cup R_2$.
5. $R = R_1 \circ R_2$.
6. $R = R_1^*$.

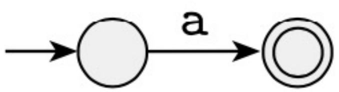
For the last three cases, we use the constructions given in the proofs that the class of regular languages is closed under the regular operations. In other words, we construct the NFA for R from the NFAs for R_1 and R_2 (or just R_1 in case 6) and the appropriate closure construction.

Example 1

Convert the regular expression $(ab \cup a)^*$ to an NFA

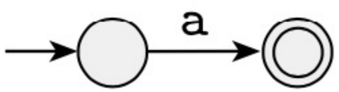
Example 1: $(ab \cup a)^*$

a

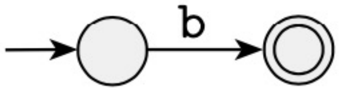


Example 1: $(ab \cup a)^*$

a

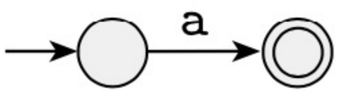


b

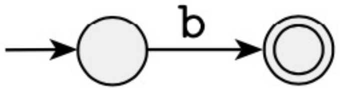


Example 1: (ab ∪ a)*

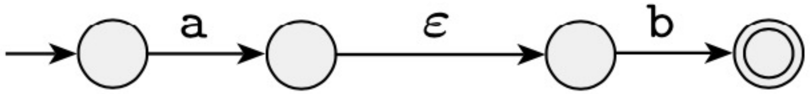
a



b

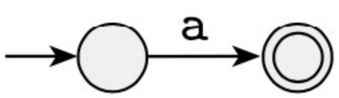


ab

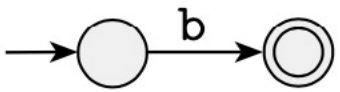


Example 1: (ab ∪ a)*

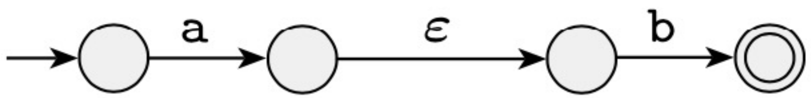
a



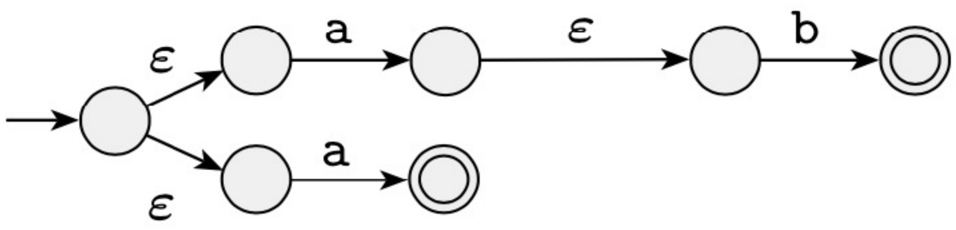
b



ab

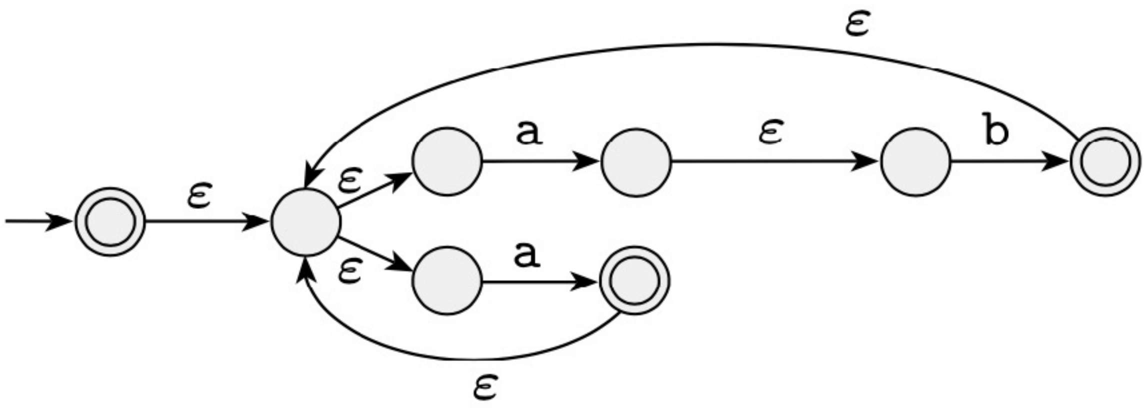


ab ∪ a



Example 1: $(ab \cup a)^*$

$(ab \cup a)^*$

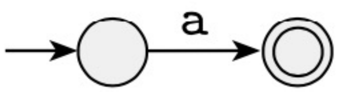


Example 2

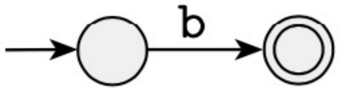
Convert the regular expression $(a \cup b)^* aba$ to an NFA

