Regulární výraz:

• ∅ je RV značící prázdnou množinu (prázdný jazyk)
• ε je RV značící jazyk { ε }
• a, kde a ∈ Σ, je RV značící jazyk { a }
• Nechť r a s jsou regulární výrazy značící po řadě jazyky L_r a L_s, potom:
    • (r.s) je RV značící jazyk $L = L_r L_s$
    • (r + s) je RV značící jazyk $L = L_r \cup L_s$
    • (r *) je RV značící jazyk $L = L_r^*$

Konečný automat:

Konečný automat (KA) je pětice: M = (Q, Σ, R, s, F), kde:
    • Q je konečná množina stavů
    • Σ je vstupní abeceda
    • R je konečná množina pravidel tvaru: pa → q, kde p, q ∈ Q, a ∈ Σ ∪ {ε}
    • s ∈ Q je počáteční stav
    • F ⊆ Q je množina koncových stavů

Zásobníkový automat:

Zásobníkový automat (ZA) je sedmice: M = (Q, Σ, Γ, R, s, S, F), kde
    • Q je kone čná množina stavů
    • Σ je vstupní abeceda
    • Γ je zásobníková abeceda
    • R je konečná množina pravidel tvaru Apa→wq, kde A ∈ Γ,p,q ∈ Q,a ∈ Σ ∪{ ε}, w ∈ Γ *
    • s ∈ Q je počáteční stav
    • S ∈ Γ je počáteční symbol na zásobníku
    • F ⊆ Q je množina koncových stavů

Turingův stroj: (Údajně se nezkouší)

Turingův stroj (TS) je šestice M = (Q, Σ, Γ, R, s, F), kde
    • Q je konečná množina stavů
    • Σ je vstupní abeceda
    • Γ je pásková abeceda; Δ ∈ Γ; Σ ⊆ Γ
    • R je konečná množina pravidel tvaru: pa → qbt , kde p , q ∈ Q , a , b ∈ Γ, t ∈ { S, R, L}
    • s ∈ Q je počáteční stav • F ⊆ Q je množina koncových stavů

Ekvivalentní modely:

Dva modely pro popis formálních jazyků (např. konečné automaty) jsou ekvivalentní, pokud specifikují tentýž jazyk.

DKA:
Nechť $M = (Q, \Sigma, R, s, F)$ je KA bez ε-přechodů. M je deterministický konečný automat (DKA), pokud pro každé $pa \to q \in R$ platí, že množina $R - \{pa \to q\}$ neobsahuje žádné pravidlo s levou stranou pa.

Dostupný stav:
Nech ť $M = (Q, \Sigma, R, s, F)$ je KA. Stav $q \in Q$ je dostupný, pokud existuje $w \in \Sigma^*$, pro který platí $sw \vdash^* q$. Jinak q je nedostupný.

Ukončující stav:
Nechť $M = (Q, \Sigma, R, s, F)$ je DKA. Stav $q \in Q$ je ukončující, pokud existuje řetězec $w \in \Sigma^*$, pro který platí: $qw \vdash^* f, f \in F$. Jinak q je neukončující.

Úplný DKA:
Nechť $M = (Q, \Sigma, R, s, F)$ je DKA. M je úplný, pokud pro libovolné $p \in Q, a \in \Sigma$ existuje právě jedno pravidlo $pa \to q \in R$ pro nějaké $q \in Q$. Jinak M je neúplný.

Dobře specifikovaný KA:
Nechť $M = (Q, \Sigma, R, s, F)$ je úplný DKA. Pak M je dobře specifikovaný KA (DSKA), pokud:
1) Q nemá nedostupné stavy
2) Q má maximálně jeden neukončující stav

Bezkontextová gramatika:
BKG je čtveřice $G = (N, T, P, S)$, kde
- N je abeceda neterminálů
- T je abeceda terminálů, při čemž $N \cap T = \emptyset$
- P je konečná množina pravidel tvaru $A \to x$, kde $A \in N, x \in (N \cup T)^*$
- $S \in N$ je počáteční neterminál

Jazyk generovaný BKG:
Nechť $G = (N, T, P, S)$ je BKG. Jazyk generovaný BKG G, L(G), je definován: $L(G) = \{w: w \in T^*, S \Rightarrow^* w\}$

Bezkontextový jazyk:
Nechť L je jazyk. L je bezkontextový jazyk (BKJ), pokud existuje bezkontextová gramatika, která generuje tento jazyk L

Gramatická nejednoznačnost:
Nech ť $G = (N, T, P, S)$ je BKG. Pokud existuje řetězec $x \in L (G)$ s více jak jedním derivačním stromem, potom G je nejednoznačná. Jinak G je jednoznačná.

