# PARALLEL COCKE-YOUNGER-KASAMI-BASED PARSING

**Zbyněk Sopuch**

Master Degree Programme (2), FIT BUT

E-mail: xsopuc00@stud.fit.vutbr.cz

Supervised by: Alexander Meduna

E-mail: meduna@fit.vutbr.cz

**Abstract**: We deal with text processing and syntax analysis every day, and new areas are emerging. Therefore, we need new effective methods which fit into these areas. In this work, we explore the topics of parallel grammars and E0L-systems. The goal is to improve the Cocke-Younger-Kasami algorithm and present a stronger algorithm of analysis.

**Keywords**: E0L-system, parallel grammar, CYK algorithm, parsing

## 1. ÚVOD

Bezkontextové jazyky (BKJ) patří v praxi mezi programátory nejčastěji používané. Jsou zde ale oblasti a problémy, které spadají do vyšší třídy. Praktická využitelnost kontextových jazyků je však snížena mimo jiné například o exponenciální složitost rozhodnutí příslušnosti řetězce do jazyka.

V této práci se zabýváme možností zvýšení síly analýzy nikoliv změnou struktury gramatických pravidel, ale zavedením určitých mechanizmů, které omezí či naopak vynutí část derivací a zvýší tak vyjadřovací sílu jazyka na úroveň mezi BKJ a kontextovými jazyky. Využijeme k tomu paralelismu, který do algoritmu zavedou L-systémy, a zesílíme metodu Cocke-Younger-Kasami (CYK) analýzy pro bezkontextové gramatiky (BKG) na oblast jazyků generovaných E0L-systémy.

## 2. PREREKVIZITY

U čtenáře předpokládáme hlubší znalosti teorie formálních jazyků. Základní definice a pojmy z této oblasti jsou k nalezení například v [1], [3].

### 2.1. PARALELNÍ GRAMATIKY

U *sekvenčního postupu* derivace aplikujeme v každém kroku pouze jedno pravidlo gramatiky na jeden výskyt symbolu. Tento postup najdeme u všech gramatik z Chomského hiearchie. Naopak *paralelní postup* se vyznačuje tím, že v každém kroku derivace jsou najednou přepsány všechny symboly slova. Lze k tomu použít různá pravidla, ale i stejná opakovaně. Více informací viz [2].

### 2.2. L-SYSTÉMY

L-systémy (Lindenmayerovy systémy) sloužily původně jako matematický model pro simulaci paralelních dějů v přírodě. Cílem bylo spojit do jednoho celku strukturu problému spolu s pravidly pro jeho vývoj. Jako nosič paralelismu použijeme základní variantu E0L-systému, které dokážou popsat celou třídu BKJ a část třídy jazyků kontextových. Zařazení E0L-systémů v Chomského hierarchii zobrazuje schéma 1. Důkazy, definice a popis L-systémů lze nalézt v [2].

*Definice 1*: E0L-systém je čtveřice $G = (V, \Delta, P, w_0)$, kde

- V je abeceda systému,
- $\Delta$ je terminální abeceda systému, $\Delta \subseteq V$,
- P je konečná množina bezkontextových pravidel tvaru $A \rightarrow x$, kde $A \in V$ a $x \in V^*$,
- $w_0$ je startovací slovo (axiom).

Derivace je relace $\Rightarrow_G$ taková, kdy v každém kroku přepíšeme všechny symboly aktuálního slova pravidly z množiny P. Reflexivní a tranzitivní uzávěr této relace značíme $\Rightarrow^*_G$.

*Definice 2*: Mějme E0L-systém G = (V, Δ, P, $w_0$). E0L jazyk generovaný E0L-systémem G je:

$$L(G) = \{w: w \in \Delta^*, w_0 \Rightarrow_G^* w\}.$$

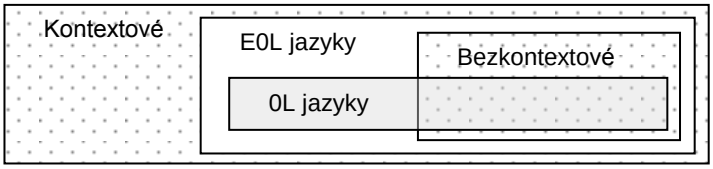


**Schéma 1**: Porovnání síly E0L-systémů

### 2.3. CYK METODA PRO BEZKONTEXTOVÉ JAZYKY

CYK metoda analyzuje syntaktickou strukturu řetězce směrem zdola nahoru. Je určená pro analýzu BKJ a využívá BKG ve speciální Chomského normální formě (CNF). Je oblíbená především pro její jednoduchost. Více informací lze nalézt v [3].

