LR Parsing

Compiler Design

CSE 504

1. Shift-Reduce Parsing
2. LR Parsers
3. SLR and LR(1) Parsers

Last modified: Fri Mar 06 2015 at 13:50:06 EST
Version: 1.7 16:58:46 2016/01/29
Compiled at 12:57 on 2016/02/26

Leftmost and Rightmost Derivations

E	$\longrightarrow$	$E + T$
E	$\longrightarrow$	T
T	$\longrightarrow$	`id`

Derivations for `id + id`:

Leftmost and Rightmost Derivations

E	$\longrightarrow$	$E + T$
E	$\longrightarrow$	T
T	$\longrightarrow$	`id`

Derivations for `id + id`:

$$E \implies E + T$$

Leftmost and Rightmost Derivations

E	$\longrightarrow$	$E + T$
E	$\longrightarrow$	T
T	$\longrightarrow$	`id`

Derivations for `id + id`:

$$\begin{aligned} E &\Longrightarrow E + T \\ &\Longrightarrow T + T \end{aligned}$$

Leftmost and Rightmost Derivations

E	$\longrightarrow$	$E+T$
E	$\longrightarrow$	T
T	$\longrightarrow$	id

Derivations for id + id:

$$\begin{aligned} E &\Longrightarrow E+T \\ &\Longrightarrow T+T \\ &\Longrightarrow \mathtt{id}+T \end{aligned}$$

Leftmost and Rightmost Derivations

E	$\longrightarrow$	$E+T$
E	$\longrightarrow$	T
T	$\longrightarrow$	id

Derivations for id + id:

$$
\begin{aligned}
E &\Longrightarrow E+T \\
&\Longrightarrow T+T \\
&\Longrightarrow \texttt{id}+T \\
&\Longrightarrow \texttt{id+id}
\end{aligned}
$$

Leftmost and Rightmost Derivations

E	$\longrightarrow$	$E+T$
E	$\longrightarrow$	T
T	$\longrightarrow$	id

Derivations for id + id:

$$\begin{aligned} E &\Longrightarrow E+T \\ &\Longrightarrow T+T \\ &\Longrightarrow \mathtt{id}+T \\ &\Longrightarrow \mathtt{id}+\mathtt{id} \end{aligned}$$

$$E \Longrightarrow E+T$$

Leftmost and Rightmost Derivations

E	$\longrightarrow$	$E+T$
E	$\longrightarrow$	T
T	$\longrightarrow$	id

Derivations for id + id:

$$\begin{aligned} E &\Longrightarrow E+T \\ &\Longrightarrow T+T \\ &\Longrightarrow \mathtt{id}+T \\ &\Longrightarrow \mathtt{id}+\mathtt{id} \end{aligned}$$

$$\begin{aligned} E &\Longrightarrow E+T \\ &\Longrightarrow E+\mathtt{id} \end{aligned}$$

Leftmost and Rightmost Derivations

E	$\longrightarrow$	$E+T$
E	$\longrightarrow$	T
T	$\longrightarrow$	id

Derivations for id + id:

$$
\begin{array}{lll|lll}
E & \Longrightarrow & E+T & E & \Longrightarrow & E+T \\
 & \Longrightarrow & T+T & & \Longrightarrow & E+\mathtt{id} \\
 & \Longrightarrow & \mathtt{id}+T & & \Longrightarrow & T+\mathtt{id} \\
 & \Longrightarrow & \mathtt{id}+\mathtt{id} & & &
\end{array}
$$

Leftmost and Rightmost Derivations

E	$\longrightarrow$	$E+T$
E	$\longrightarrow$	T
T	$\longrightarrow$	id

Derivations for id + id:

$$
\begin{array}{lll|lll}
E & \Longrightarrow & E+T & E & \Longrightarrow & E+T \\
 & \Longrightarrow & T+T & & \Longrightarrow & E+\mathtt{id} \\
 & \Longrightarrow & \mathtt{id}+T & & \Longrightarrow & T+\mathtt{id} \\
 & \Longrightarrow & \mathtt{id}+\mathtt{id} & & \Longrightarrow & \mathtt{id}+\mathtt{id}
\end{array}
$$

Leftmost and Rightmost Derivations

$$\begin{array}{lcl} E & \longrightarrow & E\texttt{+}T \\ E & \longrightarrow & T \\ T & \longrightarrow & \texttt{id} \end{array}$$

Derivations for `id + id`:

$$\begin{array}{lcl|lcl} E & \Longrightarrow & E\texttt{+}T & E & \Longrightarrow & E\texttt{+}T \\ & \Longrightarrow & T\texttt{+}T & & \Longrightarrow & E\texttt{+id} \\ & \Longrightarrow & \texttt{id+}T & & \Longrightarrow & T\texttt{+id} \\ & \Longrightarrow & \texttt{id+id} & & \Longrightarrow & \texttt{id+id} \end{array}$$

LEFTMOST

Leftmost and Rightmost Derivations

E	$\longrightarrow$	E+T
E	$\longrightarrow$	T
T	$\longrightarrow$	id

Derivations for id + id:

LEFTMOST	RIGHTMOST
$E \Longrightarrow E+T$	$E \Longrightarrow E+T$
$\Longrightarrow T+T$	$\Longrightarrow E+\texttt{id}$
$\Longrightarrow \texttt{id}+T$	$\Longrightarrow T+\texttt{id}$
$\Longrightarrow \texttt{id+id}$	$\Longrightarrow \texttt{id+id}$

Bottom-up Parsing

Given a stream of tokens w, *reduce* it to the start symbol.

E	$\longrightarrow$	E+T
E	$\longrightarrow$	T
T	$\longrightarrow$	`id`

Parse input stream: `id + id`:

Bottom-up Parsing

Given a stream of tokens w, *reduce* it to the start symbol.

E	$\longrightarrow$	E+T
E	$\longrightarrow$	T
T	$\longrightarrow$	`id`

Parse input stream: `id + id`:

`id + id`

Bottom-up Parsing

Given a stream of tokens w, *reduce* it to the start symbol.

E	$\longrightarrow$	E+T
E	$\longrightarrow$	T
T	$\longrightarrow$	id

Parse input stream: id + id:

id + id

T + id

Bottom-up Parsing

Given a stream of tokens w, *reduce* it to the start symbol.

E	$\longrightarrow$	E+T
E	$\longrightarrow$	T
T	$\longrightarrow$	`id`

Parse input stream: `id + id`:

`id + id`

T `+ id`

E `+ id`

Bottom-up Parsing

Given a stream of tokens w, *reduce* it to the start symbol.

E	$\longrightarrow$	E+T
E	$\longrightarrow$	T
T	$\longrightarrow$	id

Parse input stream: id + id:

id + id
T + id
E + id
E + T

Bottom-up Parsing

Given a stream of tokens w, *reduce* it to the start symbol.

E	$\longrightarrow$	E+T
E	$\longrightarrow$	T
T	$\longrightarrow$	id

Parse input stream: id + id:

id + id
T + id
E + id
E + T
E

Bottom-up Parsing

Given a stream of tokens w, *reduce* it to the start symbol.

E	$\longrightarrow$	E+T
E	$\longrightarrow$	T
T	$\longrightarrow$	`id`

Parse input stream: `id + id`:

`id + id`
T + `id`
E + `id`
E + T
E

Reduction $\equiv$ Derivation^{-1}.

Shift-Reduce Parsing: An Example

E	$\longrightarrow$	$E + T$
E	$\longrightarrow$	T
T	$\longrightarrow$	id

Stack	Input Stream	Action

Shift-Reduce Parsing: An Example

$$
\begin{array}{lcl}
E & \longrightarrow & E+T \\
E & \longrightarrow & T \\
T & \longrightarrow & \texttt{id}
\end{array}
$$

Stack	Input Stream	Action
$	id + id $	

Shift-Reduce Parsing: An Example

E	$\longrightarrow$	$E + T$
E	$\longrightarrow$	T
T	$\longrightarrow$	id

Stack	Input Stream	Action
$	id + id $	**shift**

Shift-Reduce Parsing: An Example

E	$\longrightarrow$	E+T
E	$\longrightarrow$	T
T	$\longrightarrow$	id

Stack	Input Stream	Action
$	id + id $	**shift**
$ id	+ id $	

Shift-Reduce Parsing: An Example

E	$\longrightarrow$	E+T
E	$\longrightarrow$	T
T	$\longrightarrow$	id

Stack	Input Stream	Action
$	id + id $	**shift**
$ id	+ id $	**reduce by** $T \longrightarrow$ id

Shift-Reduce Parsing: An Example

E	$\longrightarrow$	E+T
E	$\longrightarrow$	T
T	$\longrightarrow$	id

Stack	Input Stream	Action
\$	id + id \$	**shift**
\$ id	+ id \$	**reduce by** $T \longrightarrow$ id
\$ T	+ id \$	

Shift-Reduce Parsing: An Example

E	$\longrightarrow$	E+T
E	$\longrightarrow$	T
T	$\longrightarrow$	id

STACK	INPUT STREAM	ACTION
\$	id + id \$	**shift**
\$ id	+ id \$	**reduce by** $T \longrightarrow$ id
\$ T	+ id \$	reduce by $E \longrightarrow T$
\$ E	+ id \$	

Shift-Reduce Parsing: An Example

E	$\longrightarrow$	E+T
E	$\longrightarrow$	T
T	$\longrightarrow$	id

Stack	Input Stream	Action
\$	id + id \$	**shift**
\$ id	+ id \$	**reduce by** $T \longrightarrow$ id
\$ T	+ id \$	reduce by $E \longrightarrow T$
\$ E	+ id \$	shift
\$ E +	id \$	

Shift-Reduce Parsing: An Example

E	$\longrightarrow$	E+T
E	$\longrightarrow$	T
T	$\longrightarrow$	id

Stack	Input Stream	Action
\$	id + id \$	**shift**
\$ id	+ id \$	**reduce by** $T \longrightarrow$ id
\$ T	+ id \$	reduce by $E \longrightarrow T$
\$ E	+ id \$	shift
\$ E +	id \$	shift
\$ E + id	\$	

Shift-Reduce Parsing: An Example

E	$\longrightarrow$	E+T
E	$\longrightarrow$	T
T	$\longrightarrow$	id

Stack	Input Stream	Action
\$	id + id \$	**shift**
\$ id	+ id \$	**reduce by** $T \longrightarrow$ id
\$ T	+ id \$	reduce by $E \longrightarrow T$
\$ E	+ id \$	shift
\$ E +	id \$	shift
\$ E + id	\$	reduce by $T \longrightarrow$ id
\$ E + T	\$	

Shift-Reduce Parsing: An Example

E	$\longrightarrow$	$E+T$
E	$\longrightarrow$	T
T	$\longrightarrow$	id

Stack	Input Stream	Action
\$	id + id \$	**shift**
\$ id	+ id \$	**reduce by** $T \longrightarrow$ id
\$ T	+ id \$	reduce by $E \longrightarrow T$
\$ E	+ id \$	shift
\$ E +	id \$	shift
\$ E + id	\$	reduce by $T \longrightarrow$ id
\$ E + T	\$	reduce by $E \longrightarrow E+T$
\$ E	\$	

Shift-Reduce Parsing: An Example

E	$\longrightarrow$	$E+T$
E	$\longrightarrow$	T
T	$\longrightarrow$	id

Stack	Input Stream	Action
\$	id + id \$	**shift**
\$ id	+ id \$	**reduce by** $T \longrightarrow$ id
\$ T	+ id \$	reduce by $E \longrightarrow T$
\$ E	+ id \$	shift
\$ E +	id \$	shift
\$ E + id	\$	reduce by $T \longrightarrow$ id
\$ E + T	\$	reduce by $E \longrightarrow E+T$
\$ E	\$	**ACCEPT**

Handles

"A structure that furnishes a means to perform reductions"

E	$\longrightarrow$	E+T
E	$\longrightarrow$	T
T	$\longrightarrow$	id

Parse input stream: id + id:

Handles

"A structure that furnishes a means to perform reductions"

E	$\longrightarrow$	E+T
E	$\longrightarrow$	T
T	$\longrightarrow$	id

Parse input stream: id + id:

$\boxed{\texttt{id}}$ + id

Handles

"A structure that furnishes a means to perform reductions"

E	$\longrightarrow$	E+T
E	$\longrightarrow$	T
T	$\longrightarrow$	id

Parse input stream: id + id:

$\boxed{\texttt{id}}$ + id

$\boxed{T}$ + id

Handles

"A structure that furnishes a means to perform reductions"

