Script generated by TTT

Title: Simon: Compilerbau (27.05.2013)

Date: Mon May 27 14:34:00 CEST 2013

Duration: 76:07 min

Pages: 28

Characteristic Automaton

Observation:
The set of viable prefixes from $(N \cup T)^*$ for (admissible) items can be computed from the content of the shift-reduce parser's pushdown with the help of a finite automaton:

States: Items
Start state: $[S' \to \bullet\, S]$
Final states: $\{[B \to \gamma \bullet] \mid B \to \gamma \in P\}$
Transitions:

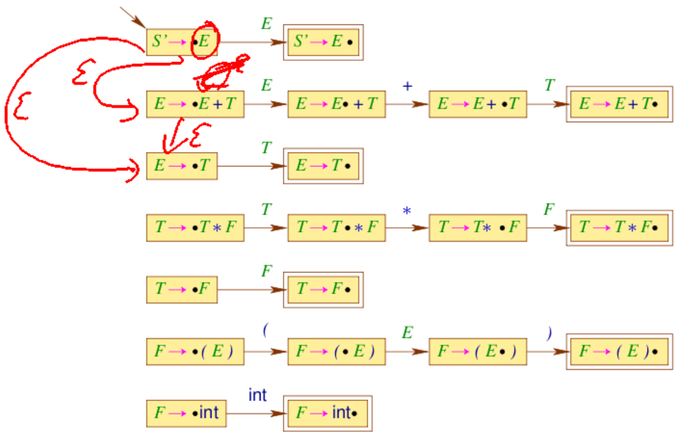
$$(1) \quad ([A \to \alpha \bullet X\, \beta], X, [A \to \alpha\, X \bullet \beta]), \quad X \in (N \cup T), A \to \alpha\, X\, \beta \in P;$$
$$(2) \quad ([A \to \alpha \bullet B\, \beta], \epsilon, [B \to \bullet\, \gamma]), \quad A \to \alpha\, B\, \beta\ ,\ B \to \gamma \in P;$$

The automaton $c(G)$ is called characteristic automaton for G.

Characteristic Automaton

for example:

$$E \to E + T \mid T$$
$$T \to T * F \mid F$$
$$F \to (\ E\ ) \mid \text{int}$$



Characteristic Automaton

Observation:
The set of viable prefixes from $(N \cup T)^*$ for (admissible) items can be computed from the content of the shift-reduce parser's pushdown with the help of a finite automaton:

States: Items
Start state: $[S' \to \bullet\, S]$
Final states: $\{[B \to \gamma \bullet] \mid B \to \gamma \in P\}$
Transitions:

$$(1) \quad ([A \to \alpha \bullet X\, \beta], X, [A \to \alpha\, X \bullet \beta]), \quad X \in (N \cup T), A \to \alpha\, X\, \beta \in P;$$
$$(2) \quad ([A \to \alpha \bullet B\, \beta], \epsilon, [B \to \bullet\, \gamma]), \quad A \to \alpha\, B\, \beta\ ,\ B \to \gamma \in P;$$

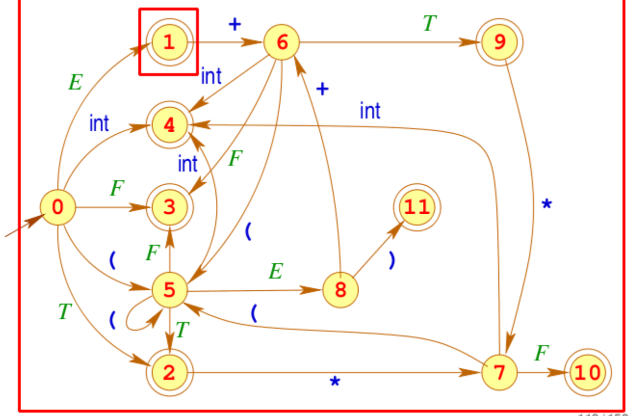
The automaton $c(G)$ is called characteristic automaton for G.

