

Compiler Construction

Lecture 6: SLR(1) and LR(1)

Jeremy Yallop

`jeremy.yallop@cl.cam.ac.uk`

Lent 2025

Bottom-up parsing components

Recap & plan



DFA

SLR(1)

Parsing
with states

SLR(1)
limits

LR(1)

Configurations $\$ \alpha, w \$$ with **actions**:

$$\begin{aligned} \$ \alpha, xw \$ & \xrightarrow{\text{shift } x} \$ \alpha x, w \$ \\ \$ \alpha \beta, w \$ & \xrightarrow{\text{reduce } A \rightarrow \beta} \$ \alpha A, w \$ \end{aligned}$$

Items $A \rightarrow \beta \bullet \gamma$ with **transitions**:

$$A \rightarrow \beta \bullet c \gamma \xrightarrow{c} A \rightarrow \beta c \bullet \gamma$$

$$A \rightarrow \beta \bullet B \gamma \xrightarrow{B} A \rightarrow \beta B \bullet \gamma$$

$$A \rightarrow \beta \bullet B \gamma \xrightarrow{\epsilon} B \rightarrow \bullet \alpha_i$$

A parsing algorithm:

$c := \text{NextToken}()$

while true:

$\alpha :=$ the stack

if $A \rightarrow \beta \bullet c \gamma \in \delta_G(q_0, \alpha)$
then **SHIFT** c ; $c := \text{NextToken}()$

if $A \rightarrow \beta \bullet \in \delta_G(q_0, \alpha)$
then **REDUCE** via $A \rightarrow \beta$

if $S \rightarrow \beta \bullet \in \delta_G(q_0, \alpha)$
then **ACCEPT** (if no more input)

if none of the above
then **ERROR**

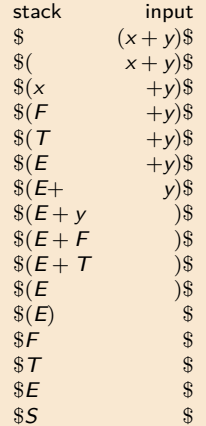
non-deterministic

Recap & plan

SLR(1)

Parsing with states

LR(1)



Example: $\delta_G(S \rightarrow \bullet E, (E = \{F \rightarrow (E\bullet)\}, \text{so shift })$.

Making the algorithm deterministic

Recap & plan



DFA

SLR(1)

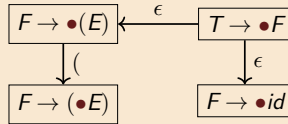
Parsing
with states

SLR(1)
limits

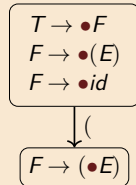
LR(1)

Two sources of **nondeterminism**:

1. The **NFA**



Solution: convert to a **DFA**



2. **Conflicts**

$\$ \alpha E + T, x \$$	$\xrightarrow{\text{reduce } E \rightarrow E + T}$
$\$ \alpha E + T, x \$$	$\xrightarrow{\text{reduce } E \rightarrow T}$
$\$ \alpha E + T, x \$$	$\xrightarrow{\text{shift } x}$

Solution: make a **deterministic choice**

using the input (lookahead)	using the grammar (FIRST and FOLLOW)
---------------------------------------	--

Two approaches: SLR(1) and LR(1)

The DFA

The easy part: NFA $\rightarrow$ DFA

Recap &
plan

DFA

● ○ ○

SLR(1)

Parsing
with states

SLR(1)
limits

LR(1)

Grammar G'_2

$S \rightarrow E$

$E \rightarrow E + T \mid T$

$T \rightarrow T * F \mid F$

$F \rightarrow (E) \mid \text{id}$

$S \rightarrow \bullet E$

$E \rightarrow \bullet E + T$

$E \rightarrow \bullet T$

$T \rightarrow \bullet T * F$

$T \rightarrow \bullet F$

$F \rightarrow \bullet (E)$

$F \rightarrow \bullet \text{id}$

DFA start state = ϵ -closure($\{S \rightarrow \bullet E\}$) =

Transition function:

$$\delta(I, X) = \epsilon\text{-closure}(\{A \rightarrow \alpha X \bullet \beta \mid A \rightarrow \alpha \bullet X \beta \in I\})$$

(**NB**: this is just the powerset/subset construction for converting NFAs to DFAs.)

Some DFA transitions for grammar G_2

Recap &
plan

DFA



SLR(1)

Parsing
with states

SLR(1)
limits

LR(1)

Grammar G'_2

$S \rightarrow E$

$E \rightarrow E + T \mid T$

$T \rightarrow T * F \mid F$

$F \rightarrow (E) \mid \text{id}$

$F \rightarrow (\bullet E)$

Some DFA transitions for grammar G_2

Recap &
plan

DFA



SLR(1)

Parsing
with states

SLR(1)
limits

LR(1)

Grammar G'_2

$S \rightarrow E$

$E \rightarrow E + T \mid T$

$T \rightarrow T * F \mid F$

$F \rightarrow (E) \mid \text{id}$

$F \rightarrow (\bullet E)$

$E \rightarrow \bullet E + T$

$E \rightarrow \bullet T$

Some DFA transitions for grammar G_2

Recap &
plan

DFA



SLR(1)

Parsing
with states

SLR(1)
limits

LR(1)

Grammar G'_2

$S \rightarrow E$

$E \rightarrow E + T \mid T$

$T \rightarrow T * F \mid F$

$F \rightarrow (E) \mid \text{id}$

$F \rightarrow (\bullet E)$

$E \rightarrow \bullet E + T$

$E \rightarrow \bullet T$

$T \rightarrow \bullet T * F$

$T \rightarrow \bullet F$

Some DFA transitions for grammar G_2

Recap &
plan

DFA



SLR(1)

Parsing
with states

SLR(1)
limits

LR(1)

Grammar G'_2

$S \rightarrow E$

$E \rightarrow E + T \mid T$

$T \rightarrow T * F \mid F$

$F \rightarrow (E) \mid \text{id}$

$F \rightarrow (\bullet E)$

$E \rightarrow \bullet E + T$

$E \rightarrow \bullet T$

$T \rightarrow \bullet T * F$

$T \rightarrow \bullet F$

$F \rightarrow \bullet (E)$

$F \rightarrow \bullet \text{id}$

Some DFA transitions for grammar G_2

Recap &
plan

DFA



SLR(1)

Parsing
with states

SLR(1)
limits

LR(1)

Grammar G'_2

$S \rightarrow E$

$E \rightarrow E + T \mid T$

$T \rightarrow T * F \mid F$

$F \rightarrow (E) \mid \text{id}$

$F \rightarrow (\bullet E)$

$E \rightarrow \bullet E + T$

$E \rightarrow \bullet T$

$T \rightarrow \bullet T * F$

$T \rightarrow \bullet F$

$F \rightarrow \bullet (E)$

$F \rightarrow \bullet \text{id}$

id

$F \rightarrow \text{id} \bullet$

Some DFA transitions for grammar G_2

Recap &
plan

DFA



SLR(1)

Parsing
with states

SLR(1)
limits

LR(1)