Pumping Lemma:
Nechť L je RJ. Pak existuje $k \geq 1$ takové, že: pokud $z \in L$ a $|z| \geq k$, pak existuje u,v,w:z = uvw,
1) $v \neq \varepsilon$
2) $|uv| \leq k$
3) pro každé $m \geq 0$, $uv^m w \in L$

Chomského normální forma (CNF):
Nech ť G = (N, T, P, S) je BKG. G je v Chomského normální formě, pokud každé pravidlo z P má jeden ze tvarů:
- $A \rightarrow BC$, kde $A, B, C \in N$;
- $A \rightarrow a$, kde $A \in N$, $a \in T$;

Greibachové normální forma (GNF):
Nechť G = (N, T, P, S) je BKG. G je v Greibachové normální formě, pokud každé pravidlo z P má následující tvar:
- $A \rightarrow a x$, kde $A \in N$, $a \in T$, $x \in N^*$

Uzávěrové vlastnosti (RJ):
Třída regulárních jazyků je uzavřená vůči operaci o, pokud výsledek operace o na libovolné regulární jazyky je opět regulární jazyk.

Uzávěrové vlastnosti (BKJ):
Třída bezkontextových jazyků je uzavřená vůči operaci o, pokud výsledek operace o na libovolné bezkontextové jazyky je opět bezkontextový jazyk.

Vnitřní nejednoznačnost:
BKJ L je vnitřně nejednoznačný, pokud L není generován žádnou jednoznačnou BKG.

Minimální KA:
Nechť M je DSKA. Potom, M je minimální KA, pokud M obsahuje pouze rozlišitelné stavy.

Přijímaný jazyk ZA:
Nechť $M = (Q, \Sigma, \Gamma, R, s, S, F)$ je ZA.
1) Jazyk přijímaný ZA M přechodem do koncového stavu, značen jako L(M)f , je definován: $L(M)f = \{w: w \in \Sigma^*, Ssw \vdash^* zf, z \in \Gamma^*, f \in F\}$
2) Jazyk přijímaný ZA M vyprázdněním zásobníku, značen jako L(M)ε, je definován: $L(M)\varepsilon = \{w: w \in \Sigma^*, Ssw \vdash^* zf, z = \varepsilon, f \in Q\}$
3) Jazyk přijímaný ZA M přechodem do koncového stavu a vyprázdněním zásobníku, značen jako L(M)fε, je definován: $L(M)f\varepsilon = \{w: w \in \Sigma^*, Ssw \vdash^* zf, z = \varepsilon, f \in F\}$

Deterministický TS: **(Údajně se nezkouší)**
Nechť $M = (Q, \Sigma, \Gamma, R, s, F)$ je TS. M je deterministický TS, pokud pro každé pravidlo tvaru $pa \to qbt \in R$ platí, že množina $R - \{pa \to qbt\}$ neobsahuje žádné pravidlo s levou stranou pa.

Obecná gramatika:
je čtveřice $G = (N, T, P, S)$, kde

- N je abeceda neterminálů
- T je abeceda terminálů, při čemž $N \cap T = \emptyset$
- P je konečná množina pravidel tvaru $x \to y$, kde $x \in (N \cup T)^*N(N \cup T)^*$, $y \in (N \cup T)^*$
- $S \in N$ je počáteční neterminál

Kontextová gramatika:
Nech ť $G = (N, T, P, S)$ je obecná gramatika. G je kontextová gramatika (KG), pokud každé pravidlo $x \to y \in P$ splňuje podmínku: $|x| \leq |y|$.

Kontextové jazyky:
Nechť L je jazyk. L je kontextový jazyk, pokud existuje lineárně ohraničený automat M takový, pro který platí: $L = L(M)$.

