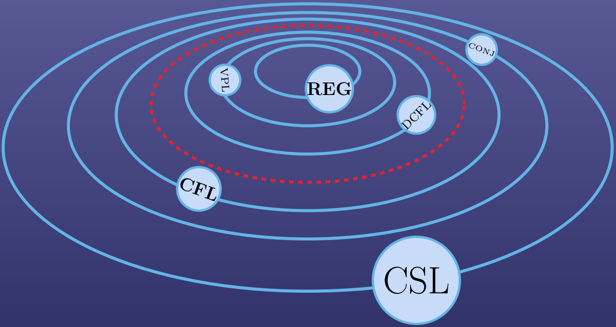
Bauman Moscow State University
Th. Computer Science Dept.

Finite State Machines and Regular Expressions

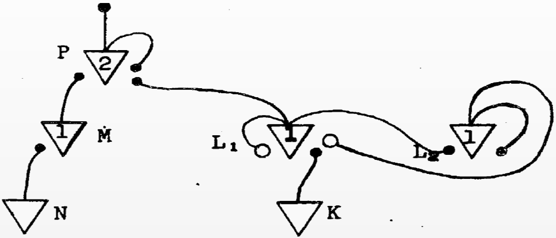


Antonina Nepeivoda
a_nevod@mail.ru

Lecture Outline

1. **Basic Notions**
2. **Closures and Determinisation**
   - ε-Removal by Closure
   - Subset Construction and Determinisation
3. **From NFA to Regular Expressions**
   - Solving Language Equations
   - State Eliminating Method
4. **From Regular Expressions to NFA**

Reminder: Neural Networks by McCulloch–Pitts



• — excitatory signal;

◦ — inhibitory signal;

▽ — an input neuron;

k — an inner neuron firing whenever none of the inhibitory signals and at least k of excitatory signals fire.

Naturally imitate: disjunction, conjunction, negation, iteration, concatenation.

Regular Expressions by Kleene

👀 Academic Definition

Given alphabet Σ, a regular expression is either a letter in Σ, ε, or a result of following operations, where r_1, r_2 are regular expressions:

- $r_1 \mid r_2$ — *union (alternation).* $\mathscr{L}(r_1 \mid r_2) = \mathscr{L}(r_1) \cup \mathscr{L}(r_2)$;
- $r_1\, r_2$ — *concatenation (sequencing).* $\mathscr{L}(r_1\, r_2) = \left\{\omega_1\omega_2 \mid \omega_1 \in \mathscr{L}(r_1)\ \&\ \omega_2 \in \mathscr{L}(r_2)\right\}$;
- $(r_1)^*$ — *iteration (0 or more concatenations of r_1 with itself);* $\mathscr{L}\left((r_1)^*\right) = \{\varepsilon\} \bigcup\limits_{i=1}^{\infty} \mathscr{L}(r_1)$.

Syntactic Sugar

- r^+ — positive iteration (shortcut for $r\, r^*$);
- $r?$ — option (shortcut for $(r \mid \varepsilon)$).

Regular Expressions by Kleene

👀 Academic Definition

Given alphabet Σ, a regular expression is either a letter in Σ, ε, or a result of following operations, where r_1, r_2 are regular expressions:

- $r_1 \mid r_2$ — *union (alternation).* $\mathscr{L}(r_1 \mid r_2) = \mathscr{L}(r_1) \cup \mathscr{L}(r_2)$;
- $r_1\, r_2$ — *concatenation (sequencing).* $\mathscr{L}(r_1\, r_2) = \left\{\omega_1\omega_2 \mid \omega_1 \in \mathscr{L}(r_1) \,\&\, \omega_2 \in \mathscr{L}(r_2)\right\}$;
- $(r_1)^*$ — *iteration (0 or more concatenations of r_1 with itself);* $\mathscr{L}\big((r_1)^*\big) = \{\varepsilon\} \bigcup_{i=1}^{\infty} \mathscr{L}(r_1)$.