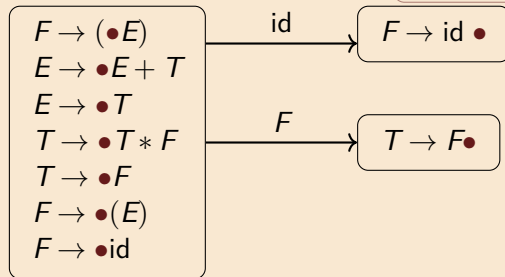
Grammar G'_2

$S \rightarrow E$

$E \rightarrow E + T \mid T$

$T \rightarrow T * F \mid F$

$F \rightarrow (E) \mid \text{id}$



Some DFA transitions for grammar G_2

Recap &
plan

DFA

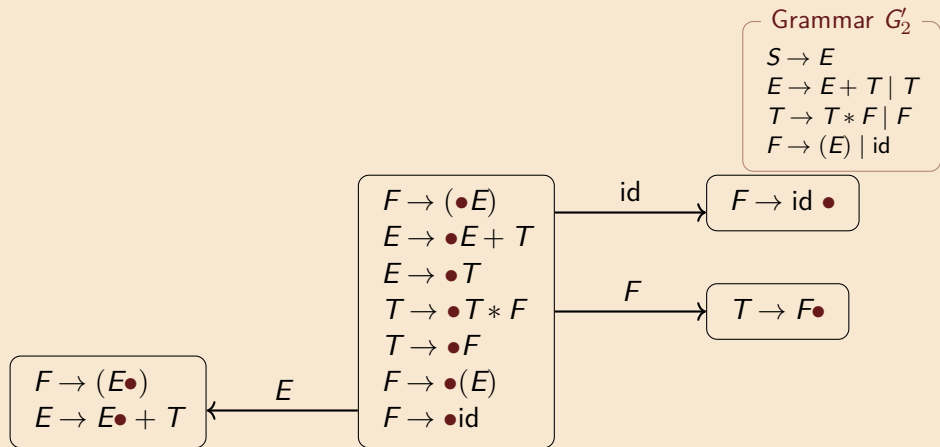


SLR(1)

Parsing
with states

SLR(1)
limits

LR(1)



Some DFA transitions for grammar G_2

Recap &
plan

DFA

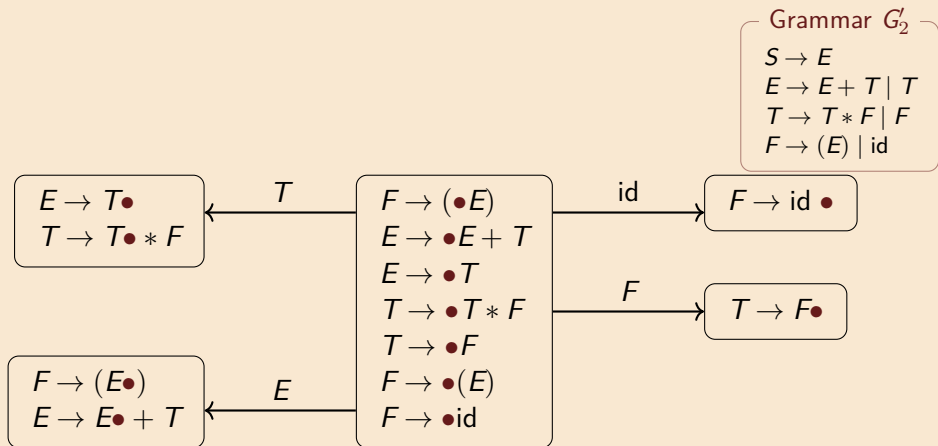


SLR(1)

Parsing
with states

SLR(1)
limits

LR(1)



Some DFA transitions for grammar G_2

Recap &
plan

DFA

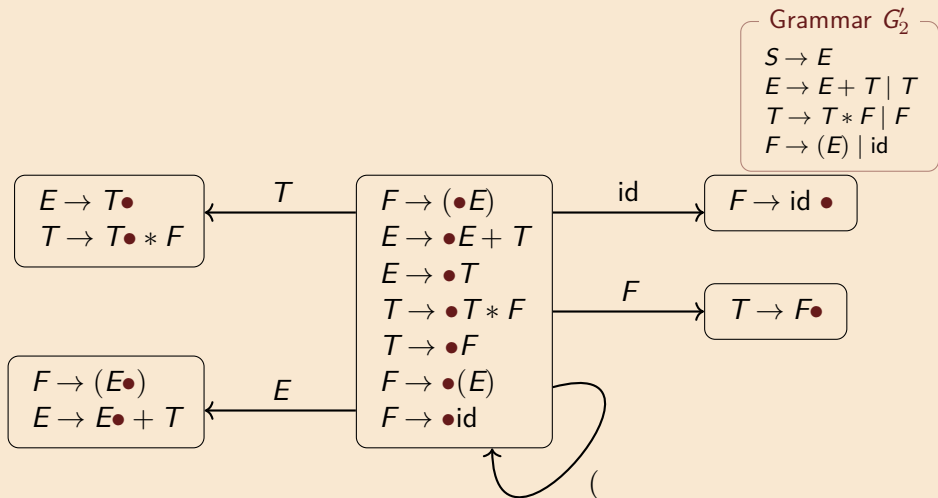


SLR(1)

Parsing
with states

SLR(1)
limits

LR(1)



Full DFA for the stack language of G_2

Recap &
plan

DFA

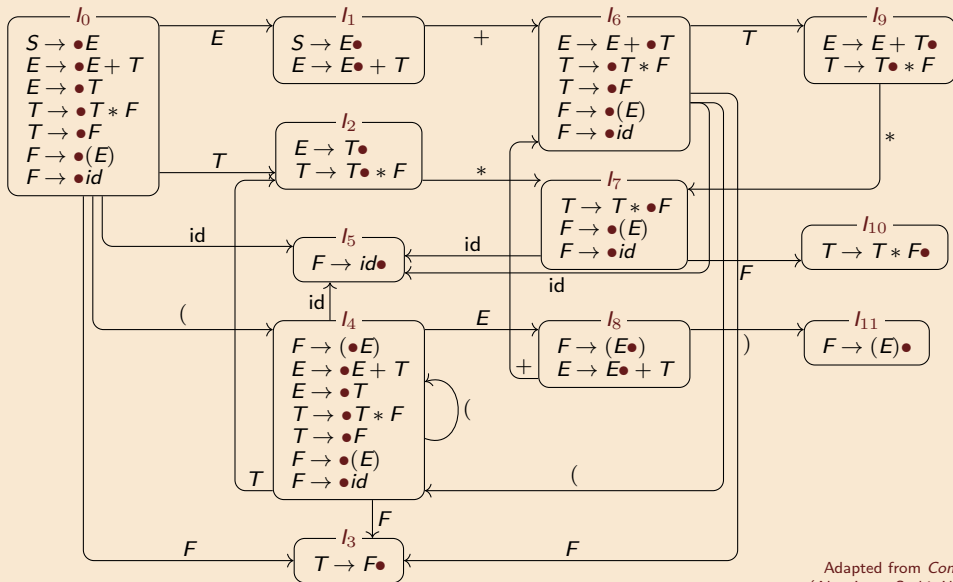
● ● ●

SLR(1)

Parsing
with states

SLR(1)
limits

LR(1)



Adapted from *Compilers*
(Aho, Lam, Sethi, Ullman)

SLR(1)

Resolving shift/reduce conflicts

Recap &
plan

DFA

SLR(1)

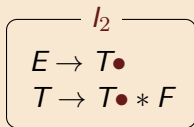


Parsing
with states

SLR(1)
limits

LR(1)

The state I_2 has a **shift/reduce conflict**.



