UNIVERZITA KOMENSKÉHO V BRATISLAVE
FAKULTA MATEMATIKY, FYZIKY A
INFORMATIKY

Dospelé jazyky k Lindenmayerovým systémom

Rok predloženia: 2012

Jozef Fekiač

UNIVERZITA KOMENSKÉHO V BRATISLAVE
FAKULTA MATEMATIKY, FYZIKY A
INFORMATIKY

Dospelé jazyky k Lindenmayerovým systémom

Bakalárska práca

Študijný program: Informatika
Študijný odbor: 9.2.1 Informatika
Číslo študijného odboru: 2508
Školiace pracovisko: Katedra Informatiky FMFI
Školitel: RNDr. Mária Pastorová

Bratislava, 2011

Jozef Fekiač

64598018

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Jozef Fekiač
Študijný program: informatika (Jednoodborové štúdium, bakalársky I. st., denná forma)
Študijný odbor: 9.2.1. informatika
Typ záverečnej práce: bakalárska
Jazyk záverečnej práce: slovenský

Názov: Dospelé jazyky k Lindenmayerovým systémom

Cieľ: Skúmať niektoré typy Lindenmayerových systémov bez interakcie, i s interakciou, vzhľadom na dospelosť. Porovnanie vybraných tried jazykov.

Vedúci: RNDr. Mária Pastorová
Katedra: FMFI.KI - Katedra informatiky

Dátum zadania: 24.10.2011

Dátum schválenia: 25.10.2011

doc. RNDr. Daniel Olejár, PhD.
garant študijného programu

.....
študent

.....
vedúci práce

Abstrakt

Práca sa zaoberá jazykmi generovanými L-systémami 0L, dospelými jazykmi triedy A0L, jazykmi generovanými L-systémami s viacero tabuľkami pravidiel na odvodzovanie T0L a dospelými jazyky triedy AT0L.
Ďalej jazykmi generovanými systémami s interakciou IL, triedou dospelých jazykov AIL a vzťahmi medzi triedami jazykov generovanými L-systémami a jazykmi Chomského hierarchie.

Abstrakt

The thesis deals with languages generated by L-systems 0L, adult languages of the A0L set, languages generated by L-systems with more tables of inferencing rules T0L and adult languages of AT0L set. Furthermore with languages generated by systems with interactions IL, set of adult languages AIL and relationships between languages generated by L-systems and languages of Chomsky hierarchy.

Čestné prehlásenie

Čestne prehlasujem, že som túto prácu vypracoval sám s použitím uvedených zdrojov.

Poďakovanie
Ďakujem RNDr. Márii Pastorovej za poskytnutie potrebnej literatúry, materiálov a rád, rovnako za morálnu podporu pri písaní práce.

Obsah

Úvod

Aristid Lindenmayer (17. November 1925 - 30 Október 1989), maďarský biológ, ktorý v roku 1968 vytvoril typ formálnych jazykov ktoré teraz nazývame L-systémy, alebo lindenmayerove systémy.
S pomocou nich sa dá modelovať správanie bunkových systémov rastlín.
Lindenmayer sa zaoberal najmä kvasinkami, vláknitými hubami, a skúmal rôzne druhy rias.
Typický zaujímavý príklad z jeho skúmania pre demonštráciu popisnej sily najjednoduchších L-systémov (túto triedu konkrétne označujeme 0L, Lindenmayerove systémy bez interakcie - presná definícia bude uvedená neskôr) bola modrozelená riasa *Anabaena Catenula.*
Táto riasa je význačná tým, že počet jej buniek v jednotlivých okamihoch po delení tvorí prvky fibonacciho postupnosti.
Iné možnosti poskytované L-systémami sú napr. kreslenie fraktálov.

Svojou podstatou je každý systém spätý s množinou slov - jazykom, podobne ako sú so svojim jazykom späté formálne gramatiky. Rozdiel je ale v odvodzovaní, ktoré prebieha paralelne.
V prírode sa veľmi často vyskytujú organizmy, ktoré sa zrodia v počiatočnej forme, postupne sa rozvíjajú, až nakoniec dospejú a ich bunky len regenerujú. Toto motivovalo vznik teórie okolo dospelých jazykov, nakoľko Lindenmayerových systémoch sa vyskytuje dospelé správanie slov tomuto podobné.

Práca má tri kapitoly.
V prvej kapitole sa zaoberáme najjednoduchšími 0L systémami s jednou tabuľkou pravidiel, ich dospelými jazykmi a rozšíreniami.
V prvej kapitole sa zaoberáme o niečo zložitejšími T0L systémami kde je tabuľiek pravidiel viac, ich dospelými jazykmi a rozšíreniami.
V tretej kapitole skúmame L-systémy s interakciou a k nim dospelé jazyky.

1 Dospelé jazyky k 0L systémom

0L označuje Lindenmayerove systémy bez interakcie, t.j. pravidlá na ľavej strane majú iba jeden symboj, teda sa prepisujú bez ohľadu na kontext. V tejto kapitole budeme skúmať aj rozšírenia 0L systémov o ďalšie tabuľky pravidiel (T0L systémy), vyčlenenie terminálov v abecede (E0L systémy, ET0L systémy)

1.1 Základné pojmy a tvrdenia

Aby sme mohli skúmať vlastnosti 0L systémov, potrebujeme mať presne definované ich základné vlastnosti. Ako každé systémy generujúce formálne jazyky, aj 0L systémy obsahujú pravidlá na prepis symbolov, množinu symbolov a úvodné slovo.

Definícia 1.1.1. ***0L systém*** *je definovaný ako trojica $\mathcal{S}$=(V,P,x), kde*
V≠{}, V je konečná množina symbolov(nazývame ju aj abeceda),
P je množina pravidiel, $P \subseteq V \times V^*$, ktorá je tiež konečná, zároveň musí byť úplná (t.j. $\forall a \in V \exists (a, \alpha) \in P$)
Pozn.: Pre jednoduchšie porozumenie budeme túto dvojicu (a, α) zapisovať v tvare $a \to \alpha$). Aby sme ušetrili miesto, môžeme zapisovať pravidlá, ktoré majú ľavú stranu rovnakú, napr $a \to \alpha, a \to \beta$ skrátene vo forme $a \to \alpha \mid \beta$.
Množinu P nazývame občas aj tabuľka pravidiel.
V prípade že tabuľka pravidiel obsahuje vždy práve jedno pravidlo pre každý symbol z abecedy V, ide o podskupinu 0L systémov, nazývanú D0L(deterministické 0L).

$x \in V^+$ je axióma, počiatočné slovo.

Definícia 1.1.2. ***Krok odvodenia*** *v 0L systéme $\mathcal{S}$=(V,P,x) je definovaný nasledovne:*
Nech $v = a_1a_2a_3...a_k \in V^*, a_i \in V, i = 1...k \in \mathbb{N}$
$w \in V^*$
potom $v \implies w$ (v systéme $\mathcal{S}$) vtedy a len vtedy,
ak $\exists w = \alpha_1\alpha_2\alpha_3...\alpha_k, k \in \mathbb{N}, a_i \to \alpha_i (i = 1...k)$
Viac krokov znázorňujeme $v \overset{k}{\implies} w$ pre daný počet krokov k resp. $v \overset{+}{\implies} w$ pre ľubovoľný kladný počet krokov.

Definícia 1.1.3. ***Jazyk generovaný systémom $\mathcal{S}$=(V,P,x)*** *je množina $L(\mathcal{S})' = \{v \in V^* \mid x \overset{*}{\implies} w\}$ kde $\overset{*}{\implies}$ je reflexívno-tranzitívny uzáver.*

Najjednoduchší príklad na 0L systém sú bunky, ktoré zdvojnásobia svoj počet každým krokom odvodenia, pričom máme iba jeden druh buniek.

Príklad 1.1.1. *Definujme Systém* $\mathcal{S}_1$*=(V,P,x)*
kde V*=*$\{a\}$
P*=*$\{a \to aa\}$
x*=*a

Uveďme niekoľko krokov odvodenia:

x*=*$a \implies aa \implies aaaa \implies aaaaaaaa \implies ...$

Je zjavné, že takýto systém generuje jazyk $L_1 = \{a^{2^n} | n \geq 0\}$

Nech $\mathcal{S}_2$*=(V',P',x'),* V'*=*V*,* x'*=*x*. Uvažujme, že chceme mať jazyk* $L_2 = L(\mathcal{S}_2)$*, ktorý obsahuje okrem slov z* L_1 *aj prázdne slovo (toto budeme označovať obvyklým značením:* ϵ *).*
Keďže axióma x musí byť neprázdne slovo, musí existovať pravidlo ktoré vymazáva symboly. Naša abeceda obsahuje len jeden symbol, teda S' obsahuje pravidlo $a \to \epsilon$*(označme ho (2)).*
Pre generovanie jazyka L_1 *potrebujeme pravidlo* $a \to aa$*(označme ho (1)).*
Pokiaľ budeme mať obidve pravidlá v P', vzniknú v $\mathcal{S}'$*aj slová mimo* L_2*.*
Uveďme jedno také odvodenie :
$a \implies aa \implies aaaa \implies aaaaaa\epsilon = aaaaaa = a^6$

V prvom kroku odvodenia aplikovalo pravidlo (1), v druhom kroku sa aplikovalo pravidlo (1) na obidva symboly, v treťom kroku sa aplikovalo pravidlo (1) na prvé 3 symboly, na štvrtý sme použili pravidlo (2). Slovo a^6 *jednoznačne nie je v* L_2*, teda* $L(\mathcal{S}') \neq L_2$*. Obe spomenuté pravidlá bezpodmienečne potrebujeme na vygenerovanie slov z* L_2*, ale okrem nich sa generujem* $\mathcal{S}'$ *aj iné slová teda* L_2 *nepatrí do triedu jazykov 0L.*
Ďalší prípad jazyka ktorý sa nedá vygenerovať 0L systémom je $L_3 = \{a, a^2\}$
Voľba axiómy je na nás. Vezmime nový systém $\mathcal{S}_3 = (V_3, P_3, x_3)$
1) x_3 *=*a
V tomto prípade bude tabuľka pravidiel obsahovať len pravidlo (1). Ale nič nevylučuje odvodzovanie ďalších slov zo slova a^2 *odvodením z axiómy, čiže voľba axiómy bola pravdepodobne nesprávna.*
2) x_3 *=*aa
V tomto prípade potrebujeme pravidlo ktoré nám vymaže symbol a, čiže P bude pozostávať z pravidla (2), a pravidla $a \to a$*, aby sme mali možnosť aj zachovať symbol(zmazať potrebujeme len jeden). Existuje ale odvodenie, ktoré zmaže oba symboly, a vznikne nám prázdne slovo ktoré nie je v* L_3*.*
Teda ani L_3 *sa nedá vygenerovať 0L systémom.*

Ďalej preskúmajme systémy ktoré obsahujú viac ako jeden druh symbolov.