Canonical LR(0)-Automaton

The canonical $LR(0)$-automaton $LR(G)$ is created from $c(G)$ by:

1. performing arbitrarily many ϵ-transitions after every consuming transition
2. performing the powerset construction

... for example:



Canonical LR(0)-Automaton

Therfore we determine:

$$q_0 = \{[S' \to \bullet E], [E \to \bullet E + T], [E \to \bullet T], [T \to \bullet T * F]\} , [T \to \bullet F], [F \to \bullet (E)], [F \to \bullet \text{int}]\}$$

$$q_1 = \delta(q_0, E) = \{[S' \to E \bullet], [E \to E \bullet + T]\}$$

$$q_2 = \delta(q_0, T) = \{[E \to T \bullet], [T \to T \bullet * F]\}$$

$$q_3 = \delta(q_0, F) = \{[T \to F \bullet]\}$$

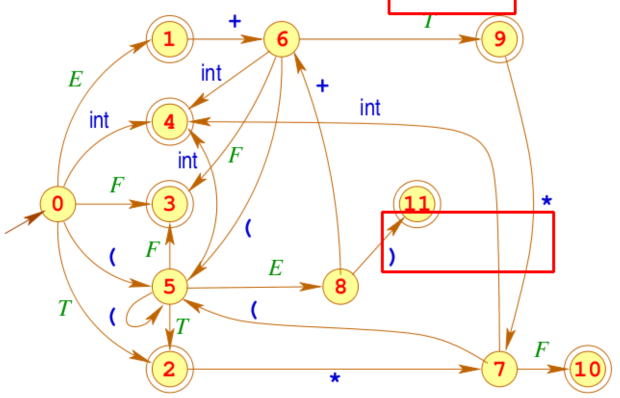
$$q_4 = \delta(q_0, \text{int}) = \{[F \to \text{int} \bullet]\}$$

Canonical LR(0)-Automaton

The canonical $LR(0)$-automaton $LR(G)$ is created from $c(G)$ by:

1. performing arbitrarily many ϵ-transitions after every consuming transition
2. performing the powerset construction

... for example:



Canonical LR(0)-Automaton

$$q_5 = \delta(q_0, () = \{[F \to (\bullet E)], [E \to \bullet E + T], [E \to \bullet T], [T \to \bullet T * F], [T \to \bullet F], [F \to \bullet (E)], [F \to \bullet \text{int}]\}$$

$$q_6 = \delta(q_1, +) = \{[E \to E + \bullet T], [T \to \bullet T * F], [T \to \bullet F], [F \to \bullet (E)], [F \to \bullet \text{int}]\}$$

$$q_7 = \delta(q_2, *) = \{[T \to T * \bullet F], [F \to \bullet (E)], [F \to \bullet \text{int}]\}$$

$$q_8 = \delta(q_5, E) = \{[F \to (E \bullet)]\} [E \to E \bullet + T]\}$$

$$q_9 = \delta(q_6, T) = \{[E \to E + T \bullet], [T \to T \bullet * F]\}$$

$$q_{10} = \delta(q_7, F) = \{[T \to T * F \bullet]\}$$

$$q_{11} = \delta(q_8,)) = \{[F \to (E) \bullet]\}$$

Canonical LR(0)-Automaton

$$
\begin{aligned}
q_5 &= \delta(q_0, (\,) = \{[F \to (\bullet E)],\ [E \to \bullet E + T],\ [E \to \bullet T],\ [T \to \bullet T * F],\ [T \to \bullet F],\ [F \to \bullet (E)],\ [F \to \bullet \text{int}]\}\\
q_6 &= \delta(q_1, +) = \{[E \to E + \bullet T],\ [T \to \bullet T * F],\ [T \to \bullet F],\ [F \to \bullet (E)],\ [F \to \bullet \text{int}]\}\\
q_7 &= \delta(q_2, *) = \{[T \to T * \bullet F],\ [F \to \bullet (E)],\ [F \to \bullet \text{int}]\}\\
q_8 &= \delta(q_5, E) = \{[F \to (E \bullet)],\ [E \to E \bullet + T]\}\\
q_9 &= \delta(q_6, T) = \{[E \to E + T \bullet],\ [T \to T \bullet * F]\}\\
q_{10} &= \delta(q_7, F) = \{[T \to T * F \bullet]\}\\
q_{11} &= \delta(q_8,)\,) = \{[F \to (E) \bullet]\}
\end{aligned}
$$

Canonical LR(0)-Automaton

Therfore we determine:

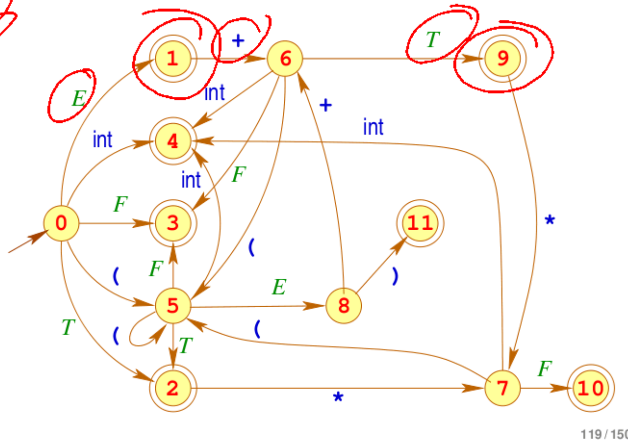
$$
\begin{aligned}
q_0 &= \{[S' \to \bullet E],\ [E \to \bullet E + T],\ [E \to \bullet T],\ [T \to \bullet T * F]\},\ [T \to \bullet F],\ [F \to \bullet (E)],\ [F \to \bullet \text{int}]\}\\
q_1 &= \delta(q_0, E) = \{[S' \to E \bullet],\ [E \to E \bullet + T]\}\\
q_2 &= \delta(q_0, T) = \{[E \to T \bullet],\ [T \to T \bullet * F]\}\\
q_3 &= \delta(q_0, F) = \{[T \to F \bullet]\}\\
q_4 &= \delta(q_0, \text{int}) = \{[F \to \text{int} \bullet]\}
\end{aligned}
$$

Canonical LR(0)-Automaton

The canonical $LR(0)$-automaton $LR(G)$ is created from $c(G)$ by:

1. performing arbitrarily many ϵ-transitions after every consuming transition
2. performing the powerset construction

... for example:



Canonical LR(0)-Automaton

Therfore we determine:

$$
\begin{aligned}
q_0 &= \{[S' \to \bullet E],\ [E \to \bullet E + T],\ [E \to \bullet T],\ [T \to \bullet T * F]\},\ [T \to \bullet F],\ [F \to \bullet (E)],\ [F \to \bullet \text{int}]\}\\
q_1 &= \delta(q_0, E) = \{[S' \to E \bullet],\ [E \to E \bullet + T]\}\\
q_2 &= \delta(q_0, T) = \{[E \to T \bullet],\ [T \to T \bullet * F]\}\\
q_3 &= \delta(q_0, F) = \{[T \to F \bullet]\}\\
q_4 &= \delta(q_0, \text{int}) = \{[F \to \text{int} \bullet]\}
\end{aligned}
$$

120 / 150

LR(0)-Parser

Idea for a parser:

- The parser manages a viable prefix $\alpha = X_1 \dots X_m$ on the pushdown and uses $LR(G)$, to identify reduction spots.
- It can reduce with $A \to \gamma$, if $[A \to \gamma \bullet]$ is admissible for α

Optimization:

We push the states instead of the X_i in order not to process the pushdown's content with the automaton anew all the time. Reduction with $A \to \gamma$ leads to popping the uppermost $|\gamma|$ states and continue with the state on top of the stack and input A.

Attention:

This parser is only deterministic, if each final state of the canonical $LR(0)$-automaton is conflict free.

Canonical LR(0)-Automaton

Observation:

The canonical $LR(0)$-automaton can be created directly from the grammar.
Therefore we need a helper function δ^*_ϵ (ϵ-closure)

$$\delta^*_\epsilon(q) = q \cup \{[B \to \bullet\gamma] \mid \exists [A \to \alpha \bullet B' \beta'] \in q, \; \beta \in (N \cup T)^* : B' \to^* B\,\beta\}$$

We define:

States: Sets of items;

Start state: $\delta^*_\epsilon \{[S' \to \bullet S]\}$

Final states: $\{q \mid \exists A \to \alpha \in P : \; [A \to \alpha \bullet] \in q\}$

Transitions: $\delta(q, X) = \delta^*_\epsilon \{[A \to \alpha X \bullet \beta] \mid [A \to \alpha \bullet X \beta] \in q\}$

LR(0)-Parser

... for example:

$$q_1 = \{[S' \to E \bullet], [E \to E \bullet + T]\}$$

$$q_2 = \{[E \to T \bullet], [T \to T \bullet * F]\} \qquad q_9 = \{[E \to E + T \bullet], [T \to T \bullet * F]\}$$

$$q_3 = \{[T \to F \bullet]\} \qquad q_{10} = \{[T \to T * F \bullet]\}$$

$$q_4 = \{[F \to \text{int} \bullet]\} \qquad q_{11} = \{[F \to (E) \bullet]\}$$

The final states q_1, q_2, q_9 contain more then one admissible item
$\Rightarrow$ non deterministic!

LR(0)-Parser

The construction of the $LR(0)$-parser:

States: $Q \cup \{f\}$ (f fresh)

Start state: q_0

Final state: f

Transitions:

Shift: $(p, a, p\,q)$ if $q = \delta(p, a) \neq \emptyset$

Reduce: $(p\,q_1 \dots q_m, \epsilon, p\,q)$ if $[A \to X_1 \dots X_m \bullet] \in q_m$, $q = \delta(p, A)$

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

with $LR(G) = (Q, T, \delta, q_0, F)$.

