Regular expressions

are a formalism for describing a certain class of languages: declarative rather than computational view.

Definition: given an alphabet $\Sigma$, regular expressions are strings over the alphabet $\Sigma \cup \{+, *, (, ), \epsilon, \emptyset\}$ defined inductively as follows:
- basis: $\epsilon$, $\emptyset$, and each $a \in \Sigma$ is a R.E.
- inductive step: if $E$ and $F$ are R.E., then so are:
  - $E + F$ (union)
  - $E \cdot F$ (concatenation)
  - $E^*$ (closure)
  - $(E)$ (parentheses)

Example: $\epsilon \cdot (a+b)^* \cdot b^* \cdot \epsilon$

Definition: language $L(E)$ defined by a R.E. $E$

is also defined inductively:
- $L(\epsilon) = \{\epsilon\}$ empty word
- $L(\emptyset) = \emptyset$ empty language
- for each $a \in \Sigma \quad L(a) = \{a\}$
- $L(E + F) = L(E) \cup L(F)$
- $L((E)) = L(E)$
- $L(E \cdot F) = L(E) \cdot L(F)$ ... concatenation

concatenation of two languages $L_1$ and $L_2$:

$L_1 \cdot L_2 = \{w \mid w = x \cdot y, x \in L_1, y \in L_2\}$

Example: $E = \epsilon + 1 \implies L(E) = \{\epsilon, 1\}$

$F = \epsilon + 0 + 1 \implies L(F) = \{\epsilon, 0, 1\}$

$G = E \cdot F = \implies L(G) = \{\epsilon, 0, 1, 10, 11\}$

$= (\epsilon + 1) \cdot (\epsilon + 0 + 1)$
\( L(E^* ) = (L(E))^* \) ... closure

**Closure of a language \( L \)?

We first define the powers of a language \( L \):

- \( L^0 = \{ \varepsilon \} \)
- \( L^k = L^{k-1} \cdot L \)

Hence \( L^k = \{ w \mid w = x_1 \cdots x_k, \text{with } \forall x_i, x_k \in L \} \)

Closure of \( L \): \( L^* = L^0 \cup L^1 \cup L^2 \cup \cdots = \bigcup_{k=0}^{\infty} L^k \)

Example:

- \( E = 0 + 1 \) \( \Rightarrow L(E) = \{ 0, 1 \} \)
- \( F = E^* \) \( \Rightarrow L(F) = \text{set of all binary strings} \)
- \( E = 0 \cdot 0 \) \( \Rightarrow L(E^*) = \{ \varepsilon, 00, 0000, 000000, \ldots \} \)
  
  = all even-length strings of 0's

Positive closure of a language \( L \)

\( L^+ = L^1 \cup L^2 \cup \cdots \)

We can introduce a positive closure operator on \( \text{RE} \):

\( L(E^+) = (L(E))^+ \)

**Note:** we have to distinguish between an expression \( E \) and the language \( L(E) \) defined by \( E \)

When we write \( E = F \), we usually mean not syntactic equality, but equality of the corresponding languages, i.e.: \( L(E) = L(F) \).

In other words, equality is in the algebra of \( \text{RE} \).

**Precedence of operators:**

\[
\begin{array}{c|c|c}
\text{high} & \ast & \text{example: } E + F \cdot G^* = E + (F \cdot (G^*)) \\
\downarrow & . & \\
\text{low} & + & \\
\end{array}
\]
Algebraic laws for R.E.

Similar to the laws for arithmetic expressions, we can express laws for R.E.: treat + as sum - as product

- associativity of + and *:
  \[(E \cdot F) \cdot G = E \cdot (F \cdot G) = E \cdot F \cdot G\]
  \[(E + F) + G = E + (F + G) = E + F + G\]

- commutativity of +:
  \[E + F = F + E\]

Note: * is not commutative: \[E \cdot F \neq F \cdot E\]

- distributivity:

  1) Left distributive law of \(*\) over +:
  \[E \cdot (F + G) = E \cdot F + E \cdot G\]

  2) Right:
  \[(F + G) \cdot E = F \cdot E + G \cdot E\]

Proof of (1): the law actually holds for arbitrary languages, and does not require \(E, F, G\) to be R.E.

Hence, we prove: for arbitrary languages \(L, M, N\):

\[L \cdot (M \cup N) = L \cdot M \cup L \cdot N\]

We show that for a string \(w\) we have \(w \in L \cdot (M \cup N)\) if \(w \in L \cdot M \cup L \cdot N\)

"Only if": \(w \in L \cdot (M \cup N) \implies w = k \cdot y\) with \(k \in L, y \in M \cup N\)

Since \(y \in M \cup N\), either \(y \in M\) or \(y \in N\) (or both).