E	$\longrightarrow$	E+T
E	$\longrightarrow$	T
T	$\longrightarrow$	id

Parse input stream: id + id:

$\boxed{\texttt{id}}$ + id

$\boxed{T}$ + id

E + $\boxed{\texttt{id}}$

Handles

"A structure that furnishes a means to perform reductions"

E	$\longrightarrow$	E+T
E	$\longrightarrow$	T
T	$\longrightarrow$	id

Parse input stream: id + id:

$\boxed{\texttt{id}}$ + id

$\boxed{T}$ + id

E + $\boxed{\texttt{id}}$

$\boxed{E + T}$

Handles

"A structure that furnishes a means to perform reductions"

E	$\longrightarrow$	E+T
E	$\longrightarrow$	T
T	$\longrightarrow$	id

Parse input stream: id + id:

$\boxed{\texttt{id}}$ + id

$\boxed{T}$ + id

E + $\boxed{\texttt{id}}$

$\boxed{E + T}$

E

Handles

Handles are substrings of sentential forms:

1. A substring that matches the right hand side of a production
2. Reduction using that rule can lead to the start symbol
3. The rule forms one step in a rightmost derivation of the string

Handles

Handles are substrings of sentential forms:

1. A substring that matches the right hand side of a production
2. Reduction using that rule can lead to the start symbol
3. The rule forms one step in a rightmost derivation of the string

$$E \implies \boxed{E + T}$$

Handles

Handles are substrings of sentential forms:

1. A substring that matches the right hand side of a production
2. Reduction using that rule can lead to the start symbol
3. The rule forms one step in a rightmost derivation of the string

$$
\begin{aligned}
E &\Longrightarrow \boxed{E + T} \\
&\Longrightarrow E + \boxed{\texttt{id}}
\end{aligned}
$$

Handles

Handles are substrings of sentential forms:

1. A substring that matches the right hand side of a production
2. Reduction using that rule can lead to the start symbol
3. The rule forms one step in a rightmost derivation of the string

$$
\begin{aligned}
E &\Longrightarrow \boxed{E + T} \\
&\Longrightarrow E + \boxed{\texttt{id}} \\
&\Longrightarrow \boxed{T} + \texttt{id}
\end{aligned}
$$

Handles

Handles are substrings of sentential forms:

1. A substring that matches the right hand side of a production
2. Reduction using that rule can lead to the start symbol
3. The rule forms one step in a rightmost derivation of the string

$$
\begin{aligned}
E &\Longrightarrow \boxed{E + T} \\
&\Longrightarrow E + \boxed{\texttt{id}} \\
&\Longrightarrow \boxed{T} + \texttt{id} \\
&\Longrightarrow \boxed{\texttt{id}} + \texttt{id}
\end{aligned}
$$

Handles

Handles are substrings of sentential forms:

1. A substring that matches the right hand side of a production
2. Reduction using that rule can lead to the start symbol
3. The rule forms one step in a rightmost derivation of the string

$$
\begin{aligned}
E &\Longrightarrow \boxed{E + T} \\
&\Longrightarrow E + \boxed{\texttt{id}} \\
&\Longrightarrow \boxed{T} + \texttt{id} \\
&\Longrightarrow \boxed{\texttt{id}} + \texttt{id}
\end{aligned}
$$

Handle Pruning: replace handle by corresponding LHS.

Shift-Reduce Parsing

Bottom-up parsing

- **Shift:** Construct leftmost handle on top of stack
- **Reduce:** Identify handle and replace by corresponding RHS
- **Accept:** Continue until string is reduced to start symbol and input token stream is empty
- **Error:** Signal parse error if no handle is found.

Implementing Shift-Reduce Parsers

- Stack to hold grammar symbols (corresponding to tokens seen thus far).
- Input stream of yet-to-be-seen tokens.
- Handles appear on top of stack.
- Stack is initially empty (denoted by $).
- Parse is successful if stack contains only the start symbol when the input stream ends.

Preparing for Shift-Reduce Parsing

1. Identify a handle in string.
   Top of stack is the *rightmost* end of the handle. What is the leftmost end?

Preparing for Shift-Reduce Parsing

1. Identify a handle in string.
   Top of stack is the *rightmost* end of the handle. What is the leftmost end?
2. If there are multiple productions with the handle on the RHS, which one to choose?

Preparing for Shift-Reduce Parsing

1. Identify a handle in string.
   Top of stack is the *rightmost* end of the handle. What is the leftmost end?
2. If there are multiple productions with the handle on the RHS, which one to choose?

Preparing for Shift-Reduce Parsing

1. Identify a handle in string.
   Top of stack is the *rightmost* end of the handle. What is the leftmost end?
2. If there are multiple productions with the handle on the RHS, which one to choose?

Construct a parsing table, just as in the case of LL(1) parsing.

Shift-Reduce Parsing: Derivations

Stack	Input Stream	Action
$	id + id $	shift
$ id	+ id $	reduce by $T \longrightarrow$ id
$ T	+ id $	reduce by $E \longrightarrow T$
$ E	+ id $	shift
$ E +	id $	shift
$ E + id	$	reduce by $T \longrightarrow$ id
$ E + T	$	reduce by $E \longrightarrow E+T$
$ E	$	**ACCEPT**

Left to Right Scan of input

Rightmost Derivation in reverse.

A Simple Example of LR Parsing

S	$\longrightarrow$	BC
B	$\longrightarrow$	a
C	$\longrightarrow$	a

Stack	Input Stream	Action
$	a a $	

A Simple Example of LR Parsing

S	$\longrightarrow$	BC
B	$\longrightarrow$	a
C	$\longrightarrow$	a

Stack	Input Stream	Action
\$	a a \$	shift
\$ a	a \$	

A Simple Example of LR Parsing

S	$\longrightarrow$	BC
B	$\longrightarrow$	a
C	$\longrightarrow$	a

Stack	Input Stream	Action
\$	a a \$	shift
\$ a	a \$	reduce by $B \longrightarrow$ a
\$ B	a \$	

A Simple Example of LR Parsing

S	$\longrightarrow$	BC
B	$\longrightarrow$	a
C	$\longrightarrow$	a

Stack	Input Stream	Action
\$	a a \$	shift
\$ a	a \$	reduce by $B \longrightarrow$ a
\$ B	a \$	shift
\$ B a	\$	

A Simple Example of LR Parsing

S	$\longrightarrow$	BC
B	$\longrightarrow$	a
C	$\longrightarrow$	a

Stack	Input Stream	Action
\$	a a \$	shift
\$ a	a \$	reduce by $B \longrightarrow$ a
\$ B	a \$	shift
\$ B a	\$	reduce by $C \longrightarrow$ a
\$ B C	\$	

A Simple Example of LR Parsing

S	$\longrightarrow$	BC
B	$\longrightarrow$	a
C	$\longrightarrow$	a

Stack	Input Stream	Action
\$	a a \$	shift
\$ a	a \$	reduce by $B \longrightarrow$ a
\$ B	a \$	shift
\$ B a	\$	reduce by $C \longrightarrow$ a
\$ B C	\$	reduce by $S \longrightarrow BC$
\$ S	\$	**ACCEPT**

A Simple Example of LR Parsing: A Detailed Look

S'	$\longrightarrow$	S	
S	$\longrightarrow$	B	C
B	$\longrightarrow$	a	
C	$\longrightarrow$	a	

STACK	INPUT	STATE	ACTION
$	a a $		

A Simple Example of LR Parsing: A Detailed Look

S'	$\longrightarrow$	S
S	$\longrightarrow$	$B\ C$
B	$\longrightarrow$	a
C	$\longrightarrow$	a

STACK	INPUT	STATE	ACTION
		$S' \longrightarrow \bullet\ S$	
\$	a a \$		

A Simple Example of LR Parsing: A Detailed Look

S'	$\longrightarrow$	S
S	$\longrightarrow$	$B\ C$
B	$\longrightarrow$	a
C	$\longrightarrow$	a

STACK	INPUT	STATE	ACTION
		$S' \longrightarrow \bullet\ S$	
\$	a a \$	$S \longrightarrow \bullet\ BC$	

A Simple Example of LR Parsing: A Detailed Look

S'	$\longrightarrow$	S
S	$\longrightarrow$	$B\ C$
B	$\longrightarrow$	a
C	$\longrightarrow$	a

STACK	INPUT	STATE	ACTION
		$S' \longrightarrow \bullet\ S$	
\$	a a \$	$S \longrightarrow \bullet\ BC$	
		$B \longrightarrow \bullet$ a	

A Simple Example of LR Parsing: A Detailed Look

S'	$\longrightarrow$	S
S	$\longrightarrow$	$B\ C$
B	$\longrightarrow$	a
C	$\longrightarrow$	a

STACK	INPUT	STATE	ACTION
$	a a $	$S' \longrightarrow \bullet\ S$ $S \longrightarrow \bullet\ BC$ $B \longrightarrow \bullet$ a	shift
$ a	a $		

A Simple Example of LR Parsing: A Detailed Look

S'	$\longrightarrow$	S
S	$\longrightarrow$	$B\ C$
B	$\longrightarrow$	a
C	$\longrightarrow$	a

STACK	INPUT	STATE	ACTION
$	a a $	$S' \longrightarrow \bullet\ S$ $S \longrightarrow \bullet\ BC$ $B \longrightarrow \bullet$ a	shift
$ a	a $	$B \longrightarrow$ a$\bullet$	

A Simple Example of LR Parsing: A Detailed Look

S'	$\longrightarrow$	S
S	$\longrightarrow$	$B\ C$
B	$\longrightarrow$	a
C	$\longrightarrow$	a

STACK	INPUT	STATE	ACTION
\$	a a \$	$S' \longrightarrow \bullet\ S$ $S \longrightarrow \bullet\ BC$ $B \longrightarrow \bullet$ a	shift
\$ a	a \$	$B \longrightarrow$ a$\bullet$	reduce by 3
\$ B	a \$		

A Simple Example of LR Parsing: A Detailed Look

S'	$\longrightarrow$	S
S	$\longrightarrow$	$B\ C$
B	$\longrightarrow$	a
C	$\longrightarrow$	a

Stack	Input	State	Action
\$	a a \$	$S' \longrightarrow \bullet\ S$ $S \longrightarrow \bullet\ BC$ $B \longrightarrow \bullet$ a	shift
\$ a	a \$	$B \longrightarrow$ a$\bullet$	reduce by 3
\$ B	a \$	$S \longrightarrow B\bullet\ C$	

A Simple Example of LR Parsing: A Detailed Look

S'	$\longrightarrow$	S
S	$\longrightarrow$	$B\ C$
B	$\longrightarrow$	a
C	$\longrightarrow$	a

STACK	INPUT	STATE	ACTION
\$	a a \$	$S' \longrightarrow \bullet\ S$ $S \longrightarrow \bullet\ BC$ $B \longrightarrow \bullet$ a	shift
\$ a	a \$	$B \longrightarrow$ a$\bullet$	reduce by 3
\$ B	a \$	$S \longrightarrow B\bullet\ C$ $C \longrightarrow \bullet$ a	

A Simple Example of LR Parsing: A Detailed Look

S'	$\longrightarrow$	S
S	$\longrightarrow$	$B\ C$
B	$\longrightarrow$	a
C	$\longrightarrow$	a

STACK	INPUT	STATE	ACTION
\$	a a \$	$S' \longrightarrow \bullet\ S$ $S \longrightarrow \bullet\ BC$ $B \longrightarrow \bullet$ a	shift
\$ a	a \$	$B \longrightarrow$ a$\bullet$	reduce by 3
\$ B	a \$	$S \longrightarrow B\bullet\ C$ $C \longrightarrow \bullet$ a	shift
\$ B a	\$		

A Simple Example of LR Parsing: A Detailed Look

S'	$\longrightarrow$	S
S	$\longrightarrow$	$B\ C$
B	$\longrightarrow$	a
C	$\longrightarrow$	a

STACK	INPUT	STATE	ACTION
\$	a a \$	$S' \longrightarrow \bullet\ S$ $S \longrightarrow \bullet\ BC$ $B \longrightarrow \bullet$ a	shift
\$ a	a \$	$B \longrightarrow$ a$\bullet$	reduce by 3
\$ B	a \$	$S \longrightarrow B\bullet\ C$ $C \longrightarrow \bullet$ a	shift
\$ B a	\$	$C \longrightarrow$ a$\bullet$	

A Simple Example of LR Parsing: A Detailed Look

S'	$\longrightarrow$	S
S	$\longrightarrow$	$B\ C$
B	$\longrightarrow$	a
C	$\longrightarrow$	a