Priorities: star > concatenation > union.

$$ab^* \mid c^*d \Leftrightarrow \Big(a(b^*)\Big) \mid \Big((c^*)d\Big).$$

Terminological Clash

Academic regexes

- $|, \cdot, {}^*$ (sometimes $^+$, ?) operations;
- define regular languages;
- studied in university courses (compilers & formal languages)

REGEX (extended regexes)

- lookaheads, backreferences, etc;
- define non-context-free languages;
- used in practice (PCRE2 standart).

- Almost identical names are used for completely different (although related) notions.

Occam Razor: Non-Deterministic Finite Automata

Only excitatory signals are left on there, and all inner neurons fire whenever there is at least one input signal.

Definition

A non-deterministic finite automaton (NFA) is a tuple $\mathscr{A} = \langle Q, \Sigma, q_0, F, \delta \rangle$, *where:*

- Q — *state set;*
- Σ — *terminal alphabet;*
- $\delta : Q \times (\Sigma \cup \{\varepsilon\}) \to 2^Q$ — *transition rules;*
- $q_0 \in Q$ — *starting state;*
- $F \subseteq Q$ — *final states.*

Sometimes we use notation:
$\langle q_1, a, q_2 \rangle \in \delta \Leftrightarrow \langle q_1, a, M \rangle \in \delta \,\&\, q_2 \in M$.
Or, usually, simply: $q_1 \xrightarrow{a} q_2$.

Asymmetry of NFA Definition

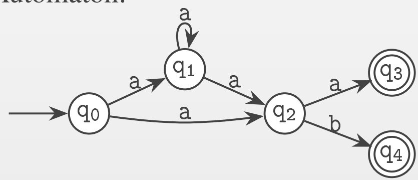
- Classical works (Kleene, Brzozowski): multiple NFA starting states are allowed.
- Modern formal language theory: the unique starting state in NFA is assumed.
- Equivalent (we can add an unique starting state with ε-transitions to the multiple states), but confusing (e.g. in Brzozowski minimisation).

Encoding into Grammars

Observation

- Transition $q_1 \xrightarrow{a} q_2$ can be seen as a rewriting rule $[q_1] \to a[q_2]$, assuming that $[q_i]$ are some intermediate constructors, while $a \in \Sigma$ is a terminal constructor.
- In order to model computation termination, for every final state q_F, we can add the rewriting rule $[q_F] \to \varepsilon$.

Automaton:



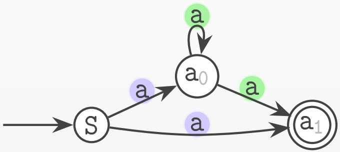
Grammar:

$$
\begin{array}{ll}
S \to \mathtt{a}[q_1] & [q_2] \to \mathtt{a}[q_3] \\
S \to \mathtt{a}[q_2] & [q_2] \to \mathtt{b}[q_4] \\
[q_1] \to \mathtt{a}[q_1] & [q_3] \to \varepsilon \\
[q_1] \to \mathtt{a}[q_2] & [q_4] \to \varepsilon
\end{array}
$$

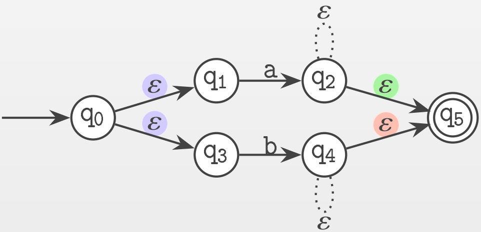
We rename the starting nonterminal $[q_0]$ to S, for uniformity.

Sources of Non-Determinism in an NFA

- Transition sets wrt (with respect to) a letter $\gamma \in \Sigma$ that are not singletons.



- ε-transitions (so-called silent actions).