The **Simple LR(1)** approach resolves the conflict using the next token c :

shift

if $c = *$

reduce with $E \rightarrow T$

only if $c \in \text{FOLLOW}(E) = \{ (, +, \$ \}$.

Deterministic SLR(1) parsing

Recap &
plan

DFA

SLR(1)



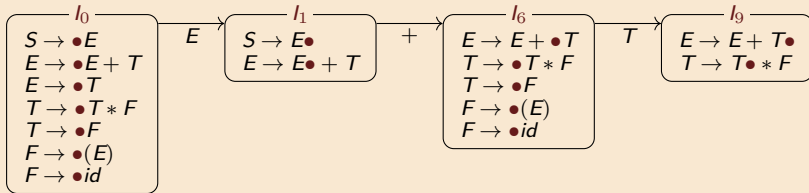
Parsing
with states

SLR(1)
limits

LR(1)

Recall: when the stack contains a , the parser is in state $\delta(l_0, a)$. For example,

$$\delta(l_0, E + T) = l_9$$



Let I be the current state and c the next token. Then:

1. When $A \rightarrow \beta \bullet c \gamma \in I$ then **shift** c onto stack
2. When $A \rightarrow \beta \bullet \in I$ and $c \in \text{FOLLOW}(A)$ then **reduce** with $A \rightarrow \beta$

Replay parsing of $(x + y)$ using SLR(1) actions

Recap &
plan

DFA

SLR(1)



Parsing
with states

SLR(1)
limits

LR(1)

stack,	input	state	action	reason
\$,	$(x + y)\$$	I_0	shift (	

Replay parsing of $(x + y)$ using SLR(1) actions

Recap &
plan

DFA

SLR(1)



Parsing
with states

SLR(1)
limits

LR(1)

stack,	input	state	action	reason
\$,	$(x + y)\$$	I_0	shift (	$F \rightarrow \bullet(E) \in \delta(I_0, \epsilon) = I_0$

Replay parsing of $(x + y)$ using SLR(1) actions

Recap &
plan

DFA

SLR(1)



Parsing
with states

SLR(1)
limits

LR(1)

stack,	input	state	action	reason
\$,	$(x + y)\$$	I_0	shift (	$F \rightarrow \bullet(E) \in \delta(I_0, \epsilon) = I_0$
\$ (,	$x + y)\$$	I_4	shift x	$F \rightarrow \bullet id \in \delta(I_0, () = I_4$

Replay parsing of $(x + y)$ using SLR(1) actions

Recap &
plan

DFA

SLR(1)



Parsing
with states

SLR(1)
limits

LR(1)

stack,	input	state	action	reason
\$,	$(x + y)\$$	I_0	shift (	$F \rightarrow \bullet(E) \in \delta(I_0, \epsilon) = I_0$
\$ (,	$x + y)\$$	I_4	shift x	$F \rightarrow \bullet \text{id} \in \delta(I_0, () = I_4$
$\$(x,$	$+y)\$$	I_5	reduce $F \rightarrow \text{id}$	$"+" \in \text{FOLLOW}(F)$

Replay parsing of $(x + y)$ using SLR(1) actions

Recap &
plan

DFA

SLR(1)



Parsing
with states

SLR(1)
limits

LR(1)

stack,	input	state	action	reason
\$,	$(x + y)\$$	l_0	shift (	$F \rightarrow \bullet(E) \in \delta(l_0, \epsilon) = l_0$
\$(\$,	$x + y)\$$	l_4	shift x	$F \rightarrow \bullet id \in \delta(l_0, () = l_4$
\$(x,	$+y)\$$	l_5	reduce $F \rightarrow id$	"+" $\in \text{FOLLOW}(F)$
\$(F,	$+y)\$$	l_3	reduce $T \rightarrow F$	"+" $\in \text{FOLLOW}(T)$

Replay parsing of $(x + y)$ using SLR(1) actions

Recap &
plan

DFA

SLR(1)



Parsing
with states

SLR(1)
limits

LR(1)

stack,	input	state	action	reason
\$,	$(x + y)$$	l_0	shift (	$F \rightarrow \bullet(E) \in \delta(l_0, \epsilon) = l_0$
\$(\$,	$x + y)$$	l_4	shift x	$F \rightarrow \bullet id \in \delta(l_0, () = l_4$
\$(x,	$+y)$$	l_5	reduce $F \rightarrow id$	$"+" \in FOLLOW(F)$
\$(F,	$+y)$$	l_3	reduce $T \rightarrow F$	$"+" \in FOLLOW(T)$
\$(T,	$+y)$$	l_2	reduce $E \rightarrow T$	$"+" \in FOLLOW(E)$

Replay parsing of $(x + y)$ using SLR(1) actions

Recap &
plan

DFA

SLR(1)



Parsing
with states

SLR(1)
limits

LR(1)

stack,	input	state	action	reason
\$,	$(x + y)\$$	l_0	shift (	$F \rightarrow \bullet(E) \in \delta(l_0, \epsilon) = l_0$
\$(\$,	$x + y)\$$	l_4	shift x	$F \rightarrow \bullet id \in \delta(l_0, () = l_4$
\$(x,	$+y)\$$	l_5	reduce $F \rightarrow id$	$"+" \in FOLLOW(F)$
\$(F,	$+y)\$$	l_3	reduce $T \rightarrow F$	$"+" \in FOLLOW(T)$
\$(T,	$+y)\$$	l_2	reduce $E \rightarrow T$	$"+" \in FOLLOW(E)$
\$(E,	$+y)\$$	l_8	shift +	$E \rightarrow E\bullet + T \in \delta(l_0, (E) = l_8$

Replay parsing of $(x + y)$ using SLR(1) actions

Recap &
plan

DFA

SLR(1)



Parsing
with states

SLR(1)
limits

LR(1)

stack,	input	state	action	reason
\$,	$(x + y)\$$	l_0	shift (	$F \rightarrow \bullet(E) \in \delta(l_0, \epsilon) = l_0$
\$ (,	$x + y)\$$	l_4	shift x	$F \rightarrow \bullet \text{id} \in \delta(l_0, () = l_4$
\$(x,	$+y)\$$	l_5	reduce $F \rightarrow \text{id}$	$"+" \in \text{FOLLOW}(F)$
\$(F,	$+y)\$$	l_3	reduce $T \rightarrow F$	$"+" \in \text{FOLLOW}(T)$
\$(T,	$+y)\$$	l_2	reduce $E \rightarrow T$	$"+" \in \text{FOLLOW}(E)$
\$(E,	$+y)\$$	l_8	shift +	$E \rightarrow E\bullet + T \in \delta(l_0, (E) = l_8$
\$(E+,	$y)\$$	l_6	shift y	$F \rightarrow \bullet \text{id} \in \delta(l_0, (E+) = l_6$

Replay parsing of $(x + y)$ using SLR(1) actions

Recap &
plan

DFA

SLR(1)