Stack	Input	State	Action
\$	a a \$	$S' \longrightarrow \bullet\ S$ $S \longrightarrow \bullet\ BC$ $B \longrightarrow \bullet$ a	shift
\$ a	a \$	$B \longrightarrow$ a$\bullet$	reduce by 3
\$ B	a \$	$S \longrightarrow B\bullet\ C$ $C \longrightarrow \bullet$ a	shift
\$ B a	\$	$C \longrightarrow$ a$\bullet$	reduce by 4
\$ B C	\$		

A Simple Example of LR Parsing: A Detailed Look

S'	$\longrightarrow$	S
S	$\longrightarrow$	$B\ C$
B	$\longrightarrow$	a
C	$\longrightarrow$	a

Stack	Input	State	Action
\$	a a \$	$S' \longrightarrow \bullet\ S$ $S \longrightarrow \bullet\ BC$ $B \longrightarrow \bullet$ a	shift
\$ a	a \$	$B \longrightarrow$ a$\bullet$	reduce by 3
\$ B	a \$	$S \longrightarrow B\bullet\ C$ $C \longrightarrow \bullet$ a	shift
\$ B a	\$	$C \longrightarrow$ a$\bullet$	reduce by 4
\$ $B\ C$	\$	$S \longrightarrow BC\bullet$	

A Simple Example of LR Parsing: A Detailed Look

S'	$\longrightarrow$	S
S	$\longrightarrow$	$B\ C$
B	$\longrightarrow$	a
C	$\longrightarrow$	a

Stack	Input	State	Action
\$	a a \$	$S' \longrightarrow \bullet\ S$ $S \longrightarrow \bullet\ BC$ $B \longrightarrow \bullet$ a	shift
\$ a	a \$	$B \longrightarrow$ a$\bullet$	reduce by 3
\$ B	a \$	$S \longrightarrow B\bullet\ C$ $C \longrightarrow \bullet$ a	shift
\$ B a	\$	$C \longrightarrow$ a$\bullet$	reduce by 4
\$ $B\ C$	\$	$S \longrightarrow BC\bullet$	reduce by 2
\$ S	\$		

A Simple Example of LR Parsing: A Detailed Look

S'	$\longrightarrow$	S
S	$\longrightarrow$	$B\ C$
B	$\longrightarrow$	a
C	$\longrightarrow$	a

Stack	Input	State	Action
\$	a a \$	$S' \longrightarrow \bullet\ S$ $S \longrightarrow \bullet\ BC$ $B \longrightarrow \bullet$ a	shift
\$ a	a \$	$B \longrightarrow$ a$\bullet$	reduce by 3
\$ B	a \$	$S \longrightarrow B\bullet\ C$ $C \longrightarrow \bullet$ a	shift
\$ B a	\$	$C \longrightarrow$ a$\bullet$	reduce by 4
\$ $B\ C$	\$	$S \longrightarrow BC\bullet$	reduce by 2
\$ S	\$	$S' \longrightarrow S\bullet$	

A Simple Example of LR Parsing: A Detailed Look

S'	$\longrightarrow$	S
S	$\longrightarrow$	$B\ C$
B	$\longrightarrow$	a
C	$\longrightarrow$	a

Stack	Input	State	Action
\$	a a \$	$S' \longrightarrow \bullet\ S$ $S \longrightarrow \bullet\ BC$ $B \longrightarrow \bullet$ a	shift
\$ a	a \$	$B \longrightarrow$ a$\bullet$	reduce by 3
\$ B	a \$	$S \longrightarrow B\bullet\ C$ $C \longrightarrow \bullet$ a	shift
\$ B a	\$	$C \longrightarrow$ a$\bullet$	reduce by 4
\$ $B\ C$	\$	$S \longrightarrow BC\bullet$	reduce by 2
\$ S	\$	$S' \longrightarrow S\bullet$	**ACCEPT**

LR Parsing: Another Example

Stack	Input	State	Action
$	id + id $		

$$
\begin{array}{lcl}
E' & \longrightarrow & E \\
E & \longrightarrow & E + T \\
E & \longrightarrow & T \\
T & \longrightarrow & \texttt{id}
\end{array}
$$

LR Parsing: Another Example

STACK	INPUT	STATE	ACTION
		$E' \longrightarrow \bullet\, E$	
$	id + id $		

E'	$\longrightarrow$	E
E	$\longrightarrow$	$E + T$
E	$\longrightarrow$	T
T	$\longrightarrow$	id

LR Parsing: Another Example

Stack	Input	State	Action
$	id + id $	$E' \longrightarrow \bullet\, E$ $E \longrightarrow \bullet\, E+T$	

E'	$\longrightarrow$	E
E	$\longrightarrow$	$E+T$
E	$\longrightarrow$	T
T	$\longrightarrow$	id

LR Parsing: Another Example

Stack	Input	State	Action
$	id + id $	$E' \longrightarrow \bullet\ E$ $E \longrightarrow \bullet\ E+T$ $E \longrightarrow \bullet\ T$	

E'	$\longrightarrow$	E
E	$\longrightarrow$	$E+T$
E	$\longrightarrow$	T
T	$\longrightarrow$	id

LR Parsing: Another Example

Stack	Input	State	Action
$	id + id $	$E' \longrightarrow \bullet\ E$ $E \longrightarrow \bullet\ E+T$ $E \longrightarrow \bullet\ T$ $T \longrightarrow \bullet\ \texttt{id}$	

E'	$\longrightarrow$	E
E	$\longrightarrow$	$E+T$
E	$\longrightarrow$	T
T	$\longrightarrow$	id

LR Parsing: Another Example

Stack	Input	State	Action
$	id + id $	$E' \longrightarrow \bullet\ E$ $E \longrightarrow \bullet\ E+T$ $E \longrightarrow \bullet\ T$ $T \longrightarrow \bullet\ \texttt{id}$	shift
$ id	+ id $		

E'	$\longrightarrow$	E
E	$\longrightarrow$	$E+T$
E	$\longrightarrow$	T
T	$\longrightarrow$	id

LR Parsing: Another Example

Stack	Input	State	Action
$	id + id $	$E' \longrightarrow \bullet\ E$ $E \longrightarrow \bullet\ E+T$ $E \longrightarrow \bullet\ T$ $T \longrightarrow \bullet\ \texttt{id}$	shift
$ id	+ id $	$T \longrightarrow \texttt{id}\bullet$	

E'	$\longrightarrow$	E
E	$\longrightarrow$	$E+T$
E	$\longrightarrow$	T
T	$\longrightarrow$	id

LR Parsing: Another Example

E'	$\longrightarrow$	E
E	$\longrightarrow$	$E + T$
E	$\longrightarrow$	T
T	$\longrightarrow$	id

Stack	Input	State	Action
\$	id + id \$	$E' \longrightarrow \bullet\ E$ $E \longrightarrow \bullet\ E+T$ $E \longrightarrow \bullet\ T$ $T \longrightarrow \bullet$ id	shift
\$ id	+ id \$	$T \longrightarrow$ id$\bullet$	reduce by 4
\$ T	+ id \$		

LR Parsing: Another Example

E'	$\longrightarrow$	E
E	$\longrightarrow$	$E + T$
E	$\longrightarrow$	T
T	$\longrightarrow$	id

Stack	Input	State	Action
\$	id + id \$	$E' \longrightarrow \bullet\, E$ $E \longrightarrow \bullet\, E+T$ $E \longrightarrow \bullet\, T$ $T \longrightarrow \bullet$ id	shift
\$ id	+ id \$	$T \longrightarrow$ id$\bullet$	reduce by 4
\$ T	+ id \$	$E \longrightarrow T\bullet$	

LR Parsing: Another Example

E'	$\longrightarrow$	E
E	$\longrightarrow$	$E + T$
E	$\longrightarrow$	T
T	$\longrightarrow$	id

Stack	Input	State	Action
\$	id + id \$	$E' \longrightarrow \bullet\, E$ $E \longrightarrow \bullet\, E+T$ $E \longrightarrow \bullet\, T$ $T \longrightarrow \bullet$ id	shift
\$ id	+ id \$	$T \longrightarrow$ id$\bullet$	reduce by 4
\$ T	+ id \$	$E \longrightarrow T\bullet$	reduce by 3
\$ E	+ id \$		

LR Parsing: Another Example

E'	$\longrightarrow$	E
E	$\longrightarrow$	$E + T$
E	$\longrightarrow$	T
T	$\longrightarrow$	id

Stack	Input	State	Action
\$	id + id \$	$E' \longrightarrow \bullet\, E$ $E \longrightarrow \bullet\, E+T$ $E \longrightarrow \bullet\, T$ $T \longrightarrow \bullet$ id	shift
\$ id	+ id \$	$T \longrightarrow$ id$\bullet$	reduce by 4
\$ T	+ id \$	$E \longrightarrow T\bullet$	reduce by 3
\$ E	+ id \$	$E' \longrightarrow E\bullet$	

LR Parsing: Another Example

E'	$\longrightarrow$	E
E	$\longrightarrow$	$E + T$
E	$\longrightarrow$	T
T	$\longrightarrow$	id

Stack	Input	State	Action
\$	id + id \$	$E' \longrightarrow \bullet\, E$ $E \longrightarrow \bullet\, E+T$ $E \longrightarrow \bullet\, T$ $T \longrightarrow \bullet$ id	shift
\$ id	+ id \$	$T \longrightarrow$ id$\bullet$	reduce by 4
\$ T	+ id \$	$E \longrightarrow T\bullet$	reduce by 3
\$ E	+ id \$	$E' \longrightarrow E\bullet$ $E \longrightarrow E\bullet +T$	

LR Parsing: Another Example

E'	$\longrightarrow$	E
E	$\longrightarrow$	$E + T$
E	$\longrightarrow$	T
T	$\longrightarrow$	id

Stack	Input	State	Action
\$	id + id \$	$E' \longrightarrow \bullet\, E$ $E \longrightarrow \bullet\, E+T$ $E \longrightarrow \bullet\, T$ $T \longrightarrow \bullet$ id	shift
\$ id	+ id \$	$T \longrightarrow$ id$\bullet$	reduce by 4
\$ T	+ id \$	$E \longrightarrow T\bullet$	reduce by 3
\$ E	+ id \$	$E' \longrightarrow E\bullet$ $E \longrightarrow E\bullet\, +T$	shift
\$ E +	id \$		

LR Parsing: Another Example

E'	$\longrightarrow$	E
E	$\longrightarrow$	$E + T$
E	$\longrightarrow$	T
T	$\longrightarrow$	id

Stack	Input	State	Action
\$	id + id \$	$E' \longrightarrow \bullet\, E$ $E \longrightarrow \bullet\, E+T$ $E \longrightarrow \bullet\, T$ $T \longrightarrow \bullet$ id	shift
\$ id	+ id \$	$T \longrightarrow$ id$\bullet$	reduce by 4
\$ T	+ id \$	$E \longrightarrow T\bullet$	reduce by 3
\$ E	+ id \$	$E' \longrightarrow E\bullet$ $E \longrightarrow E\bullet\, +T$	shift
\$ E +	id \$	$E \longrightarrow E+\bullet\, T$	

LR Parsing: Another Example

E'	$\longrightarrow$	E
E	$\longrightarrow$	$E + T$
E	$\longrightarrow$	T
T	$\longrightarrow$	id

Stack	Input	State	Action
\$	id + id \$	$E' \longrightarrow \bullet\, E$ $E \longrightarrow \bullet\, E+T$ $E \longrightarrow \bullet\, T$ $T \longrightarrow \bullet$ id	shift
\$ id	+ id \$	$T \longrightarrow$ id$\bullet$	reduce by 4
\$ T	+ id \$	$E \longrightarrow T\bullet$	reduce by 3
\$ E	+ id \$	$E' \longrightarrow E\bullet$ $E \longrightarrow E\bullet +T$	shift
\$ E +	id \$	$E \longrightarrow E+\bullet\, T$ $T \longrightarrow \bullet$ id	

LR Parsing: Another Example

E'	$\longrightarrow$	E
E	$\longrightarrow$	$E + T$
E	$\longrightarrow$	T
T	$\longrightarrow$	id

Stack	Input	State	Action
\$	id + id \$	$E' \longrightarrow \bullet\, E$ $E \longrightarrow \bullet\, E+T$ $E \longrightarrow \bullet\, T$ $T \longrightarrow \bullet$ id	shift
\$ id	+ id \$	$T \longrightarrow$ id$\bullet$	reduce by 4
\$ T	+ id \$	$E \longrightarrow T\bullet$	reduce by 3
\$ E	+ id \$	$E' \longrightarrow E\bullet$ $E \longrightarrow E\bullet +T$	shift
\$ E +	id \$	$E \longrightarrow E+\bullet\, T$ $T \longrightarrow \bullet$ id	shift
\$ E + id	\$		

