

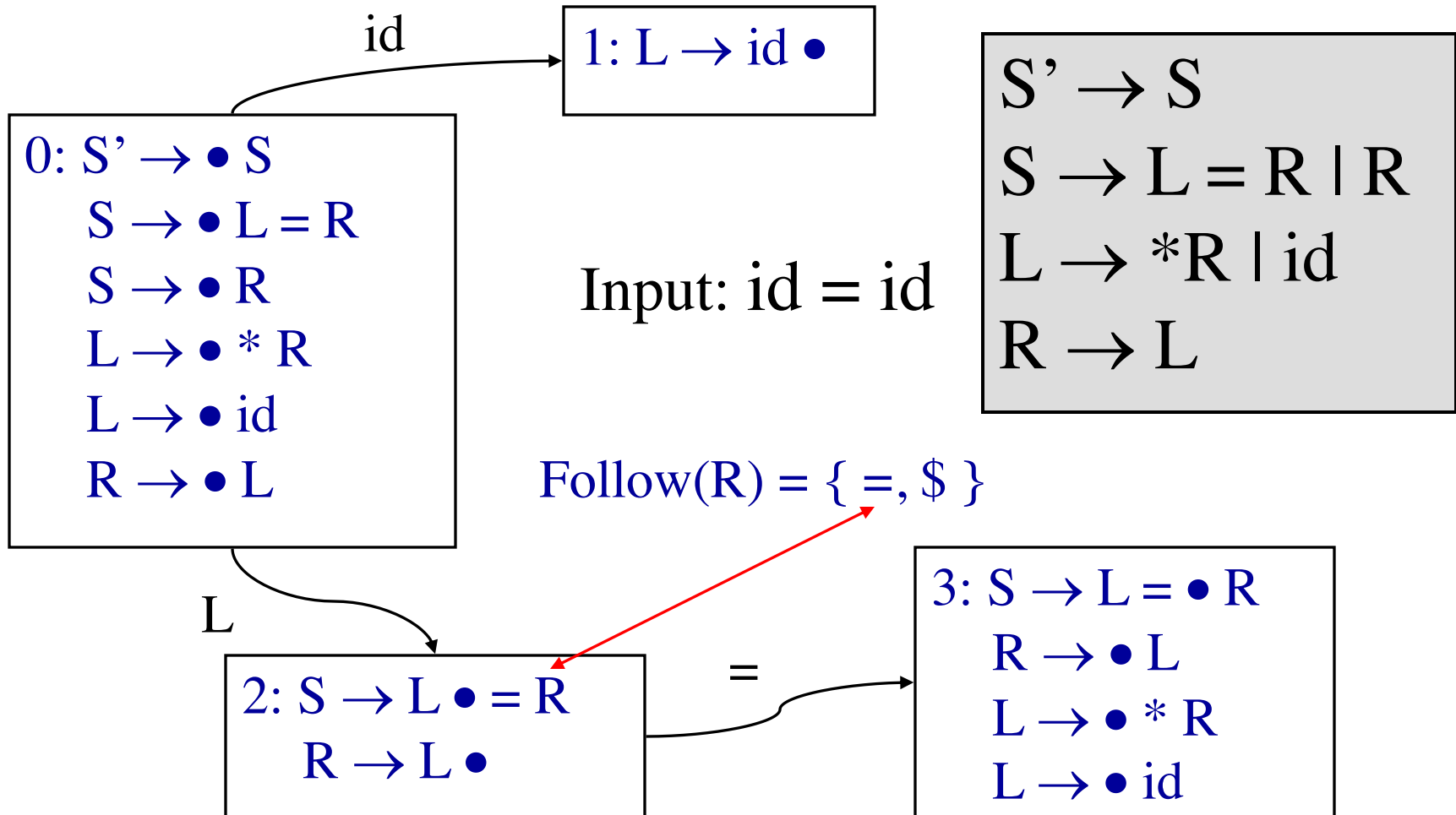
LR Parsing

CMPT 379: Compilers

Instructor: Anoop Sarkar

anoopsarkar.github.io/compilers-class

SLR limitation: lack of context



$S' \rightarrow S$

$S \rightarrow L = R \mid R$

$L \rightarrow *R \mid \text{id}$

$R \rightarrow L$

$\text{Follow}(R) = \{ =, \$ \}$

2: $S \rightarrow L \bullet = R$

$R \rightarrow L \bullet$

Find all lookaheads
for reduce $R \rightarrow L \bullet$

S'
|
 S
|
 R
|
 L
|
 id

\$

S'
|
 S
|
 L $=$ R
| | |
 id L id

\$

S'
|
 S
|
 R
|
 L
|
 $*$ R
| |
 id L
|
 id

\$

S'
|
 S
|
 L $=$ R
| | |
 $*$ R L
| | |
 L $=$ id
|
 id

\$

Problem?

No! $R \rightarrow L \bullet$ reduce
and $S \rightarrow L \bullet = R$ do
not co-occur due to
the $L \rightarrow *R$ rule

Solution: Canonical LR(1)

- Extend definition of configuration
 - Remember lookahead
- New closure method
- Extend definition of Successor

LR(1) Configurations

- $[A \rightarrow \alpha \bullet \beta, a]$ for $a \in T$ is valid for a viable prefix $\delta\alpha$ if there is a rightmost derivation $S \Rightarrow^* \delta A \eta \Rightarrow^* \delta \alpha \beta \eta$ and $(\eta = a\gamma)$ or $(\eta = \varepsilon \text{ and } a = \$)$
- Notation: $[A \rightarrow \alpha \bullet \beta, a/b/c]$
 - if $[A \rightarrow \alpha \bullet \beta, a], [A \rightarrow \alpha \bullet \beta, b], [A \rightarrow \alpha \bullet \beta, c]$ are valid configurations

LR(1) Configurations

$S \rightarrow B B$

