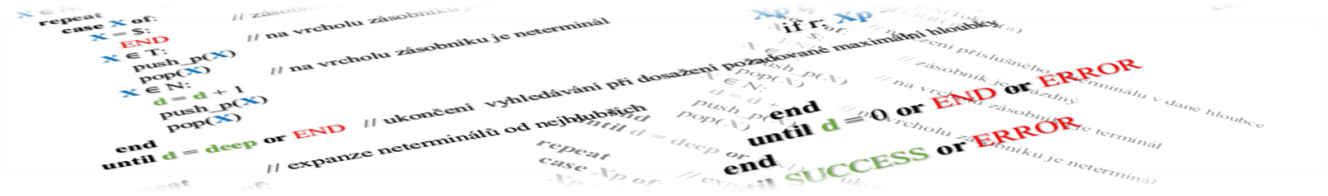
Excel@FIT 2015
http://excel.fit.vutbr.cz

Zásobníkové systémy a jejich použití v syntaktické analýze

Zdeněk Křesťan*



Abstrakt
Bude síla bezkontextových gramatik stačit i v budoucnu? Cílem tohoto článku je nastínit možné řešení tohoto problému využitím hlubokého zásobníkového automatu. Modifikací algoritmu pro prediktivní syntaktickou analýzu tímto automatem získáme algoritmus, který jednoduchým způsobem překoná sílu bezkontextových gramatik.

Klíčová slova: Hluboký zásobníkový automat — Syntaktická analýza — Překladače a gramatiky

Přiložené materiály: *N/A*

*xkrest06@stud.fit.vutbr.cz, *Faculty of Information Technology, Brno University of Technology*

1. Úvod

V dnešní době využívá většina jazyků bezkontextové gramatiky. Ty by však mohly časem nedostačovat. Je tedy nutné najít nové mechanismy syntaktické analýzy. Jako možné řešení se nabízí využití hlubokého zásobníkového automatu, který je schopný přesáhnout sílu bezkontextových gramatik. Toto řešení spočívá v nahrazení klasického zásobníkového automatu v prediktivní syntaktické analýze hlubokým zásobníkovým automatem.

Hlavním úkolem bylo modifikovat algoritmus pro prediktivní syntaktickou analýzu tak, aby pracoval s hlubokým zásobníkovým automatem a otestovat tento algoritmus různými tvary řetězců vstupních jazyků. Pro tyto jazyky je nutné vytvořit gramatiky. Tyto gramatiky využívají rozšířená pravidla o hloubku a LL tabulky pro různé hodnoty hloubky.

Například pokud se podíváme na tento jazyk L.

$$L = \{wcw | w \in \{a,b\}^*\} \quad (1)$$

Tedy dvě stejná slova w oddělená znakem c, pro který neexistuje žádná bezkontextová gramatika, která by jej generovala. Tento jazyk tedy není bezkontextový.

Algoritmus, který je detailněji popsán níže, velmi jednoduše a efektivně provede analýzu tohoto jazyka na základě pravidel a LL tabulek.

2. Hluboký zásobníkový automat

Hluboký zásobníkový automat má možnost pracovat na rozdíl od zásobníkového automatu v hloubce [1]. Kde hloubka udává maximální zanoření v zásobníku. Poprvé byl představen v článku v časopise Acta Informatika [2] v roce 2006 Alexandrem Medunou. Pracuje na stejném principu vyjmutí a expanze jako zásobníkový automat. Vyjmutí se neliší. Pokud na vstupní pásce je stejný terminál jako na vrcholu zásobníku provede se vyjmutí, ale expanze je odlišná. Na rozdíl od zásobníkového automatu, který provádí expanzi pouze neterminálu na vrcholu, je možnost provádět expanzi i neterminálů, které jsou umístěny níže v zásobníku. Expanzi lze tedy provést nad n nejvrchnějšími neterminály. Se zvyšující se hloubkou roste hierarchicky síla tohoto automatu. Síla tohoto automatu je srovnatelná se sílou

n-omezených stavových gramatik.

