Script **generated by TTT**

Title: Petter: Compiler Construction (04.06.2020) - LR(1) Parsers

Date: Wed May 27 12:17:15 CEST 2020

Duration: 15:26 min

Pages: 6

The Action Table:

During practical parsing, we want to represent states just via an integer id. However, when the canonical $LR(1)$-automaton reaches a final state, we want to know *how to reduce/shift*. Thus we introduce...

The construction of the action table:

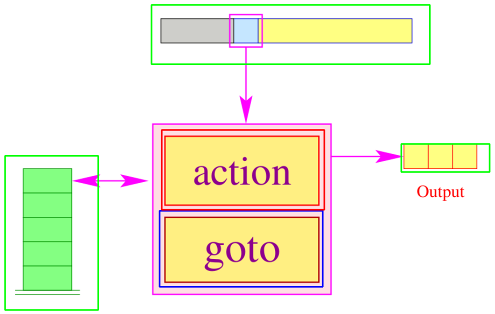
Type: action : $Q \times T \to LR(0)$-Items $\cup$ {s, error}

Reduce: action$[q, w] = [A \to \beta \bullet]$ if $[A \to \beta \bullet, w] \in q$

Shift: action$[q, w] =$ s if $[A \to \beta \bullet b\gamma, a] \in q,\ w \in$ First$_1(b\gamma) \odot_1 \{a\}$

Error: action$[q, w] =$ error else

The LR(1)-Parser:



- The goto-table encodes the transitions:

$$\text{goto}[q, X] = \delta(q, X) \in Q$$

- The action-table describes for every state q and possible lookahead w the necessary action.

The LR(1)-Parser:

The construction of the $LR(1)$-parser:

States: $Q \cup \{f\}$ (f fresh)
Start state: q_0
Final state: f
Transitions:

Shift: $(p, a, p\,q)$ if $a = w$, $\text{s} = \text{action}[p, a]$, $q = \text{goto}[p, a]$

Reduce: $(p\,q_1 \ldots q_{|\beta|}, \epsilon, p\,q)$ if $q_{|\beta|} \in F$, $[A \to \beta \bullet] = \text{action}[q_{|\beta|}, w]$, $q = \text{goto}[p, A]$

Finish: $(q_0\,p, \epsilon, f)$ if $[S' \to S \bullet, \$] \in p$

with $LR(G, 1) = (Q, T, \delta, q_0, F)$ and the lookahead w.

The LR(1)-Parser:

Possible actions are:

shift	//	Shift-operation
reduce $(A \to \gamma)$	//	Reduction with callback/output
error	//	Error

... for example:

$$
\begin{array}{lcl}
S' & \to & E \\
E & \to & E + T^{\,0} \;\mid\; T^{\,1} \\
T & \to & T * F^{\,0} \;\mid\; F^{\,1} \\
F & \to & (\,E\,)^{\,0} \;\mid\; \text{int}^{\,1}
\end{array}
$$

action	$	int	(	)	+	*
q_1	$S', 0$				s	
q_2	$E, 1$				$E, 1$	s
q_2'				$E, 1$	$E, 1$	s
q_3	$T, 1$				$T, 1$	$T, 1$
q_3'				$T, 1$	$T, 1$	$T, 1$
q_4	$F, 1$				$F, 1$	$F, 1$
q_4'				$F, 1$	$F, 1$	$F, 1$
q_9	$E, 0$				$E, 0$	s
q_9'				$E, 0$	$E, 0$	s
q_{10}	$T, 0$				$T, 0$	$T, 0$
q_{10}'				$T, 0$	$T, 0$	$T, 0$
q_{11}	$F, 0$				$F, 0$	$F, 0$
q_{11}'				$F, 0$	$F, 0$	$F, 0$

The Canonical LR(1)-Automaton

In general: We identify two conflicts for a state $q \in Q$:

Reduce-Reduce-Conflict:

q: $A \to \gamma \bullet, x$; $A' \to \gamma' \bullet, x$ with $A \neq A' \vee \gamma \neq \gamma'$

Shift-Reduce-Conflict:

q: $A \to \gamma \bullet, x$; $A' \to \alpha' \bullet a\, \beta, y$ with $a \in T$ und $x \in \{a\}$.

Such states are now called $LR(1)$-unsuited

The Canonical LR(1)-Automaton

In general: We identify two conflicts for a state $q \in Q$:

Reduce-Reduce-Conflict:

q: $A \to \gamma \bullet, x$; $A' \to \gamma' \bullet, x$ with $A \neq A' \vee \gamma \neq \gamma'$

Shift-Reduce-Conflict:

q: $A \to \gamma \bullet, x$; $A' \to \alpha' \bullet a\, \beta, y$ with $a \in T$ und $x \in \{a\} \odot_k \text{First}_k(\beta) \odot_k \{y\}$.

Such states are now called $LR(k)$-unsuited

Theorem:

A reduced contextfree grammar G is called $LR(k)$ iff the canonical $LR(k)$-automaton $LR(G, k)$ has no $LR(k)$-unsuited states.