LR(0)-Parser

Correctness:

we show:

The accepting computations of an $LR(0)$-parser are one-to-one related to those of a shift-reduce parser M_G^R.

we conclude:

- The accepted language is exactly $\mathcal{L}(G)$
- The sequence of reductions of an accepting computation for a word $w \in T$ yields a reverse rightmost derivation of G for w

LR(0)-Parser

Attention:
Unfortunately, the $LR(0)$-parser is in general non-deterministic.

We identify two reasons:

Reduce-Reduce-Conflict:
$[A \to \gamma \bullet]\,,\ [A' \to \gamma' \bullet] \in q$ with $A \neq A' \vee \gamma \neq \gamma'$

Shift-Reduce-Conflict:
$[A \to \gamma \bullet]\,,\ [A' \to \alpha \bullet a\,\beta] \in q$ with $a \in T$

for a state $q \in Q$.

Those states are called $LR(0)$-unsuited.

LR(k)-Grammars

Idea: Apply k-lookahead to solve conflicts.

Definition:
The reduced contextfree grammar G is called $LR(k)$-grammar, if for $\text{First}_k(w) = \text{First}_k(x)$ with:

$$\left.\begin{array}{lllll} S & \to_R^* & \alpha\,A\,w & \to & \alpha\,\beta\,w \\ S & \to_R^* & \alpha'\,A'\,w' & \to & \alpha\,\beta\,x \end{array}\right\} \text{ follows: } \alpha = \alpha' \wedge A = A' \wedge w' = x$$

LR(k)-Grammar

for example:

(1) $S \to A \mid B \qquad A \to a\,A\,b \mid 0 \qquad B \to a\,B\,b\,b \mid 1$

... is not $LL(k)$ for any k — but $LR(0)$:

Let $S \to_R^* \alpha\,X\,w \to \alpha\,\beta\,w$. Then $\alpha\,\underline{\beta}$ is of one of these forms:

$$\underline{A}\,,\ \underline{B}\,,\ a^n\,\underline{a\,A\,b}\,,\ a^n\,\underline{a\,B\,b\,b}\,,\ a^n\,\underline{0}\,,\ a^n\,\underline{1} \qquad (n \geq 0)$$

LR(k)-Grammar

for example:

(1) $S \to A \mid B \qquad A \to a\,A\,b \mid 0 \qquad B \to a\,B\,b\,b \mid 1$

... is not $LL(k)$ for any k — but $LR(0)$:

Let $S \to_R^* \alpha\,X\,w \to \alpha\,\beta\,w$. Then $\alpha\,\underline{\beta}$ is of one of these forms:

$$\underline{A}\,,\ \underline{B}\,,\ a^n\,\underline{a\,A\,b}\,,\ a^n\,\underline{a\,B\,b\,b}\,,\ a^n\,\underline{0}\,,\ a^n\,\underline{1} \qquad (n \geq 0)$$

(2) $S \to a\,A\,c \qquad A \to A\,b\,b \mid b$

... is also $LR(0)$:

Let $S \to_R^* \alpha\,X\,w \to \alpha\,\beta\,w$. Then $\alpha\,\underline{\beta}$ is of one of these forms:

$$a\,\underline{b}\,,\ a\,\underline{A\,b\,b}\,,\ \underline{a\,A\,c}$$

LR(k)-Grammar

for example:

(3) $S \to a\,A\,c \qquad A \to b\,b\,A \mid b$ ... is not $LR(0)$, but $LR(1)$:

Let $S \to_R^* \alpha\,X\,w \to \alpha\,\beta\,w$ with $\{y\} = \text{First}_k(w)$ then $\alpha\,\underline{\beta}\,y$ is of one of these forms:

$$a\,b^{2n}\,\underline{b}\,c\,,\ a\,b^{2n}\,\underline{b\,b\,A}\,c\,,\ \underline{a\,A\,c}$$

LR(k)-Grammar

for example:

(3) $S \to a\,A\,c \qquad A \to b\,b\,A \mid b$ ... is not $LR(0)$, but $LR(1)$:

Let $S \to_R^* \alpha\,X\,w \to \alpha\,\beta\,w$ with $\{y\} = \text{First}_k(w)$ then $\alpha\,\underline{\beta}\,y$ is of one of these forms:

$$a\,b^{2n}\,\underline{b}\,c\,,\ a\,b^{2n}\,\underline{b\,b\,A}\,c\,,\ \underline{a\,A\,c}$$