Príklad 1.1.2. *Nech $\mathcal{S}$=(V,P,x), V={a,b}, P={$a \to b \mid bb, b \to a$}, x=b*
Tu je je vidno, že tento systém generuje jazyk L={a^+} ∪ {b^+}

Príklad 1.1.3. *Nech $\mathcal{S}$=(V,P,x), V={a,b}, P={$a \to bb, b \to a$}, x=b*
Systém generuje jazyk L={$a^{2^n} \mid n \geq 0$} ∪ {$b^{2^n} \mid n \geq 0$}

Príklad 1.1.4. ***0L v praxi***
Definujme Systém $\mathcal{S}$=(V,P,x)
kde V={a, b}
x=b
P obsahuje nasledovné pravidlá:
$a \to ab$
$b \to a$
Na základe pravidiel vieme, že tento systém je skupiny D0L, čo je taká podskupina 0L, ktorá má deterministické pravidlá (prepis symbolov v jednom kroku odvodenia je jednoznačne daný. Táto skupina systémov má nižšiu popisnú schopnosť).
Vypíšme niekoľko krokov odvodenia.
$b \implies a \implies ab \implies aba \implies abaab \implies abaababa \implies ....$
Tu je zaujímavé si všimnúť nasledovnú vlastnosť systému:
dĺžka odvodeného slova po k krokoch odvodenia je rovná hodnote k-teho prvku Fibonacciho postupnosti.

1.2 Dospelý jazyk v 0L systémoch

Dospelé jazyky obsahujú len slová, ktoré sa ďalej nevyvíjajú, teda zo žiadneho slova dospelého jazyka neodvodíme nové slovo. Triedu dospelých jazykov označujeme A0L.

Definícia 1.2.1. ***Dospelý jazyk k 0L systému***
Nech $\mathcal{S}$ = (V,P,x) je 0L systém. Potom dospelým jazykom k $\mathcal{S}$ nazývame $L_A(\mathcal{S}) = \{v \in L(\mathcal{S}) \mid ak(v \implies w) tak(v = w)\}$
Slovo $v \in L_A(\mathcal{S})$ nazývame stabilné slovo.

Príklad 1.2.1.
Definujme Systém $\mathcal{S}$=(V,P,x)

kde V=$\{a, A\}$
x=A
P obsahuje nasledovné pravidlá:
$a \to a$
$A \to aA \mid a$

Je zjavné, že množina odvodených slov by narastala nasledovne:
$\{A\} \implies \{a, aA\} \implies \{a, aa, aaA\} \implies \{a, aa, aaa, aaaA\} \implies ...$
Keďže sa symbol a prepisuje len na samého seba, je zjavné, že slová obsahujúce iba symboly a sú stabilné.
To znamená, že dospelý jazyk je:
$L_A = \{a^+\}$
Tento istý jazyk vieme odvodiť v systéme 0L, ale tam by rast musel prebiehať rýchlejšie, vzhľadom na nedeterminizmus ktorý by sme museli použiť v podobe pravidiel
$a \to aa$,
$a \to a$
čo by znamenalo, že vývoj môže po k krokoch vyprodukovať až slovo dĺžky 2^k, *v dospelom jazyku je najdlhšie stabilné slovo po k krokoch dĺžky k.*

Príklad 1.2.2.
Sú ale jazyky ktoré vieme odvodiť ako dospelý jazyk triedy A0L, ale nevieme odvodiť ako jazyk generovaný 0L.

Ideálny príklad na toto je $L = \{a^n b^n | n > 0\}$

Definujme teda systém 0L ku ktorému bude tento jazyk dospelým jazykom.

$\mathcal{S}$ = *(V,P,x),* $V = \{A, a, b\}, x = A, P = \{a \to a, b \to b, A \to aAb, A \to ab\}$
V každom kroku odvodenia sa okolo symbolu A zjavia v rovnakom počte symboly a, b, popritom A môže ale nemusí zaniknúť.
Keďže a, b sú nemenné počas odvádzania, slová obsahujúce výhradne a,b patria do dospelého jazyka.
$L_A(\mathcal{S}) = L$

Príklad 1.2.3. *Ďalší zaujímavý príklad je jazyk*$L = \{a^k b^n | k > n > 0\}$. *Tento bude dospelým jazykom k L systému z predchádzajúceho príkladu, s tým rozdielom, že do množiny P pridáme pravidlo* $A \to aA$. *Tým zaručíme v každom okamihu možnosť pridať ďalšie samostatné znaky a do výsledného slova.*

Definícia 1.2.2. ***Dospelá abeceda***
Nech $\mathcal{S}$ = *(V,P,x) je 0L potom dospelou abecedou nazývame množinu*

$V_A(\mathcal{S}) = \{a|(a \in V) \wedge (\exists v \in L_A(S) : \#_a(v) > 0\}$
Označme abc(v) $= \{a \in V \mid \#_a(v) > 0)\}$ *potom* $V_A(S) = \bigcup_{v \in L_A} abc(v)$
Pre $a \in V$ *Budeme označovať* X_a, *ak platí* $a \overset{+}{\Longrightarrow} X_a \Rrightarrow X_a$

Definícia 1.2.3. *Relácia* $\Rrightarrow$ *je definovaná nasledovne:*
$X \Rrightarrow Y \overset{def.}{\Leftrightarrow} ((x \implies y) \wedge (x \implies y')) \implies (y' = y)$,
Relácia $\overset{n}{\Rrightarrow}$ *obdobne.*
[*Pozn.: Potom* $L_A(\mathcal{S}) = \{w \in L(\mathcal{S}) \mid w \Rrightarrow w\}$.
Pre $a \in V$ *budeme označovať dané slovo* X_a *ak platí* $a \overset{+}{\Longrightarrow} X_a \Rrightarrow X_a$]

Pozn.: Existuje efektívnejší algoritmus hľadania V_A:

$V_1 = \bigcup_{a \in V, a \overset{+}{\Longrightarrow} x_a \Rrightarrow x_a} abc(X_a)$
$V_2 = \{a \in V_1 \mid \exists v \in V^+ \cap L(\mathcal{S}) : \#_a v > 0\}$
$V_A = V_3 = \bigcup_{a \in V_2} a \overset{n}{\Rrightarrow} x_a$

Príklad 1.2.4. ***Aplikácia algoritmu, nájdenie dospelej abecedy k nejakému jazyku.***
Nech $\mathcal{S}$ *= (V,P,x) je 0L Systém, V=(a,b,c,A)*
$x = A$
$P = \{ A \to Aa \mid a \mid dB$
$a \to ab$
$b \to c$
$c \to \epsilon$
$B \to B^2$ *}*
$V_1 = \{a, b, c, d\}$, *pretože v konečnom dôsledku vedie odvodenie z každého z daných písmen vedie k nejakému stabilnému slovu.*

$V_2 = \{a, b, c\}$, *každé z týchto písmen je dosiahnuteľné z axiómy, vyskytuje sa v jazyku generovanom 0L*

$V_3 = \{a, b, c\}$, *pretože pri symbole A nemáme zaručené pre konečné n, že na n krokov sa určite v každom prípade dostaneme do stabilného slova. Táto množina písmen spĺňa aj pôvodnú definíciu dospelej abecedy.*

Výsledný jazyk: $L(S) = \{(abc)^n \mid n > 0\}$

Je vhodné si všimnúť, že aj symboly, ktoré sa prepisujú, ale sú členom regenerujúcej reťaze (v tomto prípade abc), patria do dospelej abecedy.

Lema 1.2.1. *Nech $\mathcal{S} = (V, P, x)$ je 0L. Nech $a \in V_A, a \to \alpha \in P$. Potom platia nasledovné tvrdenia:*
1)$(\#_a\alpha \leq 1) \wedge (a \to \beta \in P) \implies (\beta = \alpha)$
2)$(\#_a\alpha = 0) \implies X_a = \epsilon$
3)Nech |V|=m, potom $\underset{a\in V_a}{\forall} \underset{n\leq m}{\exists} a \overset{n}{\Longrightarrow} X_a \Rrightarrow X_a$
Dôkaz:
1) SPOROM. Nech $w \in L_A$ pričom $\#_a w \geq 1$ (toto môžeme predpokladať, lebo $a \in V_A$, a pravidlo $a \to \beta$ pričom $\alpha \neq \beta$. Keďže je slovo w z dospelého jazyka, musí podľa definície platiť pre ľubovoľné odvodenie v $\mathcal{S}$, že w ostane nemenné. Vezmime teda odvodenie, ktoré obsahuje pravidlo $a \to \alpha$:
$w = w_1 a w_2 \implies w_1'\alpha w_2' = w$, *pričom* $w_1 \implies w_1', w_2 \implies w_2'$.
Ak v tomto odvodení nahradíme práve tento prepis podľa pravidla $a \to \alpha$ pravidlom $a \to \beta$,aby vyzeral nasledovne(odvodenia na časti w_1 a w_2 ostanú rovnaké):
$w = w_1 a w_2 \implies w_1'\beta w_2'$.

Ak $\alpha \neq \beta$ tak $w_1'\alpha w_2' \neq w_1'\beta w_2'$ čo je spor s predpokladom $w \in L_A$.

2)Sporom: Nech $(\#_a\alpha = 0) \wedge X_a \neq \epsilon$
Nech existuje $w \in L_A, \#_a(w) > 0$. Z definície platí, že ľubovoľným odvodením dostaneme to isté slovo: $w \implies w$. Vezmime pevný symbol
$b \in V_A, \#_b(w) > 0$ *musí existovať odvodenie* $b \overset{+}{\Longrightarrow} w', \#_a(w') > 0$, *lebo inak by sa hodnota $\#_a(w)$ menila, čo je v slove dospelého jazyka neprípustné.*

Postup I: Po istom konečnom počte krokov odvodenia vznikne zo symbolu b slovo zaručene obsahujúce symbol a, z ktorého následne konečným počtom odvodení odvodíme X_a. Z definície X_a vieme, že nezaniká.

