SYNTAX ANALYSIS BASED ON THE PDA WITH DEEP PUSHDOWN EXPANSIONS

Jiří Šimáček, Bachelor Degree Programme (3)
Dept. of Information Systems, FIT, BUT
E-mail: xsimac00@stud.fit.vutbr.cz

Supervised by: Prof. Alexander Meduna

ABSTRACT

This paper discusses construction of a parser based on the generalized pushdown automaton which is able to make expansions deeper in the pushdown. First of all the formalism (state grammar) used to describe appropriate class of languages is introduced. The model for the parser of a language generated by such a grammar is explained as well. Finally, the method how to design the deep top-down parser according to the given state grammar is shown.

1 ÚVOD

Překladače programovacích jazyků bývají často založeny na modelu tříúrovňové analýzy – lexikální, syntaktické a sémantické. Každá úroveň se vyznačuje typem jazyka, který je schopna rozpoznávat. Lexikální analýza zpracuje vstupní text a vytvoří jednotnou reprezentaci lexikálních prvků. Obvykle pracuje s jazykem typu 3 (regulární). Její výstup přebírá syntaktická analýza, která rozpoznává jednotlivé jazykové konstrukce a výstup předává sémantické analýze. Syntaktická analýza obvykle pracuje s jazykem tybu 2 (bezkontextový). Sémantická analýza zkoumá ty aspekty, které není schopna vyřešit syntaktická analýza, protože je nelze popsat jazykem příslušného typu. Jazyk sémantické analýzy nebývá obvykle striktně definován pomocí zavedených formalismů a způsob implementace sémantické analýzy se oproti lexikální a syntaktické analýze může značně odlišovat (v rámci různých implementací). To je také jeden z důvodů, proč může být výhodné provádět syntaktickou analýzu nad obecnějším jazykem a zjednodušit tak úlohu sémantické analýzy. Zobecnění zásobníkového automatu na automat, který může provádět expanze neterminálů hlouběji v zásobníku, je jednou z možností, jak takového zvýšení síly syntaktické analýzy dosáhnout.

2 STAVOVÁ GRAMATIKA

Stavová gramatika je šestice $G = (V, W, T, P, s_0, S)$, kde V je *úplná abeceda*, W je konečná množina *stavů*, $T \subseteq V$ je abeceda terminálů, $P \subseteq (W \times (V - T)) \times (W \times V^+)$ je konečná relace, $s_0 \in W$ je počáteční stav a $S \in (V - T)$ je počáteční neterminál. Místo $(q, A, r, y) \in P$ lze psát $(q, A) \to (r, y) \in P$. Pokud $x \in V^*$, $x = uAv, u, v \in V^*, A \in (V - T)$ a současně $(q, A) \to (r, y) \in P$, pak A je *potenciálně derivovatelný* v (q, x). Navíc pokud A je nejlevější potenciálně derivovatelný neterminál v (q, x), pak G provede *derivační krok* z (q, uAv) do (r, uyv), zkráceně $(q, uAv) \Rightarrow (r, uyv)[(q, A) \to (r, y)]$. Pokud je počet neterminálů v uA nejvýše $n \geq 1$, pak je derivační krok $(q, uAv) \Rightarrow (r, uyv)[(q, A) \to (r, y)]$ *n-ohraničený*, zkráceně $(q, uAv)\ {}_n\Rightarrow (r, uyv)[(q, A) \to (r, y)]$. $\Rightarrow^*$ označuje reflexivní a tranzitivní uzávěr relace P (neformálně libovolný počet derivačních kroků – $(q, x) \Rightarrow (r, y) \Rightarrow \ldots \Rightarrow (s, z)$). Jazyk generovaný gramatikou G je definován jako $L(G) = \{w \in T^* | (s_0, S) \Rightarrow^* (r, w)\}$. Navíc pro každé $n \geq 1$ definujeme $L(G, n) = \{w \in T^* | (s_0, S)\ {}_n\Rightarrow^* (r, w)\}$.

2.1 PŘÍKLAD

$$
\begin{array}{rcl}
G & = & (\{a, b, c, d, A, C, S\}, \{f, r, s\}, \{a, b, c, d\}, P, s, S) \\
P & = & \{\ (s, S) \to (r, AC) \\
 & & \quad (r, A) \to (s, aAb), (r, A) \to (f, ab), (s, C) \to (r, cCd), (f, C) \to (f, cd)\ \ \}
\end{array}
$$

$$
\begin{array}{llllll}
(s, S) & \Rightarrow & (r, AC) & [(s, S) \to (r, AC)] & \Rightarrow & (s, aAbC) \quad [(r, A) \to (s, aAb)] \\
 & \Rightarrow & (r, aAbcCd) & [(s, C) \to (r, cCd)] & \Rightarrow & (f, aabbcCd) \quad [(r, A) \to (f, ab)] \\
 & \Rightarrow & (f, aabbccdd) & [(f, C) \to (f, cd)] & &
\end{array}
$$