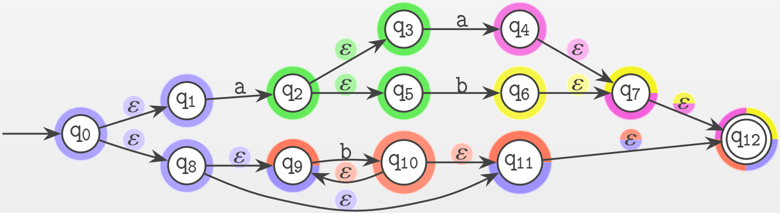


Closures

Given $\omega \in \Sigma^*$, a ω-closure of a state q in NFA $\mathscr{A}$ is a set of states reachable from q by the action ω.

We say that ω is in the language of the NFA $\mathscr{A}$ ($\omega \in \mathscr{L}(\mathscr{A})$) $\Leftrightarrow$ ω-closure of the starting state of $\mathscr{A}$ contains a final state.

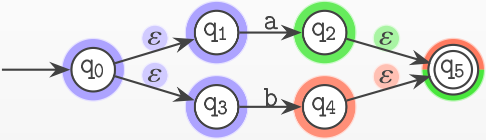
Special case: ε-closures: sets of states reachable via doing nothing.



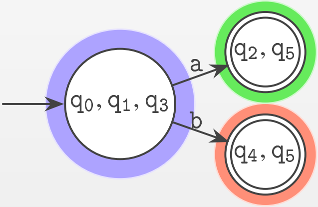
Given such closures, they can be considered as new «states».

Simple Example of ε-Removal

An NFA $\mathscr{A}$ with the ε-closures of its states being highlighted:



The closures are then merged into single states, and given a transition from $q_i \xrightarrow{\gamma} q_j$, where q_i belongs to closure $M(q_i)$, and q_j to $M(q_j)$, transition $M(q_i) \xrightarrow{\gamma} M(q_j)$ is added.



A closure is marked as a final $\Leftrightarrow$ it contains at least one final state.

ε-Closures and Chain Rules

- *Any transition* $q_i \xrightarrow{\varepsilon} q_j$ *corresponds to* ***a chain rule*** $[q_i] \to [q_j]$ *in the corresponding grammar* G.
- *state* ε*-closure is a closure set of the corresponding non-terminal* N:
  $C(N) = \left\{N_i \mid \exists N'_1, \dots N'_k \left(N \to N'_1 \;\&\; \cdots \;\&\; N'_k \to N_i\right)\right\}$
  I.e. $\langle N, N_i \rangle$ *are pairs in* ***a transitive closure*** $\to_c^+$ *of the relation* $\to_c$:
  $A_i \to_c A_j \Leftrightarrow (A_i \to A_j \in G)$.
- *Before removing all chain rules, for every* $N' \in C(N)$ *and a non-chain rule* $N' \to \Phi$*, we add the transition* $N \to \Phi$ *to the set of grammar rules. Exactly as in the* ε*-closure algorithm for NFA.*

Initial grammar:

$S \to Q_1$ $\quad S \to Q_3$ $\quad Q_1 \to \mathtt{a}Q_2$

$Q_3 \to \mathtt{b}Q_4$ $\quad Q_2 \to Q_5$ $\quad Q_4 \to Q_5$

$Q_5 \to \varepsilon$

After removing chain rules:

$S \to \mathtt{a}Q_2$ $\quad S \to \mathtt{b}Q_4$

$Q_2 \to \varepsilon$ $\quad Q_4 \to \varepsilon$

Note: unreachable non-terminals Q_1, Q_3, Q_5 are deleted from the resulting grammar.

ω-Closures and Subset Construction

The closure sets wrt transitions by non-ε actions can be also merged in similar sense.

Subset Automaton Construction

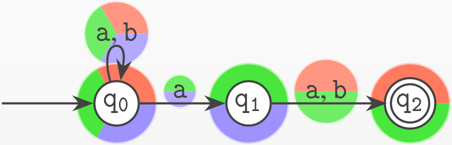
Let an ε-free NFA $\mathscr{A}$ be given. Its **subset automaton** $D(\mathscr{A})$ can be constructed as follows.