Definition 2.1
Hluboký zásobníkový automat je definován jako uspořádaná sedmice ${}_{deep}M = (Q, \sum, \Gamma, R, s, S, F)$, *kde*

- *deep je maximální hloubka, ve které může být provedena expanze neterminálu*
- Q *je konečná neprázdná množina stavů*
- $\sum$ *je množina vstupních symbolů (vstupní abeceda)*
- Γ *je množina zásobníkových symbolů (zásobníková abeceda), kde* $\sum \subset \Gamma$, $\$ \in (\Gamma - \sum)$, *symbol* \$ *označuje dno zásobníku*
- R *je konečná množina pravidel tvaru:* ${}_{d}pN \rightarrow q$, $0 < d <= deep$ *(d značí hloubku prováděné expanze),* $p, q \in Q$, $N \in (\Gamma - \sum)$
- $s \in Q$ *je počáteční stav*
- $S \in \Gamma$ *je počáteční zásobníkový symbol*
- $F \subseteq Q$ *je množina všech koncových stavů*

3. Prediktivní, tabulkou řízená syntaktická analýza

Syntaktický analyzátor u této metody syntaktické analýzy se skládá ze zásobníku symbolů, vstupní pásky a prediktivní tabulky. Tato metoda je v tomto článku modifikována, proto si ji přiblížíme. Analyzátor na základě symbolu na vrcholu zásobníku a aktuálního symbolu ze vstupní pásky provádí dvě základní operace: vyjmutí a expanzi pravidla. Analýza končí buď vyprázdně- ním zásobní- ku a vstupní pásky nebo při syntaktické chybě, kdy pro danou kombinaci symbolů neexistuje záznam v tabulce. Vyjmutí je provedeno pokud na vrcholu zásobníku je shodný terminální symbol jako je aktuální symbol ze vstupní pásky. Pokud se na vrcholu zásobníku nachází neterminální symbol je podle tabulky zjištěno příslušné pravidlo a provedena expanze podle tohoto pravidla. Základem je tedy obyčejný zásobníkový automat, který umožňuje expanzi pouze neterminálu na vrcholu zásobníku, a proto je výsledný algoritmus slabší než při využití hlubokého zásobníkového automatu.

4. Algoritmus prediktivní syntaktické analýzy s hlubokým zásobníkovým automatem

Mnou vytvořený algoritmus vychází z algoritmu pro prediktivní syntaktickou analýzu, tedy provádí vyjmutí a expanze. Nachází-li se na vrcholu zásobníku stejný terminál jako je aktuální znak (terminál) ze vstupní pásky je tento terminál vyjmut. U expanze je postup odlišný. Jak je již psáno výše, expanze probíhá od nejhlubších neterminálů na zásobníku. Je tedy nutné všechny tyto neterminály od příslušné hloubky získat a provést jejich expanzi. K tomuto úkolu je použit pomocný zásobník, do kterého jsou postupně vkládány terminály i neterminály ze zásobníku. Tímto způsobem získáme požado- vaný neterminál, který můžeme expandovat pomocí pravidla získaného z LL tabulky. Algoritmus končí úspěšně, pokud vyprázdní zásobník a na vstupní pásce nezůstane žádný znak.

4.1 Rozbor algoritmu krok po kroku

Algoritmus je zobrazen v pseudokódu podle konvence, která je převzata z [3].

Popis jednotlivých proměnných a funkcí.

- X - vrchol zásobníku
- X_p - vrchol pomocného zásobníku
- a - terminál z aktuální pozice čtecí hlavy, která čte zleva doprava
- $deep$ - maximální hloubka pro expanzi
- d - aktuální hloubka
- **push, pop** - funkce pro vložení/vyjmutí ze zásobníku
- **push_p, pop_p** - funkce pro vložení/vyjmutí z pomocného zásobníku

Algoritmus pracuje v cyklu, který končí, pokud je zásobník prázdný a jsou přečteny všechny symboly ze vstupní pásky nebo pokud dojde k nalezení chyby při kontrole vstupního řetězce. Pokud se na vrcholu zásobníku nachází terminál, který odpovídá aktuálnímu znaku ze vstupní pásky je vyjmut. Při výskytu neterminálu na vrcholu zásobníku je nutné nalézt další neterminály do příslušné hloubky v zásobníku a následně tyto neterminály expandovat.

Toto hledání zajišťuje první vnořený cyklus (řádky 13 - 22 v 1), který postupně bere symboly z vrcholu zásobníku. Pokud narazí na terminál, tak jej odstraní z vrcholu a vloží na pomocný zásobník. Totéž provádí s neterminály, kde navíc při každé této operaci zvyšuje aktuální hloubku. Tento cyklus končí po dosažení požadované hloubky nebo vyprázdněním zásobníku. Po tomto cyklu jsou všechny neterminály, které mají být expandovány umís- těny v pomocném zásobníku.