LR Parsing: Another Example

E'	$\longrightarrow$	E
E	$\longrightarrow$	$E + T$
E	$\longrightarrow$	T
T	$\longrightarrow$	id

Stack	Input	State	Action
\$	id + id \$	$E' \longrightarrow \bullet\, E$ $E \longrightarrow \bullet\, E+T$ $E \longrightarrow \bullet\, T$ $T \longrightarrow \bullet$ id	shift
\$ id	+ id \$	$T \longrightarrow$ id$\bullet$	reduce by 4
\$ T	+ id \$	$E \longrightarrow T\bullet$	reduce by 3
\$ E	+ id \$	$E' \longrightarrow E\bullet$ $E \longrightarrow E\bullet +T$	shift
\$ E +	id \$	$E \longrightarrow E+\bullet\, T$ $T \longrightarrow \bullet$ id	shift
\$ E + id	\$	$T \longrightarrow$ id$\bullet$	

LR Parsing: Another Example

E'	$\longrightarrow$	E
E	$\longrightarrow$	$E + T$
E	$\longrightarrow$	T
T	$\longrightarrow$	id

Stack	Input	State	Action
\$	id + id \$	$E' \longrightarrow \bullet\, E$ $E \longrightarrow \bullet\, E+T$ $E \longrightarrow \bullet\, T$ $T \longrightarrow \bullet$ id	shift
\$ id	+ id \$	$T \longrightarrow$ id$\bullet$	reduce by 4
\$ T	+ id \$	$E \longrightarrow T\bullet$	reduce by 3
\$ E	+ id \$	$E' \longrightarrow E\bullet$ $E \longrightarrow E\bullet +T$	shift
\$ E +	id \$	$E \longrightarrow E+\bullet\, T$ $T \longrightarrow \bullet$ id	shift
\$ E + id	\$	$T \longrightarrow$ id$\bullet$	reduce by 4
\$ E + T	\$		

LR Parsing: Another Example

E'	$\longrightarrow$	E
E	$\longrightarrow$	$E + T$
E	$\longrightarrow$	T
T	$\longrightarrow$	id

Stack	Input	State	Action
\$	id + id \$	$E' \longrightarrow \bullet\, E$ $E \longrightarrow \bullet\, E+T$ $E \longrightarrow \bullet\, T$ $T \longrightarrow \bullet$ id	shift
\$ id	+ id \$	$T \longrightarrow$ id$\bullet$	reduce by 4
\$ T	+ id \$	$E \longrightarrow T\bullet$	reduce by 3
\$ E	+ id \$	$E' \longrightarrow E\bullet$ $E \longrightarrow E\bullet +T$	shift
\$ E +	id \$	$E \longrightarrow E+\bullet\, T$ $T \longrightarrow \bullet$ id	shift
\$ E + id	\$	$T \longrightarrow$ id$\bullet$	reduce by 4
\$ E + T	\$	$E \longrightarrow E+T\bullet$	

LR Parsing: Another Example

E'	$\longrightarrow$	E
E	$\longrightarrow$	$E+T$
E	$\longrightarrow$	T
T	$\longrightarrow$	id

Stack	Input	State	Action
\$	id + id \$	$E' \longrightarrow \bullet\, E$ $E \longrightarrow \bullet\, E+T$ $E \longrightarrow \bullet\, T$ $T \longrightarrow \bullet$ id	shift
\$ id	+ id \$	$T \longrightarrow$ id$\bullet$	reduce by 4
\$ T	+ id \$	$E \longrightarrow T\bullet$	reduce by 3
\$ E	+ id \$	$E' \longrightarrow E\bullet$ $E \longrightarrow E\bullet +T$	shift
\$ E +	id \$	$E \longrightarrow E+\bullet\, T$ $T \longrightarrow \bullet$ id	shift
\$ E + id	\$	$T \longrightarrow$ id$\bullet$	reduce by 4
\$ E + T	\$	$E \longrightarrow E+T\bullet$	reduce by 2
\$ E	\$		

LR Parsing: Another Example

E'	$\longrightarrow$	E
E	$\longrightarrow$	$E+T$
E	$\longrightarrow$	T
T	$\longrightarrow$	id

Stack	Input	State	Action
\$	id + id \$	$E' \longrightarrow \bullet\, E$ $E \longrightarrow \bullet\, E+T$ $E \longrightarrow \bullet\, T$ $T \longrightarrow \bullet$ id	shift
\$ id	+ id \$	$T \longrightarrow$ id$\bullet$	reduce by 4
\$ T	+ id \$	$E \longrightarrow T\bullet$	reduce by 3
\$ E	+ id \$	$E' \longrightarrow E\bullet$ $E \longrightarrow E\bullet +T$	shift
\$ E +	id \$	$E \longrightarrow E+\bullet\, T$ $T \longrightarrow \bullet$ id	shift
\$ E + id	\$	$T \longrightarrow$ id$\bullet$	reduce by 4
\$ E + T	\$	$E \longrightarrow E+T\bullet$	reduce by 2
\$ E	\$	$E \longrightarrow E\bullet +T$	

LR Parsing: Another Example

E'	$\longrightarrow$	E
E	$\longrightarrow$	$E+T$
E	$\longrightarrow$	T
T	$\longrightarrow$	id

Stack	Input	State	Action
\$	id + id \$	$E' \longrightarrow \bullet\, E$ $E \longrightarrow \bullet\, E+T$ $E \longrightarrow \bullet\, T$ $T \longrightarrow \bullet$ id	shift
\$ id	+ id \$	$T \longrightarrow$ id$\bullet$	reduce by 4
\$ T	+ id \$	$E \longrightarrow T\bullet$	reduce by 3
\$ E	+ id \$	$E' \longrightarrow E\bullet$ $E \longrightarrow E\bullet +T$	shift
\$ E +	id \$	$E \longrightarrow E+\bullet\, T$ $T \longrightarrow \bullet$ id	shift
\$ E + id	\$	$T \longrightarrow$ id$\bullet$	reduce by 4
\$ E + T	\$	$E \longrightarrow E+T\bullet$	reduce by 2
\$ E	\$	$E \longrightarrow E\bullet +T$ $E' \longrightarrow E\bullet$	

LR Parsing: Another Example

E'	$\longrightarrow$	E
E	$\longrightarrow$	$E + T$
E	$\longrightarrow$	T
T	$\longrightarrow$	id

Stack	Input	State	Action
\$	id + id \$	$E' \longrightarrow \bullet\, E$ $E \longrightarrow \bullet\, E+T$ $E \longrightarrow \bullet\, T$ $T \longrightarrow \bullet$ id	shift
\$ id	+ id \$	$T \longrightarrow$ id$\bullet$	reduce by 4
\$ T	+ id \$	$E \longrightarrow T\bullet$	reduce by 3
\$ E	+ id \$	$E' \longrightarrow E\bullet$ $E \longrightarrow E\bullet +T$	shift
\$ E +	id \$	$E \longrightarrow E+\bullet\, T$ $T \longrightarrow \bullet$ id	shift
\$ E + id	\$	$T \longrightarrow$ id$\bullet$	reduce by 4
\$ E + T	\$	$E \longrightarrow E+T\bullet$	reduce by 2
\$ E	\$	$E \longrightarrow E\bullet +T$ $E' \longrightarrow E\bullet$	**ACCEPT**

States of an LR parser

I_0:	$E' \longrightarrow \bullet\, E$
	$E \longrightarrow \bullet\, E + T$
	$E \longrightarrow \bullet\, T$
	$T \longrightarrow \bullet\, \texttt{id}$

States of an LR parser

I_0:	$E' \longrightarrow \bullet\, E$
	$E \longrightarrow \bullet\, E + T$
	$E \longrightarrow \bullet\, T$
	$T \longrightarrow \bullet\, \texttt{id}$

Item: A production with "$\bullet$" somewhere on the RHS. Intuitively,

States of an LR parser

I_0:	$E' \longrightarrow \bullet\, E$
	$E \longrightarrow \bullet\, E + T$
	$E \longrightarrow \bullet\, T$
	$T \longrightarrow \bullet\, \texttt{id}$

Item: A production with "•" somewhere on the RHS. Intuitively,

- grammar symbols <u>before</u> the "•" are on stack;

States of an LR parser

I_0:	$E' \longrightarrow \bullet\, E$
	$E \longrightarrow \bullet\, E + T$
	$E \longrightarrow \bullet\, T$
	$T \longrightarrow \bullet\, \texttt{id}$

Item: A production with "•" somewhere on the RHS. Intuitively,

- grammar symbols <u>before</u> the "•" are on stack;
- grammar symbols <u>after</u> the "•" represent symbols in the input stream.

States of an LR parser

I_0:	$E' \longrightarrow \bullet\, E$
	$E \longrightarrow \bullet\, E + T$
	$E \longrightarrow \bullet\, T$
	$T \longrightarrow \bullet\, \texttt{id}$

Item: A production with "•" somewhere on the RHS. Intuitively,

- grammar symbols <u>before</u> the "•" are on stack;
- grammar symbols <u>after</u> the "•" represent symbols in the input stream.

Item set: A set of items; corresponds to a <u>state</u> of the parser.

States of an LR parser (contd.)

I_0	$E' \longrightarrow \bullet\ E$ $E \longrightarrow \bullet\ E + T$ $E \longrightarrow \bullet\ T$ $T \longrightarrow \bullet\ \texttt{id}$	Initial State $= closure(\{E' \longrightarrow \bullet\ E\})$

Closure:

- What other items are "equivalent" to the given item?

States of an LR parser (contd.)

I_0	$E' \longrightarrow \bullet\ E$ $E \longrightarrow \bullet\ E+T$ $E \longrightarrow \bullet\ T$ $T \longrightarrow \bullet\ \texttt{id}$	Initial State $= closure(\{E' \longrightarrow \bullet\ E\})$

Closure:

- What other items are "equivalent" to the given item?
- Given an item $A \longrightarrow \alpha\bullet\ B\beta$, $closure(A \longrightarrow \alpha\bullet\ B\beta)$ is the smallest set that contains

States of an LR parser (contd.)

I_0	$E' \longrightarrow \bullet\ E$ $E \longrightarrow \bullet\ E+T$ $E \longrightarrow \bullet\ T$ $T \longrightarrow \bullet\ \texttt{id}$	Initial State $= closure(\{E' \longrightarrow \bullet\ E\})$

Closure:

- What other items are "equivalent" to the given item?
- Given an item $A \longrightarrow \alpha\bullet\ B\beta$, $closure(A \longrightarrow \alpha\bullet\ B\beta)$ is the smallest set that contains
  1. the item $A \longrightarrow \alpha\bullet\ B\beta$, and

States of an LR parser (contd.)

I_0	$E' \longrightarrow \bullet\ E$ $E \longrightarrow \bullet\ E+T$ $E \longrightarrow \bullet\ T$ $T \longrightarrow \bullet\ \texttt{id}$	Initial State $= closure(\{E' \longrightarrow \bullet\ E\})$

Closure:

- What other items are "equivalent" to the given item?
- Given an item $A \longrightarrow \alpha\bullet\ B\beta$, $closure(A \longrightarrow \alpha\bullet\ B\beta)$ is the smallest set that contains
  1. the item $A \longrightarrow \alpha\bullet\ B\beta$, and
  2. every item in $closure(B \longrightarrow \bullet\ \gamma)$ for every production $B \longrightarrow \gamma \in G$

States of an LR parser (contd.)

I_0	$E' \longrightarrow \bullet\ E$ $E \longrightarrow \bullet\ E\texttt{+}T$ $E \longrightarrow \bullet\ T$ $T \longrightarrow \bullet\ \texttt{id}$	Initial State $= closure(\{E' \longrightarrow \bullet\ E\})$
I_3	$T \longrightarrow \texttt{id}\bullet$	$= goto(I_0, \texttt{id})$

Goto:

- $goto(I, X)$ specifies the next state to visit.

States of an LR parser (contd.)

I_0	$E' \longrightarrow \bullet\ E$ $E \longrightarrow \bullet\ E{+}T$ $E \longrightarrow \bullet\ T$ $T \longrightarrow \bullet\ \texttt{id}$	Initial State $= closure(\{E' \longrightarrow \bullet\ E\})$
I_3	$T \longrightarrow \texttt{id}\bullet$	$= goto(I_0, \texttt{id})$

Goto:

- $goto(I, X)$ specifies the next state to visit.
  - <u>X is a terminal:</u> when the next symbol on input stream is X.