Pretože platí aj $w \overset{+}{\Longrightarrow} w$, *určite po tomto konečnom počte odvodení v postupe I bude stále prítomný symbol b. Keďže týmto postupom vieme vyrobiť symbol a, počet symbolov z X_a bude vo w narastať, čo je spor s $w \in L_A$.*

3) Na základe bodu 1) tejto lemy vieme, že pre každý symbol dospelej abecedy existuje práve jedno pravidlo z P kde je na ľavej strane. Z definície dospelej abecedy a stabilných slov je zjavné, že symboly z α sú rovnako z dospelej abecedy. Pre každý symbol $p \in X_a$ vieme presne určiť poradie pravidiel pomocou ktorých bol odvodený. Nakoľko môže byť symbolov dospelej abecedy najviac m, musí byť aj týchto pravidiel ktoré viedli k odvodeniu p najviac m, čím sme

dokázali tvrdenie.

Veta 1. *Pre každý 0L systém $\mathcal{S}$ existuje ekvivalentný 0L, že $L_A(\mathcal{S}) = L_A(\mathcal{S}')$ a pre všetky symboly $a \in V_A(\mathcal{S})$ existuje jediné pravidlo $a \to a$ v systéme $\mathcal{S}'$*

Dôkaz: Konštrukciou.
Zaveďme nasledovný homomorfizmus:
$h(a)=\{$
$X_a \mid a \in V_A,$
$a \mid a \notin V_A \}$
S jeho pomocou vygenerujeme novú množinu pravidiel ktorá bude tvaru:
$P' = \{a \to a \mid a \in V_A\} \cup \{a \to h(\alpha) \mid a \notin V_A\}$
Dôležité je pomocou homomorfizmu vygenerovať novú axiómu.

Veta 2. ***Platí A0L = L(CF)***
$A0L \subseteq L(CF)$
Nech $G = (V_G, \Sigma, R, \sigma)$ je bezkontextová gramatika. Pre ľubovoľný systém 0L $\mathcal{S} = \{V, P, x\}$ vieme vytvoriť ekvivalentný systém pomocou Vety 1, ktorý bude mať dôležité vlastnosti pre konštrukciu bezkontextovej gramatiky.
Druhý krok bude identifikovať terminály a neterminály.
Terminály budú všetky symboly z dospelej abecedy. $V_G = V_A$
Neterminály budú ostatné symboly: $\Sigma = V - V_A$
Pravidlá v R budú identické s pravidlami v P, s výnimkou pravidiel z P, ktoré majú na ľavej strane symbol z V_A. Tie budú vynechané, následne ešte treba pridať pravidlo $\sigma \to x$ aby sme dostali obvyklý tvar gramatiky:
$R = \{a \to \alpha \mid a \notin V_A\} \cup \{\sigma \to x\}$

Príklad 1.2.5. ***Bezkontextová gramatika na základe A0L***
Budeme postupovať podla vety 2.
Nech $\mathcal{S} = (V, P, x), V = \{X, A, B, a, b, c, d\}$
Pre demonštráciu uvedieme viacero druhov pravidiel

$P = \{X \to A \mid B, A \to aAbAa \mid a, B \to aBc \mid b, a \to a, b \to bc, c \to d, d \to \epsilon\}$
$x = XaXb$
Pretransformujeme 0L systém podľa Vety 1. Zistíme si, ako vyzerá homomorfizmus h pre tento systém, musíme zistiť X_a pre znaky z dospelej abecedy.
Je zjavné, že a, b patrí tiež do dospelej abecedy, rovnako c a d. (Toto si môžeme overiť aj algoritmom na hľadanie dospelej abecedy).
$X_a = a \quad X_b = bcd \quad X_c = \epsilon \quad X_d = \epsilon$

$P = \{X \to A \mid B, A \to aAbcdAa \mid a, B \to aB \mid bcd, a \to a, b \to b, c \to c, d \to d\}$
$x = XaXbcd$
Konštrukcia gramatiky:
Musíme pridať pravidlo $\sigma \to x$ *kde za* x *dosadíme aktuálnu axiómu.*
$\sigma \to XaXbcd$
Vynecháme pravidlá pre písmená dospelej abecedy
Určíme terminály a neterminály .

Terminály: $\{a,b,c,d\}$*, táto množina je rovnaká ako dospelá abeceda.*
Neterminály: $\{\sigma,X,A,B\}$

Čiže bezkontextová gramatika byde vyzerať nasledovne:

$G=(V,\Sigma,R,\sigma)$ *kde* $V=\{\sigma,X,A,B\}$, $\Sigma=\{a,b,c,d\}$,

$R=\{ \sigma \to XaXbcdX \to A \mid B, A \to aAbcdAa \mid a, B \to aB \mid bcd, \}$

Veta 3. ***Platí*** $A0L - \mathcal{L}(0L) \neq \emptyset \ \wedge \mathcal{L}(0L) - A0L \neq \emptyset$

Nasledujúce dve triedy jazykov sú neporovnateľné. Ako dôkaz uvedieme príklad na jazyk triedy 0L ktorý nie je v triede A0L a naopak, čo nie je prekvapujúce, pretože tr. bezkontextových jazykov a tr. jazykov 0L sú neporovnateľné.

Nech $\mathcal{S} = (V, P, x)$ *je 0L. Nech* $L_1 = \{a^{2^n} \mid n \geq 0\}$. *Aby* $L_1 = L_A(S)$, *muselo by existovať nejaké pravidlo/pravidlá ktoré by vytvorili stabilné slovo z bežného nestabilného.*
$A \to a \in P$.
Ak by takéto pravidlo neexistovalo, nemáme ako vygenerovať najkratšie slovo v L_1. *Stabilné slová zložené iba zo symbolov a, ktoré sú ďalej nemenné. Môžeme uvažovať* $x=A$, *pretože pokiaľ by bola axióma dlhšia, muselo by existovať pravidlo ktoré má na pravej strane prázdne slovo, čo by zaručene viedlo výskytu prázdneho slova v jazyku.*
Na základe pravidla $A \to a$ *môžeme prehlásiť, že slová v jazyku generovanom 0L systéme, ktoré nie sú dospelé, musia mať jednoznačne tvar* $A^{2^n}, n \geq 0$, *lebo inak by určite v dospelom jazyku bolo aj slovo mimo* L_1.
Slová tohoto tvaru vytvoríme z axiómy odvodením pomocou pravidla $A \to AA$, *nech je teda v* P. *Iným by sme nedostali všetky slová spomenutého tvaru.*
Tu ale nastáva problém:
Nasledovné odvodenie v $\mathcal{S}$ *vygeneruje stabilné slovo, ktoré nie je v* L_1: $A \to AA \to AAa \to aaa$

Tým sme dokázali, že existuje jazyk z $\mathcal{L}(0L)$ ktorý nie je z $\mathcal{L}(A0L)$.
Príklad na jazyk z ktorý je z $\mathcal{L}(A0L)$ a zároveň nie je z $\mathcal{L}(0L)$ je $L = \{a^n b^n | n > 0\}$, ktorý je riešený v príklade 1.2.2
Ďalší zaujímavý príklad na jazyk z A0L ktorý nie je 0L je $L_2 = \{a, a^2\}$
Je generovaný systémom $\mathcal{S}_2 = (\{A, a\}, \{A \to a \mid aa, a \to a\}, a)$

1.3 E0L

Rozšírením 0L systémov o terminálne symboly (terminály) získame E0L systémy.

Definícia 1.3.1.
E0L systém $\mathcal{S}$ je definovaný ako štvorica (V, P, x, Σ) , kde platí, že $\mathcal{S}' = (V, P, x)$ je 0L systém, $\emptyset \neq \Sigma \subseteq V$ a $L(S) = L(S') \cap \Sigma^*$.

Na základe jednoduchej úvahy vieme dokázať nasledovnú vetu

Veta 4. *Platí $A0L \subsetneq \mathcal{L}(E0L)$*
Dôkaz:
Pre ľubovoľný systém 0L $\mathcal{S} = (V, P, x)$ vieme skonštruovať E0L systém $\mathcal{S}' = (V', P', x', \Sigma)$ nasledovným spôsobom tak, že bude platiť $L_A(\mathcal{S}) = L(\mathcal{S}')$:
Na základe jednej z predošlých viet môžeme predpokladať,že k systému $\mathcal{S}$ existuje $\mathcal{S}_\S = (V_x, P_x, x_x)$ ktorých dospelé jazyky A0L sú rovnaké, pričom platí $\underset{a \in L_A}{\forall} a \to \alpha \implies a = \alpha$ Potom E0L systém bude vyzerať nasledovne:
$V_A(\mathcal{S}) = \Sigma, P_x = P', x_x = x'$.
Aby sme ukázali, že E0L má vyššiu popisnú schopnosť ako A0L, uvedieme nasledovný príklad:
Nech $\mathcal{S}_1 = (V_1, P_1, x_1, \Sigma_1), V_1 = \{S, a, b\}$,
$P_1 = \{S \to a, S \to b, a \to aa, b \to bbb\}, x_1 = S, \Sigma_1 = \{a, b\}$
Je zjavné, že tento jazyk nepatrí do A0L, pretože ani a^{2^n} nevieme vygenerovať ako dospelý jazyk k 0L systému.

Veta 5. *$L \in \mathcal{L}(E0L)$* ***je z triedy bezkontextových jazykov ak k nemu existuje E0L systém $\mathcal{S}$ taký, že $L(\mathcal{S}) = L$ a zároveň pre terminály $a \in \Sigma$ existujú iba pravidlá $a \to a$*** *K E0L jazyku $\mathcal{S} = (V, P, x, \Sigma)$ jednoznačne existuje 0L $\mathcal{S}' = (V', P', x')$ taký, že dospelý jazyk k $\mathcal{S}'$, a platí $V{=}V \cup \{F\}, x = x'$.*
Aby $L_A(\mathcal{S}') = L(\mathcal{S})$, musíme zaručiť, aby $V_A(\mathcal{S}') = \Sigma$. To zabezpečíme pridaním pravidiel do P'.
P'$=P \cup \{a \to F \mid a \in V - \Sigma\} \cup F \to FF$, pričom F je nový symbol.
Týmito pravidlami zaručíme, že neterminálne symboly nebudú v dospelej

abecede. Sporom: Vezmime neterminálny symbol s, ktorý je v dospelej abecede L_A.
Teda existuje stabilné slovo ktoré tento symbol obsahuje. Určite ale existuje pravidlo pre s, ktoré vedie do symbolu F, ktorý sa neobmedzene množí. To znamená, že zo stabilného slova odvodíme nové slovo, pretože symbol F nie je v dospelej abecede. Toto je spor.