Gramatiky pro regulární jazyky:
Jsou pravé lineární gramatiky

Konfigurace:
Nechť $M = (Q, \Sigma, R, s, F)$ je KA. Konfigurace KA M je řetězec $\chi \in Q\Sigma^*$

Konfigurace ZA:
Nech ť $M = (Q, \Sigma, \Gamma, R, s, S, F)$ je ZA. Konfigurace ZA M je řet ězec $\chi \in \Gamma^* Q \Sigma^*$

Derivační krok u BKG:
Nechť $G = (N, T, P, S)$ je BKG. Nechť $u, v \in (N \cup T)^*$ a $p = A \to x \in P$. Potom, uAv přímo derivuje uxv za použití p v G, zapsáno $uAv \Rightarrow uxv$ [p] nebo zjednodušeně $uAv \Rightarrow uxv$.

Přijímaný jazyk:
Nechť $M = (Q, \Sigma, R, s, F)$ je KA. Jazyk přijímaný konečným automatem M, L(M), je definován:
$L(M) = \{w: w \in \Sigma^*, sw \vdash^* f, f \in F\}$

Nejlevější derivace:
Nechť $G = (N, T, P, S)$ je BKG, nechť $u \in T^*$, $v \in (N \cup T)^*$, $p = A \to x \in P$ je pravidlo. Pak uAv přímo derivuje uxv za pomocí nejlevější derivace užitím pravidla p v G, zapsáno jako: $uAv \Rightarrow_{lm} uxv$ [p]

Nejpravější derivace:
Nechť G = (N, T, P, S) je BKG, nechť u ∈ (N ∪ T)*, v ∈ T*, p = A → x ∈ P je pravidlo. Pak uAv přímo derivuje uxv za pomocí nejpravější derivace užitím pravidla p v G, zapsáno jako: uAv ⇒rm uxv [p]

Deterministický ZA:
Nech ť M = (Q, Σ, Γ, R, s, S, F) je ZA. M je deterministický ZA, pokud pro každé pravidlo tvaru Apa → wq ∈ R platí, že množina R – {Apa → wq} neobsahuje žádné pravidlo s levou stranou Apa nebo Ap.

Rozšířený ZA:
Rozšířený zásobníkový automat (RZA) je sedmice M = (Q, Σ, Γ, R, s, S, F), kde Q, Σ, Γ, s, S, F jsou definovány stejně jako u ZA a R je konečná množina pravidel tvaru: vpa → wq, kde v, w ∈ Γ*, p, q ∈ Q, a ∈ Σ ∪ {ε}

LL-Gramatika bez ε-pravidel:
Nechť G = (N, T, P, S) je BKG bez ε-pravidel. G je LL gramatika, pokud pro každé a ∈ T a A ∈ N existuje maximálně jedno pravidlo A → X1 X2... Xn ∈ P takové, že: a ∈ First (X1 X2... Xn)

LL-Gramatika s ε-pravidly:
Nech ť G = (N, T, P, S) je BKG. G je LL-gramatika, pokud pro každé a ∈ T a každé A ∈ N existuje maximálně jedno A-pravidlo tvaru A → X1 X2... Xn ∈ P a platí: a ∈ Predict (A → X1 X2... Xn)

Boolova algebra jazyků:
Nechť je třída jazyků uzavřena vůči sjednocení, průniku a doplňku. Potom tato třída tvoří Boolovu algebru jazyků.

Rozlišné stavy:
Nechť M = (Q, Σ, R, s, F) je DSKA a nechť p, q ∈ Q, p ≠ q. Stavy p a q jsou rozlišitelné pokud existuje řetězec w ∈ Σ* takový, že: pw |–* p' and qw |–* q', kde p', q' ∈ Q a ((p' ∈ F a q' ∉ F) nebo (p' ∉ F a q' ∈ F)). Jinak stavy p a q jsou nerozlišitelné.

Dostupné symboly:
Nechť G = (N, T, P, S) je BKG. Symbol X ∈ N ∪ T je dostupný, pokud existuje u, v ∈ (N ∪ T)*, takové, že: S ⇒* uXv. Jinak je X nedostupný.

