Parsing $LR(0)$ Grammars

Parsing $LR(0)$

- Our goal is now to build a deterministic parser automaton for an $LR(0)$ grammar that can very efficiently decide which actions (reduce or shift) it has to do to parse a given input

- For this, we will first build the *goto-automaton* (usually called the *goto function*). This automaton will later help us to build the parser automaton

Constructing the goto-automaton

- Example grammar: $S' \to S$ $\quad S \to B|C$
  $B \to aB|b$ $\quad C \to aC|c$
- The states of the goto-automaton tells us how far we have progressed in parsing the right-hand sides of rules
- In the initial state, we have not read anything so far (i.e., ε). Therefore, we are just at the beginning of the right-hand side of $S' \to S$. We write that as
  $$[S' \to \cdot S]$$
  This special notation is called an ***item***
- Since S can be expanded to B and C (and those to aB and b and aC and c) without reading any input, we also include those in the initial state:
  $\{[S' \to\cdot S], [S \to\cdot B], [S \to\cdot C], [B \to\cdot aB], [B \to\cdot b], [C \to\cdot aC], [C \to\cdot c]\}$
- The set $\{[S' \to\cdot S], [S \to\cdot B], [S \to\cdot C], [B \to\cdot aB], [B \to\cdot b], [C \to\cdot aC], [C \to\cdot c]\}$ is called the $LR(0)$ ***item set*** of ε, written as $LR(0)(\varepsilon)$

Constructing the goto-automaton, part 2

- Example grammar: $S' \to S \qquad S \to B|C$
  $B \to aB|b \qquad C \to aC|c$

- The goto-automaton reads terminal and non-terminal symbols
- Reading the non-terminal symbol C moves the automaton to a new state

Again, the dot $\cdot$ indicates how far we have progressed in reading the right-hand side

$$LR(0)(\varepsilon) = \left\{ \begin{matrix} [S' \to \cdot\, S], [S \to \cdot\, B], \\ [S \to \cdot\, C], [B \to \cdot\, aB], \\ [B \to \cdot\, b], [C \to \cdot\, aC], \\ [C \to \cdot\, c] \end{matrix} \right\} \xrightarrow{\text{C}} \{[S \to C\, \cdot]\}$$

- The set $\{[S \to C\, \cdot]\}$ is the $LR(0)$ item set of C, written as $LR(0)(C)$

Constructing the goto-automaton, part 3

- Example grammar: $S' \to S$ $\quad S \to B|C$ $\quad B \to aB|b$ $\quad C \to aC|c$
- We build the states for all possible inputs from the initial state

$$LR(0)(S) = \{[S' \to S\,\cdot]\}$$

$$LR(0)(\varepsilon) = \left\{\begin{matrix} [S' \to\cdot\, S], [S \to\cdot\, B], \\ [S \to\cdot\, C], [B \to\cdot\, aB], \\ [B \to\cdot\, b], [C \to\cdot\, aC], \\ [C \to\cdot\, c] \end{matrix}\right\}$$

$$LR(0)(\varepsilon) \xrightarrow{S} LR(0)(S) \qquad LR(0)(\varepsilon) \xrightarrow{B} LR(0)(B) \qquad LR(0)(\varepsilon) \xrightarrow{C} LR(0)(C)$$

$$LR(0)(\varepsilon) \xrightarrow{b} LR(0)(b) \qquad LR(0)(\varepsilon) \xrightarrow{c} LR(0)(c) \qquad LR(0)(\varepsilon) \xrightarrow{a} LR(0)(a)$$

