A Unifying Model for Lookahead LR Parsing

Manuel E. Bermudez

University of Florida
U.S.A.
LR Parser Construction

1.- Build LR(0) Automaton:

\[
\begin{align*}
S & \rightarrow S \\
S & \rightarrow E \\
E & \rightarrow Ac \\
A & \rightarrow a \\
E & \rightarrow ad \\
\end{align*}
\]

2.- Add lookahead sets:

\[
\begin{align*}
\text{LA (7, } A \rightarrow a) & = \{ c \} \\
\text{Disjoint. Grammar is LALR(1).} \\
\text{LA (7, } d) & = \{ d \}
\end{align*}
\]
Problems:

- Good diagnostics are difficult to produce.
- Required changes contort semantic processing.
- Remaining conflicts are few and difficult to correct.
- Many different lookahead algorithms.
- Systems are completely incompatible.
- No unifying algorithms.
LAR(M,C,L) Parsers

**Central Premise:** Lookahead information can be obtained by “simulating-ahead” the LR(0) parser’s moves, with three degrees of freedom:

- **M:** How much of the stack is to be simulated.
- **C:** Whether the context implied by the inconsistent state is to be taken into consideration during the simulation.
- **L:** The amount of lookahead, i.e. the duration of the simulation.
Strategy

Build one “Lookahead” FSA for each inconsistent LR(0) state.

![Diagram of a non-deterministic finite automaton (NFA)]
Notation:

• \([q:X_1 \ldots X_n]\) : a sequence of states:

\[
\begin{array}{cccc}
q & X_1 & X_2 & X_n \\
\rightarrow & \rightarrow & \rightarrow \\
\end{array}
\]

• Top\([q:X_1 \ldots X_n]\) denotes \(r\).
• \([X_1 \ldots X_n]\) denotes \([\text{Start}:X_1 \ldots X_n]\)
• \([\alpha]x:n\) is a “[stack]input:lookahead” configuration.

LAR(M,C,L) parser moves:

• Shift: \([\alpha]tz:0 \leftarrow [\alpha t]z:0\)
• Reduce: \([\alpha \omega]z:0 \leftarrow [\alpha A]z:0\)
• LA-Scan: \([\alpha]wtz:n \leftarrow [\alpha]wtz:n+1\)
• LA-Shift: \([\alpha]tz:n \leftarrow [\alpha t]z:0\)
• LA-Reduce: \([\alpha \omega]z:n \leftarrow [\alpha A]z:0\)
Example

LR(0):

LAA$_5$:

[1:]afebd $\downarrow$:0
$\vdash$ [1:a]febd $\downarrow$:0
$\vdash$ [1:a]febd $\downarrow$:1
$\vdash$ [1:a]febd $\downarrow$:2
$\vdash$ [1:a]febd $\downarrow$:3
$\vdash$ [1:a]febd $\downarrow$:4
$\vdash$ [1:af]ebd $\downarrow$:0
$\vdash$ [1:afe]bd $\downarrow$:0
$\vdash$ [1:afe]bd $\downarrow$:0
$\vdash$ [1:afeb]d $\downarrow$:0
$\vdash$ [1:afD]d $\downarrow$:0
$\vdash$ [1:afD]d $\downarrow$:0
$\vdash$ [1:X] $\downarrow$:0
$\vdash$ [1:S] $\downarrow$:0
Construction of Lookahead Automata

- “Simulate-ahead” LR(0) parser’s actions, starting at the conflicting state.

- Simulation keeps track of top (at most M) states of LR(0) parser’s stack (as best it can).

- Begin by “forcing” each conflicting action.

- Simulate all applicable moves:
  -- Reduction chain “collected” in LA state.
  -- Shift-moves are transitions in LA Automaton.

- Stop if L is exceeded, or if conflict is resolved.
Case I. L too small: LAA not reduced

LR(0):

1. S
   2. ↓S→X
   3. X
      4. A
         5. A→a
         6. ↑S→S ⊥
    7. D
       8. D
           9. e
              10. b
                 11. d
                     12. X→AfDc
                     13. X→afDd
    14. D→eb

LAR(5, true, 3)LAA₅:

[5:ε], f
[1:A], A→a

[5:f], f
[1:Af], A→a

[5:fe], f
[1:Afe], A→a

[5:feb], f
[1:Afeb], A→a

[5:fD], f
[1:AfD], A→a

[1:AfDc], A→a

[5:fDd], f
Case II. M too small: LAA not acyclic
Definition:
A grammar is $\text{LAR}(M,C,L)$ if all lookahead automata are **acyclic** and **reduced**, i.e. if final states are always reachable.

**Theorem:**

If $G$ is $\text{LAR}(M,C,L)$, then

$$L(\text{LR}(0)) = L(\text{LAR}(M,C,L)),$$

i.e. the parser is correct.

**Theorem:**

- $\text{LAR}(\infty,\text{true},k) \equiv \text{LALR}(k)$.
- $\text{LAR}(M,\text{true},k) \equiv \text{NQLALR}(k)$.
- $\text{LAR}(\infty,\text{false},k) \equiv \text{SLR}(k)$
- $\text{LAR}(M,\text{false},k) \equiv \text{NQSLR}(k)$. 
Grammar Class Relationships
Conclusions

- LAR(M,C,L) allows single and multiple symbol lookahead.

- Can obtain LALR(k), SLR(k), NQLALR(k), etc. via appropriate settings of M,C, and L.

- Can manipulate the parameters to suit the grammar, rather than the other way around.

- Can progressively attempt larger amounts of lookahead.

- Better understood relationships among grammar classes.

- Simultaneous availability of several techniques.