Jazyk přijímaný TS: (Údajně se nezkouší)
Nech ť M = (Q , Σ , Γ , R , s , F) je TS. Jazyk přijímaný TS M, L (M), je definován: L (M) = { w: w ∈ Σ * , s w |– * xfy; x, y ∈ Γ * , f ∈ F} ∪ { ε: s Δ |– * xfy; x, y ∈ Γ * , f ∈ F }

Ukončující symboly:
Nechť G = (N, T, P, S) je BKG. Symbol $X \in N \cup T$ je ukončující, pokud existuje řetězec $w \in T^*$, pro který platí: $X \Rightarrow^* w$. Jinak je X neukončující.

Jazyk L přijme sám sebe:
LPřijmeSámSebe = {δ(M): M je DTS, δ(M) ∈ L(M)}

Jazyk L nepřijme sám sebe:
LNepřijmeSámSebe = {0, 1}* – LPřijmeSámSebe

Rozhodnutelný jazyk:
Nechť L je jazyk. Pokud existuje DTS M, který vždy zastaví a pro který platí L = L (M), potom L je rozhodnutelný jazyk.

Rekurzivně spočetný jazyk:
Nechť L je jazyk. L je rekurzivně spočetný jazyk, pokud existuje Turingův stroj M takový, pro který platí: L = L(M).

Abeceda:
Konečná, neprázdná množina elementů, které nazýváme symboly.

Řetězec:
Nechť Σ je abeceda.
1) ε je řetězec nad abecedou Σ
2) pokud x je řetězec nad Σ a $a \in \Sigma$, potom xa je řetězec nad abecedou Σ

Konkatenace řetězců:
Nechť x a y jsou dva řetězce nad abecedou Σ. Konkatenace x a y je řetězec xy.

Reverzace řetězce:
Nechť x je řetězec nad abecedou Σ.
Reverzace řetězce x, reversal(x), je definována:
1) pokud x = ε pak reversal(ε) = ε 2) pokud $x = a_1 \ldots a_n$ pak $\text{reversal}(a_1 \ldots a_n) = a_n \ldots a_1$ pro $n \geq 1$ a $a_i \in \Sigma$ pro všechna $i = 1, \ldots, n$

Prefix řetězce:
Nechť x a y jsou dva řetězce nad abecedou Σ; x je prefixem y, pokud existuje řetězec z nad abecedou Σ, přičemž platí xz = y.

Sufix řetězce:
Nechť x a y jsou dva řetězce nad abecedou Σ; x je sufixem y, pokud existuje řetězec z nad abecedou Σ, přičemž platí zx = y.

Podřetězec:
Nechť x a y jsou dva řetězce nad abecedou Σ. x je podřetězec y, pokud existují řetězce z, z' nad abecedou Σ přičemž platí zxz' = y.

Jazyk:
Nechť Σ* značí množinu všech řetězců nad Σ. Každá podmnožina L ⊆ Σ* je jazyk nad Σ
Konečný a nekonečný jazyk:
Jazyk L je konečný, pokud L obsahuje konečný počet řetězců, jinak je nekonečný.

Sjednocení jazyků:
Nechť L1 a L2 jsou dva jazyky nad Σ. Sjednocení jazyků L a L , L ∪ L , je definováno:
L1 ∪ L2 = {x: x ∈ L1 nebo x ∈ L2}

Průnik jazyků:
Nechť L1 a L2 jsou dva jazyky nad Σ. Průnik jazyků L a L , L ∩ L , je definován:
L1 ∩ L2 = {x: x ∈ L1 a x ∈ L2}

Rozdíl jazyků:
Nechť L1 a L2 jsou dva jazyky nad Σ. Rozdíl jazyků L1 a L2, L1 – L2, je definován:
L1 – L2 = {x: x ∈ L1 a x ∉ L2}

Doplněk jazyka:
Nechť L je jazyk nad abecedou Σ. Doplněk jazyka L, L, je definován:
L = Σ* – L

Konkatenace jazyků:
Nechť L1 a L2 jsou dva jazyky nad Σ. Konkatenace jazyků L a L , L L , je definována jako:
L1L2 = {xy: x ∈ L1 a y ∈ L2}

Reverzace jazyka:
Nechť L je jazyk nad abecedou Σ. Reverzace jazyka L, reverse(L), je definována:
reverse(L) = {reverse(x): x ∈ L}