If \(y \in M\), then \(w = k \cdot y \in L \cdot M\), hence \(w \in L \cdot M \cup L \cdot N\)

(similarly for \(y \in N\))

"If": \(w \in L \cdot M \cup L \cdot N\), hence either \(w \in L \cdot M\) or \(w \in L \cdot N\).

If \(w \in L \cdot M\), then \(w = k \cdot y\) with \(k \in L, y \in M\). (Similarly, whenever \(y \in M\), and \(w = x \cdot y\) \(x \in L \cdot (M \cup N)\).)
Example: \( 0 \cdot 0 + 01^* = 0 \cdot (0 + 1^*) \)

we can factor out \( e \) \( 0 \) from the union

What about \( 0 + 0 \cdot 1^* \)?

if we factor out \( e \) \( 0 \), what remains of the summand on the left?

\[
0 + 0 \cdot 1^* = 0 \cdot e + 0 \cdot 1^* = 0 \cdot (e + 1^*) = 0 \cdot 1^*
\]

\( \therefore \) identity

- identities and annihilators (hold for arbitrary languages)
  - \( \emptyset + E = E + \emptyset = E \)
  - \( e \cdot E = E \cdot e = E \)
  - \( \emptyset \cdot E = E \cdot \emptyset = \emptyset \)

- idempotency
  - \( E + E = E \)
  - \( (E^*)^* = E^* \)  
  Proof: Exercise 3.4.1 f

- other laws for closure (already seen)
  - \( \emptyset^* = \emptyset \)
  - \( e^* = e \)
  - \( E^+ = E \cdot E^* = E^* E \)
  - \( E^* = E^+ + e \)

  Note: if \( e \in L(E) \), then \( E^* = E^+ \)

Exercise 3.4.4) Prove that \( (E^* F^*)^* = (E + F)^* \)
Exercise 3.1.1 Write R.E.'s for the following languages:

a) \( \{ w \in \{c, b, c\}^* \mid w \text{ contains at least one } c \text{ and at least one } b \} \)

b) \( \{ w \in \{0, 1\}^* \mid w's \text{ tenth symbol from the right is } 0 \} \)

c) \( \{ w \in \{0, 1\}^* \mid w \text{ contains at most one pair of consecutive } 1's \} \)

Exercise 3.1.2 Write R.E.'s for the following languages:

a) The set of all strings over \( \{0, 1\} \) s.t. every pair of adjacent 0's appears before any pair of adjacent 1's

b) The set of strings of 0's and 1's whose number of 0's is divisible by 5

Solutions:

3.1.1 e) \( (c^* a^* (a+c)^* b^* (c+b+c)^*)^* + (c^* b^* (b+c)^* a^* (a+b+c)^*)^* \)

b) \( (0+1)^* \text{ 1 \( \overbrace{- (0+1) \ldots - (0+1)}^{3 \text{ times}} \) } (0+1) \)

c) \( 0^*(1.0^+)^* \cdot 1.1. (0^+.1) .0^* + 0^* .(1.0^+) .(0+1)^* \)

on, simplify \( (0+1)^* (E+1) .(0+01)^* \)

3.1.2 e) \( \underbrace{(0+10)^* \ldots (1+10)^*}_{\text{no pair of adjacent 1's}} \cdot \underbrace{(1+10)^*}_{\text{no pair of adjacent 0's}} \)

b) \( (1 \cdot 0 .1^* .0 .1^* .0 .1^* .0 .1^*)^* \)
What is the relationship between the classes of languages studied so far?

\[ \varepsilon\text{-NFA} \iff \text{NFA} \iff \text{DFA} \iff \text{regular languages} \iff \text{R.E.} \]

**Theorem:** \((\text{R.E.} \rightarrow \varepsilon\text{-NFA})\)

For every R.E. \( E \) there is an \( \varepsilon\text{-NFA} \( A_E \) s.t. \( L(A_E) = L(E) \).

**Proof:** let us call an \( \varepsilon\text{-NFA} \) simple if
- it has only one final state
- the initial state has no incoming arcs
- the final --- outgoing ---

We show by structural induction that for each R.E. \( E \) there is a simple \( \varepsilon\text{-NFA} \( A_E \) s.t. \( L(E) = L(A_E) \).