Example 2: $(a \cup b)^* aba$

a

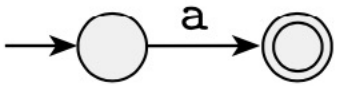


b

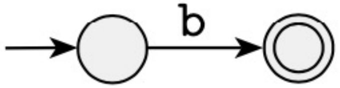


Example 2: $(a \cup b)^* aba$

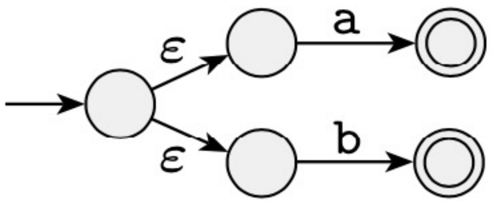
a



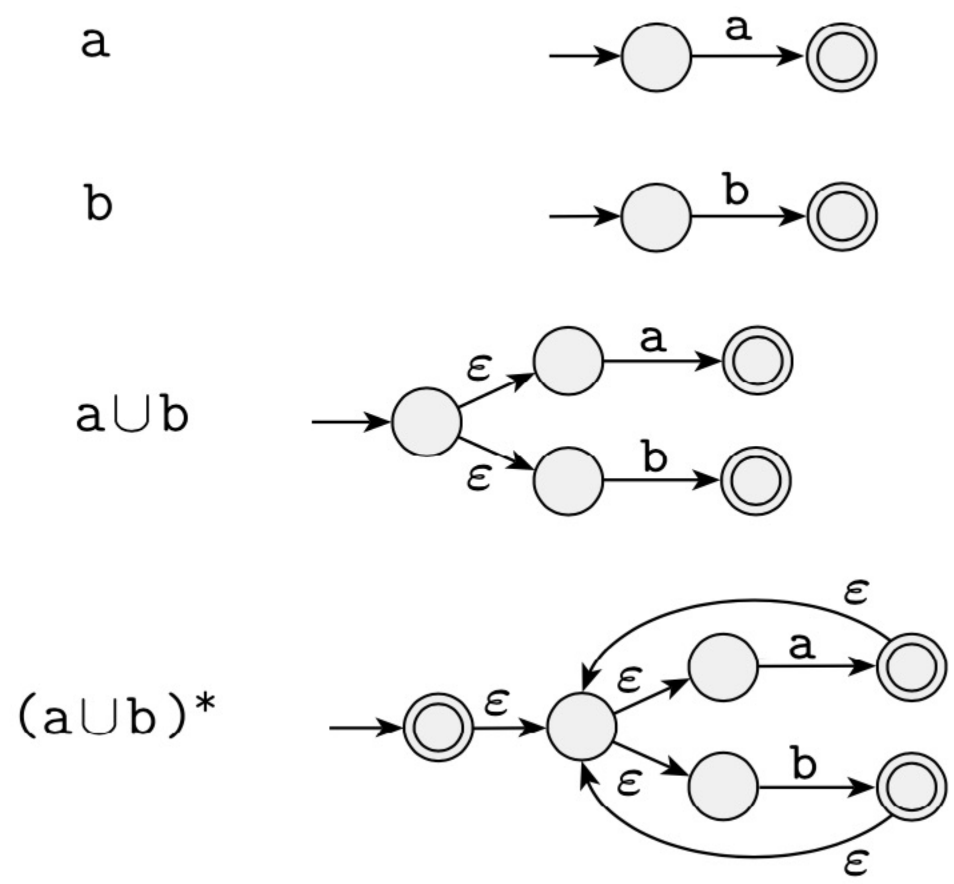
b



$a \cup b$

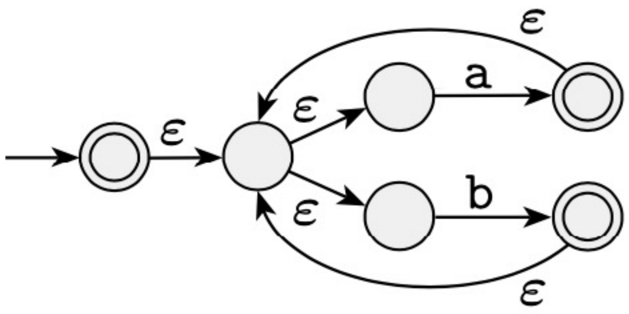


Example 2: $(a \cup b)^* aba$

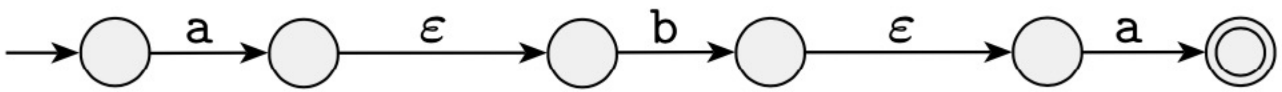


Example 2: $(a \cup b)^* aba$

(a∪b)*



aba



Example 2: $(a \cup b)^* aba$