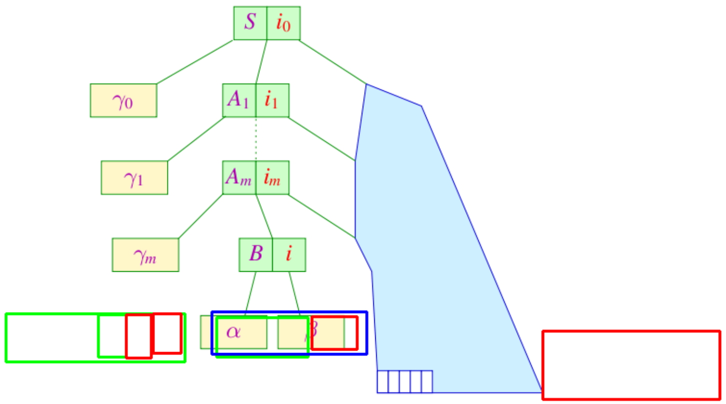
(4) $S \to a\,A\,c \qquad A \to b\,A\,b \mid b$ ... is not $LR(k)$ for any $k \geq 0$:

Consider the rightmost derivations:

$$S \to_R^* a\,b^n\,A\,b^n\,c \to a\,b^n\,\underline{b}\,b^n\,c$$

Ende der Präsentation. Klicken Sie zum Schließen.

LR(1)-Items



... with $\gamma_0 \ldots \gamma_m = \gamma$

The Characteristic LR(1)-Automaton

The automaton $c(G, 1)$:

States: $LR(1)$-items

Start state: $[S' \to \bullet\, S,\ \epsilon]$

Final states: $\{[B \to \gamma \bullet,\ x] \mid B \to \gamma \in P, x \in \text{Follow}_1(B)\}$

Transitions:

(1) $([A \to \alpha \bullet X\, \beta,\ x], X, [A \to \alpha\, X \bullet \beta,\ x]), \quad X \in (N \cup T)$

(2) $([A \to \alpha \bullet B\, \beta,\ x], \epsilon, [B \to \bullet\, \gamma,\ x']),$
$A \to \alpha\, B\, \beta\ ,\quad B \to \gamma \in P, x' \in \text{First}_1(\beta) \odot \{x\};$

This automaton works like $c(G)$ — but additionally manages a 1-prefix from Follow_1 of the left-hand sides.

The canonical LR(1)-automaton

The canonical $LR(1)$-automaton $LR(G, 1)$ is created from $c(G, 1)$, by performing arbitrarily many ϵ-transitions and then making the resulting automaton deterministic ...

The canonical LR(1)-automaton

The canonical $LR(1)$-automaton $LR(G, 1)$ is created from $c(G, 1)$, by performing arbitrarily many ϵ-transitions and then making the resulting automaton deterministic ...
But again, it can be constructed directly from the grammar
Analoguously to $LR(0)$, we need a helper function:

$$\delta^*_\epsilon(q) = q \cup \{[B \to \bullet\, \gamma,\ x] \mid \exists\, [A \to \alpha \bullet B'\, \beta',\ x'] \in q, \beta \in (N \cup T)^* : B' \to^* B\, \beta \wedge x \in \text{First}_1(\beta\, \beta') \odot \{x'\}\}$$

The canonical LR(1)-automaton

The canonical $LR(1)$-automaton $LR(G,1)$ is created from $c(G,1)$, by performing arbitrarily many ϵ-transitions and then making the resulting automaton deterministic ...

But again, it can be constructed directly from the grammar

Analoguously to $LR(0)$, we need a helper function:

$$\delta^*_\epsilon(q) = q \cup \{[B \to \bullet\,\gamma,\, x] \mid \exists\, [A \to \alpha \bullet B'\,\beta',\, x'] \in q, \beta \in (N \cup T)^* : B' \to^* B\,\beta \,\wedge\, x \in \mathsf{First}_1(\beta\,\beta') \odot \{x'\}\}$$

Then, we define:

States: Sets of $LR(1)$-items;

Start state: $\delta^*_\epsilon\,\{[S' \to \bullet\, S,\, \epsilon]\}$

Final states: $\{q \mid \exists A \to \alpha \in P : [A \to \alpha\,\bullet,\, x] \in q\}$

Transitions: $\delta(q, X) = \delta^*_\epsilon\,\{[A \to \alpha\, X \bullet\, \beta,\, x] \mid [A \to \alpha \bullet X\,\beta,\, x] \in q\}$