$$LR(0)(B) = \{[S \to B\,\cdot]\}$$

$$LR(0)(C) = \{[S \to C\,\cdot]\}$$

$$LR(0)(b) = \{[B \to b\,\cdot]\}$$

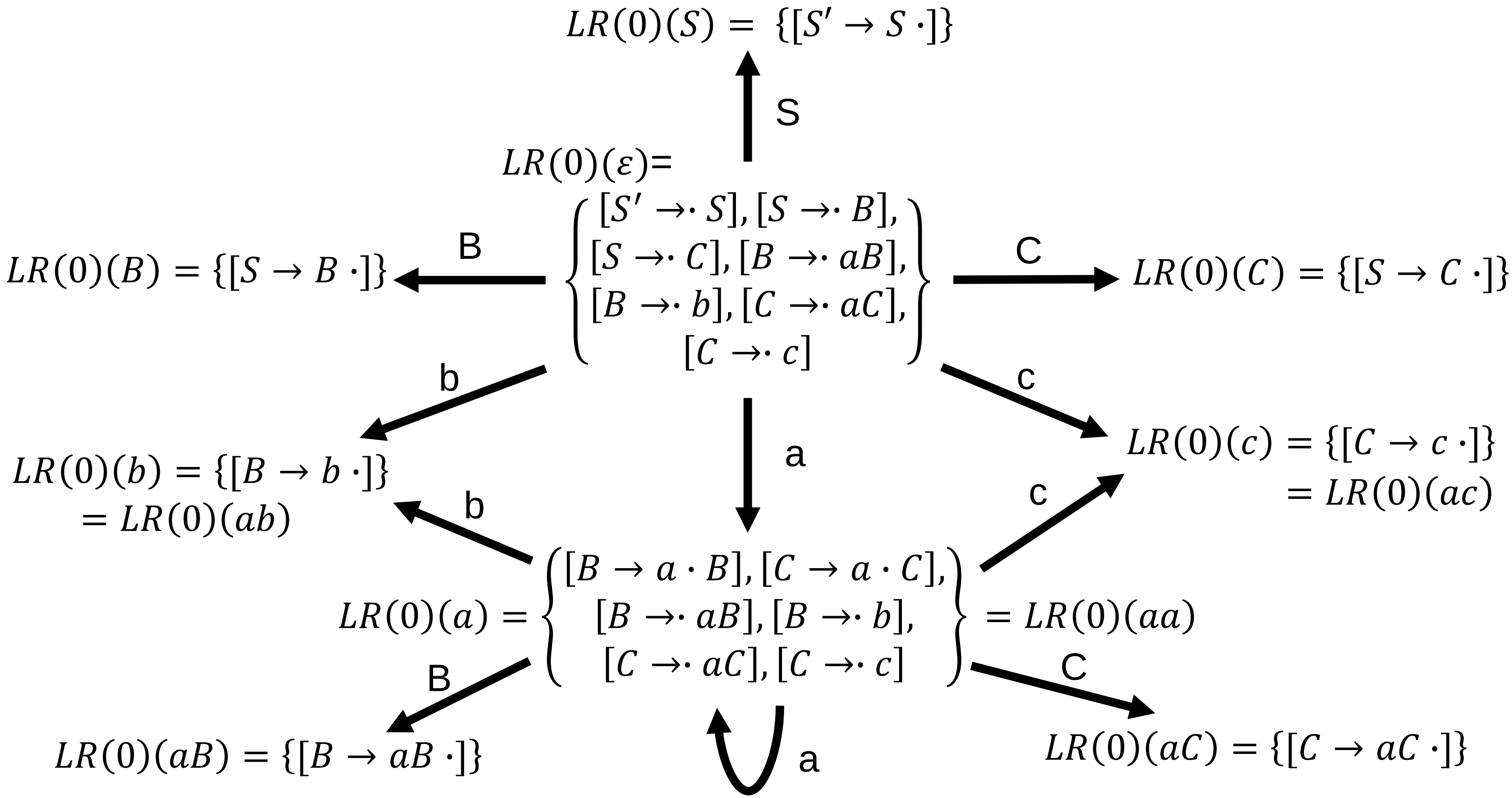
$$LR(0)(c)\ \{[C \to c\,\cdot]\}$$

$$LR(0)(a) = \left\{\begin{matrix} [B \to a \cdot B], [C \to a \cdot C], \\ [B \to\cdot\, aB], [B \to\cdot\, b], \\ [C \to\cdot\, aC], [C \to\cdot\, c] \end{matrix}\right\}$$

Again, we expand the first non-terminal following the dot ·

Constructing the goto-automaton, part 4

- We continue adding transitions and states until we are done
- Not shown in this picture: any other input γ will lead to a state $LR(0)(\gamma) = \{\}$



Item sets

- Formal definition of the *item set*:
  Given $G =< \Sigma, N, P, S >$ and a derivation $S \Rightarrow_r^* \alpha A w \Rightarrow_r \alpha\beta_1\beta_2 w$,
  $[A \rightarrow \beta_1 \cdot \beta_2]$ is in $LR(0)(\alpha\beta_1)$