Parsing
with states

SLR(1)
limits

LR(1)

stack,	input	state	action	reason
\$,	$(x + y)$$	l_0	shift (	$F \rightarrow \bullet(E) \in \delta(l_0, \epsilon) = l_0$
\$ (,	$x + y)$$	l_4	shift x	$F \rightarrow \bullet \text{id} \in \delta(l_0, () = l_4$
\$(x,	$+y)$$	l_5	reduce $F \rightarrow \text{id}$	$"+" \in \text{FOLLOW}(F)$
\$(F,	$+y)$$	l_3	reduce $T \rightarrow F$	$"+" \in \text{FOLLOW}(T)$
\$(T,	$+y)$$	l_2	reduce $E \rightarrow T$	$"+" \in \text{FOLLOW}(E)$
\$(E,	$+y)$$	l_8	shift +	$E \rightarrow E\bullet + T \in \delta(l_0, (E) = l_8$
\$(E+,	$y)$$	l_6	shift y	$F \rightarrow \bullet \text{id} \in \delta(l_0, (E+) = l_6$
\$(E + y,	)\$	l_5	reduce $F \rightarrow \text{id}$	$"*)" \in \text{FOLLOW}(F)$

Replay parsing of $(x + y)$ using SLR(1) actions

Recap &
plan

DFA

SLR(1)



Parsing
with states

SLR(1)
limits

LR(1)

stack,	input	state	action	reason
\$,	$(x + y)$$	l_0	shift (	$F \rightarrow \bullet(E) \in \delta(l_0, \epsilon) = l_0$
\$ (,	$x + y)$$	l_4	shift x	$F \rightarrow \bullet \text{id} \in \delta(l_0, () = l_4$
\$(x,	$+y)$$	l_5	reduce $F \rightarrow \text{id}$	$"+" \in \text{FOLLOW}(F)$
\$(F,	$+y)$$	l_3	reduce $T \rightarrow F$	$"+" \in \text{FOLLOW}(T)$
\$(T,	$+y)$$	l_2	reduce $E \rightarrow T$	$"+" \in \text{FOLLOW}(E)$
\$(E,	$+y)$$	l_8	shift +	$E \rightarrow E\bullet + T \in \delta(l_0, (E) = l_8$
\$(E+,	$y)$$	l_6	shift y	$F \rightarrow \bullet \text{id} \in \delta(l_0, (E+) = l_6$
\$(E + y,	)\$	l_5	reduce $F \rightarrow \text{id}$	$")" \in \text{FOLLOW}(F)$
\$(E + F,	)\$	l_3	reduce $T \rightarrow F$	$")" \in \text{FOLLOW}(T)$

Replay parsing of $(x + y)$ using SLR(1) actions

Recap &
plan

DFA

SLR(1)



Parsing
with states

SLR(1)
limits

LR(1)

stack,	input	state	action	reason
\$,	$(x + y)$$	l_0	shift (	$F \rightarrow \bullet(E) \in \delta(l_0, \epsilon) = l_0$
\$ (,	$x + y)$$	l_4	shift x	$F \rightarrow \bullet \text{id} \in \delta(l_0, () = l_4$
\$(x,	$+y)$$	l_5	reduce $F \rightarrow \text{id}$	$"+" \in \text{FOLLOW}(F)$
\$(F,	$+y)$$	l_3	reduce $T \rightarrow F$	$"+" \in \text{FOLLOW}(T)$
\$(T,	$+y)$$	l_2	reduce $E \rightarrow T$	$"+" \in \text{FOLLOW}(E)$
\$(E,	$+y)$$	l_8	shift +	$E \rightarrow E\bullet + T \in \delta(l_0, (E) = l_8$
\$(E+,	$y)$$	l_6	shift y	$F \rightarrow \bullet \text{id} \in \delta(l_0, (E+) = l_6$
\$(E + y,	)\$	l_5	reduce $F \rightarrow \text{id}$	$")" \in \text{FOLLOW}(F)$
\$(E + F,	)\$	l_3	reduce $T \rightarrow F$	$")" \in \text{FOLLOW}(T)$
\$(E + T,	)\$	l_9	reduce $E \rightarrow E + T$	$")" \in \text{FOLLOW}(E)$

Replay parsing of $(x + y)$ using SLR(1) actions

Recap &
plan

DFA

SLR(1)



Parsing
with states

SLR(1)
limits

LR(1)

stack,	input	state	action	reason
\$,	$(x + y)$$	l_0	shift (	$F \rightarrow \bullet(E) \in \delta(l_0, \epsilon) = l_0$
\$ (,	$x + y)$$	l_4	shift x	$F \rightarrow \bullet id \in \delta(l_0, () = l_4$
\$(x,	$+y)$$	l_5	reduce $F \rightarrow id$	$"+" \in FOLLOW(F)$
\$(F,	$+y)$$	l_3	reduce $T \rightarrow F$	$"+" \in FOLLOW(T)$
\$(T,	$+y)$$	l_2	reduce $E \rightarrow T$	$"+" \in FOLLOW(E)$
\$(E,	$+y)$$	l_8	shift +	$E \rightarrow E\bullet + T \in \delta(l_0, (E) = l_8$
\$(E+,	$y)$$	l_6	shift y	$F \rightarrow \bullet id \in \delta(l_0, (E+) = l_6$
\$(E + y,	)\$	l_5	reduce $F \rightarrow id$	$")" \in FOLLOW(F)$
\$(E + F,	)\$	l_3	reduce $T \rightarrow F$	$")" \in FOLLOW(T)$
\$(E + T,	)\$	l_9	reduce $E \rightarrow E + T$	$")" \in FOLLOW(E)$
\$(E,	)\$	l_8	shift)	$E \rightarrow (E\bullet) \in \delta(l_0, (E) = l_8$

Replay parsing of $(x + y)$ using SLR(1) actions

Recap &
plan

DFA

SLR(1)



Parsing
with states

SLR(1)
limits

LR(1)

stack,	input	state	action	reason
\$,	$(x + y)$$	l_0	shift (	$F \rightarrow \bullet(E) \in \delta(l_0, \epsilon) = l_0$
\$(\$,	$x + y)$$	l_4	shift x	$F \rightarrow \bullet id \in \delta(l_0, () = l_4$
\$(x,	$+y)$$	l_5	reduce $F \rightarrow id$	$"+" \in FOLLOW(F)$
\$(F,	$+y)$$	l_3	reduce $T \rightarrow F$	$"+" \in FOLLOW(T)$
\$(T,	$+y)$$	l_2	reduce $E \rightarrow T$	$"+" \in FOLLOW(E)$
\$(E,	$+y)$$	l_8	shift +	$E \rightarrow E\bullet + T \in \delta(l_0, (E) = l_8$
\$(E+,	$y)$$	l_6	shift y	$F \rightarrow \bullet id \in \delta(l_0, (E+) = l_6$
\$(E + y,	)\$	l_5	reduce $F \rightarrow id$	$")" \in FOLLOW(F)$
\$(E + F,	)\$	l_3	reduce $T \rightarrow F$	$")" \in FOLLOW(T)$
\$(E + T,	)\$	l_9	reduce $E \rightarrow E + T$	$")" \in FOLLOW(E)$
\$(E,	)\$	l_8	shift)	$E \rightarrow (E\bullet) \in \delta(l_0, (E) = l_8$
\$(E),	\$	l_{11}	reduce $F \rightarrow (E)$	$"$" \in FOLLOW(F)$

Replay parsing of $(x + y)$ using SLR(1) actions

Recap &
plan

DFA

SLR(1)