States of an LR parser (contd.)

I_0	$E' \longrightarrow \bullet\ E$ $E \longrightarrow \bullet\ E{+}T$ $E \longrightarrow \bullet\ T$ $T \longrightarrow \bullet\ \texttt{id}$	Initial State $= closure(\{E' \longrightarrow \bullet\ E\})$
I_3	$T \longrightarrow \texttt{id}\bullet$	$= goto(I_0, \texttt{id})$

Goto:

- $goto(I, X)$ specifies the next state to visit.
  - <u>X is a terminal:</u> when the next symbol on input stream is X.
  - <u>X is a nonterminal:</u> when the last reduction was to X.

States of an LR parser (contd.)

I_0	$E' \longrightarrow \bullet\ E$ $E \longrightarrow \bullet\ E{+}T$ $E \longrightarrow \bullet\ T$ $T \longrightarrow \bullet\ \texttt{id}$	Initial State $= closure(\{E' \longrightarrow \bullet\ E\})$
I_3	$T \longrightarrow \texttt{id}\bullet$	$= goto(I_0, \texttt{id})$

Goto:

- $goto(I, X)$ specifies the next state to visit.
  - <u>X is a terminal:</u> when the next symbol on input stream is X.
  - <u>X is a nonterminal:</u> when the last reduction was to X.
- $goto(I, X)$ contains all items in $closure(A \longrightarrow \alpha X\bullet\ \beta)$ for every item $A \longrightarrow \alpha\bullet\ X\beta \in I$.

Collection of LR(0) Item Sets

The canonical collection of LR(0) item sets, $\mathcal{C} = \{I_0, I_1, \ldots\}$ is the smallest set such that

- $closure(\{S' \longrightarrow \bullet\ S\}) \in \mathcal{C}$.
- $I \in \mathcal{C} \Rightarrow \forall X,\ goto(I, X) \in \mathcal{C}$.

Canonical LR(0) Item Sets: An Example

E'	$\longrightarrow$	E	E	$\longrightarrow$	T
E	$\longrightarrow$	E+T	T	$\longrightarrow$	id

I_0	$= closure(\{E' \longrightarrow \bullet\ E\})$

Canonical LR(0) Item Sets: An Example

E'	$\longrightarrow$	E	E	$\longrightarrow$	T
E	$\longrightarrow$	E+T	T	$\longrightarrow$	id

I_0	$= closure(\{E' \longrightarrow \bullet\ E\})$	$E' \longrightarrow \bullet\ E$

Canonical LR(0) Item Sets: An Example

E'	$\longrightarrow$	E	E	$\longrightarrow$	T
E	$\longrightarrow$	E+T	T	$\longrightarrow$	id

I_0	$= closure(\{E' \longrightarrow \bullet\ E\})$	$E' \longrightarrow \bullet\ E$ $E \longrightarrow \bullet\ E$+$T$

Canonical LR(0) Item Sets: An Example

E'	$\longrightarrow$	E	E	$\longrightarrow$	T
E	$\longrightarrow$	$E+T$	T	$\longrightarrow$	id

I_0	$= closure(\{E' \longrightarrow \bullet\ E\})$	$E' \longrightarrow \bullet\ E$ $E \longrightarrow \bullet\ E+T$ $E \longrightarrow \bullet\ T$

Canonical LR(0) Item Sets: An Example

E'	$\longrightarrow$	E	E	$\longrightarrow$	T
E	$\longrightarrow$	$E+T$	T	$\longrightarrow$	id

I_0	$= closure(\{E' \longrightarrow \bullet\ E\})$	$E' \longrightarrow \bullet\ E$ $E \longrightarrow \bullet\ E+T$ $E \longrightarrow \bullet\ T$ $T \longrightarrow \bullet\ \texttt{id}$

Canonical LR(0) Item Sets: An Example

E'	$\longrightarrow$	E	E	$\longrightarrow$	T
E	$\longrightarrow$	$E+T$	T	$\longrightarrow$	id

I_0	$= closure(\{E' \longrightarrow \bullet\ E\})$	$E' \longrightarrow \bullet\ E$ $E \longrightarrow \bullet\ E+T$ $E \longrightarrow \bullet\ T$ $T \longrightarrow \bullet\ \texttt{id}$
I_1	$= goto(I_0, E)$	

Canonical LR(0) Item Sets: An Example

E'	$\longrightarrow$	E	E	$\longrightarrow$	T
E	$\longrightarrow$	$E+T$	T	$\longrightarrow$	id

I_0	$= closure(\{E' \longrightarrow \bullet\ E\})$	$E' \longrightarrow \bullet\ E$ $E \longrightarrow \bullet\ E+T$ $E \longrightarrow \bullet\ T$ $T \longrightarrow \bullet$ id
I_1	$= goto(I_0, E)$	$E' \longrightarrow E\bullet$

Canonical LR(0) Item Sets: An Example

E'	$\longrightarrow$	E	E	$\longrightarrow$	T
E	$\longrightarrow$	$E+T$	T	$\longrightarrow$	id

I_0	$= closure(\{E' \longrightarrow \bullet\ E\})$	$E' \longrightarrow \bullet\ E$ $E \longrightarrow \bullet\ E+T$ $E \longrightarrow \bullet\ T$ $T \longrightarrow \bullet\ \texttt{id}$
I_1	$= goto(I_0, E)$	$E' \longrightarrow E\bullet$ $E \longrightarrow E\bullet\ +T$

Canonical LR(0) Item Sets: An Example

E'	$\longrightarrow$	E	E	$\longrightarrow$	T
E	$\longrightarrow$	$E+T$	T	$\longrightarrow$	id

I_0	$= closure(\{E' \longrightarrow \bullet\ E\})$	$E' \longrightarrow \bullet\ E$ $E \longrightarrow \bullet\ E+T$ $E \longrightarrow \bullet\ T$ $T \longrightarrow \bullet\ \mathtt{id}$
I_1	$= goto(I_0, E)$	$E' \longrightarrow E\bullet$ $E \longrightarrow E\bullet\ +T$
I_2	$= goto(I_0, T)$	

Canonical LR(0) Item Sets: An Example

E'	$\longrightarrow$	E	E	$\longrightarrow$	T
E	$\longrightarrow$	$E+T$	T	$\longrightarrow$	id

I_0	$= closure(\{E' \longrightarrow \bullet\ E\})$	$E' \longrightarrow \bullet\ E$ $E \longrightarrow \bullet\ E+T$ $E \longrightarrow \bullet\ T$ $T \longrightarrow \bullet\ \texttt{id}$
I_1	$= goto(I_0, E)$	$E' \longrightarrow E\bullet$ $E \longrightarrow E\bullet\ +T$
I_2	$= goto(I_0, T)$	$E \longrightarrow T\bullet$

Canonical LR(0) Item Sets: An Example

E'	$\longrightarrow$	E	E	$\longrightarrow$	T
E	$\longrightarrow$	$E+T$	T	$\longrightarrow$	id

I_0	$= closure(\{E' \longrightarrow \bullet\ E\})$	$E' \longrightarrow \bullet\ E$ $E \longrightarrow \bullet\ E+T$ $E \longrightarrow \bullet\ T$ $T \longrightarrow \bullet\ \texttt{id}$
I_1	$= goto(I_0, E)$	$E' \longrightarrow E\bullet$ $E \longrightarrow E\bullet\ +T$
I_2	$= goto(I_0, T)$	$E \longrightarrow T\bullet$
I_3	$= goto(I_0, \texttt{id})$	

Canonical LR(0) Item Sets: An Example

E'	$\longrightarrow$	E	E	$\longrightarrow$	T
E	$\longrightarrow$	$E+T$	T	$\longrightarrow$	id

I_0	$= closure(\{E' \longrightarrow \bullet\ E\})$	$E' \longrightarrow \bullet\ E$ $E \longrightarrow \bullet\ E+T$ $E \longrightarrow \bullet\ T$ $T \longrightarrow \bullet\ \texttt{id}$
I_1	$= goto(I_0, E)$	$E' \longrightarrow E\bullet$ $E \longrightarrow E\bullet\ +T$
I_2	$= goto(I_0, T)$	$E \longrightarrow T\bullet$
I_3	$= goto(I_0, \texttt{id})$	$T \longrightarrow \texttt{id}\bullet$

Canonical LR(0) Item Sets: An Example

E'	$\longrightarrow$	E	E	$\longrightarrow$	T
E	$\longrightarrow$	$E+T$	T	$\longrightarrow$	id

I_0	$= closure(\{E' \longrightarrow \bullet\ E\})$	$E' \longrightarrow \bullet\ E$ $E \longrightarrow \bullet\ E+T$ $E \longrightarrow \bullet\ T$ $T \longrightarrow \bullet\ \texttt{id}$
I_1	$= goto(I_0, E)$	$E' \longrightarrow E\bullet$ $E \longrightarrow E\bullet\ +T$
I_2	$= goto(I_0, T)$	$E \longrightarrow T\bullet$
I_3	$= goto(I_0, \texttt{id})$	$T \longrightarrow \texttt{id}\bullet$
I_4	$= goto(I_1, +)$	

Canonical LR(0) Item Sets: An Example

E'	$\longrightarrow$	E	E	$\longrightarrow$	T
E	$\longrightarrow$	$E+T$	T	$\longrightarrow$	id

I_0	$= closure(\{E' \longrightarrow \bullet\ E\})$	$E' \longrightarrow \bullet\ E$ $E \longrightarrow \bullet\ E+T$ $E \longrightarrow \bullet\ T$ $T \longrightarrow \bullet\ \texttt{id}$
I_1	$= goto(I_0, E)$	$E' \longrightarrow E\bullet$ $E \longrightarrow E\bullet\ +T$
I_2	$= goto(I_0, T)$	$E \longrightarrow T\bullet$
I_3	$= goto(I_0, \texttt{id})$	$T \longrightarrow \texttt{id}\bullet$
I_4	$= goto(I_1, +)$	$E \longrightarrow E+\bullet\ T$

Canonical LR(0) Item Sets: An Example

E'	$\longrightarrow$	E	E	$\longrightarrow$	T
E	$\longrightarrow$	$E+T$	T	$\longrightarrow$	id

I_0	$= closure(\{E' \longrightarrow \bullet\ E\})$	$E' \longrightarrow \bullet\ E$ $E \longrightarrow \bullet\ E+T$ $E \longrightarrow \bullet\ T$ $T \longrightarrow \bullet\ \texttt{id}$
I_1	$= goto(I_0, E)$	$E' \longrightarrow E\bullet$ $E \longrightarrow E\bullet\ +T$
I_2	$= goto(I_0, T)$	$E \longrightarrow T\bullet$
I_3	$= goto(I_0, \texttt{id})$	$T \longrightarrow \texttt{id}\bullet$
I_4	$= goto(I_1, +)$	$E \longrightarrow E+\bullet\ T$ $T \longrightarrow \bullet\ \texttt{id}$

Canonical LR(0) Item Sets: An Example

E'	$\longrightarrow$	E	E	$\longrightarrow$	T
E	$\longrightarrow$	$E+T$	T	$\longrightarrow$	id

I_0	$= closure(\{E' \longrightarrow \bullet\ E\})$	$E' \longrightarrow \bullet\ E$ $E \longrightarrow \bullet\ E+T$ $E \longrightarrow \bullet\ T$ $T \longrightarrow \bullet\ \texttt{id}$
I_1	$= goto(I_0, E)$	$E' \longrightarrow E\bullet$ $E \longrightarrow E\bullet\ +T$
I_2	$= goto(I_0, T)$	$E \longrightarrow T\bullet$
I_3	$= goto(I_0, \texttt{id})$	$T \longrightarrow \texttt{id}\bullet$
I_4	$= goto(I_1, +)$	$E \longrightarrow E+\bullet\ T$ $T \longrightarrow \bullet\ \texttt{id}$
I_5	$= goto(I_4, T)$	

Canonical LR(0) Item Sets: An Example

E'	$\longrightarrow$	E	E	$\longrightarrow$	T
E	$\longrightarrow$	$E+T$	T	$\longrightarrow$	id

