



- ## LR(0) parsing
- Construct transition relation between states
 - Use algorithms **Initial item set** and **Next item set**
 - States are set of LR(0) items
 - Shift items of the form $P \rightarrow \alpha \bullet S \beta$
 - Reduce items of the form $P \rightarrow \alpha \bullet$
 - Construct parsing table
 - If every state contains no conflicts use LR(0) parsing algorithm
 - If states contain conflict
 - Rewrite grammar or
 - Resolve conflict or
 - Use stronger parsing technique

ϵ closure algorithm for LR(0) item sets for a grammar G

Data definitions:
A set S of LR(0) items.

Initializations:
S is prefilled externally with one or more LR(0) items.

Inference rules:
If S holds an item of the form $P \rightarrow \alpha \bullet N \beta$, then for each production rule $N \rightarrow \gamma$ in G, S must also contain the item $N \rightarrow \bullet \gamma$.

p. 155

FUNCTION Initial item set RETURNING an item set:
SET New item set TO Empty;
// Initial contents – obtain from the start symbol;
FOR EACH production rule $S \rightarrow \alpha$ for the start symbol S;
SET New item set TO New item set + item $S \rightarrow \bullet \alpha$;
RETURN ϵ closure (New item set);

p. 158

FUNCTION Next item set (Item set, Symbol) RETURNING an item set:
SET New item set TO Empty;
// Initial contents – obtain from token moves;
FOR EACH item $N \rightarrow \alpha \bullet S \beta$ IN item set;
IF S = Symbol;
SET New item set TO New item set + $N \rightarrow \alpha S \bullet \beta$;
RETURN ϵ closure (New item set);

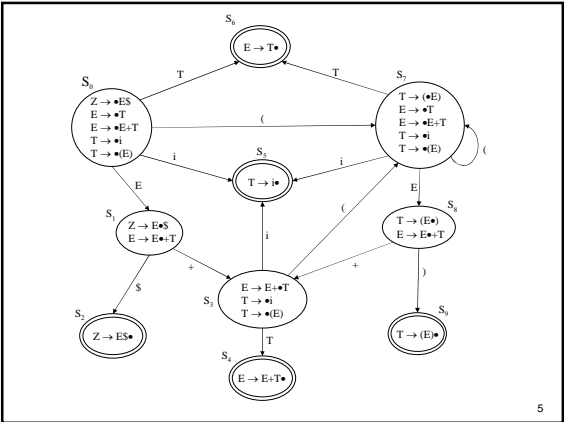
p. 158

Example: LR(0) for grammar G

Grammar G:
 $Z \rightarrow ES$
 $E \rightarrow T$
 $E \rightarrow E + T$
 $T \rightarrow i$
 $T \rightarrow (E)$

Convention:
 non-terminals denoted by upper-case letters
 terminals denoted by lower-case letters

Precomputed LR(0) items:
 1: $S \rightarrow \bullet ES$
 2: $S \rightarrow E \bullet S$
 3: $S \rightarrow ES \bullet$
 4: $E \rightarrow \bullet T$
 5: $E \rightarrow T \bullet$
 6: $E \rightarrow \bullet E + T$
 7: $E \rightarrow E \bullet + T$
 8: $E \rightarrow E + \bullet T$
 9: $E \rightarrow E + T \bullet$
 10: $T \rightarrow \bullet i$
 11: $T \rightarrow i \bullet$
 12: $T \rightarrow \bullet (E)$
 13: $T \rightarrow (\bullet E)$
 14: $T \rightarrow (E \bullet)$
 15: $T \rightarrow (E) \bullet$



| state | GOTO table symbol | | | | | | | ACTION table |
|-------|-------------------|---|---|---|----|---|---|--------------|
| | i | + | (|) | \$ | E | T | |
| 0 | 5 | | 7 | | | 1 | 6 | shift |
| 1 | | 3 | | | 2 | | | shift |
| 2 | Z → ES | | | | | | | reduce |
| 3 | 5 | | 7 | | | | 4 | shift |
| 4 | E → E + T | | | | | | | reduce |
| 5 | T → i | | | | | | | reduce |
| 6 | E → T | | | | | | | reduce |
| 7 | 5 | | 7 | | | 8 | 6 | shift |
| 8 | | 3 | | | 9 | | | shift |
| 9 | T → (E) | | | | | | | reduce |

LR(0) Parsing with a push-down automaton

```

IMPORT input token [1..]: // from the lexical analyzer
SET Input token index TO 1;
SET Reduction stack TO Empty stack;
PUSH Start State ON Reduction stack;

While Reduction stack ≠ {Start state, Start symbol, End state}
  SET State TO Top of Reduction stack;
  SET Action TO Action table [State];
  IF Action = "shift":
    // Do a shift move:
    SET shifted token TO Input token [Input token index];
    SET Input token index TO Input token index + 1; // shifted
    PUSH Shifted token ON Reduction stack;
    Set New State TO Goto table [State, Shifted token .class];
    PUSH New state ON Reduction stack; // can be empty
  ELSE IF Action = ("reduce", N→α) :
    // Do a reduction move:
    Pop the symbols of α from the Reduction stack;
    SET State TO Top of Reduction stack; // update state
    PUSH N ON Reduction stack;
    SET New state TO Goto table [State, N]
    PUSH New State ON Reduction stack; // cannot be empty
  ELSE Action = Empty:
    ERROR "Error at token ", Input token [Input token index];

```

p. 162 7

LR(0) parsing of i+i

| Stack | Input | Action |
|-------------------------|----------|-------------------------------|
| S_0 | i + i \$ | shift |
| $S_0 i S_5$ | + i \$ | reduce by $T \rightarrow i$ |
| $S_0 T S_6$ | + i \$ | reduce by $E \rightarrow T$ |
| $S_0 E S_1$ | + i \$ | shift |
| $S_0 E S_1 + S$ | i \$ | shift |
| $S_0 E S_1 + S_3 i S_3$ | \$ | reduce by $T \rightarrow i$ |
| $S_0 E S_1 + S_3 T S_4$ | \$ | reduce by $E \rightarrow E+T$ |
| $S_0 E S_1$ | \$ | shift |
| $S_0 E S_1 \$ S_2$ | | reduce by $Z \rightarrow ES$ |
| $S_0 Z$ | | stop |

8