Parsing
with states

SLR(1)
limits

LR(1)

stack,	input	state	action	reason
\$,	$(x + y)$$	l_0	shift (	$F \rightarrow \bullet(E) \in \delta(l_0, \epsilon) = l_0$
$$(,$	$x + y)$$	l_4	shift x	$F \rightarrow \bullet id \in \delta(l_0, () = l_4$
$$(x,$	$+y)$$	l_5	reduce $F \rightarrow id$	$"+" \in FOLLOW(F)$
$$(F,$	$+y)$$	l_3	reduce $T \rightarrow F$	$"+" \in FOLLOW(T)$
$$(T,$	$+y)$$	l_2	reduce $E \rightarrow T$	$"+" \in FOLLOW(E)$
$$(E,$	$+y)$$	l_8	shift +	$E \rightarrow E\bullet + T \in \delta(l_0, (E) = l_8$
$$(E+,$	$y)$$	l_6	shift y	$F \rightarrow \bullet id \in \delta(l_0, (E+) = l_6$
$$(E + y,$	$)$$	l_5	reduce $F \rightarrow id$	$")" \in FOLLOW(F)$
$$(E + F,$	$)$$	l_3	reduce $T \rightarrow F$	$")" \in FOLLOW(T)$
$$(E + T,$	$)$$	l_9	reduce $E \rightarrow E + T$	$")" \in FOLLOW(E)$
$$(E,$	$)$$	l_8	shift)	$E \rightarrow (E\bullet) \in \delta(l_0, (E) = l_8$
$$(E),$	$$$	l_{11}	reduce $F \rightarrow (E)$	$"$" \in FOLLOW(F)$
$$F,$	$$$	l_3	reduce $T \rightarrow F$	$"$" \in FOLLOW(T)$

Replay parsing of $(x + y)$ using SLR(1) actions

Recap &
plan

DFA

SLR(1)



Parsing
with states

SLR(1)
limits

LR(1)

stack,	input	state	action	reason
\$,	$(x + y)$$	l_0	shift (	$F \rightarrow \bullet(E) \in \delta(l_0, \epsilon) = l_0$
\$(\$,	$x + y)$$	l_4	shift x	$F \rightarrow \bullet id \in \delta(l_0, () = l_4$
\$(x,	$+y)$$	l_5	reduce $F \rightarrow id$	$"+" \in FOLLOW(F)$
\$(F,	$+y)$$	l_3	reduce $T \rightarrow F$	$"+" \in FOLLOW(T)$
\$(T,	$+y)$$	l_2	reduce $E \rightarrow T$	$"+" \in FOLLOW(E)$
\$(E,	$+y)$$	l_8	shift +	$E \rightarrow E\bullet + T \in \delta(l_0, (E) = l_8$
\$(E+,	$y)$$	l_6	shift y	$F \rightarrow \bullet id \in \delta(l_0, (E+) = l_6$
\$(E + y,	)\$	l_5	reduce $F \rightarrow id$	$")" \in FOLLOW(F)$
\$(E + F,	)\$	l_3	reduce $T \rightarrow F$	$")" \in FOLLOW(T)$
\$(E + T,	)\$	l_9	reduce $E \rightarrow E + T$	$")" \in FOLLOW(E)$
\$(E,	)\$	l_8	shift)	$E \rightarrow (E\bullet) \in \delta(l_0, (E) = l_8$
\$(E),	\$	l_{11}	reduce $F \rightarrow (E)$	$"$" \in FOLLOW(F)$
\$F,	\$	l_3	reduce $T \rightarrow F$	$"$" \in FOLLOW(T)$
\$T,	\$	l_2	reduce $F \rightarrow E$	$"$" \in FOLLOW(E)$

Replay parsing of $(x + y)$ using SLR(1) actions

Recap &
plan

DFA

SLR(1)



Parsing
with states

SLR(1)
limits

LR(1)

stack,	input	state	action	reason
\$,	$(x + y)$$	l_0	shift (	$F \rightarrow \bullet(E) \in \delta(l_0, \epsilon) = l_0$
$$(,$	$x + y)$$	l_4	shift x	$F \rightarrow \bullet id \in \delta(l_0, () = l_4$
$$(x,$	$+y)$$	l_5	reduce $F \rightarrow id$	$"+" \in FOLLOW(F)$
$$(F,$	$+y)$$	l_3	reduce $T \rightarrow F$	$"+" \in FOLLOW(T)$
$$(T,$	$+y)$$	l_2	reduce $E \rightarrow T$	$"+" \in FOLLOW(E)$
$$(E,$	$+y)$$	l_8	shift +	$E \rightarrow E\bullet + T \in \delta(l_0, (E) = l_8$
$$(E+,$	$y)$$	l_6	shift y	$F \rightarrow \bullet id \in \delta(l_0, (E+) = l_6$
$$(E + y,$	)\$	l_5	reduce $F \rightarrow id$	$")" \in FOLLOW(F)$
$$(E + F,$	)\$	l_3	reduce $T \rightarrow F$	$")" \in FOLLOW(T)$
$$(E + T,$	)\$	l_9	reduce $E \rightarrow E + T$	$")" \in FOLLOW(E)$
$$(E,$	)\$	l_8	shift)	$E \rightarrow (E\bullet) \in \delta(l_0, (E) = l_8$
$$(E),$	\$	l_{11}	reduce $F \rightarrow (E)$	$"$" \in FOLLOW(F)$
$$F,$	\$	l_3	reduce $T \rightarrow F$	$"$" \in FOLLOW(T)$
$$T,$	\$	l_2	reduce $F \rightarrow E$	$"$" \in FOLLOW(E)$
$$E,$	\$	l_1	reduce $S \rightarrow E$	$"$" \in FOLLOW(S)$

Parsing with states

Idea: don't restart the DFA at every step

Recap &
plan

DFA

SLR(1)

Parsing
with states



SLR(1)
limits

LR(1)

Previous approach

Maintain a **stack of symbols**

\$ (E + id

At each step:

Use the **full stack**
to find items

New approach

Maintain a **stack of states**

0 4 1 6 5

At each step:

Use the **top of the stack**
to find actions

LR parsing with DFA states on the stack

Recap &
plan

DFA

SLR(1)

Parsing
with states



SLR(1)
limits

LR(1)

```
tok := NextToken()
```

```
while true:
```

```
    state := TopStackState()
```

```
    if ACTION[state, tok] = SHIFT state
```

```
        then push state
```

```
            tok := NextToken()
```

```
    else if ACTION[state, tok] = REDUCE  $A \rightarrow \beta$ 
```

```
        then pop  $|\beta|$  states
```

```
            push GOTO[TopStackState(), A]
```

```
    else if ACTION[state, tok] = ACCEPT
```

```
        then accept and exit
```

```
    else ERROR
```

Constructing ACTION and GOTO for SLR(1)

Recap &
plan

DFA

SLR(1)

Parsing
with states



SLR(1)
limits

LR(1)

If $A \rightarrow \alpha \bullet a \beta \in I_i$ and $\delta(I_i, a) = I_j$

then $\text{ACTION}[i, a] = \text{SHIFT } j$.