**Basis:** \( E = \varepsilon, \phi, \epsilon \) for some \( \epsilon \in \Sigma \)

\[ \begin{align*}
A_{\varepsilon} & \rightarrow q_0 \xrightarrow{\varepsilon} q_1 \\
A_{\phi} & \rightarrow q_0 \xrightarrow{} q_0 \\
A_{\epsilon} & \rightarrow q_0 \xrightarrow{\epsilon} q_1
\end{align*} \]

**Inductive case:**
1. \( E = F \cup G \)
2. \( E = F \cdot G \)
3. \( E = F^* \)
4. \( E = \left( F \right) \)
By I.H., there are simple $\varepsilon$-NFA$\omega$s for $F$ and $G$.

1) $E = F \sqcap G$

$L(A_E) = L(A_F) \cup L(A_G) = L(F) \cup L(G) = L(F \sqcap G) = L(E)$

by I.H.

2) $E = F \cdot G$

$L(A_E) = L(A_F) \cdot L(A_G) = L(F) \cdot L(G) = L(F \cdot G) = L(E)$

by I.H.

3) $E = F^*$

$L(A_E) = L(A_F)^* = L(E)$

by I.H.

3) $E = (F)$

$A_E = A_F$
Example: \( E = Q^* \in L \subseteq \Sigma \)

\[
E \rightarrow O \rightarrow E \rightarrow E \rightarrow E \rightarrow E
\]

\[
E \rightarrow O \rightarrow E \rightarrow E \rightarrow E \rightarrow E
\]

\[
E \rightarrow O \rightarrow E \rightarrow E \rightarrow E \rightarrow E
\]

\[
OPTIMAL
\]

**Theorem (DFA \( \rightarrow \) R.E.)**

For every DFA \( A \) there is a R.E. \( E_A \) s.t. \( L(E_A) = L(A) \)

**Proof:** Let \( A = (Q, \Sigma, \delta, q_0, F) \)

We assume without loss of generality (w.l.o.g.) that \( Q = \{q_1, q_2, \ldots, q_n\} \)

Let us define \( L_{ij} = \{ w \mid \hat{\delta}(q_i, w) = q_j \} = \{ w \mid w \text{ takes } A \text{ from } q_i \text{ to } q_j \} \)

Note that \( L_{ij} = L(A_{ij}) \) with \( A_{ij} = (Q, \Sigma, \delta, q_i, \{q_j\}) \)

We aim at constructing R.E.'s \( E_{ij} \) for \( L_{ij} \).

Then we can take \( E_A = \Sigma \bigcup E_{ij} \), since

\[
L(E_A) = U_{q_j \in F} L(E_{ij}) = U_{q_j \in F} \{ w \mid \hat{\delta}(q_i, w) = q_j \} = L(A)
\]

**How can we compute \( E_{ij} \)?**

Let us define \( \forall i, j \in \{1, \ldots, n\}, \forall k \in \{0, \ldots, n\} \)

\[
L_{ik} = \{ w \mid A \text{ goes from } q_i \text{ to } q_k \text{ on input } w,
\text{passing only through } q_1, \ldots, q_{i-1} \text{ and } q_{i+1}, \ldots, q_n \text{ as intermediate states} \}
\]
Example:

\[
\begin{array}{c}
q_9 \xrightarrow{e} q_3 \xrightarrow{c} q_5 \\
q_4 \xrightarrow{f} q_5
\end{array}
\]

\[
a b c \in L_{15}^3 \]
\[
def \notin L_{15}^3 \quad \text{but} \quad def \in L_{15}^4
\]
\[
def \in L_{15}^4
\]
\[
L_{12} = \{e, d\}
\]
\[
L_{15}^3 = \{abc, dbc\}
\]
\[
L_{15}^4 = L_{15}^5 = \{abc, dbc, def, def\}
\]

Note: \(L_{ij}^+ = L_{ij}\)

Hence, we are done if we can construct REs \(E_{ij}^k\) for \(L_{ij}^k\).

We can simply take \(E_{ij} = E_{ij}^k\), and hence \(E_A = \sum_{q_j \in F} E_{ij}^k\).