*Algoritmus 1*: **Vstupem** je řetězec $x = a_1a_2...a_n$ a BKG G = (N, T, P, *S*) v CNF, **výstupem** je rozhodnutí o příslušnosti řetězce *x* do jazyka L(G). Metoda je definovaná následovně:

1. Vytvořme množiny CYK[*i*,*j*] = ∅, pro $1 \le i \le j \le n$, kde $n = |x|$.
2. Pro všechna *i* od 1 do *n*, kde existuje $X \rightarrow a_i \in P$, $a_i \in T$, vložíme *X* do CYK[*i*,*i*].
3. Opakujeme bod 3.1, dokud dochází ke změně obsahu množin CYK a zároveň $S \notin$ CYK[1,*n*]
   3.1. Pokud $B \in$ CYK[*i*,*j*], $C \in$ CYK[*j*+1,*k*], $A \rightarrow BC \in P$, pak vložíme *A* do CYK[*i*,*k*].
4. Pokud $S \in$ CYK[1,*n*], tak $x \in L(G)$, jinak $x \notin L(G)$.

## 3. CYK METODA NAD E0L-SYSTÉMY

Proč vlastně stavět syntaktickou analýzu nad E0L-systémy? Jednak jsou silnější než BKG, což je náš hlavní cíl, a jednak nám přinášejí výše zmíněný paralelismus, který v dnešní době rozvoje paralelních výpočtů může skrývat značný potenciál. Důležitou stránkou také je, že E0L-systémy používají stále pouze bezkontextový tvar pravidel.

### 3.1. ANALÝZA MODIFIKACE

Stěžejní problém paralelismu, který bylo potřeba vyřešit, je generování všech symbolů řetězce na stejné úrovni derivačního stromu. Mimo jiné to znamená, že při přechodu na další úroveň stromu musí být aplikována pravidla na všechny symboly aktuální úrovně. Abychom tento princip zajistili a nešlo generovat/substituovat (záleží na směru pohledu) symbol z úrovně jiné, než vzdálené právě o jednu úroveň, vytváříme vždy kopii pracovní úrovně a provádíme analýzu pouze nad ní. Pozitivní na tomto přístupu je, že nám to zajistí vynucení přepsání všech potřebných symbolů v každé úrovni zároveň, neboť pokud by některý nebyl přepsán, nikdy již nemůžeme dojít ke startujícímu axiomu.

Další změnou je tvar pravidel v CNF. Pravidla mohou nabýt binárního tvaru A → BC (kde A, B, C jsou neterminály), případně unárního A → *a* (kde *a* je terminál). Pomineme prozatím rozdělení na terminály a neterminály. Prakticky se jedná o podmnožinu množiny všech bezkontextových pravidel, proto vyžadování této formy pravidel definici E0L-systému neodporuje.

Problémem zůstává algoritmické převedení obecného E0L-systému na E0L-systém s pravidly v CNF. Bohužel tyto systémy nepokrývají stejnou jazykovou třídu, protože paralelismus v kombinaci s pouze binárními pravidly nám nedovolí vygenerovat například jazyk $L = \{a^n: n > 1\}$. Navíc právě rozdíl terminály/neterminály je u E0L-systémů velmi tenký, neboť oproti klasickému pojetí lze terminály umístit i na levou stranu pravidla. Využijeme toho k řešení našeho problému a povolíme neomezená unární pravidla typu A → B, kde $\{A, B\} \subseteq V$, namísto původního terminální-

ho smyslu u symbolu B. Jelikož jsme tím porušili CNF, tak takovouto modifikovanou formu budeme nazývat CNF-U (s unárními pravidly).