- q_0 becomes the starting state $\{q_0\}$ of $D(\mathscr{A})$.
- Given a state M in $D(\mathscr{A})$ and $\gamma \in \Sigma$, construct a closure set $M_\gamma = \{q_i \mid \exists q_j \in M(q_j \xrightarrow{\gamma} q_i)\}$. If M_γ is non-empty and does not yet introduced as a state of $D(\mathscr{A})$, add it to set of states of $D(\mathscr{A})$.
- The final states of $D(\mathscr{A})$ are labelled with the sets containing at least one final state of $\mathscr{A}$.

In fact, the states of $D(\mathscr{A})$ are ω-closures of $\mathscr{A}$-states, where $\omega \in \Sigma^*$.

Subset Automaton: a Simple Example

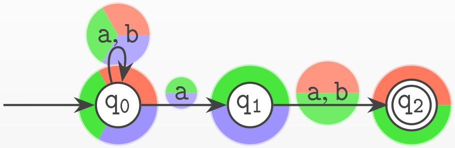
Let us consider the following NFA with γ-closures of its states:



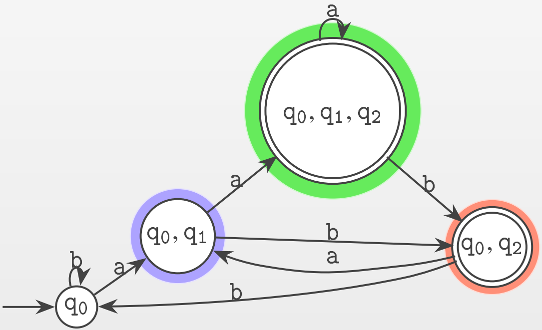
- $\mathtt{a}$-closure of starting state $\{q_0\}$ is $\{q_0, q_1\}$.
- $\mathtt{b}$-closure of starting state $\{q_0\}$ is the state $\{q_0\}$ itself.
- $\mathtt{a}$-closure of the state $\{q_0, q_1\}$ is $\{q_0, q_1, q_2\}$.
- $\mathtt{b}$-closure of the state $\{q_0, q_1\}$ is $\{q_0, q_2\}$.
- $\mathtt{a}$-closure of the state $\{q_0, q_1, q_2\}$ is $\{q_0, q_1, q_2\}$ itself, while $\mathtt{b}$-closure is the state $\{q_0, q_2\}$.
- $\mathtt{a}$-closure of the state $\{q_0, q_2\}$ is $\{q_0, q_1\}$, while $\mathtt{b}$-closure is $\{q_0\}$.

Subset Automaton: a Simple Example

Let us consider the following NFA with γ-closures of its states:



Hence, its subset automaton is:



Deterministic Finite Automata

Definition

A deterministic finite automaton (DFA) is a tuple $\mathscr{A} = \langle Q, \Sigma, q_0, F, \delta \rangle$, where:

- *Q is a state set, Σ is a terminal alphabet;*
- *δ is a transition set $\langle q_i, \gamma, q_j \rangle$, where $q_i, q_j \in Q$, $\gamma \in \Sigma$, and for any q_i, γ there is at most one q_j such that $q_i \xrightarrow{\gamma} q_j \in \delta$;*
- *$q_0 \in Q$ is a starting state, $F \subseteq Q$ is a set of final states.*

Language $\mathscr{L}(\mathscr{A})$ of DFA $\mathscr{A}$ is a set $\left\{\omega \mid \exists q \in F(q_0 \xrightarrow{\omega} q)\right\}$, i.e. there exists a final state that is ω-closure of q_0.

By construction, the subset automaton has no non-determinism in the transition set:

- ε-transitions are eliminated in the preliminary ε-free NFA;
- the non-singleton transition sets are processed in the subset construction.

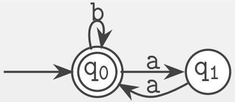
Traps and Trims

- **A trap state** is a state s.t. any its ω-closure is non-final.