Mocnina jazyka:
Nechť L je jazyk nad abecedou Σ.
Pro i ≥ 0, i-tá mocnina jazyka L, Li, je definována:
1) L0 = {ε}
2) pro i ≥ 1: Li = LL^{i-1}

Iterace jazyka:
Nechť L je jazyk nad abecedou Σ.
Iterace jazyka L, L*, a pozitivní iterace jazyka L, L+, jsou definovány:

$$L^* = \bigcup_{i=0}^{\infty} L^i, \qquad L^+ = \bigcup_{i=1}^{\infty} L^i$$

Přechod:
Nechť pax a qx jsou dvě konfigurace KA M, kde p, q ∈ Q, a ∈ Σ ∪ {ε} a x ∈ Σ*. Nechť r = pa → q ∈ R je pravidlo. Potom M může provést přechod z pax do qx za použití r, zapsáno pax |– qx [r] nebo zjednodušeně pax |– qx

Sekvence přechodů:
Nechť χ je konfigurace. M provede nula přechodů z χ do χ; zapisujeme:
χ |– 0 χ [ε] nebo zjednodušeně χ |– 0 χ
Nechť χ0, χ1, ..., χn je sekvence přechodů konfigurací pro n ≥ 1 a χi-1 |– χi [ri], ri ∈ R pro všechna i = 1, ..., n, což znamená: i ∈ R pro všechna i = 1, ..., n, což znamená: χ0 |– χ1 [r1] |– χ2 [r2] … |– χn [rn] Pak M provede n-přechodů z χ0 do χn; zapisujeme: χ0 |– n χn [r1... rn] nebo zjednodušeně χ0 |– n χn
Pokud χ0 |–n χn [ρ] pro nějaké n ≥ 1, pak
χ0 |–+ χn [ρ].
Pokud χ0 |–n χn [ρ] pro nějaké n ≥ 0,
pak χ0 |–* χn [ρ].

ε-uzávěr:
Pro každý stav p ∈ Q je definován ε-uzávěr(p): ε-uzávěr(p) = {q: q ∈ Q, p |–* q}

Sekvence derivačních kroků:
Nechť u ∈ (N ∪ T)*. G provede nula derivačních kroků z u do u; zapisujeme: u ⇒0 u [ε] nebo zjednodušeně u ⇒0 u
Nechť u0,…,un ∈ (N ∪ T)*, n ≥ 1 a ui-1 ⇒ ui [pi], pi ∈ P pro všechna i = 1,…, n, což znamená: u0 ⇒ u1 [p1]⇒ u2 [p2] … ⇒ u_n [pn] Pak, G provede n derivačních kroků z u_0 do u_n; zapisujeme: u0 $\Rightarrow^n u_n$ [p1... pn] nebo zjednodušeně $u_0 \Rightarrow^n u_n$

Přechod u ZA:
Nechť xApay a xwqy jsou dvě konfigurace ZA M, kde x, w ∈ Γ*, A ∈ Γ, p, q ∈ Q, a ∈ Σ ∪ {ε} a y ∈ Σ*. Nechť r = Apa → wq ∈ R je pravidlo. Potom M může provést přechod z xApay do xwqy za použití r, zapsáno xApay |– xwqy [r] nebo zjednodušeně xApay |– xwqy.

Přechod u RZA:
Nechť xvpay a xwqy jsou dvě konfigurace RZA M, kde x, v, w ∈ Γ*, p, q ∈ Q, a ∈ Σ ∪ {ε} a y ∈ Σ*. Nechť r = vpa → wq ∈ R je pravidlo. Potom M může provést přechod z xvpay do xwqy za použití r, zapsáno: xvpay |– xwqy [r] nebo xvpay |– xwqy

Množina first:
Nechť $G = (N, T, P, S)$ je BKG. Pro každé $x \in (N \cup T)^*$ je definováno First(x) jako: First(x) = $\{a: a \in T, x \Rightarrow^* ay; y \in (N \cup T)^*\}$.

Množina empty:
Nechť $G = (N, T, P, S)$ je BKG. Empty(x) = $\{\varepsilon\}$ if $x \Rightarrow^* \varepsilon$; jinak Empty(x) = $\emptyset$, kde $x \in (N \cup T)^*$.