I_0	$= closure(\{E' \longrightarrow \bullet\ E\})$	$E' \longrightarrow \bullet\ E$ $E \longrightarrow \bullet\ E+T$ $E \longrightarrow \bullet\ T$ $T \longrightarrow \bullet\ \texttt{id}$
I_1	$= goto(I_0, E)$	$E' \longrightarrow E\bullet$ $E \longrightarrow E\bullet\ +T$
I_2	$= goto(I_0, T)$	$E \longrightarrow T\bullet$
I_3	$= goto(I_0, \texttt{id})$	$T \longrightarrow \texttt{id}\bullet$
I_4	$= goto(I_1, +)$	$E \longrightarrow E+\bullet\ T$ $T \longrightarrow \bullet\ \texttt{id}$
I_5	$= goto(I_4, T)$	$E \longrightarrow E+T\bullet$

Canonical LR(0) Item Sets: An Example

E'	$\longrightarrow$	E	E	$\longrightarrow$	T
E	$\longrightarrow$	$E+T$	T	$\longrightarrow$	id

I_0	$= closure(\{E' \longrightarrow \bullet\ E\})$	$E' \longrightarrow \bullet\ E$ $E \longrightarrow \bullet\ E+T$ $E \longrightarrow \bullet\ T$ $T \longrightarrow \bullet\ \texttt{id}$
I_1	$= goto(I_0, E)$	$E' \longrightarrow E\bullet$ $E \longrightarrow E\bullet\ +T$
I_2	$= goto(I_0, T)$	$E \longrightarrow T\bullet$
I_3	$= goto(I_0, \texttt{id})$	$T \longrightarrow \texttt{id}\bullet$
I_4	$= goto(I_1, +)$	$E \longrightarrow E+\bullet\ T$ $T \longrightarrow \bullet\ \texttt{id}$
I_5	$= goto(I_4, T)$	$E \longrightarrow E+T\bullet$

LR Action Table

E'	$\longrightarrow$	E	E	$\longrightarrow$	T
E	$\longrightarrow$	E+T	T	$\longrightarrow$	id

	id	+	$
0	S, 3		
1		S, 4	A
2	$R3$	$R3$	$R3$
3	$R4$	$R4$	$R4$
4	S, 3		
5	$R2$	$R2$	$R2$

LR Goto Table

E'	$\longrightarrow$	E	E	$\longrightarrow$	T
E	$\longrightarrow$	E+T	T	$\longrightarrow$	id

	E	T
0	1	2
1		
2		
3		
4		5
5		

LR Parsing: States and Transitions

Action Table:

	id	+	$
0	S, 3		
1		S, 4	A
2	$R3$	$R3$	$R3$
3	$R4$	$R4$	$R4$
4	S, 3		
5	$R2$	$R2$	$R2$

E'	$\longrightarrow$	E	E	$\longrightarrow$	T
E	$\longrightarrow$	E+T	T	$\longrightarrow$	id

State Stack	Symbol Stack	Input	Action
$ 0	$	id + id $	

Goto Table:

	E	T
0	1	2
1		
2		
3		
4		5
5		

LR Parsing: States and Transitions

Action Table:

	id	+	$
0	S, 3		
1		S, 4	A
2	$R3$	$R3$	$R3$
3	$R4$	$R4$	$R4$
4	S, 3		
5	$R2$	$R2$	$R2$

E'	$\longrightarrow$	E	E	$\longrightarrow$	T
E	$\longrightarrow$	E+T	T	$\longrightarrow$	id

State Stack	Symbol Stack	Input	Action
$ 0	$	id + id $	shift, 3

Goto Table:

	E	T
0	1	2
1		
2		
3		
4		5
5		

LR Parsing: States and Transitions

Action Table:

	id	+	$
0	S, 3		
1		S, 4	A
2	$R3$	$R3$	$R3$
3	$R4$	$R4$	$R4$
4	S, 3		
5	$R2$	$R2$	$R2$

E'	$\longrightarrow$	E	E	$\longrightarrow$	T
E	$\longrightarrow$	E+T	T	$\longrightarrow$	id

State Stack	Symbol Stack	Input	Action
$ 0	$	id + id $	shift, 3
$ 0 3	$ id	+ id $	

Goto Table:

	E	T
0	1	2
1		
2		
3		
4		5
5		

LR Parsing: States and Transitions

Action Table:

	id	+	$
0	S, 3		
1		S, 4	A
2	$R3$	$R3$	$R3$
3	$R4$	$R4$	$R4$
4	S, 3		
5	$R2$	$R2$	$R2$

Goto Table:

	E	T
0	1	2
1		
2		
3		
4		5
5		

E'	$\longrightarrow$	E	E	$\longrightarrow$	T
E	$\longrightarrow$	E+T	T	$\longrightarrow$	id

State Stack	Symbol Stack	Input	Action
$ 0	$	id + id $	shift, 3
$ 0 3	$ id	+ id $	reduce by 4

LR Parsing: States and Transitions

Action Table:

	id	+	$
0	S, 3		
1		S, 4	A
2	$R3$	$R3$	$R3$
3	$R4$	$R4$	$R4$
4	S, 3		
5	$R2$	$R2$	$R2$

Goto Table:

	E	T
0	1	2
1		
2		
3		
4		5
5		

E'	$\longrightarrow$	E	E	$\longrightarrow$	T
E	$\longrightarrow$	E+T	T	$\longrightarrow$	id

State Stack	Symbol Stack	Input	Action
$ 0	$	id + id $	shift, 3
$ 0 3	$ id	+ id $	reduce by 4
$ 0	$	+ id $	

LR Parsing: States and Transitions

Action Table:

	id	+	$
0	S, 3		
1		S, 4	A
2	$R3$	$R3$	$R3$
3	$R4$	$R4$	$R4$
4	S, 3		
5	$R2$	$R2$	$R2$

Goto Table:

	E	T
0	1	2
1		
2		
3		
4		5
5		

E'	$\longrightarrow$	E	E	$\longrightarrow$	T
E	$\longrightarrow$	E+T	T	$\longrightarrow$	id

State Stack	Symbol Stack	Input	Action
$ 0	$	id + id $	shift, 3
$ 0 3	$ id	+ id $	reduce by 4
$ 0	$ T	+ id $	

LR Parsing: States and Transitions

Action Table:

	id	+	$
0	S, 3		
1		S, 4	A
2	$R3$	$R3$	$R3$
3	$R4$	$R4$	$R4$
4	S, 3		
5	$R2$	$R2$	$R2$

Goto Table:

	E	T
0	1	2
1		
2		
3		
4		5
5		

E'	$\longrightarrow$	E	E	$\longrightarrow$	T
E	$\longrightarrow$	E+T	T	$\longrightarrow$	id

State Stack	Symbol Stack	Input	Action
$ 0	$	id + id $	shift, 3
$ 0 3	$ id	+ id $	reduce by 4
$ 0 2	$ T	+ id $	

LR Parsing: States and Transitions

Action Table:

	id	+	$
0	S, 3		
1		S, 4	A
2	$R3$	$R3$	$R3$
3	$R4$	$R4$	$R4$
4	S, 3		
5	$R2$	$R2$	$R2$

Goto Table:

	E	T
0	1	2
1		
2		
3		
4		5
5		

E'	$\longrightarrow$	E	E	$\longrightarrow$	T
E	$\longrightarrow$	E+T	T	$\longrightarrow$	id

State Stack	Symbol Stack	Input	Action
$ 0	$	id + id $	shift, 3
$ 0 3	$ id	+ id $	reduce by 4
$ 0 2	$ T	+ id $	reduce by 3
$ 0	$	+ id $	

LR Parsing: States and Transitions

Action Table:

	id	+	$
0	S, 3		
1		S, 4	A
2	$R3$	$R3$	$R3$
3	$R4$	$R4$	$R4$
4	S, 3		
5	$R2$	$R2$	$R2$

Goto Table:

	E	T
0	1	2
1		
2		
3		
4		5
5		

E'	$\longrightarrow$	E	E	$\longrightarrow$	T
E	$\longrightarrow$	E+T	T	$\longrightarrow$	id

State Stack	Symbol Stack	Input	Action
$ 0	$	id + id $	shift, 3
$ 0 3	$ id	+ id $	reduce by 4
$ 0 2	$ T	+ id $	reduce by 3
$ 0	$ E	+ id $	

LR Parsing: States and Transitions

Action Table:

	id	+	$
0	S, 3		
1		S, 4	A
2	$R3$	$R3$	$R3$
3	$R4$	$R4$	$R4$
4	S, 3		
5	$R2$	$R2$	$R2$

Goto Table:

	E	T
0	1	2
1		
2		
3		
4		5
5		

E'	$\longrightarrow$	E	E	$\longrightarrow$	T
E	$\longrightarrow$	E+T	T	$\longrightarrow$	id

State Stack	Symbol Stack	Input	Action
$ 0	$	id + id $	shift, 3
$ 0 3	$ id	+ id $	reduce by 4
$ 0 2	$ T	+ id $	reduce by 3
$ 0 1	$ E	+ id $	

LR Parsing: States and Transitions

Action Table:

	id	+	$
0	S, 3		
1		S, 4	A
2	$R3$	$R3$	$R3$
3	$R4$	$R4$	$R4$
4	S, 3		
5	$R2$	$R2$	$R2$

Goto Table:

	E	T
0	1	2
1		
2		
3		
4		5
5		

E'	$\longrightarrow$	E	E	$\longrightarrow$	T
E	$\longrightarrow$	E+T	T	$\longrightarrow$	id

State Stack	Symbol Stack	Input	Action
$ 0	$	id + id $	shift, 3
$ 0 3	$ id	+ id $	reduce by 4
$ 0 2	$ T	+ id $	reduce by 3
$ 0 1	$ E	+ id $	shift, 4
$ 0 1 4	$ E +	id $	

LR Parsing: States and Transitions

Action Table:

	id	+	$
0	S, 3		
1		S, 4	A
2	$R3$	$R3$	$R3$
3	$R4$	$R4$	$R4$
4	S, 3		
5	$R2$	$R2$	$R2$

Goto Table:

	E	T
0	1	2
1		
2		
3		
4		5
5		

E'	$\longrightarrow$	E	E	$\longrightarrow$	T
E	$\longrightarrow$	E+T	T	$\longrightarrow$	id

State Stack	Symbol Stack	Input	Action
$ 0	$	id + id $	shift, 3
$ 0 3	$ id	+ id $	reduce by 4
$ 0 2	$ T	+ id $	reduce by 3
$ 0 1	$ E	+ id $	shift, 4
$ 0 1 4	$ E +	id $	shift, 3
$ 0 1 4 3	$ E + id	$	

LR Parsing: States and Transitions

Action Table:

	id	+	$
0	S, 3		
1		S, 4	A
2	$R3$	$R3$	$R3$
3	$R4$	$R4$	$R4$
4	S, 3		
5	$R2$	$R2$	$R2$

Goto Table:

	E	T
0	1	2
1		
2		
3		
4		5
5		

E'	$\longrightarrow$	E	E	$\longrightarrow$	T
E	$\longrightarrow$	E+T	T	$\longrightarrow$	id

State Stack	Symbol Stack	Input	Action
$ 0	$	id + id $	shift, 3
$ 0 3	$ id	+ id $	reduce by 4
$ 0 2	$ T	+ id $	reduce by 3
$ 0 1	$ E	+ id $	shift, 4
$ 0 1 4	$ E +	id $	shift, 3
$ 0 1 4 3	$ E + id	$	reduce by 4
$ 0 1 4 5	$ E + T	$	

LR Parsing: States and Transitions

Action Table:

	id	+	$
0	S, 3		
1		S, 4	A
2	$R3$	$R3$	$R3$
3	$R4$	$R4$	$R4$
4	S, 3		
5	$R2$	$R2$	$R2$

Goto Table:

	E	T
0	1	2
1		
2		
3		
4		5
5		

E'	$\longrightarrow$	E	E	$\longrightarrow$	T
E	$\longrightarrow$	E+T	T	$\longrightarrow$	id

State Stack	Symbol Stack	Input	Action
$ 0	$	id + id $	shift, 3
$ 0 3	$ id	+ id $	reduce by 4
$ 0 2	$ T	+ id $	reduce by 3
$ 0 1	$ E	+ id $	shift, 4
$ 0 1 4	$ E +	id $	shift, 3
$ 0 1 4 3	$ E + id	$	reduce by 4
$ 0 1 4 5	$ E + T	$	reduce by 2
$ 0 1	$ E	$	

LR Parsing: States and Transitions

Action Table:

	id	+	$
0	S, 3		
1		S, 4	A
2	$R3$	$R3$	$R3$
3	$R4$	$R4$	$R4$
4	S, 3		
5	$R2$	$R2$	$R2$

Goto Table:

	E	T
0	1	2
1		
2		
3		
4		5
5		

E'	$\longrightarrow$	E	E	$\longrightarrow$	T
E	$\longrightarrow$	E+T	T	$\longrightarrow$	id

State Stack	Symbol Stack	Input	Action
$ 0	$	id + id $	shift, 3
$ 0 3	$ id	+ id $	reduce by 4
$ 0 2	$ T	+ id $	reduce by 3
$ 0 1	$ E	+ id $	shift, 4
$ 0 1 4	$ E +	id $	shift, 3
$ 0 1 4 3	$ E + id	$	reduce by 4
$ 0 1 4 5	$ E + T	$	reduce by 2
$ 0 1	$ E	$	**ACCEPT**

LR Parser

```
while (true) {
  switch (action(state_stack.top(), current_token)) {
    case shift s':
      symbol_stack.push(current_token);
      state_stack.push(s');
      next_token();
    case reduce A ⟶ β:
      pop |β| symbols off symbol_stack and state_stack;
      symbol_stack.push(A);
      state_stack.push(goto(state_stack.top(), A));
    case accept: return;
    default: error;
}}
```

LR Parsing: A review

E'	$\longrightarrow$	E	E	$\longrightarrow$	T
E	$\longrightarrow$	E+T	T	$\longrightarrow$	id

Table-driven shift reduce parsing:

- Shift: Move **terminal** symbols from input stream to stack.
- Reduce: Replace top elements of stack that form an instance of the RHS of a production with the corresponding LHS
- Accept: Stack top is the start symbol when the input stream is exhausted

Table constructed using LR(0) Item Sets.