Veta 6. ***Ku každému E0L systému*** $\mathcal{S}$ ***existuje ekvivalentný synchronizovaný*** $\mathcal{S}'$***, kde*** $v \in L(\mathcal{S})$ ***sa odvodia synchronizovane (t.j.*** $\underset{a\in\Sigma}{\forall} a \overset{+}{\Longrightarrow} v, v \notin \Sigma^+$)
Dôkaz:
Zaveďme nasledovný homomorfizmus:
$h(a) = \{$
$\bar{a} \mid a \in \Sigma$
$a \mid a \in V - \Sigma$
$\}$
$\bar{\Sigma} = \{\bar{a} \mid a \in \Sigma\}$, *tieto symboly sa nevyskytujú vo* V
$\mathcal{S}' = (V', P', x', \Sigma)$
$V' = V \cup \bar{\Sigma} \cup \{F\}$, *kde* $F \notin V$
x'=h(x)
$P' = \{a \to F \mid a \in \Sigma \cup \{F\}\} \cup \{\bar{a} \to h(\alpha) \mid a \in V, a \to \alpha \in P\} \cup \{\bar{a} \to a \mid a \in \Sigma\}$ *Pre slová* $v \in L(\mathcal{S})$ *vždy v* $\mathcal{S}'$ *pred vznikom terminálneho slova existuje neterminálne, ktoré má symboly s pruhom. Pri nesynchronizovanom odvodení sa terminálny symbol, ktorý sa prepísal skôr, pri nasledovnom kroku odvodenia prepíše na zamietací symbol, ktorý je neterminálny. Tým zaručíme, že nesynchronizovaným odvodením nevzniknú slová jazyka* $L(\mathcal{S}')$.

Príklad 1.3.1. *Uvedieme príklad na jednoduchý systém E0L, ktorý bude v synchronizovanom tvare.*
Nech $L = \{a^n b^n c^n \mid n > 0\}$. *Tento jazyk vieme pomerne jednoducho vygenerovať E0L systémom, pokiaľ zavedieme správnu axiómu a odvodzovacie pravidlá.*
Nech $\mathcal{S} = \{V, p, x, \Sigma\}$,
$V = \{A, B, C, a, b, c, \bar{a}, \bar{b}, \bar{c}, F\}$
$x = ABC$,
$P = \{A \to A\bar{a} \mid a$
$B \to B\bar{b} \mid b$
$C \to C\bar{c} \mid c$
$\bar{a} \to \bar{a} \mid a$
$\bar{b} \to \bar{b} \mid b$
$\bar{c} \to \bar{c} \mid c$

$r \rightarrow F, r \in \{a, b, c, F\}\}\Sigma = \{a, b, c\}$ *Týmito pravidlami máme zaručené, že pokiaľ symboly A,B,C, ktoré sú zodpovedné za rast slova, sa premenia na terminálne, tak už to musia urobiť všetky naraz. V ďalšom kroku odvodenia všetky terminálne znaky sa zmenia na neterminálny zamietací symbol F. Z definície E0L do jazyka patria len slová zložené z terminálov, teda zaručene sa nám budú odvodzovať všetky slová synchronizovane.*

Lema 1.3.1. ***Platí*** $\mathcal{L}(E0L) - \mathcal{L}(T0L) = A_1 \neq \emptyset \ \wedge \mathcal{L}(T0L) - \mathcal{L}(E0L) = A_2 \neq \emptyset$
Príklad na jazyk z množiny $A_1 : L_1\{a, a^2\}$
Príklad na jazyk z množiny $A_2 : L_2\{a^{2^n 3^m}\}$
L_1 *nie je z* $\mathcal{L}(T0L)$*, pretože už musí existovať pravidlo na množenie symbolov a, čo by viedlo k neobmedzenému množeniu, a naopak, ak by sme na to išli z opačnej strany, pravidlo na zmazanie symbolov a by zmazalo v jednom okamihu všetky, a teda by vzniklo aj prázdne slovo, ktoré nie je z jazyka* L_1
L_2 *nie je z* $\mathcal{L}(E0L)$*, pretože by vyžadoval nezávislé použitie pravidiel* $a \rightarrow aa$, $a \rightarrow aaa$*, čo inak ako použitím viacerých tabuliek nevieme.*

2 Dospelé jazyky k T0L systémom

2.1 Základné pojmy a tvrdenia

T0L systémy sú 0L systémy, ktoré sú rozšírené o ďalšie tabuľky pravidiel.

Definícia 2.1.1. ***T0L Systém*** *je definovaný ako trojica* $\mathcal{S} = (V, \mathcal{P}, x)$
kde platí: $\mathcal{P}$ *je konečná množina tabuliek pravidel, pričom platí* $\forall_{P \in \mathcal{P}}(V, P, x)$ *je 0L systém*

Definícia 2.1.2. ***Odvodenie v T0L***
Vezmime S z definície 1.2.1, nech $v \in V^+, w \in V^*$, *tak odvodenie z v do w (označujeme* $v \overset{+}{\Longrightarrow} w$*)existuje,*
ak existujú $P_1, P_2, ..., P_n \in \mathcal{P}, n > 0$ *také, že* $w \underset{P_1}{\Longrightarrow} w_1 \underset{P_2}{\Longrightarrow} w_2 ... \underset{P_n}{\Longrightarrow} w$

Definícia 2.1.3. ***Jazyk generovaný T0L systémom***
Nech $\mathcal{S}$ *z definície 1.2.1, tak jazyk generovaný T0L systémom je nasledovná množina* $L(\mathcal{S}) = \{w \in V^* \mid x \overset{*}{\Longrightarrow} w\}$

Je zjavné, že popisná sila T0L systémov je vyššia, ako popisná sila 0L, nakoľko 0L systém je T0L systém s jednou tabuľkou pravidiel.

Príklad 2.1.1. ***Čo môžeme spraviť v T0L ale v 0L nie***
Chceli by sme mať jazyk generovaný T0L systémom, ktorý by sa skladal zo zjednotenia dvoch 0L systémov. Ako príklad vezmeme nasledovné dva jazyky generovateľné 0L systémami:

$L_1 = \{a^{2^k} \mid k \geq 0\}$

$L_2 = \{b^l \mid l \geq 0\}$

$L_3 = L_1 \cup L_2$ *Pri 0L systémoch máme k dispozícii iba jednu tabuľku pravidiel. Slová každého z jazykov* L_1 *aj* L_2 *obsahujú len jedno špecifické písmeno pre daný jazyk (Abeceda jazyka je jednopísmenová), zároveň sú tieto písmená rôzne. Ak by sme sa pokúsili jazyk* L_3 *vygenerovať 0L jazykom, narazili by sme na nasledovné problémy:*

1. aká by mala byť axióma?
2. ak sú slová zložené iba z jedného druhu písmen, ako sa dostaneme k druhému?
3. ako zariadime, že nám nevzniknú iné slová ako slová z jazyka L_3*?*

Odpovede na problémy:

1. Pokiaľ by sme modelovali iba L_1, musela by byť jednoznačne jednopísmenová. Ak by sme dali viacpísmenovú axiómu, nebolo by možné vygenerovať slová kratšie ako axióma, ktoré by s istotou patrili do L_1. Toto je z dôvodu, že jediná možnosť ako vymazávať znaky, je pravidlom $a \to \epsilon$, ktoré nevieme v 0L presne kontrolovať. Vyberme teda náhodné slovo z L_1 ako axiómu. Napríklad a^8. Aby sme zaručili, že vieme vygenerovať nasledujúce slovo v jazyku, musí existovať v systéme pravidlo $a \to aa$. Ale zároveň chceme získať aj kratšie slová, teda existuje aj skracujúce pravidlo. Iné ako $a \to \epsilon$ v 0L neexistuje. Dôsledkom použitia týchto dvoch pravidiel vieme nasledovne získať slovo mimo jazyka. Uvedieme príklad odvodenia takého slova z axiómy:
slovo $\boldsymbol{a\ a\ a\ a\ a\ a\ a\ a}$ *prepíšeme jedným z možných spôsobov podľa pravidiel ktorých existenciu v 0L systéme sme vydedukovali:*
$\boldsymbol{aa}\ \epsilon\ \epsilon\ \epsilon\ \epsilon\ \epsilon\ \epsilon\ \epsilon$
Takto sme dostali kratšie slovo z jazyka L_1
Ale vedľajší účinok je nasledovné odvodenie z axiómy:
$\boldsymbol{aa\ aa\ aa}\ \epsilon\ \epsilon\ \epsilon\ \epsilon\ \epsilon$
toto slovo zaručene nepatrí do jazyka L_1.

Pri systéme generujúcom L_2 tvar axiómy nehrá veľkú rolu. Jediná podmienka v tomto prípade je, aby bola slovom z L_2, nakoľko tento jazyk obsahuje prázdne slovo.
Z vlastností by sme vedeli nasledovnými pravidlami vygenerovať všetky kratšie slová jazyka L_2 ako axióma:
$b \to \epsilon \mid b$
Vysvetlenie: pri každom symbole teraz vybranej axiómy sa rohodne, či po kroku odvodenia symbol zanikne, alebo sa zachová. Z hľadiska počtu písmen v slove teda máme možnosť zmazať ľubovoľný počet písmen slova, čo nám zaručuje nedeterminizmus použitia pravidiel.
Zaručiť ale musíme aj rast slova, čiže pridáme pravidlo ktoré pridáva symboly. V tomto prípade je úplne nepodstatné, koľko ich pridáme, nakoľko máme pravidlo ktoré môže v ďalšom kroku odvodenia zachovať ich počet, prípadne znížiť o 1.
$b \to b^k, k > 1$ *pre ľubovoľné pevne zvolené k zaručí správny rast.*

2. Na túto otázku máme rôzne odpovede. Ale pravidlo na zmenu symbolu má zaiste v tomto prípade tvar $a \to b^k, k > 0$. Je to preto, že ak by sme uvažovali jazyk $L_1' = \{b^{2^k} \mid k \geq 0\}$, tak by sme si všimli nasledovné: $L_1 \subset L_2$. V inom prípade musíme vyriešit, či by sme prepisom nedostali slová mimo jazyka L_2. Ak by sme chceli prepisovať z jazyka L_2 do L_1, zistíme, že by sme museli platiť $\forall k \exists l : b^k \implies a^{2^l}$
3. Tu začína byť vidno rozdiel v popisnej sile 0L a T0L, najmä však nutnosť

použitia T0L systému na generovanie jazyka L_3
Ak by sme pravidlá popísané v bodoch 1 a 2 mali v 0L systéme, nevedeli by sme zaručiť vlastnosť, že sa nám všetky symboly a prepíšu naraz pravidlom $a \to b^k, k > 0$. Pokiaľ chceme zaručiť správne generovanie jazyka L_3, musíme zaviesť druhú tabuľku, ktorá bude zaručovať, že sa prepis stane naraz pre všetky symboly.