Trim DFA

- For any q_i, γ there is **at most one** q_j s.t. $q_i \xrightarrow{\gamma} q_j \in \delta$;
- is naturally constructed via subset technique;
- default in RoFL course, useful for most operations.

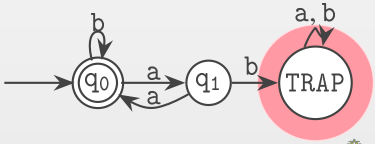
Trim DFA example



Complete DFA

- For any q_i, γ there is **exactly one** q_j s.t. $q_i \xrightarrow{\gamma} q_j \in \delta$;
- usually requires introducing **trap (sink) states**;
- useful for constructing complementation.

DFA with the trap state for $\Sigma = \{\mathtt{a}, \mathtt{b}\}$



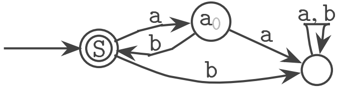
Complementation and Traps

- By switching finality of all states in DFA $\mathscr{A}$, we can construct a DFA $\mathscr{A}'$ accepting exactly the set of words that are rejected by the initial DFA, i.e. $\mathscr{L}(\mathscr{A}') = \Sigma^* \setminus \mathscr{L}(\mathscr{A})$.
- The language complementation requires complete DFA.

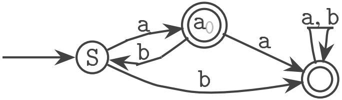
Initial trim DFA for $(\mathtt{ab})^*$

Invalid complement DFA recognising $\mathtt{a}(\mathtt{ba})^*$

Complete DFA for $(\mathtt{ab})^*$



Valid compement DFA



- Without a trap state, complementation operation loses words starting with b, or containing either aa or bb.

One More Encoding: Equations

Sometimes it is convenient to gather all the right-hand sides of the rules with a same left-hand side together. Then, if we replace $\to$ by $=$ sign, we get an equation system determining non-terminal languages:

$$\begin{array}{ll} S \to \mathtt{a}[q_1] & [q_2] \to \mathtt{a}[q_3] \\ S \to \mathtt{a}[q_2] & [q_2] \to \mathtt{b}[q_4] \\ [q_1] \to \mathtt{a}[q_1] & [q_3] \to \varepsilon \\ [q_1] \to \mathtt{a}[q_2] & [q_4] \to \varepsilon \end{array} \quad \longrightarrow \quad \begin{array}{l} S = \mathtt{a}[q_1] \mid \mathtt{a}[q_2] \\ [q_1] = \mathtt{a}[q_1] \mid \mathtt{a}[q_2] \\ [q_2] = \mathtt{a}[q_3] \mid \mathtt{b}[q_4] \\ [q_3] = \varepsilon \\ [q_4] = \varepsilon \end{array}$$

If there is no rule part $[q_1] = \mathtt{a}[q_1]$, these languages could be found by exhaustive substitutions of the right-hand sides.

E.g. $\mathscr{L}([q_3]) = \mathscr{L}([q_4]) = \{\varepsilon\}$, while
$\mathscr{L}([q_2]) = \{\mathtt{a}\mathscr{L}([q_3])\} \cup \{\mathtt{b}\mathscr{L}([q_4])\} = \{\mathtt{a}, \mathtt{b}\}$.

How to deal with self-referring rules as $[q_1] = \mathtt{a}[q_1]$*?*

Arden's Lemma

👀 Theorem

If a language $\mathscr{L}$ satisfies the equation $\mathscr{L} = \mathscr{L}_1\mathscr{L} \cup \mathscr{L}_2$, where $\varepsilon \notin \mathscr{L}_1$, then $\mathscr{L} = \mathscr{L}_1^*\mathscr{L}_2$.

Proof: Let us consider arbitrary $\omega \in \mathscr{L}$.

