CSE443
Compilers

Dr. Carl Alphonce
alphonce@buffalo.edu
343 Davis Hall

http://www.cse.buffalo.edu/faculty/alphonce/SP17/CSE443/index.php
https://piazza.com/class/iyn4ndqa1s3ei
Phases of a compiler

Figure 1.6, page 5 of text
Building the finite control for a bottom-up parser

- Build a finite state machine, whose states are sets of items
- Build a table \((M)\) incorporating shift/reduce decisions
Augment grammar

Given a grammar

\[ G = (N,T,P,S) \]

we augment to a grammar

\[ G' = (N \cup \{S'\}, T, P \cup \{S' \rightarrow S\}, S'), \text{ where } S' \not\in N \]

\[ G' \] has exactly one rule with \[ S' \] on left.
CLOSURE(I)

- I is a set of items
- CLOSURE(I) fixed point construction

\[
\text{CLOSURE}_0(I) = I \\
\text{repeat } \{ \\
\text{CLOSURE}_{i+1}(I) = \text{CLOSURE}_i(I) \cup \{ B \rightarrow \gamma \mid A \rightarrow \alpha \beta \in \text{CLOSURE}_i(I) \text{ and } B \rightarrow \gamma \in P \} \\
\} \text{ until } \text{CLOSURE}_{i+1}(I) = \text{CLOSURE}_i(I)
\]
Terminology

- **Kernel items**: $S' \rightarrow \bullet S$ and all items with $\bullet$ not at left edge

- **Non-kernel items**: all items with $\bullet$ at left edge, except $S' \rightarrow S$
GOTO(I,X)

- \textbf{GOTO(I,X)} is the closure of the set of items \( A \rightarrow \alpha X \beta \) s.t. \( A \rightarrow \alpha X \beta \in I \)

- \textbf{GOTO(I,X)} construction for \( G' \) (figure 4.32)

\begin{verbatim}
void items(G') {
    C = { CLOSURE( { S' \rightarrow •S } ) }
    repeat {
        for each set of items \( I \in C \)
        for each grammar symbols \( X \in (NUT) \)
        if ( GOTO(I,X) is not empty and not already in C )
            add GOTO(I,X) to C
    } until no new sets of items are added to C
}
\end{verbatim}
<table>
<thead>
<tr>
<th>Grammar G</th>
<th>Augmented Grammar G'</th>
</tr>
</thead>
<tbody>
<tr>
<td>S' -&gt; E</td>
<td>S' -&gt; E</td>
</tr>
<tr>
<td>E -&gt; E + T</td>
<td>E -&gt; E + T</td>
</tr>
<tr>
<td>E -&gt; T</td>
<td>E -&gt; T</td>
</tr>
<tr>
<td>T -&gt; T * F</td>
<td>T -&gt; T * F</td>
</tr>
<tr>
<td>T -&gt; F</td>
<td>T -&gt; F</td>
</tr>
<tr>
<td>F -&gt; (E)</td>
<td>F -&gt; (E)</td>
</tr>
<tr>
<td>F -&gt; id</td>
<td>F -&gt; id</td>
</tr>
</tbody>
</table>
Compute items(G') requires several steps, start with CLOSURE( { S'→•E } )

<table>
<thead>
<tr>
<th>SET OF ITEMS (I)</th>
<th>i</th>
<th>CLOSURE_i(I)</th>
</tr>
</thead>
<tbody>
<tr>
<td>{ S' → • E }</td>
<td>0</td>
<td>{ S' → • E }</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>CLOSURE_0(I) \cup { E → • E + T , E → • T }</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>CLOSURE_1(I) \cup { T → • T * F , T → • F }</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>CLOSURE_2(I) \cup { F → • (E) , F → • id }</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>CLOSURE_3(I) \cup \emptyset</td>
</tr>
</tbody>
</table>
This gives us the first state of the finite state machine, $I_0$.

\[
\begin{array}{|c|}
\hline
S' \rightarrow \circ E \\
E \rightarrow \circ E + T \\
E \rightarrow \circ T \\
T \rightarrow \circ T \times F \\
T \rightarrow \circ F \\
F \rightarrow \circ (E) \\
F \rightarrow \circ id \\
\hline
\end{array}
\]

**kernel item**

non-kernel items are computed from CLOSURE(kernel), and therefore do not need to be explicitly stored.
Next we compute $\text{GOTO}(I_0, X) \forall X \in N \cup T$

$N \cup T = \{ E, T, F, +, *, (, ), id \}$

N.B. - augmented start symbol $S'$ can be ignored.

$\text{GOTO}(I_0, E) = \text{CLOSURE}( \{ S' \rightarrow E \cdot, E \rightarrow E \cdot + T \} )$

$= \{ S' \rightarrow E \cdot, E \rightarrow E \cdot + T \}$

N.B. there is no non-terminal after the $\cdot$, so no new items are added by \text{CLOSURE} operation.

$I_1$

\begin{align*}
S' & \rightarrow E \cdot \\
E & \rightarrow E \cdot + T
\end{align*}

only kernel items
\[ \text{GOTO}(I_0, T) = \text{CLOSURE}( \{ \ E \rightarrow T \odot, \ T \rightarrow T \odot \ast F \ \} ) \]

\[ = \{ \ E \rightarrow T \odot, \ T \rightarrow T \odot \ast F \ \} \]

\[ I_2 \]

\[ \begin{align*}
E & \rightarrow T \odot \\
T & \rightarrow T \odot \ast F
\end{align*} \]

**N.B.** there is no non-terminal after the \( \odot \), so no new items are added by \text{CLOSURE} operation

**only kernel items**
\[ GOTO(I_0, F) = \text{CLOSURE}( \{ T \rightarrow F \circ \} ) \]

\[ = \{ T \rightarrow F \circ \} \]

N.B. there is no non-terminal after the \( \circ \), so no new items are added by CLOSURE operation

only kernel items
\[ \text{GOTO}(I_0, \mathit{())} = \text{CLOSURE}(\{ F \rightarrow (\cdot E) \}) \]

\[ = \{ F \rightarrow (\cdot E) \} \cup \{ E \rightarrow \cdot E + T, E \rightarrow \cdot T \} \cup \{ T \rightarrow \cdot T \cdot F, T \rightarrow \cdot F \} \cup \{ F \rightarrow \cdot (E), F \rightarrow \cdot \text{id} \} \]

N.B. there is a non-terminal after the \( \cdot \), so new items are added by CLOSURE operation.

**Kernel Items**

- \( F \rightarrow (\cdot E) \)

**Non-Kernel Items**

- \( E \rightarrow \cdot E + T \)
- \( E \rightarrow \cdot T \)
- \( T \rightarrow \cdot T \cdot F \)
- \( T \rightarrow \cdot F \)
- \( F \rightarrow \cdot (E) \)
- \( F \rightarrow \cdot \text{id} \)
\( \text{GOTO}(I_0, \text{id}) = \text{CLOSURE}( \{ T \rightarrow \text{id} \} ) \)

\[ = \{ T \rightarrow \text{id} \} \]

N.B. there is no non-terminal after the \( \bullet \), so no new items are added by \text{CLOSURE} operation

\( \text{Is} \)

\[ T \rightarrow \text{id} \bullet \]

only kernel items

\( \text{GOTO}(I_0, \_ \_ ) = \text{GOTO}(I_0, + ) = \text{GOTO}(I_0, * \_ ) = \emptyset \)
The finite state machine as far as we developed it in lecture last class.

**EXERCISE**: complete the machine by computing \( \text{GOTO}(I_k, X) \) until no new states are added.

**HINT**: there will be 11 states in all.