Výsledný T0L systém generujúci jazyk L_3:

$\mathcal{S} = (V, \mathcal{P}, x), \qquad V=\{a, b\}, \qquad \mathcal{P} = \{P_1, P_2\}$
$P_1 = \{a \to aa, b \to b\}$
$P_2 = \{a \to b, b \to bb \mid b \mid \epsilon\}$
$x=a$

Poznámka: Stačilo by zvoliť nesprávne podmienky na k a l, a už by nebolo možné urobiť takéto zjednotenie jazykov.

Veta 7. ***Platí*** $\mathcal{L}(0L) \subsetneq \mathcal{L}(T0L)$
Dôkaz:
Podľa definície je zjavné, že T0L systém s jednou tabuľkou je ekvivalentný 0L systému, teda $\mathcal{L}(0L) \subseteq \mathcal{L}(T0L)$ Ostáva vylúčiť rovnosť týchto dvoch skupín jazykov.
Na základe predchádzajúceho príkladu je zjavné, že existuje jazyk z T0L ktorý nie je z 0L.
Uvedieme ale ešte jeden, elegantnejší.
$\mathcal{S} = (V, \mathcal{P}, x), \qquad V=\{a\}, \qquad \mathcal{P} = \{P_1, P_2\}$
$P_1 = \{a \to aa\} \qquad P_2 = \{a \to aaa\}$
$x=a$

Výsledný jazyk v tomto prípade je

$L = \{a^{2^k 3^l} \mid k, l > 0\}$

Príklad 2.1.2.
Vezmime pevné m, $m \geq 1$ a nasledovné jazyky:

$L'_1 = \{a^{2^k} \mid k \geq m\},$
$L'_2 = \{b^l \mid l \geq 0\}$
$L'_3 = L_1 \cup L_2$
Na základe predchádzajúceho príkladu vieme určiť, že axióma môže byť najmenšie slovo z jazyka L'_1, pretože jediné nepríjemnosti nám robilo pravidlo

$a \to \epsilon$.

Pri zachovaní podmienky na premennú l v L_2 môžeme jednoducho vydedukovať, že systém z predchádzajúceho príkladu nie je nutné meniť, až na axiómu. Výsledný T0L systém generujúci jazyk L'_3:

$\mathcal{S} = (V, \mathcal{P}, x)$, V=$\{a, b\}$, $\mathcal{P} = \{P_1, P_2\}$
$P_1 = \{a \to aa, b \to b\}$
$P_2 = \{a \to b, b \to bb \mid b \mid \epsilon\}$
x=a^{2^m}

Príklad 2.1.3.
Vezmime pevné m,n,x prirodzené, platí $m \geq 1$ a nasledovné jazyky:

$L'_1 = \{a^{2^k} \mid k \geq m, m \geq 0\}$,
$L'_2 = \{b^l \mid l \geq n\}$
$L'_3 = L_1 \cup L_2$

L_3 je generovateľné T0L systémom práve vtedy, keď je zvolené $n = 2^k * x$

Odôvodnenie je jednoduché: pokiaľ by sme z axiómy priamo spravili krok odvodenia podľa pravidla $a \to b^x$, vznikne nám najkratšie slovo z jazyka L'_2. Ak by nebolo najkratšie, bolo by nutné zmenšovať počet b, a tým by sme dostali slová ktoré sú mimo jazyka L'_2 nakoľko jeho najkratšie nie je prázdne slovo, a pravidlom na zmenšovanie počtu b by sme sa k prázdnemu slovu určite dostali.

T0L systém potom vyzerá nasledovne:

$\mathcal{S} = (V, \mathcal{P}, x)$, V=$\{a, b\}$, $\mathcal{P} = \{P_1, P_2\}$
$P_1 = \{a \to aa, b \to b\}$
$P_2 = \{a \to b^x, b \to bb \mid b\}$
x=a^{2^m}

Príklad 2.1.4.
Viacerými tabuľkami máme zabezpečenú aj možnosť kontrolovaného rastu: Vezmime T0L systém s abecedou a,b, axiómou a, a nasledovnými tabuľkami pravidiel:

$P_1 = \{a \to bb, b \to b\}$ $P_2 = \{b \to aa, a \to a\}$

Je zjavné, že generovaný jazyk bude $L = \{a^{2^{2n}}, b^{2^{2n+1}} \mid n \geq 0\}$. V tomto prípade vygenerovaný jazyk síce je z triedy 0L, ale počet krokov odvodenia

nutný na odvodenie slova je neurčitý - v konečnom dôsledku je každé slovo z jazyka odvodené.
Ak by sme tento jazyk generovali 0L systémom, bolo by isté, že slovo dĺžky 2^k by bolo odvodené po k krokoch. Táto vlastnosť vyplýva z potreby deterministických pravidiel, ak chceme generovať slová z L.
(Nedeterministické pravidlá by celkom iste viedli k možnosti narastaniu dĺžky slova pomalšie ako exponenciálne v závislosti od počtu krokov.)

Príklad 2.1.5.
Viaceré tabuľky nám rozširujú možnosti synchrónneho rastu pre niektoré jazyky.
$L_1 = \{a^{2^n}b^{2^n} \mid n \geq 0\}$ *odvodiť v 0L systéme síce vieme.*
(Pravidlá $a \to aa, b \to bb$*, axióma* $x = ab$*.)*
$L_2 = \{a^k b^{2^n} \mid n \geq 0, k \geq 2^n\}$ *je pre ľubovoľné prirodzené k nemožné. V 0L máme jedinú šancu toto ovplyvniť pridaním druhého pravidla pre a, ktoré bude pridávať niekoľko symbolov a v niektorom kroku. Lenže to dáva pre k isté obmedzenia, pretože v 0L systéme ktorý by generoval jazyk* L_2 *premenná n zaručene udáva počet krokov. Ak by existovalo pravidlo na zachovanie počtu b v danom kroku odvodenia, nedeterminizmus z pravidiel* $b \to b \mid bb$ *by zaručil, že platí* $\exists w \forall n \ \#_b(w) \neq 2^n$*, Teda by sa generovali aj slová mimo jazyka* L_2*. Takýto jazyk ale vieme vygenerovať T0L systémom:*

Vezmime T0L systém $\mathcal{S}_1$ *s abecedou a,b, axiómou a, a nasledovnými tabuľkami pravidiel:*

$$P_1 = \{a \to aa, b \to bb\} \qquad P_2 = \{a \to a \mid aa, b \to b\}$$

Pravidlá pre a z tabuľky P_2 *nám zaručia zvyšovanie premennej k nezávisle od n Pravidlá z tabuľky* P_1 *nám zaručia zvyšovanie n nezávisle od k.*

Ďalší jazyk ktorý sa bude riešiť obdobným spôsobom je
$L_3 = \{a^{2^{n+k}}b^{2^n} \mid n \geq 0, k \geq 0\}$.
Vezmime T0L systém $\mathcal{S}_2$ *s abecedou a,b, axiómou a, a nasledovnými tabuľkami pravidiel:*

$$P_1 = \{a \to aa, b \to bb\} \qquad P_2 = \{a \to aa, b \to b\}$$

Definícia 2.1.4. *T_n0L budeme označovať T0L systémy s maximálne n tabuľkami.*

Veta 8. ***Platí*** $\mathcal{L}(T_n0L) \subsetneq \mathcal{L}(T_{n+1}0L)$

Dôkaz:
Z definície je zjavné, že $\mathcal{L}(T_n0L) \subseteq \mathcal{L}(T_{n+1}0L)$
Dokážeme, že tieto dve množiny pre ľubovoľné n nie sú rovnaké.
Pre každé n existujú unikátne prvočísla $p_0, p_1, p_2, ..., p_n$

Vezmime jazyk $L=\{a^{p_0^{n_0}p_1^{n_1}...p_n^{n_n}}\}$
Je zjavné, že takýto jazyk nevieme skonštruovať T0L systémom s n tabuľkami, ale T0L systém s n+1 tabuľkami áno. Pre každé $p_i, i < n+1$ *je nutné, aby existovala jedna tabuľka s jediným pravidlom* $a \implies a^{p_i}$.
Nakoľko je prvočísel je n+1, potrebovali by sme n+1 tabuliek.

2.2 Dospelosť jazykov k T0L systémom

Dospelosť jazyka je pri T0L systémoch definovaná podobne ako pri 0L.

Definícia 2.2.1. ***Dospelý jazyk AT0L***
Nech $\mathcal{S} = (V, \mathcal{P}, x)$ *je T0L systém, kde* $\mathcal{P} = \{P_1, ..., P_n\}$, *kde n je počet tabuliek T0L systému. Potom dospelým jazykom k* $\mathcal{S}$ *nazývame*
$L_A(\mathcal{S}) = \{v \in L(\mathcal{S}) \mid ak(v \underset{P_k}{\Longrightarrow} w) tak(v = w)\}$

Takéto slovo v nazývame stabilné slovo.

Pri dospelých jazykoch k TOL je dospelá abeceda definovaná podobne ako pri A0L.

Definícia 2.2.2. ***Dospelá abeceda k AT0L jazyku*** *Vezmime* $\mathcal{S}$ *z predchádzajúcej definície. Potom platí* $V_A(S) = \{a | \exists w : w \in L_A(\mathcal{S}) \wedge \#_a(w) > 0\}$

Príklad 2.2.1. *Nech* $L = \{a^n b^n c^n | n > 0\}$. *Tento jazyk nepatrí do skupiny jazykov* $\mathcal{L}(CF)$, *teda nepatrí ani do A0L. Vieme ale nájsť taký T0L systém* $\mathcal{S}$, *že* $L_A(\mathcal{S}) = L$
Nech teda $\mathcal{S} = (V, \mathcal{P}, x), V = \{a, b, c, A, B\}, x = AB, \mathcal{P} = \{P_1, P_2\}$
$P_1 = \{A \to \epsilon, B \to \epsilon\} \cup \{a \to a | a \in \{a, b, c\}\}$
$P_2 = \{A \to aAb, B \to Bc\} \cup \{a \to a | a \in \{a, b, c\}\}$ *Je zjavné, že po každom kroku odvodenia podľa* P_2 *narastie počet symbolov a,b,c rovnakou rýchlosťou. Po odvodení podľa* P_1 *vzniknú stabilné slová, nakoľko iné A,B prostredníctvom ktorých by sme mohli generovať ďalšie písmená, nebudú existovať v odvodenom slove.*

Príklad 2.2.2. *Platí* $L_2 = \{a^{2^n}, a^{3^n} \mid n \geq 0\} \in AT0L$ *Tento jazyk nevieme generovať T0L systémom. Je ale dospelým jazykom k nasledujúcemu T0L systému:*
$\mathcal{S} = (V, \mathcal{P}, x)$
$\mathcal{P} = \{P_1, P_2, P_3\}$
$V = \{A, B, a, S\}$
$P_1 = \{A \to A^2, B \to B^3, a \to a, S \to S\}$
$P_2 = \{S \to A \mid B, A \to a, B \to a, a \to a\}$
$x = S$