If $A \rightarrow \alpha \bullet \in I_i$ and $A \neq S$

then for each $a \in \text{FOLLOW}(A)$,

$\text{ACTION}[i, a] = \text{REDUCE } A \rightarrow \alpha$

If $S \rightarrow \alpha \bullet \in I_i$

then $\text{ACTION}[i, \$] = \text{ACCEPT}$

If $\delta(I_i, A) = I_j$

then $\text{GOTO}[i, A] = j$

(ACTION resolves conflicts; GOTO records nonterminal transitions $I_i \xrightarrow{A} I_j$)

ACTION and GOTO for G'_2

Recap &
plan

DFA

SLR(1)

Parsing
with states



SLR(1)
limits

LR(1)

STATE	ACTION						GOTO		
	id	+	*	(	)	\$	<i>E</i>	<i>T</i>	<i>F</i>
0	s5			s4			1	2	3
1		s6				acc			
2		r2	s7		r2	r2			
3		r4	r4		r4	r4			
4	s5			s4			8	2	3
5		r6	r6		r6	r6			
6	s5			s4				9	3
7	s5			s4					10
8		s6			s11				
9		r1	s7		r1	r1			
10		r3	r3		r3	r3			
11		r5	r5		r5	r5			

Adapted from *Compilers*
(Aho, Lam, Sethi, Ullman)

Example parse

Recap &
plan

DFA

SLR(1)

Parsing
with states



SLR(1)
limits

LR(1)

STACK	SYMBOLS	INPUT	ACTION
0		id * id + id\$	shift

Example parse

Recap &
plan

DFA

SLR(1)

Parsing
with states



SLR(1)
limits

LR(1)

STACK	SYMBOLS	INPUT	ACTION
0		id * id + id\$	shift
0 5	id	* id + id\$	reduce $F \rightarrow id$ GOTO[0, F] = 3

Example parse

Recap &
plan

DFA

SLR(1)

Parsing
with states



SLR(1)
limits

LR(1)

STACK	SYMBOLS	INPUT	ACTION
0		id * id + id\$	shift
0 5	id	* id + id\$	reduce $F \rightarrow id$ GOTO[0, F] = 3
0 3	F	* id + id\$	reduce $T \rightarrow F$ GOTO[0, T] = 2

Example parse

Recap &
plan

DFA

SLR(1)

Parsing
with states



SLR(1)
limits

LR(1)

STACK	SYMBOLS	INPUT	ACTION
0		id * id + id\$	shift
0 5	id	* id + id\$	reduce $F \rightarrow id$ GOTO[0, F] = 3
0 3	F	* id + id\$	reduce $T \rightarrow F$ GOTO[0, T] = 2
0 2	T	* id + id\$	shift

Example parse

Recap &
plan

DFA

SLR(1)

Parsing
with states

SLR(1)
limits

LR(1)

STACK	SYMBOLS	INPUT	ACTION
0		id * id + id\$	shift
0 5	id	* id + id\$	reduce $F \rightarrow id$ GOTO[0, F] = 3
0 3	F	* id + id\$	reduce $T \rightarrow F$ GOTO[0, T] = 2
0 2	T	* id + id\$	shift
0 2 7	$T *$	id + id\$	shift

Example parse

Recap &
plan

DFA

SLR(1)

Parsing
with states



SLR(1)
limits

LR(1)

STACK	SYMBOLS	INPUT	ACTION	
0		id * id + id\$	shift	
0 5	id	* id + id\$	reduce $F \rightarrow id$	GOTO[0, F] = 3
0 3	F	* id + id\$	reduce $T \rightarrow F$	GOTO[0, T] = 2
0 2	T	* id + id\$	shift	
0 2 7	$T *$	id + id\$	shift	
0 2 7 5	$T * id$	+ id\$	reduce $F \rightarrow id$	GOTO[7, F] = 10

Example parse

Recap &
plan

DFA

SLR(1)

Parsing
with states



SLR(1)
limits

LR(1)

STACK	SYMBOLS	INPUT	ACTION
0		id * id + id\$	shift
0 5	id	* id + id\$	reduce $F \rightarrow id$ GOTO[0, F] = 3
0 3	F	* id + id\$	reduce $T \rightarrow F$ GOTO[0, T] = 2
0 2	T	* id + id\$	shift
0 2 7	$T *$	id + id\$	shift
0 2 7 5	$T * id$	+ id\$	reduce $F \rightarrow id$ GOTO[7, F] = 10
0 2 7 10	$T * F$	+ id\$	reduce $T \rightarrow T * F$ GOTO[0, T] = 2

Example parse

Recap &
plan

DFA

SLR(1)

Parsing
with states

SLR(1)
limits

LR(1)

STACK	SYMBOLS	INPUT	ACTION	
0		id * id + id\$	shift	
0 5	id	* id + id\$	reduce $F \rightarrow id$	GOTO[0, F] = 3
0 3	F	* id + id\$	reduce $T \rightarrow F$	GOTO[0, T] = 2
0 2	T	* id + id\$	shift	
0 2 7	$T *$	id + id\$	shift	
0 2 7 5	$T * id$	+ id\$	reduce $F \rightarrow id$	GOTO[7, F] = 10
0 2 7 10	$T * F$	+ id\$	reduce $T \rightarrow T * F$	GOTO[0, T] = 2
0 2	T	+ id\$	reduce $E \rightarrow T$	GOTO[0, E] = 1

Example parse

Recap &
plan

DFA

SLR(1)

Parsing
with states

SLR(1)
limits

LR(1)

STACK	SYMBOLS	INPUT	ACTION
0		id * id + id\$	shift
0 5	id	* id + id\$	reduce $F \rightarrow id$ GOTO[0, F] = 3
0 3	F	* id + id\$	reduce $T \rightarrow F$ GOTO[0, T] = 2
0 2	T	* id + id\$	shift
0 2 7	$T *$	id + id\$	shift
0 2 7 5	$T * id$	+ id\$	reduce $F \rightarrow id$ GOTO[7, F] = 10
0 2 7 10	$T * F$	+ id\$	reduce $T \rightarrow T * F$ GOTO[0, T] = 2
0 2	T	+ id\$	reduce $E \rightarrow T$ GOTO[0, E] = 1
0 1	E	+ id\$	shift

Example parse

Recap &
plan

DFA

SLR(1)

Parsing
with states



SLR(1)
limits

LR(1)

STACK	SYMBOLS	INPUT	ACTION
0		id * id + id\$	shift
0 5	id	* id + id\$	reduce $F \rightarrow id$ GOTO[0, F] = 3
0 3	F	* id + id\$	reduce $T \rightarrow F$ GOTO[0, T] = 2
0 2	T	* id + id\$	shift
0 2 7	$T *$	id + id\$	shift
0 2 7 5	$T * id$	+ id\$	reduce $F \rightarrow id$ GOTO[7, F] = 10
0 2 7 10	$T * F$	+ id\$	reduce $T \rightarrow T * F$ GOTO[0, T] = 2
0 2	T	+ id\$	reduce $E \rightarrow T$ GOTO[0, E] = 1
0 1	E	+ id\$	shift
0 1 6	$E +$	id\$	shift

Example parse

Recap &
plan

DFA

SLR(1)

Parsing
with states



SLR(1)
limits

LR(1)

STACK	SYMBOLS	INPUT	ACTION
0		id * id + id\$	shift
0 5	id	* id + id\$	reduce $F \rightarrow id$ GOTO[0, F] = 3
0 3	F	* id + id\$	reduce $T \rightarrow F$ GOTO[0, T] = 2
0 2	T	* id + id\$	shift
0 2 7	$T *$	id + id\$	shift
0 2 7 5	$T * id$	+ id\$	reduce $F \rightarrow id$ GOTO[7, F] = 10
0 2 7 10	$T * F$	+ id\$	reduce $T \rightarrow T * F$ GOTO[0, T] = 2
0 2	T	+ id\$	reduce $E \rightarrow T$ GOTO[0, E] = 1
0 1	E	+ id\$	shift
0 1 6	$E +$	id\$	shift
0 1 6 5	$E + id$	\$	reduce $F \rightarrow id$ GOTO[6, F] = 3