Na vrcholu pomocného zásobníku se tedy nachází neterminál z největší hloubky. Druhý vnořený cyklus (řádky 23 - 34 v 1) pracuje na podobném principu jako předchozí. Pokud narazí na terminál, tak je vkládá zpět do zásobníku, ale v případě neterminálu se provede expanze podle pravidla. Výsledek expanze, tedy pravá strana pravidla je obrácena a vložena na hlavní zásobník. Jaké pravidlo má být použito se určí pomocí dotazu do LL-tabulky, kde se na základě neterminálu, aktuálního terminálu ze vstupní pásky

Algoritmus 1: Modifikovaný algoritmus prediktivní syntaktické analýzy

```
1  d = 0; deep =<max. hloubka gramm.>;
2  push($); push_p($);
3  nextToken(a) ; // funkce vracející další terminál
4  repeat
5      switch X do
6          case X = $
7              if a = $ then SUCCESS ;
8              else ERROR ;
9          case X ∈ T
10             if X = a then
11                 pop(X); nextToken(a);
12             else ERROR ;
13         case X ∈ N
14             repeat// cyklus na vyhledání neterminálů
15                 switch X do
16                     case X = $
17                         END;
18                     case X ∈ T
19                         push_p(X); pop(X);
20                     case X ∈ N
21                         d = d + 1; push_p(X);
                           pop(X);
22                 endsw
23             until d == deep or END;
24             repeat// cyklus na expanzi neterminálů
25                 switch X_p do
26                     case X = $
27                         END
28                     case X ∈ T
29                         push_p(X); pop(X);
30                     case X ∈ N
31                         if X_p → x ∈ tabulka[X_p,
                           a, d] then
32                             push(reversal(x));
                               pop_p(X_p);
                               d = d − 1;
33                         else ERROR ;
34                 endsw
35             until d == 0 or END or ERROR;
36     endsw
37 until SUCCESS or ERROR;
```

a aktuální hloubky určí číslo pravidla nebo při chybné struktuře určí syntaktická chyba.

Ukázka dotazu do LL-tabulky.

$$pravidlo = tabulka[X_p, a, d] \tag{2}$$

Výsledek expanze je vložen do zásobníku a již expandovaný neterminál je odstraněn z vrcholu pomocného zásobníku. Tento cyklus postupně expanduje všechny neterminály až do vyprázdnění pomocného zásobníku.

5. Modifikovaná LL gramatika

Výše uvedený algoritmus využívá LL gramatiky, u kterých na rozdíl od běžných LL gramatik jednotlivá pravidla obsahují hloubku, která je klíčová pro použití pravidla. Hloubka je znázorněna indexem na začátku každého pravidla.

Ukázka pravidel pro jazyk $L = \{wcw | w \in \{a,b\}^*\}$:

1. ${}_1S \rightarrow XY$
2. ${}_1X \rightarrow aX$
3. ${}_1X \rightarrow bX$
4. ${}_2Y \rightarrow aY$
5. ${}_2Y \rightarrow bY$
6. ${}_1X \rightarrow c$
7. ${}_2Y \rightarrow \varepsilon$

Pro použití pravidla je tedy nutné mít nejen samotný neterminál, ale i hloubku, ve které se tento neterminál nachází. Na základě těchto znalostí je vytvořena LL-tabulka, která je třírozměrná. K základním výběrům pomocí neterminálu a znaku (terminálu) ze vstupní pásky přibývá ještě hloubka. Hloubka ve dvourozměrném zobrazení je znázorněna hodnotou v první buňce tabulky (levý horní roh).

Ukázka LL tabulky pro jazyk $L = \{wcw | w \in \{a,b\}^*\}$:

1	a	b	c	$
X	2	3	6	
Y				
S	1	1	1	
$				☺

2	a	b	c	$
X				
Y	4	5	7	
S				
$				

U této gramatiky je maximální hloubka dvě. To znamená, že při každé expanzi budou expandovány až dva nejvrchnější neterminály na zásobníku.

6. Závěr

Cílem tohoto článku bylo navrhnout jednoduchý, ale dostatečně silný algoritmus tabulkou řízené prediktivní syntaktické analýzy. Důležité je si tedy zapamatovat, že využitím hlubokého zásobníkového automatu

získáme větší sílu než při použití bezkontextových gramatik. Zde navržený algoritmus je jednoduchý na pochopení a výukovou demonstraci.

Literatura

[1] Alexander Meduna and Petr Zemek. *Regulated Grammars and Automata*. Springer US, 2014.

[2] Alexander Meduna. Deep pushdown automata. *Acta Informatica*, 2006(98):114–124, 2006.

[3] Alexander Meduna. *Formal Languages and Computation*. Taylor and Francis. Taylor & Francis Informa plc, 2014.