- If $\omega \in \mathscr{L}_2$, then the statement trivially holds.
- Otherwise, $\exists \omega_1 \in \mathscr{L}_1, \omega' \in \mathscr{L} (\omega = \omega_1\omega')$. The suffix ω' also belongs to $\mathscr{L}_1\mathscr{L} \cup \mathscr{L}_2$, and $|\omega'| < |\omega|$, since $\omega_1 \neq \varepsilon$. Now we can repeat the same reasoning for ω', and due to finiteness of $|\omega|$ and well-foundedness of $(\mathbb{N}, <)$ we will eventually get $\omega' \in \mathscr{L}_2$. $\square$

Arden's lemma allows one to solve the equation systems in Gaussian style, via non-terminal elimination + substitution, assuming there are no chain rules in the grammar.

Equation Solving Example

Let us construct the language of the grammar:

$$S \to aT \quad S \to aS$$

$$T \to aT \quad T \to bT \quad T \to bF \quad F \to \varepsilon$$

First, construct the system and substitute F:

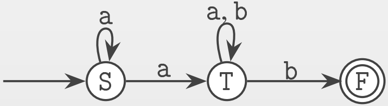
$$S = (aS) \mid (aT)$$

$$T = ((a \mid b)T) \mid b(\varepsilon)$$

Solve the second equation: $T = (a \mid b)^* b$
Then substitute the solution: $S = (aS) \mid (a(a \mid b)^* b)$.

The resulting language is: $S = a^* a (a \mid b)^* b$

The NFA that corresponds to the grammar is given below:



Equation Solving Example

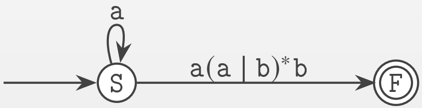
Let us construct the language of the grammar:

$$S \to aT \quad S \to aS$$

$$T \to aT \quad T \to bT \quad T \to bF \quad F \to \varepsilon$$

The resulting language is: $S = a^*a(a \mid b)^*b$

The NFA that corresponds to the grammar is given below .
After solving T-based equation and substituting F value, in fact we again constructed an NFA, whose transitions are marked with regexes.



If we assume that S is preceded by the "very starting state" S', then $\mathscr{L}(S)$ can be also considered as a transition in the NFA containing only S' and F states.

Finding NFA Language

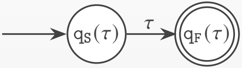
The extended NFAs allow one to use transitions marked with regexes.

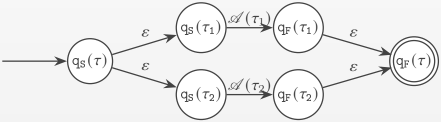
State Exclusion Method

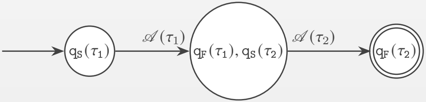
- For the sake of uniformity, we introduce "the very starting state" S, having ε-transition to q_0, and "the very final state" T, having ingoing ε-transitions from $q \in F$. All the states except S and T are now ordinary.
- In order to exclude the state q s.t. $q \xrightarrow{\tau} q$, for all pairs q_A, q_B, where $q_A \xrightarrow{\Phi} q$, $q \xrightarrow{\Psi} q_B$, add the transition $q_A \xrightarrow{\Phi(\tau)^*\Psi} q_B$, then we can delete q.
- When only S and T are left, where $S \xrightarrow{\rho} T$, the expression ρ is the regex equivalent to the NFA.

Modular Construction: Thompson NFA

Any regular expression τ has a recursive structure. Let us use this structure to model the corresponding NFA.

- $\tau = \gamma, \gamma \in \Sigma \Rightarrow \mathscr{A}(\tau)$ is 

- $\tau = \tau_1 \mid \tau_2 \Rightarrow \mathscr{A}(\tau)$ is 

- $\tau = \tau_1\tau_2 \Rightarrow \mathscr{A}(\tau)$ is 

- $\tau = {\tau_1}^* \Rightarrow \mathscr{A}(\tau)$ is