Example parse

Recap &
plan

DFA

SLR(1)

Parsing
with states



SLR(1)
limits

LR(1)

STACK	SYMBOLS	INPUT	ACTION
0		id * id + id\$	shift
0 5	id	* id + id\$	reduce $F \rightarrow id$ GOTO[0, F] = 3
0 3	F	* id + id\$	reduce $T \rightarrow F$ GOTO[0, T] = 2
0 2	T	* id + id\$	shift
0 2 7	$T *$	id + id\$	shift
0 2 7 5	$T * id$	+ id\$	reduce $F \rightarrow id$ GOTO[7, F] = 10
0 2 7 10	$T * F$	+ id\$	reduce $T \rightarrow T * F$ GOTO[0, T] = 2
0 2	T	+ id\$	reduce $E \rightarrow T$ GOTO[0, E] = 1
0 1	E	+ id\$	shift
0 1 6	$E +$	id\$	shift
0 1 6 5	$E + id$	\$	reduce $F \rightarrow id$ GOTO[6, F] = 3
0 1 6 3	$E + F$	\$	reduce $T \rightarrow F$ GOTO[6, T] = 9

Example parse

Recap &
plan

DFA

SLR(1)

Parsing
with states

SLR(1)
limits

LR(1)

STACK	SYMBOLS	INPUT	ACTION
0		id * id + id\$	shift
0 5	id	* id + id\$	reduce $F \rightarrow id$ GOTO[0, F] = 3
0 3	F	* id + id\$	reduce $T \rightarrow F$ GOTO[0, T] = 2
0 2	T	* id + id\$	shift
0 2 7	$T *$	id + id\$	shift
0 2 7 5	$T * id$	+ id\$	reduce $F \rightarrow id$ GOTO[7, F] = 10
0 2 7 10	$T * F$	+ id\$	reduce $T \rightarrow T * F$ GOTO[0, T] = 2
0 2	T	+ id\$	reduce $E \rightarrow T$ GOTO[0, E] = 1
0 1	E	+ id\$	shift
0 1 6	$E +$	id\$	shift
0 1 6 5	$E + id$	\$	reduce $F \rightarrow id$ GOTO[6, F] = 3
0 1 6 3	$E + F$	\$	reduce $T \rightarrow F$ GOTO[6, T] = 9
0 1 6 9	$E + T$	\$	reduce $E \rightarrow E + T$ GOTO[0, E] = 1

Example parse

Recap &
plan

DFA

SLR(1)

Parsing
with states



SLR(1)
limits

LR(1)

STACK	SYMBOLS	INPUT	ACTION
0		id * id + id\$	shift
0 5	id	* id + id\$	reduce $F \rightarrow id$ GOTO[0, F] = 3
0 3	F	* id + id\$	reduce $T \rightarrow F$ GOTO[0, T] = 2
0 2	T	* id + id\$	shift
0 2 7	$T *$	id + id\$	shift
0 2 7 5	$T * id$	+ id\$	reduce $F \rightarrow id$ GOTO[7, F] = 10
0 2 7 10	$T * F$	+ id\$	reduce $T \rightarrow T * F$ GOTO[0, T] = 2
0 2	T	+ id\$	reduce $E \rightarrow T$ GOTO[0, E] = 1
0 1	E	+ id\$	shift
0 1 6	$E +$	id\$	shift
0 1 6 5	$E + id$	\$	reduce $F \rightarrow id$ GOTO[6, F] = 3
0 1 6 3	$E + F$	\$	reduce $T \rightarrow F$ GOTO[6, T] = 9
0 1 6 9	$E + T$	\$	reduce $E \rightarrow E + T$ GOTO[0, E] = 1
0 1	E	\$	accept

The limits of SLR(1)

Recap &
plan

DFA

SLR(1)

Parsing
with states

SLR(1)
limits



LR(1)

A new example grammar (for *assignment expressions*):

$$G_4 = \langle N_4, T_4, P_4, S' \rangle$$

$$N_4 = \{S', S, L, R\}$$

$$T_4 = \{*, :=, id\}$$

$$\begin{aligned} P_4 : \quad & S' \rightarrow S \$ \\ & S \rightarrow L := R \mid R \\ & L \rightarrow *R \mid id \\ & R \rightarrow L \end{aligned}$$

LR(0) DFA for grammar G_4

Recap &
plan

DFA

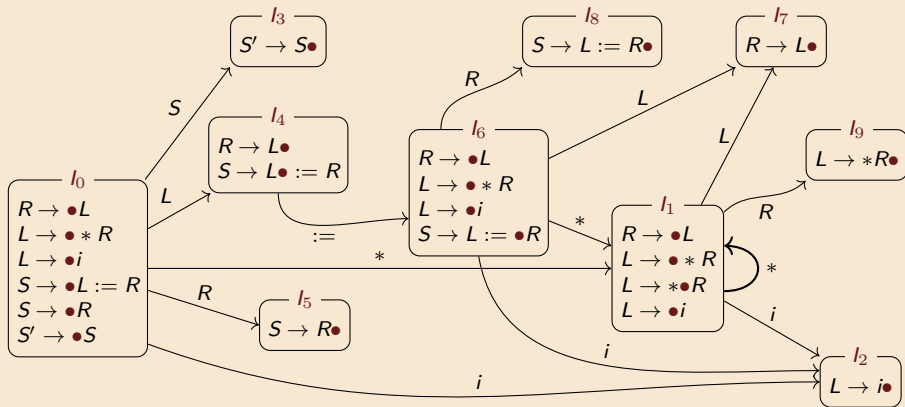
SLR(1)

Parsing
with states

SLR(1)
limits



LR(1)



LR(0) DFA for grammar G_4

Recap &
plan

DFA

SLR(1)

Parsing
with states

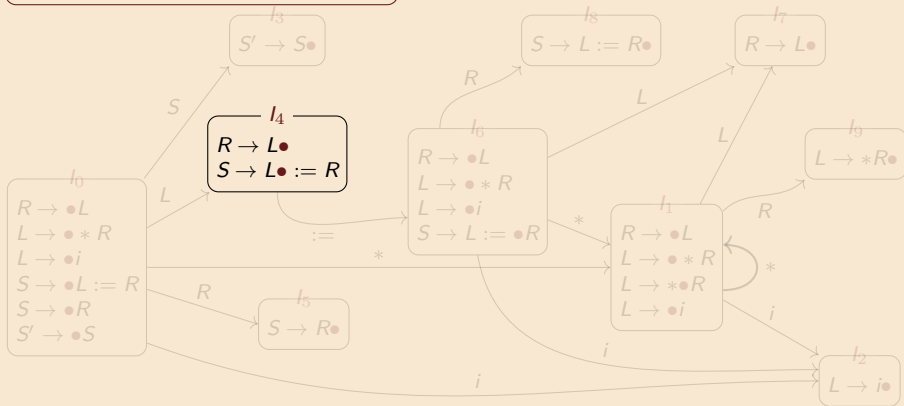
SLR(1)
limits



LR(1)

Ambiguity

In state 4 there is a shift/reduce conflict
between $S \rightarrow L \bullet := R$ and $R \rightarrow L \bullet$



SLR(1) cannot resolve this conflict.