Formálně složitější bude také rozhodnutí o příslušnosti řetězce do jazyka, tzn. v případě, že jsme dosáhli multikořene derivačního složeného ze všech symbolů $w_0$ ve správném pořadí. Pojmem „multikořen“ pouze zdůrazňujeme, že kořen může být tvořen více než jedním symbolem. Lze transformovat E0L-systému se startujícím řetězcem $w_0$ na systém se startujícím symbolem $S'$ pomocí pravidla $S' \rightarrow w_0$, ale nelze zaručit, že bude vyhovovat CNF-U. Museli bychom vygenerovat celý nový derivační strom jednoznačně generující z $S'$ řetězec $w_0$, nebo opět tuto formu porušit.

### 3.2. ALGORITMUS ANALÝZY E0L-SYSTÉMŮ CYK METODOU

*Algoritmus 2*: **Vstupem** je E0L-systém G = (V, Δ, P, $w_0$) s pravidly v CNF-U a vstupní slovo $x = a_1a_2...a_n$, **výstupem** rozhodnutí o příslušnosti řetězce $x$ do jazyka generovaného systémem G. Metoda je definována následovně:

1. Předpokládejme, že $1 \leq i \leq j \leq k \leq n$, kde $n = |x|$.
2. Pokud neplatí, že $\{a_1, a_2, \ldots, a_n\} \subseteq \Delta$, tak $x \notin L(G)$ a algoritmus končí.
3. Vytvořme množiny $M[i,j] = \emptyset$.
4. Pro všechna $i$, $1 \leq i \leq n$, provedeme $M[i,i] = \{x[i]\}$, $x[i]$ značí $i$-tý symbol slova.
5. Opakujeme, dokud jsou změny a zároveň jsme nedosáhli multikořene stromu $w_0$ (viz poslední bod algoritmu: *Podmínka nalezení startujícího axiomu*):
   5.1. Vytvořme pomocné množiny $MP[i,j] = \emptyset$.
   5.2. Pokud $B \in M[i,j]$ a $C \in M[j+1,k]$ a zároveň $A \rightarrow BC \in P$, vložme $A$ do $MP[i,k]$.
   5.3. Pokud $B \in M[i,j]$ a zároveň $A \rightarrow B \in P$, $B \in V$, vložme $A$ do $MP[i,j]$.
   5.4. Pro všechny množiny $M[i,j]$, kde $1 \leq i \leq j \leq n$, proveďme $M[i,j] = MP[i,j]$.
6. *Podmínka nalezení startujícího axiomu* $w_0$: Je-li $w_0[g]$ $g$-tý symbol ze startujícího axiomu $w_0$ a pro všechna $g$, kde $1 \leq g \leq |w_0|-1$, platí, že $w_0[g] \in M[i,j]$ a zároveň $w_0[g+1] \in M[j+1,k]$ a zároveň $w_0[1] \in M[1,j]$ a $w_0[|w_0|] \in M[i,n]$, tak $x \in L(G)$, jinak $x \notin L(G)$.

## 4. ZÁVĚR

Všechny požadované změny ze sekce 3.1 jsme promítli do algoritmu 1 a získali tím metodu analýzy syntaktické struktury řetězce, algoritmus 2, který lze aplikovat nad jazyky generované E0L-systémy. Proto dokážeme analyzovat i některé jazyky z vyšší třídy než BKJ (viz schéma 1), zároveň však využívá gramatická pravidla jednoduchého bezkontextového tvaru, přidáváme do procesu paralelismus a zachováváme podobnost s klasickou CYK metodou.

Dokázali jsme tím, že lze zesílit standardní analytické metody i bez změny tvaru pravidel na složitější (například kontextový) tvar. Právě paralelismus a změna řízení generování jazyka otevírá nové možnosti, kdy jednoduché, člověku blízké gramatiky využíváme ke generování stále složitějších syntaktických struktur.

## REFERENCE


[1] MEDUNA, Alexander. *Automata and Languages: Theory and Applications*. London, GB: Springer Verlag, 2000. 892 s. ISBN 1-85233-074-0.

[2] VAVREČKOVÁ, Šárka. *Teorie jazyků a automatů II* [online]. Opava, 2008. 89 s. Studijní opora. Slezská universita v Opavě, aktualizováno 2008-02-26 [cit. 2011-01-18] Dostupné na URL: <http://fpf.slu.cz/~vav10ui/obsahy/tja/skripta/tja2_celek.pdf>.

[3] MEDUNA, Alexander. *Elements of Compiler Design*. Boca Raton: Auerbach Publications, 2008. 286 s. ISBN 978-1-4200-6323-3.