We construct \(E_{ij}^k\) by induction on \(k\):

**Basis:** we construct \(E_{ij}^0\) for all \(i, j \in \{1, ..., n\}\)

- Since \(k=0\), we cannot go through any intermediate state.
- 2 cases: each with 2 sub-cases:
  - \(i \neq j\)
    - \(E_{ij}^0 = \varepsilon q_i \ldots q_j \)
    - \(E_{ij}^0 = \emptyset\)
  - \(i = j\)
    - \(E_{ii}^0 = \varepsilon + e_i \ldots e_i \)
    - \(E_{ii}^0 = \varepsilon\)
Induction: assume we have constructed $E^{k-1}_{ij}, \forall k, j \in \{1, \ldots, n\}$ we show how to construct $E^k_{ij}$.

Observe:
- $L^k_{ij}$ will include $L^{k-1}_{ij}$
- it additionally will contain those words that lead through $q_k$ at least once, when going from $q_i$ to $q_j$:

$$w = x_1 x_2 \ldots x_n \text{ where: } x_n \text{ represents transitions going at most through } \{x_1, \ldots, x_{k-1}\}$$

Then $x_1 \in L^{k-1}_{ik}$, $x_2, \ldots, x_{n-1} \in L^{k-1}_{kk}$, $x_n \in L^{k-1}_{kj}$

$$\Rightarrow w \in L^{k-1}_{ik} \cdot (L^{k-1}_{kk})^* \cdot L^{k-1}_{kj}$$

$$\Rightarrow E^k_{ij} = E^{k-1}_{ij} + E^{k-1}_{ik} \cdot (E^{k-1}_{kk})^* \cdot E^{k-1}_{kj}$$

\[\text{Example:}\]

\[\begin{array}{cccc}
0 & E^0_{11} & E^0_{12} & E^0_{21} & E^0_{22} \\
0 & \varepsilon + 0 & 1 & \emptyset & \varepsilon + 0 + 1 \\
1 & (\varepsilon + 0) + (\varepsilon + 0)(\varepsilon + 0)^* + 1 = 0^*1 & \emptyset & \emptyset & \emptyset \\
2 & \text{not needed} & \text{not needed} & \text{not needed} & \text{not needed} \\
\end{array}\]

\[S(A) = S(E^2_{12}) = S(E^2_{22}) = \emptyset \cdot (0^*1)^* \]

\[E^1_{11} = E^1_{12} = E^1_{22} = \emptyset \cdot (0^*1)^* \]

\[\text{OPTIONAL}\]
Theorem \( (NFA \rightarrow R.E.) \)

For every NFA \( A \) there is a R.E. \( E_A \) s.t. \( L(E_A) = L(A) \)

Proof sketch: We show how to construct \( E_A \) by eliminating states of \( A \).

Consider the elimination of a state \( s \):

If there was a path from state \( p \) to state \( q \) over \( s \), after eliminating \( s \) the path does no longer exist.

We have to compensate for that.

We add a regular expression “connecting” \( p \) and \( q \) and capturing the missing path.

\[
\begin{align*}
\text{We can eliminate in this way all states except initial and} \\
\text{final states:} \\
\text{Strategy: Assume} \ A \ \text{has the final states} q_1, \ldots, q_k. \\
\text{For each final state} q_i, \text{eliminate all states except} q_i, q_0.
\end{align*}
\]

a) If \( q_i \neq q_0 \), we are left with:

\[
E_i = (R + S \cdot U^* \cdot T)^* \cdot S \cdot U^* 
\]

b) If \( q_i = q_0 \in F \), we must eliminate all states except \( q_0 \).

We are left with \( R \) the corresponding R.E.is

\[
E_i = R^* 
\]

Then \( E_A \) is the sum of all \( E_i \):

\[
E_A = E_1 + E_2 + \cdots + E_k 
\]
Example

We view all edge labels as R.E.'s (missing labels mean $\emptyset$)

Eliminate $B$:

$E_B = \emptyset + 1 \cdot \emptyset^* (0+1) = 1 \cdot \emptyset^* (0+1) = 1 \cdot (0+1)$

Eliminate $C$:

$E_C = \emptyset + 1 \cdot (0+1) \cdot \emptyset^* (0+1) = 1 \cdot (0+1) \cdot (0+1)$

$E_1 = (0+1)^* \cdot E_C = (0+1)^* \cdot 1 \cdot (0+1) \cdot (0+1)$

Eliminate $D$:

$E_2 = (0+1)^* \cdot 1 \cdot (0+1)$

$E = E_1 + E_2 = (0+1)^* \cdot 1 \cdot (0+1) \cdot (0+1) + (0+1)^* \cdot 1 \cdot (0+1)$

$= (0+1)^* \cdot 1 \cdot (0+1) \cdot (0 \cdot 0 \cdot 1)$