Recap &
plan

DFA

SLR(1)

Parsing
with states

SLR(1)
limits



LR(1)

Suppose we see $:=$ in the input in state I_4 . Then:

shifting is valid

Since $S \rightarrow L \bullet := R \in I_4$,
and $\delta(I_4, " := ") = I_6$,

$\text{ACTION}[4, " := "] = \text{shift } 6$

reducing is valid

Since $R \rightarrow L \bullet \in I_4$
and $" := " \in \text{FOLLOW}(R)$,

$\text{ACTION}[4, " := "] = \text{reduce } R \rightarrow L$

LR(1)

Recap &
plan

DFA

SLR(1)

Parsing
with states

SLR(1)
limits

LR(1)



With SLR(1) there may be

shift-reduce
conflicts

reduce-reduce
conflicts

when ACTION and GOTO are not uniquely defined.

Options: fix the grammar, or use a more powerful parsing technique.

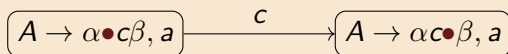
LR(1) parsing extends LR(0) items with an explicit lookahead token a :

$$\begin{array}{c} \text{LR(1) item} \\ \underbrace{A \rightarrow \alpha \bullet \beta, a} \\ \underbrace{A \rightarrow \alpha \bullet \beta}_{\text{LR(0) item}} \quad \underbrace{a}_{\text{lookahead token}} \end{array}$$

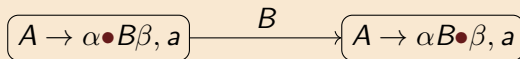
Define an NFA with LR (1) items as states

Recap &
plan

DFA



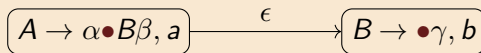
SLR(1)



Parsing
with states

For each $b \in \text{FIRST}(\beta a)$:

SLR(1)
limits



LR(1)



LR(1) DFA for grammar G_4

Recap &
plan

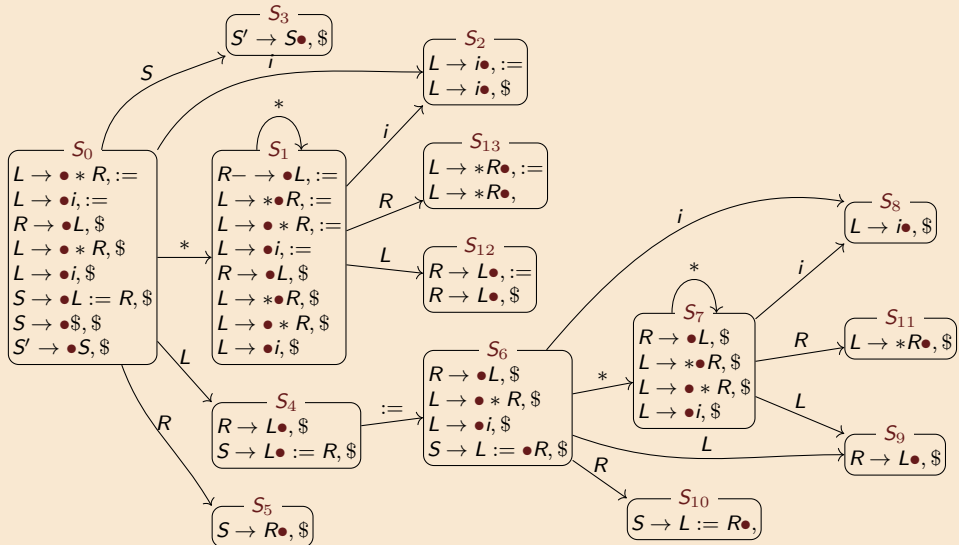
DFA

SLR(1)

Parsing
with states

SLR(1)
limits

LR(1)



LR(1) DFA for grammar G_4

Recap & plan

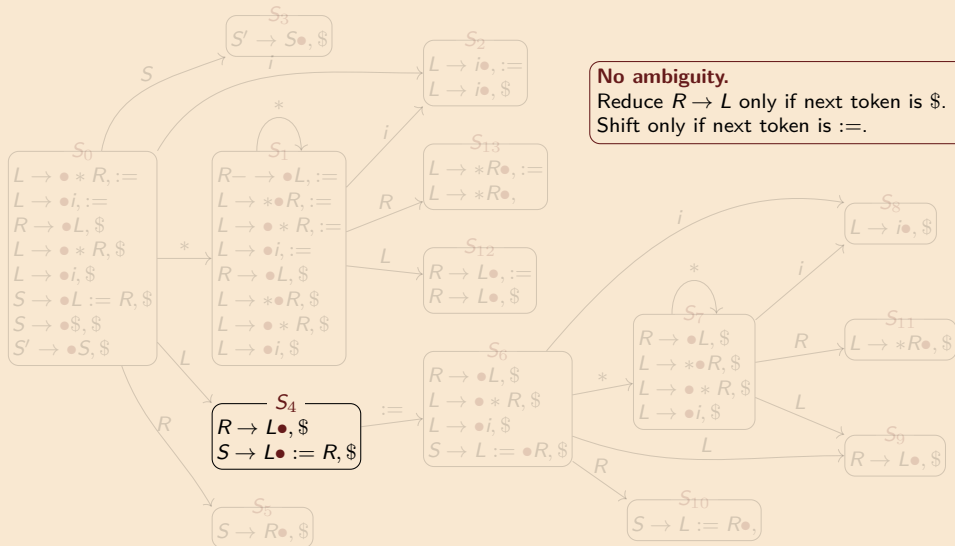
DFA

SLR(1)

Parsing with states

SLR(1) limits

LR(1)



Constructing ACTION and GOTO for LR(1)

Recap &
plan

DFA

SLR(1)

Parsing
with states

SLR(1)
limits

LR(1)

If $[A \rightarrow \alpha \bullet a \beta, b] \in I_i$ and $\delta(I_i, a) = I_j$

then $\text{ACTION}[i, a] = \text{SHIFT } j$.

If $[A \rightarrow \alpha \bullet, b] \in I_i$ and $A \neq S$,

then $\text{ACTION}[i, b] = \text{REDUCE } A \rightarrow \alpha$

If $[S \rightarrow \alpha \bullet, \$] \in I_i$

then $\text{ACTION}[i, \$] = \text{ACCEPT}$

If $\delta(I_i, A) = I_j$

then $\text{GOTO}[i, A] = j$.

Key change from SLR(1): use **lookahead** (i.e. FIRST), not FOLLOW, to select **REDUCE** actions



Recap &
plan

DFA

SLR(1)

Parsing
with states

SLR(1)
limits

LR(1)

SLR(1)

If $[A \rightarrow \alpha \bullet] \in I_i$ and $A \neq S$
then for each $a \in \text{FOLLOW}(A)$,
 $\text{ACTION}[i, a] = \text{reduce } A \rightarrow \alpha$

LR(1)

If $[A \rightarrow \alpha \bullet, b] \in I_i$ and $A \neq S$
then
 $\text{ACTION}[i, b] = \text{reduce } A \rightarrow \alpha$

NB: b used *only* for reductions, not for shifts

LR(1) more powerful than SLR(1)

LR(1) DFA may have a very large number of states.

LALR¹ optimises DFA, collapsing states (but can give strange error messages)

¹Implemented in yacc; not covered here

Next time: translation