Conflicts in Parsing Table

Grammar:

S'	$\longrightarrow$	S
S	$\longrightarrow$	$\mathtt{a}\ S$
S	$\longrightarrow$	ϵ

Conflicts in Parsing Table

Grammar:

S'	$\longrightarrow$	S
S	$\longrightarrow$	$\mathtt{a}\ S$
S	$\longrightarrow$	ϵ

Item Sets:

I_0	$= closure(\{S' \longrightarrow \bullet\ S\})$	$S' \longrightarrow \bullet\ S$ $S \longrightarrow \bullet\ \mathtt{a}\ S$ $S \longrightarrow \bullet$

Conflicts in Parsing Table

Grammar:

S'	$\longrightarrow$	S
S	$\longrightarrow$	$\mathtt{a}\ S$
S	$\longrightarrow$	ϵ

Item Sets:

I_0	$= closure(\{S' \longrightarrow \bullet\ S\})$	$S' \longrightarrow \bullet\ S$ $S \longrightarrow \bullet\ \mathtt{a}\ S$ $S \longrightarrow \bullet$
I_1	$= goto(I_0, S)$	$S' \longrightarrow S\bullet$

Conflicts in Parsing Table

Grammar:

S'	$\longrightarrow$	S
S	$\longrightarrow$	$\mathtt{a}\ S$
S	$\longrightarrow$	ϵ

Item Sets:

I_0	$= closure(\{S' \longrightarrow \bullet\ S\})$	$S' \longrightarrow \bullet\ S$ $S \longrightarrow \bullet\ \mathtt{a}\ S$ $S \longrightarrow \bullet$
I_1	$= goto(I_0, S)$	$S' \longrightarrow S\bullet$
I_2	$= goto(I_0, \mathtt{a})$	$S \longrightarrow \mathtt{a}\ \bullet\ S$ $S \longrightarrow \bullet\ \mathtt{a}\ S$ $S \longrightarrow \bullet$

Conflicts in Parsing Table

Grammar:

S'	$\longrightarrow$	S
S	$\longrightarrow$	$\mathtt{a}\ S$
S	$\longrightarrow$	ϵ

Item Sets:

I_0	$= closure(\{S' \longrightarrow \bullet\ S\})$	$S' \longrightarrow \bullet\ S$ $S \longrightarrow \bullet\ \mathtt{a}\ S$ $S \longrightarrow \bullet$
I_1	$= goto(I_0, S)$	$S' \longrightarrow S\bullet$
I_2	$= goto(I_0, \mathtt{a})$	$S \longrightarrow \mathtt{a}\ \bullet\ S$ $S \longrightarrow \bullet\ \mathtt{a}\ S$ $S \longrightarrow \bullet$
I_3	$= goto(I_2, S)$	$S \longrightarrow \mathtt{a}\ S\ \bullet$

Conflicts in Parsing Table

Grammar:

S'	$\longrightarrow$	S
S	$\longrightarrow$	$\mathtt{a}\ S$
S	$\longrightarrow$	ϵ

Action Table:

Item Sets:

I_0	$= closure(\{S' \longrightarrow \bullet\ S\})$	$S' \longrightarrow \bullet\ S$ $S \longrightarrow \bullet\ \mathtt{a}\ S$ $S \longrightarrow \bullet$
I_1	$= goto(I_0, S)$	$S' \longrightarrow S \bullet$
I_2	$= goto(I_0, \mathtt{a})$	$S \longrightarrow \mathtt{a} \bullet\ S$ $S \longrightarrow \bullet\ \mathtt{a}\ S$ $S \longrightarrow \bullet$
I_3	$= goto(I_2, S)$	$S \longrightarrow \mathtt{a}\ S \bullet$

Conflicts in Parsing Table

Grammar:

S'	$\longrightarrow$	S
S	$\longrightarrow$	$\mathtt{a}\ S$
S	$\longrightarrow$	ϵ

Item Sets:

I_0	$= closure(\{S' \longrightarrow \bullet\ S\})$	$S' \longrightarrow \bullet\ S$ $S \longrightarrow \bullet\ \mathtt{a}\ S$ $S \longrightarrow \bullet$
I_1	$= goto(I_0, S)$	$S' \longrightarrow S\bullet$
I_2	$= goto(I_0, \mathtt{a})$	$S \longrightarrow \mathtt{a}\ \bullet\ S$ $S \longrightarrow \bullet\ \mathtt{a}\ S$ $S \longrightarrow \bullet$
I_3	$= goto(I_2, S)$	$S \longrightarrow \mathtt{a}\ S\ \bullet$

Action Table:

	a	$
0	S, 2	

Conflicts in Parsing Table

Grammar:

S'	$\longrightarrow$	S
S	$\longrightarrow$	a S
S	$\longrightarrow$	ϵ

Item Sets:

I_0	$= closure(\{S' \longrightarrow \bullet\ S\})$	$S' \longrightarrow \bullet\ S$ $S \longrightarrow \bullet$ a S $S \longrightarrow \bullet$
I_1	$= goto(I_0, S)$	$S' \longrightarrow S\bullet$
I_2	$= goto(I_0,$ a$)$	$S \longrightarrow$ a $\bullet\ S$ $S \longrightarrow \bullet$ a S $S \longrightarrow \bullet$
I_3	$= goto(I_2, S)$	$S \longrightarrow$ a $S\ \bullet$

Action Table:

	a	$
0	S, 2 R 3	R 3

Conflicts in Parsing Table

Grammar:

S'	$\longrightarrow$	S
S	$\longrightarrow$	a S
S	$\longrightarrow$	ϵ

Item Sets:

I_0	$= closure(\{S' \longrightarrow \bullet\ S\})$	$S' \longrightarrow \bullet\ S$ $S \longrightarrow \bullet$ a S $S \longrightarrow \bullet$
I_1	$= goto(I_0, S)$	$S' \longrightarrow S\bullet$
I_2	$= goto(I_0,$ a$)$	$S \longrightarrow$ a $\bullet\ S$ $S \longrightarrow \bullet$ a S $S \longrightarrow \bullet$
I_3	$= goto(I_2, S)$	$S \longrightarrow$ a $S\ \bullet$

Action Table:

	a	$
0	S, 2 R 3	R 3
1		A

Conflicts in Parsing Table

Grammar:

S'	$\longrightarrow$	S
S	$\longrightarrow$	a S
S	$\longrightarrow$	ϵ

Item Sets:

I_0	$= closure(\{S' \longrightarrow \bullet\ S\})$	$S' \longrightarrow \bullet\ S$ $S \longrightarrow \bullet$ a S $S \longrightarrow \bullet$
I_1	$= goto(I_0, S)$	$S' \longrightarrow S\bullet$
I_2	$= goto(I_0,$ a$)$	$S \longrightarrow$ a $\bullet\ S$ $S \longrightarrow \bullet$ a S $S \longrightarrow \bullet$
I_3	$= goto(I_2, S)$	$S \longrightarrow$ a $S\ \bullet$

Action Table:

	a	$
0	S, 2 R 3	R 3
1		A
2	S, 2	

Conflicts in Parsing Table

Grammar:

S'	$\longrightarrow$	S
S	$\longrightarrow$	a S
S	$\longrightarrow$	ϵ

Item Sets:

I_0	$= closure(\{S' \longrightarrow \bullet\ S\})$	$S' \longrightarrow \bullet\ S$ $S \longrightarrow \bullet$ a S $S \longrightarrow \bullet$
I_1	$= goto(I_0, S)$	$S' \longrightarrow S\bullet$
I_2	$= goto(I_0,$ a$)$	$S \longrightarrow$ a $\bullet\ S$ $S \longrightarrow \bullet$ a S $S \longrightarrow \bullet$
I_3	$= goto(I_2, S)$	$S \longrightarrow$ a $S\ \bullet$

Action Table:

	a	$
0	S, 2 R 3	R 3
1		A
2	S, 2 R 3	R 3
3	R 2	R 2

Conflicts in Parsing Table

Grammar:

S'	$\longrightarrow$	S
S	$\longrightarrow$	a S
S	$\longrightarrow$	ϵ

Item Sets:

I_0	$= closure(\{S' \longrightarrow \bullet\ S\})$	$S' \longrightarrow \bullet\ S$ $S \longrightarrow \bullet$ a S $S \longrightarrow \bullet$
I_1	$= goto(I_0, S)$	$S' \longrightarrow S\bullet$
I_2	$= goto(I_0,$ a$)$	$S \longrightarrow$ a $\bullet\ S$ $S \longrightarrow \bullet$ a S $S \longrightarrow \bullet$
I_3	$= goto(I_2, S)$	$S \longrightarrow$ a $S\ \bullet$

Action Table:

	a	$
0	S, 2 R 3	R 3
1		A
2	S, 2 R 3	R 3
3	R 2	R 2

Shift-Reduce Conflict

"Simple LR" (SLR) Parsing

Constructing Action Table *action*, indexed by *states* $\times$ *terminals*, and Goto Table *goto*, indexed by *states* $\times$ *nonterminals*:

- Construct $\{I_0, I_1, \ldots, I_n\}$, the LR(0) sets of items for the grammar. For each i, $0 \leq i \leq n$, do the following:

"Simple LR" (SLR) Parsing

Constructing Action Table *action*, indexed by *states* $\times$ *terminals*, and Goto Table *goto*, indexed by *states* $\times$ *nonterminals*:

- Construct $\{I_0, I_1, \ldots, I_n\}$, the LR(0) sets of items for the grammar. For each i, $0 \leq i \leq n$, do the following:
- If $A \longrightarrow \alpha \bullet \mathtt{a}\beta \in I_i$, and $goto(I_i, \mathtt{a}) = I_j$, set $action[i, \mathtt{a}] = shift \;\; j$.

"Simple LR" (SLR) Parsing

Constructing Action Table *action*, indexed by *states* $\times$ *terminals*, and Goto Table *goto*, indexed by *states* $\times$ *nonterminals*:

- Construct $\{I_0, I_1, \ldots, I_n\}$, the LR(0) sets of items for the grammar. For each i, $0 \leq i \leq n$, do the following:
- If $A \longrightarrow \alpha \bullet \, \mathtt{a}\beta \in I_i$, and $goto(I_i, \mathtt{a}) = I_j$, set $action[i, \mathtt{a}] = shift \;\; j$.
- If $A \longrightarrow \gamma \bullet \; \in I_i$ (A is not the start symbol), <u>for each $\mathtt{a} \in FOLLOW(A)$</u>, set $action[i, \mathtt{a}] = reduce \;\; A \longrightarrow \gamma$.