Množina follow:
Nechť $G = (N, T, P, S)$ je BKG. Pro všechna $A \in N$ definujeme množinu Follow(A): Follow(A) = $\{a: a \in T, S \Rightarrow^* xAay, x, y \in (N \cup T)^*\} \cup \{\$: S \Rightarrow^* xA, x \in (N \cup T)^*\}$

Množina predict:
Nechť $G = (N, T, P, S)$ je BKG. Pro každé $A \to x \in P$ definujeme množinu Predict($A \to x$) jako:
- pokud Empty(x) = $\{\varepsilon\}$ potom: Predict($A \to x$) = First(x) $\cup$ Follow(A)
- jinak pokud Empty(x) = $\emptyset$ potom: Predict($A \to x$) = First(x)

Konfigurace TS: **(Údajně se nezkouší)**
Nech ť $M = (Q, \Sigma, \Gamma, R, s, F)$ je TS. Konfigurace TS M je řet ězec $\chi = xpy$, kde $x \in \Gamma^*$, $p \in Q$, $y \in \Gamma^*(\Gamma - \{\Delta\}) \cup \{\Delta\}$.

Stacionární přechod (TS): **(Údajně se nezkouší)**
Nechť χ, χ' jsou dvě konfigurace TS M. Potom M může provést stacionární přechod z χ do χ' použitím r, zapsáno $\chi \vdash_S \chi'$ [r] nebo zjednodušeně $\chi \vdash_S \chi'$ pokud: $\chi = xpay$, $\chi' = xqby$ a $r: pa \to qbS \in R$

Pravý přechod (TS): **(Údajně se nezkouší)**
Nechť χ, χ' jsou dvě konfigurace TS M. Potom M může provést pravý přechod z χ do χ' použitím r, zapsáno $\chi \vdash_R \chi'$ [r] nebo zjednodušeně $\chi \vdash_R \chi'$, pokud: $\chi = xpay$, $r: pa \to qbR \in R$ a současně:

(1) $\chi' = xbqy$, $y \neq \varepsilon$ nebo
(2) $\chi' = xbq\Delta$, $y = \varepsilon$

Levý přechod (TS): **(Údajně se nezkouší)**
Nechť χ, χ' jsou dvě konfigurace TS M. Potom M může provést levý přechod z χ do χ' použitím r, zapsáno $\chi \vdash_L \chi'$ [r] nebo zjednodušeně $\chi \vdash_L \chi'$ pokud:

(1) $\chi = xcpay$, $\chi' = xqcby$, $y \neq \varepsilon$ or $b \neq \Delta$, $r: pa \to qbL \in R$ nebo
(2) $\chi = xcpa$, $\chi' = xqc$, $r: pa \to q\Delta L \in R$

Přechod (TS): **(Údajně se nezkouší)**
Nechť χ, χ' jsou dvě konfigurace TS M. Potom M může provést přechod z χ do χ' použitím r, zapsáno $\chi \vdash \chi'$ [r] nebo zjednodušeně $\chi \vdash \chi'$ pokud: $\chi \vdash_X \chi'$ [r] pro nějaké $X \in \{S, R, L\}$.

TS jako výpočetní model: **(Údajně se nezkouší)**
Nechť $M = (Q, \Sigma, \Gamma, R, s, F)$ je TS. TS M vyčísluje n-ární funkci φ následujícím způsobem: $s\Delta x1\Delta x2\ldots\Delta xn \vdash^* f\Delta y$, kde $f \in F$ právě tehdy, když $\varphi(x1, x2, \ldots, xn) = y$.

Derivační krok (TS): **(Údajně se nezkouší)**
Nechť $G = (N, T, P, S)$ je NG. Nechť $u, v \in (N \cup T)^*$ a $p: x \to y \in P$. Potom, uxv přímo derivuje uyv za použití p v G, zapsáno $uxv \Rightarrow uyv\ [p]$ nebo zjednodušeně $uxv \Rightarrow uyv$.

Lineárně ohraničené automaty:
(LOA) je TS, který nemůže žádným pravidlem prodloužit pásku.

Pravé lineární gramatiky:
Nechť $G = (N, T, P, S)$ je BKG. G je pravá lineární gramatika (PLG), pokud každé pravidlo $A \to x \in P$ splňuje: $x \in T^* \cup T^*N$.