Odvodenie prebieha z axiómy pomocou druhej tabuľky, tu sa rozhodne, či sa bude generovať jazyk a^{2^n} *v prípade, že sa z axiómy odvodí slovo* A,
alebo a^{3^n} *v prípade, že sa z axiómy odvodí slovo* B.
Hodnota premennej n v stabilnom slove je podmienená počtom odvodení podľa tabuľky P_1.
Po niekoľkých odvodeniach podľa tabuľky P_1 *sa použije tabuľka* P_2 *a vznikne stabilné slovo.*

Veta 9. ***Každý jazyk z*** $\mathcal{L}(T0L)$ ***patrí do triedy*** $AT0L$

Dôkaz:
Nech existuje $L \in \mathcal{L}(T0L)$ *a* $L = L(S), \mathcal{S} = (V, P, x)$
Skonštruujeme T0L systém $(S') = (V', \mathcal{P}', x)$, *že* $L = L_A(S')$ *nasledovne: Nech* $V' = V \cup \{a_x | a \in V\}$, *nech x'=h(x), kde h je homorfizmus definovaný takto:* $h(a) = a_x, a \in V, a_x \notin V$
Pravidlá v tabuľkách upravíme nasledovne: každé pravidlo $a \to \alpha$ *premeníme na* $a' \implies h(\alpha)$.
Pridáme novú tabuľku pravidiel $P_A = \{a_x \to a | a \in V\}$. *Navyše do každej tabuľky pridáme pravidlá* $\underset{a \in V}{\forall} a \to a$
Táto tabuľka nám zaručí generovanie dospelého jazyka.
Z konštrukcie je zjavné, že až do prvého použitia tabuľky P_A *platí nasledovné:* $\underset{w \in L(\mathcal{S}) w' \in L(S')}{\forall \quad \exists} h(w) = w'$ *Použitím tabuľky* P_A *dostaneme zaručene stabilné slovo.*
Nakoľko odvodenie podľa tabuľky P_A *je totožné s inverzný homomorfizmom k h, je zjavné, že po odvodení podľa tejto tabuľky vzniknú slová z* $L(\mathcal{S})$, *a teda naša veta platí.*

Dôsledok: Každý jazyk z triedy $\mathcal{L}(0L)$ *patrí do triedy AT0L.*

2.3 ET0L

ET0L systémy sú rozšírením T0L systémov o neterminály, majú podobné vlastnosti ako E0L systémy.

Definícia 2.3.1. *ET0L systém je* $\mathcal{S} = (V, \mathcal{P}, x, \Sigma)$ *kde* $\mathcal{S}' = (V, \mathcal{P}, x)$ *je T0L a platí, že* $\emptyset \neq \Sigma \subseteq V$ *(tieto symboly obvykle nazývame terminály).*
Potom $L(\mathcal{S}) = \{v \in \Sigma^* \mid x \implies {}^*v\}(t.j. L(\mathcal{S}) = L(\mathcal{S}') \cap \Sigma^*\}$

Veta 10. ***ku každému ET0L systému*** $\mathcal{S} = (V, \mathcal{P}, x, \Sigma)$ ***existuje ekvivalentný systém*** $\mathcal{S} = (V', \mathcal{P}', x', \Sigma)$***, že*** $|\mathcal{P}'| \leq 2$

Dôkaz:
Nech $r = |\mathcal{P}| > 2$.
Potom $\mathcal{P}' = \{P'_1, P'_2\}$ *a* $\mathcal{P} = \{P_1, P_2, ...P_r\}$
Konštrukcia je nasledovná:
$V' = V \cup \{a^{(i)} \mid a \in V, i = 1, 2, 3, ..., r-1\}$
$P'_1 = \{a \to a^{(1)} \mid a \in V\} \cup \{a^{(i)} \to a^{(i+1)} \mid a \in V, i = 1, 2, 3, ..., r-1\} \cup$
$\cup \{a^{(r)} \to a^{(r)} \mid a \in V\}$

$P'_2 = \{a^{(i)} \to \alpha \mid a \to \alpha \in P_i\} \cup \{a \to a \mid a \in V\}$

V prípade použitia tohoto tvaru sa vždy na začiatku rozhodne prepismi pomocou pravidiel v P'_1*, ktorá tabuľka pôvodného ET0L systému bude použitá, a následne odvodením v tabuľke* P'_2

Príklad 2.3.1.
Vezmime jednoduchý E0L systém $\mathcal{S} = (V, \mathcal{P}, x, \Sigma)$ *kde:*
V=Σ=$\{a\}$
$\mathcal{P} = \{\{a \to a^2\}, \{a \to a^3\}, \{a \to a^5\}\}$
$x = a$
Podľa vety vytvoríme ekvivalentný ET0L systém s dvoma tabuľkami:
V'=$V \cup \{a^{(1)}, a^{(2)}, a^{(3)}\}$
$\Sigma = \{a\}$
$P'_1 = \{a \to a^{(1)}, a^{(1)} \to a^{(2)}, a^{(2)} \to a^{(3)}, a^{(3)} \to a^{(3)}\}$
$P'_2 = \{a^{(1)} \to a^2, a^{(2)} \to a^3, a^{(3)} \to a^5\}$

Veta 11. ***Ku každému ET0L systému*** $\mathcal{S} = (V, \mathcal{P}, x, \Sigma)$ ***existuje ekvivalentný synchronizovaný*** $\mathcal{S}' = (V', \mathcal{P}', x', \Sigma)$***, kde*** $v \in L(\mathcal{S})$ ***sa odvodia synchronizovane***
Dôkaz:

Nech (V', P', x', Σ) je synchronizovaný E0L systém k (V, P, x, Σ) pre každú tabuľku $P \in \mathcal{P}$

Potom $\mathcal{S}' = (V', \mathcal{P}', x', \Sigma)$ je synchronizovaný ET0L systém, kde $\mathcal{P}' = \{P' \mid P \in \mathcal{P}\}$ - množina tabuliek pravidiel zo synchronizovaných E0L systémov spomenutých vyššie.

Veta 12. ***o normálnom tvare so zamietnutím*** *Ku každému ET0L systému $\mathcal{S} = (V, P, x, \Sigma)$ existuje ekvivalentný systém $\mathcal{S}' = (V', P', S, \Sigma)$, pričom $\underset{P \in \mathcal{P}'}{\forall} \underset{a \in \Sigma}{\forall} a \to \alpha \in P \implies \alpha = a$ a existuje ukončovacia tabuľka $P_T \in \mathcal{P}$, v ktorej platí $\underset{a \in V - \Sigma}{\forall} a \to R$.*
R je zamietací symbol, $R \notin V$.
Dôkaz:
$V' = V \cup \bar{\Sigma} \cup \{S, R\}, S, R \notin V$, pričom symbol S pridávame v prípade, že axióma x bola dlhšia ako jeden znak

Zavedieme homomorfizmus
$\bar{a} = \{$
$\bar{a}, a \in \Sigma,$
$a, a \in V - \Sigma \ \}$

Nech $P_I = \{S \to \bar{X}\} \cup \{a \to a \mid a \in V' - \{S\}\}$

$P_T = \{\bar{a} \to a \mid a \in \Sigma\} \cup \{a \to a \mid a \in \Sigma \cup \{S, R\}\} \cup \{a \to R \mid a \in V - \Sigma\}$

$\mathcal{P} = \{P_I, P_T\} \cup \{\bar{P} \mid P \in \mathcal{P}\}$ *kde* $\bar{P} = \{a \to a \mid a \in \Sigma \cup \{S, R\}\} \cup \{\bar{a} \to \bar{\alpha} \mid a \to \alpha \in P\}$

V každom systéme prebehne odvodenie nasledovne:
Z axiómy vznikne axióma prepísaná homomorfizmom. Každý symbol bude mať teda nad sebou pruh.
Potom prebieha odvodenie medzi symbolmi s pruhom podľa tabuliek z pôvodného systému $\mathcal{S}$ prepísaných homomorfizmom bara. Na záver sa sa prepíšu symboly slov s pruhom na terminály, ak v pred aplikovaním homomorfizmu boli terminálmi, alebo na zamietací symbol v prípade, že terminálmi neboli. Toto zaručuje rovnosť jazykov.

(Pozn.: Symbolu R hovoríme aj zamietací symbol.
O zamietnutí hovoríme preto, že tabuľka P_T je tabuľka, ktorá určuje, kedy slovo v jazyku dospeje do konečného štádia.
Terminály sa podľa predpokladov ďalej neprepisujú, a po prepise P_T sa už slová vôbec nevyvíjajú, vzhľadom tvar pravidiel pre R, a pravidiel pre ter-

minály z množiny Σ
Zároveň vieme, že pokiaľ pred prepisom neboli všetky písmená v slove terminály s pruhom, slovo nebude v jazyku, lebo R *nie je terminál. Pre neterminály je v tabuľke* P_T *pravidlo zamietnutia.)*

Lema 2.3.1.
Pre každý T0L systém $\mathcal{S}$ *existuje ekvivalentný T0L systém* $\mathcal{S}'$ *taký, že* $L_A(\mathcal{S}) = L_A(\mathcal{S}')$*, a pre všetky symboly* $a \in V_A(\mathcal{S})$ *existuje v každej tabuľke systému* $\mathcal{S}'$ *jediné pravidlo* $a \to a$.
Dôkaz: Zavedením správneho homomorfizmu zmeníme tvar pravidiel z $\mathcal{S}$ *na výsledný tvar v* $\mathcal{S}'$*, podobne ako vo vete 1.*
$h(a) = \{$
$X_a \mid a \in V_A,$
$a \mid a \notin V_A\}$
$P' = \{a \to a \mid a \in V_A\} \cup \{a \to h(\alpha) \mid a \notin V_A\}$
Je znovu nutné vygenerovať aj správnu axiómu.
x'=h(x)

Veta 13. ***Množina jazykov generovaná ET0L systémami je ekvivalentná s množinou dospelých jazykov k T0L systémom.***

Dôkaz :
Pre každý ET0L systém existuje T0L systém, ktorý generuje dospelý jazyk ekvivalentný s jazykom generovaným ET0L systému:

Nech $\mathcal{S} = (V, \mathcal{P}, x, \Sigma)$ *je ET0L systém v normálnom tvare so zamietnutím.*
Na základe tvaru pravidiel v tabuľkách systému§ vieme, že po odvodení podľa pravidiel v tabuľke P_T *sa žiadnym ďalším odvodením nevytvorí nové slovo.*

To znamená, že pokiaľ vezmeme T0L systém $\mathcal{S}' = (V, \mathcal{P}, x)$*, v jeho dospelom jazyku budú všetky slová, ktoré vzniknú prepisom podľa tabuľky* P_T.
Slová ktoré neboli odvodené použitím tabuľky P_T *nie sú stabilné, pretože prepísaním podľa tejto tabuľky vzniknú nové slová.*
Ostáva zamedziť výskytu zamietnutých slov v dospelom jazyku k $\mathcal{S}'$*. Pridaním pravidla* $R \to RR$ *do tabuľky* P_T *dosiahneme, že zamietnuté slová z jazyka* L_S *nebudú stabilné.*
Touto konštrukciou je zaručené, že $L(\mathcal{S}) = L_A(\mathcal{S}')$.
Dokázali sme, že pre každý jazyk generovaný ET0L systémom existuje dospelý jazyk z triedy AT0L.