"Simple LR" (SLR) Parsing

Constructing Action Table *action*, indexed by *states* $\times$ *terminals*, and Goto Table *goto*, indexed by *states* $\times$ *nonterminals*:

- Construct $\{I_0, I_1, \ldots, I_n\}$, the LR(0) sets of items for the grammar. For each i, $0 \leq i \leq n$, do the following:
- If $A \longrightarrow \alpha \bullet \, \mathtt{a}\beta \in I_i$, and $goto(I_i, \mathtt{a}) = I_j$, set $action[i, \mathtt{a}] = shift \;\; j$.
- If $A \longrightarrow \gamma \bullet \; \in I_i$ (A is not the start symbol), for each $\mathtt{a} \in FOLLOW(A)$, set $action[i, \mathtt{a}] = reduce \;\; A \longrightarrow \gamma$.
- If $S' \longrightarrow S \bullet \; \in I_i$, set $action[i, \$] = accept$.

"Simple LR" (SLR) Parsing

Constructing Action Table *action*, indexed by *states* $\times$ *terminals*, and Goto Table *goto*, indexed by *states* $\times$ *nonterminals*:

- Construct $\{I_0, I_1, \ldots, I_n\}$, the LR(0) sets of items for the grammar. For each i, $0 \leq i \leq n$, do the following:
- If $A \longrightarrow \alpha \bullet \, \mathtt{a}\beta \in I_i$, and $goto(I_i, \mathtt{a}) = I_j$, set $action[i, \mathtt{a}] = shift \;\; j$.
- If $A \longrightarrow \gamma \bullet \; \in I_i$ (A is not the start symbol), for each $\mathtt{a} \in FOLLOW(A)$, set $action[i, \mathtt{a}] = reduce \;\; A \longrightarrow \gamma$.
- If $S' \longrightarrow S \bullet \; \in I_i$, set $action[i, \$] = accept$.
- If $goto(I_i, A) = I_j$ (A is a nonterminal), set $goto[i, A] = j$.

SLR Parsing Table

Grammar:

S'	$\longrightarrow$	S
S	$\longrightarrow$	$\mathtt{a}\ S$
S	$\longrightarrow$	ϵ

Item Sets:

I_0	$= closure(\{S' \longrightarrow \bullet\ S\})$	$S' \longrightarrow \bullet\ S$ $S \longrightarrow \bullet\ \mathtt{a}\ S$ $S \longrightarrow \bullet$
I_1	$= goto(I_0, S)$	$S' \longrightarrow S\bullet$
I_2	$= goto(I_0, \mathtt{a})$	$S \longrightarrow \mathtt{a}\ \bullet\ S$ $S \longrightarrow \bullet\ \mathtt{a}\ S$ $S \longrightarrow \bullet$
I_3	$= goto(I_2, S)$	$S \longrightarrow \mathtt{a}\ S\ \bullet$

SLR Parsing Table

Grammar:

S'	$\longrightarrow$	S
S	$\longrightarrow$	$\mathtt{a}\ S$
S	$\longrightarrow$	ϵ

$FOLLOW(S) = \{\$\}$

Item Sets:

I_0	$= closure(\{S' \longrightarrow \bullet\ S\})$	$S' \longrightarrow \bullet\ S$ $S \longrightarrow \bullet\ \mathtt{a}\ S$ $S \longrightarrow \bullet$
I_1	$= goto(I_0, S)$	$S' \longrightarrow S\bullet$
I_2	$= goto(I_0, \mathtt{a})$	$S \longrightarrow \mathtt{a}\ \bullet\ S$ $S \longrightarrow \bullet\ \mathtt{a}\ S$ $S \longrightarrow \bullet$
I_3	$= goto(I_2, S)$	$S \longrightarrow \mathtt{a}\ S\ \bullet$

SLR Parsing Table

Grammar:

S'	$\longrightarrow$	S
S	$\longrightarrow$	$\mathtt{a}\ S$
S	$\longrightarrow$	ϵ

$FOLLOW(S) = \{\$\}$

Item Sets:

I_0	$= closure(\{S' \longrightarrow \bullet\ S\})$	$S' \longrightarrow \bullet\ S$ $S \longrightarrow \bullet\ \mathtt{a}\ S$ $S \longrightarrow \bullet$
I_1	$= goto(I_0, S)$	$S' \longrightarrow S\bullet$
I_2	$= goto(I_0, \mathtt{a})$	$S \longrightarrow \mathtt{a}\ \bullet\ S$ $S \longrightarrow \bullet\ \mathtt{a}\ S$ $S \longrightarrow \bullet$
I_3	$= goto(I_2, S)$	$S \longrightarrow \mathtt{a}\ S\ \bullet$

SLR Action Table:

SLR Parsing Table

Grammar:

S'	$\longrightarrow$	S
S	$\longrightarrow$	$\mathtt{a}\ S$
S	$\longrightarrow$	ϵ

$FOLLOW(S) = \{\$\}$

Item Sets:

I_0	$= closure(\{S' \longrightarrow \bullet\ S\})$	$S' \longrightarrow \bullet\ S$ $S \longrightarrow \bullet\ \mathtt{a}\ S$ $S \longrightarrow \bullet$
I_1	$= goto(I_0, S)$	$S' \longrightarrow S\bullet$
I_2	$= goto(I_0, \mathtt{a})$	$S \longrightarrow \mathtt{a}\ \bullet\ S$ $S \longrightarrow \bullet\ \mathtt{a}\ S$ $S \longrightarrow \bullet$
I_3	$= goto(I_2, S)$	$S \longrightarrow \mathtt{a}\ S\ \bullet$

SLR Action Table:

	a	$
0	S, 2	

SLR Parsing Table

Grammar:

S'	$\longrightarrow$	S
S	$\longrightarrow$	$\mathtt{a}\ S$
S	$\longrightarrow$	ϵ

$FOLLOW(S) = \{\$\}$

Item Sets:

I_0	$= closure(\{S' \longrightarrow \bullet\ S\})$	$S' \longrightarrow \bullet\ S$ $S \longrightarrow \bullet\ \mathtt{a}\ S$ $S \longrightarrow \bullet$
I_1	$= goto(I_0, S)$	$S' \longrightarrow S\bullet$
I_2	$= goto(I_0, \mathtt{a})$	$S \longrightarrow \mathtt{a}\ \bullet\ S$ $S \longrightarrow \bullet\ \mathtt{a}\ S$ $S \longrightarrow \bullet$
I_3	$= goto(I_2, S)$	$S \longrightarrow \mathtt{a}\ S\ \bullet$

SLR Action Table:

	a	$
0	S, 2	R 3

SLR Parsing Table

Grammar:

S'	$\longrightarrow$	S
S	$\longrightarrow$	$\mathtt{a}\ S$
S	$\longrightarrow$	ϵ

$FOLLOW(S) = \{\$\}$

Item Sets:

I_0	$= closure(\{S' \longrightarrow \bullet\ S\})$	$S' \longrightarrow \bullet\ S$ $S \longrightarrow \bullet\ \mathtt{a}\ S$ $S \longrightarrow \bullet$
I_1	$= goto(I_0, S)$	$S' \longrightarrow S \bullet$
I_2	$= goto(I_0, \mathtt{a})$	$S \longrightarrow \mathtt{a} \bullet\ S$ $S \longrightarrow \bullet\ \mathtt{a}\ S$ $S \longrightarrow \bullet$
I_3	$= goto(I_2, S)$	$S \longrightarrow \mathtt{a}\ S \bullet$

SLR Action Table:

	a	$
0	S, 2	R 3
1		A

SLR Parsing Table

Grammar:

S'	$\longrightarrow$	S
S	$\longrightarrow$	a S
S	$\longrightarrow$	ϵ

$FOLLOW(S) = \{\$\}$

Item Sets:

I_0	$= closure(\{S' \longrightarrow \bullet\ S\})$	$S' \longrightarrow \bullet\ S$ $S \longrightarrow \bullet$ a S $S \longrightarrow \bullet$
I_1	$= goto(I_0, S)$	$S' \longrightarrow S\bullet$
I_2	$= goto(I_0,$ a$)$	$S \longrightarrow$ a $\bullet\ S$ $S \longrightarrow \bullet$ a S $S \longrightarrow \bullet$
I_3	$= goto(I_2, S)$	$S \longrightarrow$ a $S\ \bullet$

SLR Action Table:

	a	\$
0	S, 2	R 3
1		A
2	S, 2	

SLR Parsing Table

Grammar:

S'	$\longrightarrow$	S
S	$\longrightarrow$	$\mathtt{a}\ S$
S	$\longrightarrow$	ϵ

$FOLLOW(S) = \{\$\}$

Item Sets:

I_0	$= closure(\{S' \longrightarrow \bullet\ S\})$	$S' \longrightarrow \bullet\ S$ $S \longrightarrow \bullet\ \mathtt{a}\ S$ $S \longrightarrow \bullet$
I_1	$= goto(I_0, S)$	$S' \longrightarrow S\bullet$
I_2	$= goto(I_0, \mathtt{a})$	$S \longrightarrow \mathtt{a}\ \bullet\ S$ $S \longrightarrow \bullet\ \mathtt{a}\ S$ $S \longrightarrow \bullet$
I_3	$= goto(I_2, S)$	$S \longrightarrow \mathtt{a}\ S\ \bullet$

SLR Action Table:

	a	$
0	S, 2	R 3
1		A
2	S, 2	R 3
3		R 2

Deficiencies of SLR Parsing

- SLR(1) treats all occurrences of a RHS on stack as identical.
- Only a few of these reductions may lead to a successful parse.
- Example:

S	$\longrightarrow$	$A\mathtt{a}A\mathtt{b}$	$A \longrightarrow \epsilon$
S	$\longrightarrow$	$B\mathtt{b}B\mathtt{a}$	$B \longrightarrow \epsilon$

$I_0 = \{[S' \rightarrow \bullet\ S], [S \rightarrow \bullet\ A\mathtt{a}A\mathtt{b}], [S \rightarrow \bullet\ B\mathtt{b}B\mathtt{a}], [A \rightarrow \bullet\ ], [B \rightarrow \bullet\ ]\}$.

- Since $FOLLOW(A) = FOLLOW(B)$, we have reduce/reduce conflict in state 0.

LR(1) Item Sets

Construct LR(1) items of the form $A \longrightarrow \alpha \bullet \beta, \mathtt{a}$, which means:

> *The production $A \longrightarrow \alpha\beta$ can be applied when the next token on input stream is $\mathtt{a}$.*

S	$\longrightarrow$	$A\mathtt{a}A\mathtt{b}$	$A \longrightarrow \epsilon$
S	$\longrightarrow$	$B\mathtt{b}B\mathtt{a}$	$B \longrightarrow \epsilon$

An example LR(1) item set:

$$I_0 = \{[S' \rightarrow \bullet\ S, \$], [S \rightarrow \bullet\ A\mathtt{a}A\mathtt{b}, \$], [S \rightarrow \bullet\ B\mathtt{b}B\mathtt{a}, \$], [A \rightarrow \bullet\ , \mathtt{a}], [B \rightarrow \bullet\ , \mathtt{b}]\}.$$

LR(1) and LALR(1) Parsing

LR(1) parsing: Parse tables built using LR(1) item sets.

LALR(1) parsing: *Look Ahead* LR(1)

- Merge LR(1) item sets; then build parsing table.
- Typically, LALR(1) parsing tables are much smaller than LR(1) parsing table.

- $SLR(1) \subset LALR(1) \subset LR(1)$.
- $LL(1) \not\subseteq SLR(1)$, but $LL(1) \subset LR(1)$.

YACC

Yet **A**nother **C**ompiler **C**ompiler:
LALR(1) parser generator.

- Grammar rules written in a specification (`.y`) file, analogous to the regular definitions in a `lex` specification file.
- Yacc translates the specifications into a parsing function `yyparse()`.

$$\texttt{spec.y} \xrightarrow{\texttt{yacc}} \texttt{spec.tab.c}$$

- `yyparse()` calls `yylex()` whenever input tokens need to be consumed.
- `bison`: GNU variant of `yacc`.
- `ply`: Python's "yacc"; provides function `yacc()` that is similar to Yacc's `yyparse()`

Using Yacc

```
%{
  ... C headers (#include)
%}

... Yacc declarations:
       %token ...
       %union{...}
       precedences
%%
... Grammar rules with actions:

Expr:  Expr TOK_PLUS Expr
    |  Expr TOK_STAR Expr
    ;
%%
... C support functions
```

Parsing in PLY

See http://www.dabeaz.com/ply/ply.html

```
import ply.yacc as yacc
... import tokens from PLY/lexer

# precedences:
precedence = ( ('left', 'TOK_PLUS'), ('left', 'TOK_STAR') )

# Grammar rules with actions:
def p_expression_plus(p):
    'expr:  expr TOK_PLUS expr'
    pass   #action, if necessary

def p_expression_minus(p):
    'expr:  expr TOK_STAR expr'
    pass   #action, if necessary
```