Pozn.: $L(G) = L(G, 2) = \{a^n b^n c^n d^n | n \geq 1\}$

3 SYNTAKTICKÝ ANALYZÁTOR S HLUBOKÝM ZÁSOBNÍKEM

Syntaktický analyzátor s hlubokým zásobníkem je sedmice $M = (Q, \Sigma, \Gamma, R, s_0, S, F)$, kde Q je konečná množina stavů, Σ je vstupní abeceda, Γ je zásobníková abeceda, $R \subseteq (\mathbb{N} \times Q \times (\Gamma - \Sigma)) \times (Q \times \Gamma^+)$ je konečná relace, $s_0 \in Q$ je počáteční stav, $S \in \Gamma$ je počáteční zásobníkový symbol a $F \subseteq Q$ je množina koncových stavů. Místo $(m, p, A, q, v) \in R$ lze psát $mpA \to qv \in R$. R označujeme jako množinu přechodů syntaktického analyzátoru M. *Konfigurace* syntaktického analyzátoru je trojice $\Gamma^* \times Q \times \Sigma^*$. Množinu všech konfigurací označme χ. Nechť $x, y \in \chi$ jsou konfigurace. M čte svůj zásobník z x do y – symbolicky $x\ {}_p\Rightarrow y$, pokud $x = (q, az, au), y = (q, z, u)$, kde $q \in Q, a \in \Sigma, z \in \Gamma^*, u \in \Sigma^*$. Naopak M expanduje svůj zásobník z x do y podle $mqA \to rv$ – symbolicky $x\ {}_e\Rightarrow y[mqA \to rv]$, pokud $x = (q, uAz, w), y = (r, uvz, w), mqA \to rv \in R$, kde $A \in \Gamma, u, v, z \in \Gamma^*, w \in \Sigma, p, q \in Q$ a počet výskytů symbolů z $(\Gamma - \Sigma)$ v uA je m. M udělá přechod z x do y – symbolicky $x \Rightarrow y$, pokud $x\ {}_p\Rightarrow y$ nebo $x\ {}_e\Rightarrow y$. Pokud n je nejmenší kladné celé číslo takové, že pro každý přechod $mqA \to rv$ platí, že $m \geq n$, pak M je hloubky n, psáno ${}_nM$.

3.1 PŘÍKLAD

$$
\begin{array}{rcl}
{}_2M & = & (\{r, s\}, \{a, b, c\}, \{A, C, S\}, R, s, S, \{r\}) \\
R & = & \{1sS \to rAC, 1rA \to saAb, 2sC \to rcCd, 1rA \to sab, 1sC \to rcd\}
\end{array}
$$

$$
\begin{array}{lll}
(s, aabbccdd, S) & {}_e\Rightarrow & (r, aabbccdd, AC) \quad [1sS \to rAC] \\
& {}_e\Rightarrow & (s, aabbccdd, aAbC) \quad [1rA \to saAb] \\
& {}_e\Rightarrow & (r, aabbccdd, aAbcCd) \quad [2sC \to rcCd] \\
& {}_e\Rightarrow & (s, aabbccdd, aabbcCd) \quad [1rA \to sab] \\
& {}_e\Rightarrow & (r, aabbccdd, aabbccdd) \quad [1sC \to rcd] \\
& {}_p\Rightarrow^8 & (r, \varepsilon, \varepsilon)
\end{array}
$$

Pozn. 1: $L({}_2M) = \{a^n b^n c^n d^n | n \geq 1\}$
Pozn. 2: V příkladu je dno zásobníku vpravo.

3.2 KONSTRUKCE SYNTAKTICKÉHO ANALYZÁTORU

Nechť $L(G,n)$ je jazyk, pro který chceme navrhnout syntaktický analyzátor, generovaný gramatikou $G = (V, W, T, P, s_0, S)$. pref(s,n) je notace označující předponu s délky n, suf(s,n) označuje příponu s délky n a f(s) je kódování, které z s vypustí všechny terminály. Syntaktický analyzátor $M = (Q, \Sigma, \Gamma, R, s'_0, S, F)$ sestrojíme následovně:

$Q \subseteq \{\langle q,s\rangle | q \in W, s \in ((\Gamma - \Sigma) \cup \{\#\})^n\}$;
$\Sigma = T$; $\Gamma = V$; $R = \emptyset$; $s'_0 = \langle s_0, S\#^{n-1}\rangle$; $F = \{\langle q, \#^n\rangle | q \in W\}$;
for $s \in \{x | x \in (\Gamma - \Sigma)^*\#^*, |x| = n\}$, $q \in W$ **do**
  i = 0;
  changed = false;
  while $i < n$ **and** changed = false **do**
    for $(q, s[i]) \to (r,y) \in P$ **do**
      add $i\langle q,s\rangle s[i] \to \langle r, \text{pref}(\text{pref}(s,i)\,\text{f}(y)\,\text{suf}(s, n-i-1)\#^*, n)\rangle y$ **to** R;
      changed = true;
    od
    i = i + 1;
  od
od

3.3 SYNTAKTICKÁ ANALÝZA

Idea. Syntaktický analyzátor simuluje n-ohraničenou derivaci, tudíž je možné, aby si pamatoval prvních n neterminálů na zásobníku, které je možné expandovat, ve svém stavu. Při provedení přechodu analyzátor provede změnu stavu a příslušný neterminál na zásobníku nahradí řetězcem na pravé straně vybraného pravidla. Pokud vložený řetězec obsahuje pouze terminály a zásobník obsahuje alespoň n neterminálů, pak analyzátor upraví svůj stav podle podle n-tého neterminálu na zásobníku (načte do svého stavu). Pokud analyzátor přečte celý vstup a vyprázdní svůj zásobník, pak na výstup vloží PŘIJATO, v ostatních případech ZAMÍTNUTO.

REFERENCE

[1] Meduna, A.: Deep Pushdown Automata, Acta Informatica, roč. 2006, č. 98, DE, s. 114-124, ISSN 0001-5903