Druhý krok je dokázať, že ku každému dospelému jazyku z triedy AT0L existuje ET0L systém, ktorý bude generovať rovnaký jazyk.

Táto časť je pomerne jednoduchá:
Nech $\mathcal{S}_\mathcal{A} = (V, \mathcal{P}, x)$ *je T0L systém, pričom môžeme predpokladať, že pre každý symbol dospelej abecedy a existuje každej tabuľke práve jedno pravidlo tvaru* $a \to a$
Cieľom je skonštruovať ET0L systém $\mathcal{S}_\mathcal{X} = (V', \mathcal{P}', x', \Sigma)$, *ktorý bude generovať jazyk* $L_A(\mathcal{S}_\mathcal{A})$.
Konštrukcia:
$\Sigma = V_A(\mathcal{S}_\mathcal{A})$,
$V' = V$
$\mathcal{P}' = \mathcal{P}$
$x' = x$

Pre slová obsahujúce len symboly z dospelej abecedy odvoditeľné v S_A, *čiže slová dospelého jazyka* $L_A(\mathcal{S}_\mathcal{A})$, *je z tvaru pravidiel v* $\mathcal{S}_\mathcal{A}$ *zjavné, že budú existovať aj v* $L(\mathcal{S})$. *Iné slová v tomto jazyku nebudú, pretože by museli obsahovať neterminál resp. písmeno mimo dospelej abecedy, čo je podla definície jazyka generovaného ET0L systémom nezmysel.*
Týmto je dokázané, že $L_A(\mathcal{S}_\mathcal{A}) = L(\mathcal{S}_\mathcal{X})$

3 Dospelé jazyky k L-systémom s interakciou

Doteraz sme sa zaoberali systémami ktoré mali pravidlá s jedným symbolom abecedy na ľavej strane. V tejto kapitole sa budeme zaoberať takými, ktoré majú na ľavej strane jeden hlavný symbol pravidla a jeho kontext, t.j. niekoľko symbolov naľavo od hlavného symbolu a niekoľko symbolov napravo od hlavného symbolu.
Počet symbolov v ľavom alebo pravom kontexte môže byť nulový. Súčet kontextov musí byť kladný, inak by išlo o L-systém bez interakcie. Takéto systémy označujeme IL systémy.

3.1 Základné pojmy a tvrdenia

Definícia 3.1.1. ***IL systém***
je definovaný ako štvorica $\mathcal{S} = (V, g, P, x)$*, kde* $g \notin V$ *je symbol, ktorý označuje prázdny symbol kontextu,* V *je konečná abeceda,* $V \neq \emptyset, x \in V^+$ *je axióma.*
P *je množina pravidiel, platí* $P \subset (\bigcup_{i=0}^{m} \{g\}^{m-i} V^i) \times V \times (\bigcup_{j=0}^{n} V^i \{g\}^{n-j}) \times V^*$.
Premenné m,n sú v tomto prípade dĺžky ľavého resp. pravého kontextu.
Pravidlá potom zapisujeme v tvare
$\alpha < a > \beta \to \gamma$
$(\alpha, a, \beta) \to \gamma$
alebo $(\alpha, a, \beta, \gamma)$.

V prípade systémov s kontextom len na jednej strane použijeme skrátený zápis $\alpha < a \to \beta$ *resp.* $a > \alpha \to \beta$
Rovnako ako pre 0L systémy, musí platiť, že pre všetky kombinácie symbolov a kontextov musí existovať pravidlo v P.

Definícia 3.1.2. ***Kontext symbolu***
Zoberme ako príklad slovo $w{=}a_1a_2...a_k$*,* $k{>}0$ *generované (m,n)L systémom.*
Definujeme $a_i = g$ *pre* $i < 1 \vee i > k$.
Potom ľavý kontext sybolu a_i *v slove w je slovo* $l_c(a_i) = a_{i-m}a_{i-m+1}...a_{i-1}$
Pravý kontext sybolu a_i *v slove w je slovo* $r_c(a_i) = a_{i+1}a_{i+2}...a_{i+n}$

Definícia 3.1.3. ***Krok odvodenia v IL systémoch***
Nech $\mathcal{S} = (V, g, P, x)$ *je IL systém. Nech* $v{=}a_1a_2...a_k$*,* $k{>}0$*,* $v \in V^*, w = \alpha_1\alpha_2...\alpha_k$
Potom $v \underset{S}{\Longrightarrow} w$ *je krok odvodenia, ak platí*
$\underset{1 \geq i \geq k}{\forall} l_c(a_i) < a_i > r_c(a_i) \to \alpha_i \in P$

Definícia 3.1.4. ***Jazyk generovaný IL systémom*** *Nech $\mathcal{S} = (V, g, P, x)$ je IL systém. $L(\mathcal{S}) = (w \mid x \implies {}^*w)$ kde * je reflexívnotranzitívny uzáver.*

Príklad 3.1.1. *Pomerne jednoducho vieme (1,0)L aj (0,1)L systémom vygenerovať jazyk $L = (a^n b^n c^n \mid n > 0)$. Vezmime ako príklad (1,0)L systém*
$\mathcal{S} = (V, g, P, x)$
$V = \{a, b, c\}$
$P = \{g < a \to aa$
$x < a \to a \mid x \in \{a, b, c\}$
$a < b \to bb$
$x < b \to b \mid x \in \{g, b, c\}$
$b < c \to cc$
$x < a \to a \mid x \in \{g, a, c\}$
$\}$
$x{=}abc$
Tento systém zjavne generuje požadovaný jazyk. Pravidlá pre (0,1)L systém vyzerajú obdobne.

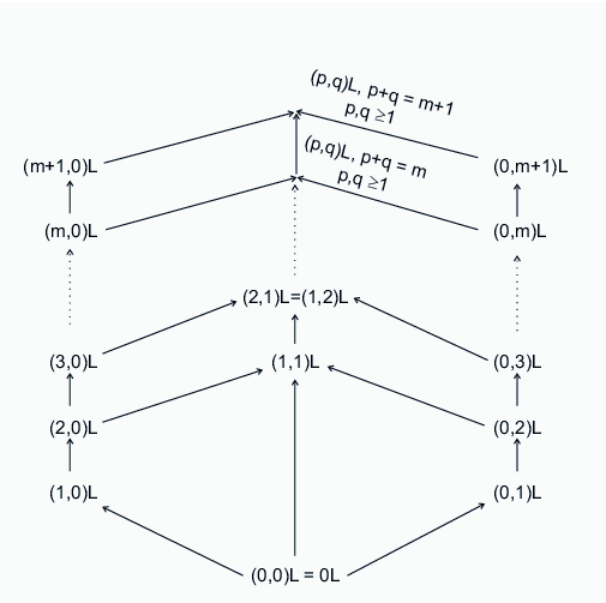
Definícia 3.1.5. ***Triedy IL systémov*** *Vo všeobecnosti označujeme triedy systémov s interakciou podľa dĺžky kontextu v pravidlách (m,n)L,*
pričom m je dĺžka ľavého kontextu a n je dĺžka pravého kontextu v pravidlách. Špeciálne označujeme zjednotenie tried jazykov (1,0)L a (0,1)L: 1L. Triedu jazykov (1,1)L označujeme 2L.

Lema 3.1.1. ***Triedy jazykov (1,0)L a (0,1)L sú neporovnateľné.***
Ukážeme príklad jazyka z triedy (1,0)L ktorý nepatrí do (0,1)L a naopak.
Nech $L_1 = \{a^3, ba^{3+n} \mid n > 0\}$
Potom zjavne existuje (1,0)L systém, ktorý generuje tento jazyk.
$(S) = (V, g, P, x)$,
$x = a^3$
$P = \{g < a \to baa,$
$a < a \to a$
$b < a \to aa$
$x < b \to b \mid x \in \{g, b, a\}\}$

Tento jazyk sa evidentne nedá spraviť (0,1)L systémom.
Voľbu axiómy máme z dvoch možností: baaa, aaa.
V prvom prípade by sme potrebovali pravidlo na zmazanie symbolu b, ale to nám znovu nezamedzí zmazať symbol b aj v iných slovách, než len axióme, lebo pravý kontext symbolu b je stále rovnaký.
V druhom prípade by sme potrebovali nejakým spôsobom k ľavému krajnému symbolu odvodením pridať symbol b. Lenže v slove aaa existuje ďalší symbol,

ktorý má rovnaký kontext ako ľavý krajný symbol, čo by znamenalo, že znovu vygenerujeme slovo mimo jazyka L_1.
Obdobne môžeme postupovať opačným smerom pre jazyk $L_2 = \{a^3, a^{3+n}b \mid n > 0\}$, ktorý bude generovateľný (0,1)L systémom, a (1,0)L systémom nebude.

Lema 3.1.2. *Je všeobecne známe podla [3], že platia vzťahy na nasledovnom diagrame:*



Interpretovať tento diagram môžeme ako orientovaný graf. Platí, že trieda jazykov (1) je podmnožinou triedy jazykov (2), pokiaľ sa vieme konečným počtom hrán dostať z vrcholu (1) reprezentujúceho triedu jazykov (1) do vrcholu (2) ktorý reprezentuje triedu jazykov (2).

3.2 Dospelý jazyk v IL systémoch

Dospelý jazyk k IL systému je definovaný presne ako 0L systému, teda nesmie existovať odvodenie zo slova dospelého jazyka také, že vznikne nové slovo.

Definícia 3.2.1. ***Dospelý jazyk k IL systému***
Nech $\mathcal{S}$ = (V,g,P,x) je IL systém. Potom dospelým jazykom k $\mathcal{S}$ nazývame $L_A(\mathcal{S}) = \{v \in L(\mathcal{S}) \mid ak(v \implies w)tak(v = w)\}$
Slovo $v \in L_A(\mathcal{S})$ nazývame stabilné slovo.