- It can be proved (not in this course) that for an $LR(0)$ grammar it always hold:
  - The set $LR(0)(\gamma)$ for any $\gamma \in X^*$ is always finite
  - The set $LR(0)(G) = \{LR(0)(\gamma) \mid \gamma \in X^*\}$ is always finite

  This means that the goto-automaton has a finite number of states and each state is a finite item set

From the goto-automaton to a parser automaton

- As already said, the goto-automaton will help us to build a deterministic parser automaton for $LR(0)$
- Here is the idea:
  - An item like $[B \to aB\ \cdot]$ tells us that we are "ready for reduction" to B because all the symbols on the right-hand side of the rule have been parsed
  - An item like $[A \to \alpha_1 \cdot B\alpha_2]$ tells us that more input (shift or ε-reduction) is needed before a reduction to A can be done

Conflicts

- It can happen that we have conflicting information in the item sets:
  - If an item set contains two items $[A \rightarrow \alpha \cdot]$ and $[B \rightarrow \beta \cdot]$ then we have a **reduce/reduce** conflict
  - If an item set contains two items $[A \rightarrow \alpha_1 \cdot a\alpha_2]$ and $[B \rightarrow \beta \cdot]$ then we have a **shift/reduce** conflict

- **Important lemma:** A grammar G is $LR(0)$ if and only if no item set in $LR(0)(G)$ contains conflicting items

Example for Shift-Reduce Conflict

- Grammar: $S' \to S$
  $S \to aA$
  $A \to S \mid \varepsilon$
- Note that this grammar is unambiguous and $LL(1)$!

$$\begin{tikzcd}
LR(0)(S) = \{[S' \to S\,\cdot]\} \\
LR(0)(\varepsilon) = \{[S' \to \cdot\, S], [S \to \cdot\, aA]\} \arrow[u, "S"] \arrow[d, "a"] \\
LR(0)(a) = \{[S \to a \cdot A], [A \to \cdot\, S], [A \to \cdot], [S \to \cdot\, aA]\} \arrow[d, "aA"] \\
LR(0)(aA) = \{[S \to aA\,\cdot]\}
\end{tikzcd}$$

Conflict

Example for Reduce-Reduce Conflict

- Grammar: $S' \rightarrow S$
  $S \rightarrow Aa \mid Bb$
  $A \rightarrow a$
  $B \rightarrow a$
- Note that this grammar is unambiguous (but not $LL(1)$)

- $LR(0)(\varepsilon) = \{[S' \rightarrow \cdot\, S], [S \rightarrow \cdot\, Aa], [S \rightarrow \cdot\, Bb], [A \rightarrow \cdot\, a], [B \rightarrow \cdot\, a]\}$
- $LR(0)(a) = \{[A \rightarrow a\, \cdot], [B \rightarrow a\, \cdot]\}$

Conflict

Deterministic Parsing Automaton $LR(0)$

- We define the deterministic parsing automaton of $LR(0)$ grammar $G = < \Sigma, N, P, S >$ with rule 0: $S' \to S$
  - Input alphabet Σ
  - Output alphabet U = the rule numbers 0,1,2,3,...
  - States $\Sigma^* \times \Gamma^* \times U^*$
  - Pushdown alphabet $\Gamma = LR(0)(G)$!! item sets !!
  - Initial state (w, I_0, ε) for $w \in \Sigma^*$ where $I_0 = LR(0)(\varepsilon)$
  - Final state final state $(\varepsilon, \varepsilon, u)$ where $u \in U^*$
  - Action depends on the item set I in the state $(x, \alpha I, z)$:
    - **Shift** $(aw, \alpha I, z) \to (w, \alpha IJ, z)$ if $[A \to \alpha_1 \cdot a\alpha_2] \in I$ and $I \underset{goto}{\overset{a}{\to}} J$
    - **Reduce** $(w, \alpha I I_1 \dots I_n, z) \to (w, \alpha IJ, zi)$ with rule $i \neq 0$ $A \to Y_1 \dots Y_n$ if $[A \to Y_1 \dots Y_n \cdot] \in I_n$ and $I \underset{goto}{\overset{A}{\to}} J$
    - **Accepting** state $(\varepsilon, I_0 I, z) \to (\varepsilon, \varepsilon, z0)$ if $[S' \to S \cdot] \in I$
    - **Error** in state $(w, \alpha I, z)$ if $I = \emptyset$