Definícia 3.2.2. ***Dospelá abeceda***
Nech $\mathcal{S}$ = (V,g,P,x) je IL potom dospelou abecedou nazývame množinu $V_A(\mathcal{S}) = \{a \mid (a \in V) \wedge (\exists v \in L_A(S) : \#_a(v) > 0\}$
Označme $abc(v) = \{a \in V \mid \#_a(v) > 0)\}$ potom $V_A(S) = \bigcup_{v \in L_A} abc(v)$

Pre $a \in V$ Budeme označovať X_a, ak platí $a \overset{+}{\Longrightarrow} X_a \Rrightarrow X_a$

Príklad 3.2.1. *Vezmime jazyk $L_1 = \{a^{2^n}\}$. Tento sme vedeli generovať 0L systémom, ale nebol dospelým jazykom triedy A0L, pretože nie je z triedy bezkontextových jazykov, ale vieme ho generovať ako dospelý jazyk k T0L systému.*
Teraz sa pokúsime ho vygenerovať ako dospelý jazyk k 2L systému.

Nech $\mathcal{S}$ = (V,g,P,x) je 2L systém,
$V = \{A, a, F\}$

$x = A$
$P = \{$
$g < A > g \to A^2 \mid a,$
$g < A > A \to A^2 \mid a$
$A < A > A \to A^2 \mid a$
$A < A > g \to A^2 \mid a$
- týmito pravidlami zaručujeme rast a dospievanie slov
$g < a > a \to a$
$g < a > g \to a$
$a < a > a \to a$
$a < a > g \to a$
- týmito pravidlami zaručíme stabilnosť slov dospelého jazyka
$A < a > x \to F,$
$x < a > A \to F,$
$x < F > x' \to F,$
$x, x' \in \{A, a, g, F\},$
$y < A > y' \to F,$
$y, y' \in \{A, a, g, F\} \wedge (y \in \{a, F\} \vee y' \in \{a, F\})$ $\}$
- týmito pravidlami zaručíme zamietnutie nesprávneho, nesynchronizovaného dospievania slov, ktorého dôsledkom by vniklo slovo mimo požadovaného jazyka.
Pokúsime sa rovnaký jazyk vygenerovať ako jazyk triedy A(1,0)L. Jednoznačne vieme rozlíšiť, kedy jazyk nedospel synchronizovane aj s absenciou pravého kontextu, pretože v tomto prípade vždy existuje aspoň jeden symbol A ktorý má ľavý kontext a alebo aspoň jeden symbol a ktorý má ľavý kontext A (čo je znakom nesynchronizovaného odvodenia).

Nech $\mathcal{S}$ = (V,g,P,x) je (0,1)L systém,
$V = \{A, a, F\}$
$x = A$
$P = \{g < A \to A^2 \mid a$
$A < A \to A^2 \mid a$
$A < a \to F$
$F < a \to F$
$a < A \to F$
$F < A \to F$
$a < a \to a$
$g < a \to a$
$y < F \to F \mid y \in V\}$

Máme aj príklad jazyka, ktorý vieme generovať ako dospelý jazyk triedy AT0L, E0L systémom, ale T0L systémom ani IL systémom ho generovať nevieme. Ukážeme, že sa dá generovať ako jazyk triedy. A(1,0)L

Príklad 3.2.2. $L = \{a^{2^n}, a^{3^n} \mid n \geq 0\}$

Nech existuje (1,0)L systém $\mathcal{S}$ = (V,g,P,x)
Aby generoval tento jazyk ako dospelý, zaručene pravidlá
$g < a \to a,\ a < a \to a$
patria do tabuľky pravidiel P.
Keďže potrebujeme rozlíšiť, či sme v aktuálnom stave odvodenia v časti jazyka a^{2^n} alebo a^{3^n}, zavedieme nové symboly A,B, počiatočný symbol S a zamietací symbol F.
x=S
V={S,A,B,a,F}
V tabuľke pravidiel budú figurovať dve pravidlá, zapísané v skrátenom tvare
$g < S \to A \mid B$
ktoré v počiatočnom stave systému rozhodnú, do ktorej vetvy jazyka bude smerovať odvodenie.
Nasledovať budú dôležité pravidlá na rozmnoženie symbolov:
$g < A \to AA$
$A < A \to AA$
$g < B \to BBB$
$B < B \to BBB$
Dôležité je synchronizovane ukončiť odvodzovanie, čím získame stabilné slovo. Nesynchronizovane odvodené slová nebudú v jazyku, pretože ich v ďalšom kroku znestabilníme zamietacím symbolom F.
Preto sú pravidlá pre F v tvare
$y < F \to FF, y \in V,$

Zamietnutie nesynchrónneho odvodenia zabezpečíme týmito pravidlami:
$g < a \to a$
$a < a \to a$
$y < a \to F$,
$a < y \to F, y \in V - \{a, g\}$

Synchrónne odvodenia vedú do stabilného slova, a to obsahuje výhradne symboly a, v inom prípade bude v odvodenom slove prítomný zamietací symbol.

Dospelosť slova zabezpečíme synchrónnym prepisom podľa týchto pravidiel v závislosti od vetvy jazyka do ktorého bude spadať stabilné slovo.
$r < A \to a, r \in \{g, A\}$
$s < B \to a, s \in \{g, B\}$
Ostatné pravidlá budú mať na pravej strane F, pretože budú prepisovať symboly ktoré majú nežiaduci kontext.

Veta 14. ***Platí vzťah*** $\mathcal{L}(IL) \subsetneq AIL$
Jazyk z predchádzajúceho príkladu sa dá použiť na dôkaz existencie dospelého jazyka triedy AIL ktorý nie je z generovateľný IL systémom.
Na základe kontextu nie je možné určiť, či je systém práve vo vetve odvodenia generujúceho a^{2^n} *alebo* a^{3^n}*, pretože kontext môže byť len konečné slovo, teda tento jazyk určite nevieme generovať systémom z triedy IL.*
Pre prehľadnosť ukážeme dôkaz len pre $\mathcal{L}((0,1)L) \subsetneq A(0,1)L$*, pre ostatné podtriedy by bol dôkaz podobný.*
Majme (1,0)L systém $\mathcal{S}$ *= (V,g,P,x).*
Potom existuje (1,0)L systém $\mathcal{S}'$ *= (V',g,P',x') taký, že platí* $L(\mathcal{S}) = L_A(\mathcal{S}')$

Konštrukcia je nasledovná:
zavedieme homomorfizmus h(x), ktorým zavedieme nové symboly, ktoré nebudú v dospelej abecede. $h(x) = \{\bar{x} \mid x \in V\}$
x'=h(x)
$V' = V \cup \{h(x) \mid x \in V\} \cup \{F\}$
$P' =$
(0) $\{h(c) < h(b) \to h(\alpha) \mid (c < b \to \alpha) \in P\} \cup$
(1) $\cup\{h(x) < h(a) \to a \mid a \in V \wedge x \in V\} \cup$
(2) $\cup\{x < a \to FF \mid x \in (V' - V) \wedge a \in V\} \cup$
(3) $\cup\{a < x \to FF \mid x \in (V' - V) \wedge a \in V \cup \{g\}\} \cup$
(4) $\cup\{b < a \to a \mid a \in V \wedge b \in V \cup \{g\}\} \cup$
(5) $\cup\{x < F \to FF \mid x \in V' \cup \{g\}\} \cup$
(6) $\cup\{F < x \to FF \mid x \in V'\}$

Pravidlá z množiny v riadku (0) zaručujú správanie systému pri odvodzovaní nedospelých slov podľa pôvodného (0,1)L systému.
V riadku (1) vidíme množinu obsahujúcu pravidlá na dospievanie.
V riadku (4) vidíme pravidlo, ktoré zachováva dospelé symboly, pokiaľ majú pre nás správny kontext, a to symboly dospelej abecedy (ktorá je presne rovnaká ako abeceda pôvodného systému).
Ostatné riadky zaručujú detekciu nesynchrónneho odvodenia.

Záver

Spísané fakty a tvrdenia by mali zrozumiteľným spôsobom ozrejmiť umiestnenie Lindenmayerových systémov v teórii formálnych jazykov a paralelných výpočtov, k nim zodpovedajúce dospelé jazyky a rozšírenia systémov zaradiť do hierarchie. Tiež by mali vysvetliť dôležité vlastnosti a porovnať jazyky generované Lindenmayerovými systémamy s Chomského hierarchiou.

Známy výsledok zverejnený v [6] je, že trieda dospelých jazykov k 0L systémom je identická s triedou bezkontextových jazykov. Ďalší známy fakt dokázaný v [2] je rovnosť triedy rekurzívne vyčísliteľných jazykov a triedy jazykov generovaných EIL systémami, ktoré sa zároveň rovnajú triede jazykov generovaných E(1,0)L systémami.

Výsledkom, ktorý nebol doteraz skúmaný, nakoľko prax tento fakt nevyžadovala, je identickosť triedy dospelých jazykov k T0L systémom a triedy jazykov generovaných ET0L systémami.

V budúcnosti by mohlo byť zaujímavé zistiť, aký vzťah je medzi jazykmi generovanými EIL systémami a dospelými jazykmi k IL systémom. Predpokladá sa ale rovnosť, podobne ako pri jazykoch generovaných systémami triedy ET0L a triedy dospelých jazykoch AT0L. To by znamenalo, že trieda dospelých jazykov AIL by bola identická s triedou rekurzívne vyčísliteľných jazykov.

Ďalšia možnosť výskumu by sa mohla orientovať na vzťah IL systémov a ich rozšírení ku kontextovým gramatikám.

Literatúra

[1] Rozenberg G. - Salomaa, A. 1992 Lindenmayer systems: impacts on theoretical computer science computer graphics and developmental biology,Berlin: Springer-Verlag, 1992 ISBN 0387553207

[2] Vaško, M. 2010 Rozšírenia Lindenmayerových systémov s interakciou, Bratislava: Univerzita Komenského, 2010.

[3] Minárik, J. 2006. Lindenmayerove systémy s interakciou. Bratislava: Univerzita Komenského, 2006

[4] Rozenberg G. - Salomaa, A. 1980. The mathematical theory of L systems. New York: Academic Press, 1980. ISBN 0125971400

[5] Hopcroft, J.E -Ullmann, J.D. 1969. Formal languages and their relation to automata. Reading: Addison-Wesley Pub. Co., 1969. ISBN 0201029839

[6] ROZENBERG, G. - SALOMAA, A. 1986. The Book of L. Berlin: Springer-Verlag, 1986. ISBN 0387160221