$B \rightarrow a B \mid b$

$S \Rightarrow BB \Rightarrow BaB \Rightarrow Bab$
 $\Rightarrow aBab \Rightarrow aaBab \Rightarrow aaBab$

- $S \Rightarrow_{rm}^* aaBab \Rightarrow_{rm} aaaBab$
- Item $[B \rightarrow a \bullet B, a]$ is valid for **viable prefix** aaa
- $S \Rightarrow_{rm}^* BaB \Rightarrow_{rm} BaaB$ ← In $BaB \Rightarrow BaaB$ the string aB is the **handle** (rhs of B)
- Also, item $[B \rightarrow a \bullet B, \$]$ is valid for viable prefix Baa

$S \Rightarrow BB \Rightarrow BaB \Rightarrow BaaB$

LR(1) Closure

Closure property:

- If $[A \rightarrow \alpha \bullet B\beta, a]$ is in set, then $[B \rightarrow \bullet \gamma, b]$ is in set if $b \in \text{First}(\beta a)$
- Compute as fixed point
- Only include contextually valid lookaheads to guide reducing to B

Starting Configuration

- Augment Grammar with S' just like for LR(0), SLR(1)
- Initial configuration set is

$$I = \text{closure}([S' \rightarrow \bullet S, \$])$$

Example: $\text{closure}([S' \rightarrow \bullet S, \$])$

$[S' \rightarrow \bullet S, \$]$

$[S \rightarrow \bullet L = R, \$]$

$[S \rightarrow \bullet R, \$]$

$[L \rightarrow \bullet * R, =]$

$[L \rightarrow \bullet \text{id}, =]$

$[R \rightarrow \bullet L, \$]$

$[L \rightarrow \bullet * R, \$]$

$[L \rightarrow \bullet \text{id}, \$]$

$S' \rightarrow S$

$S \rightarrow L = R \mid R$

$L \rightarrow *R \mid \text{id}$

$R \rightarrow L$

concisely
written as:

$S' \rightarrow \bullet S, \$$

$S \rightarrow \bullet L = R, \$$

$S \rightarrow \bullet R, \$$

$L \rightarrow \bullet * R, =/\$$

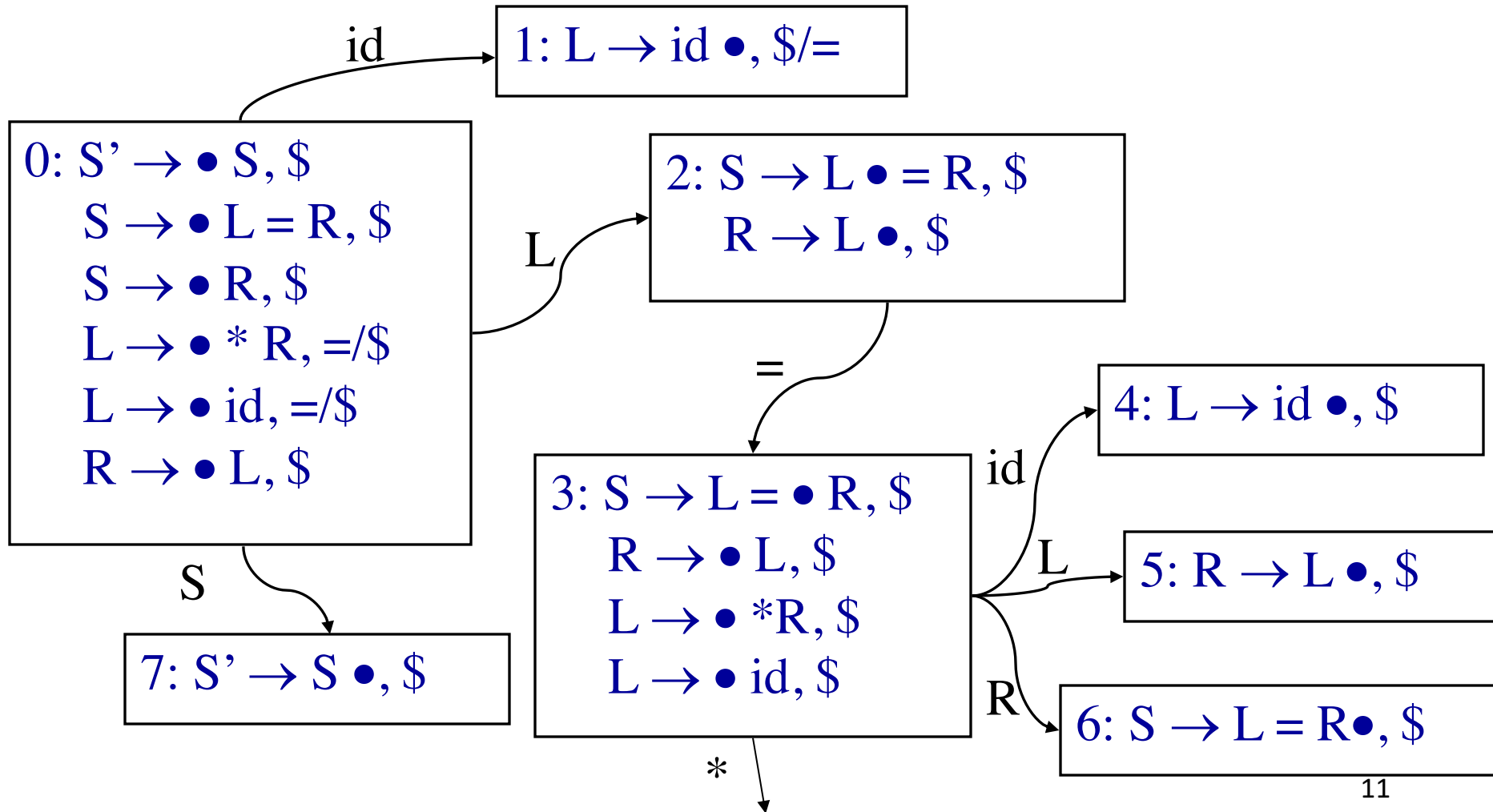
$L \rightarrow \bullet \text{id}, =/\$$

$R \rightarrow \bullet L, \$$

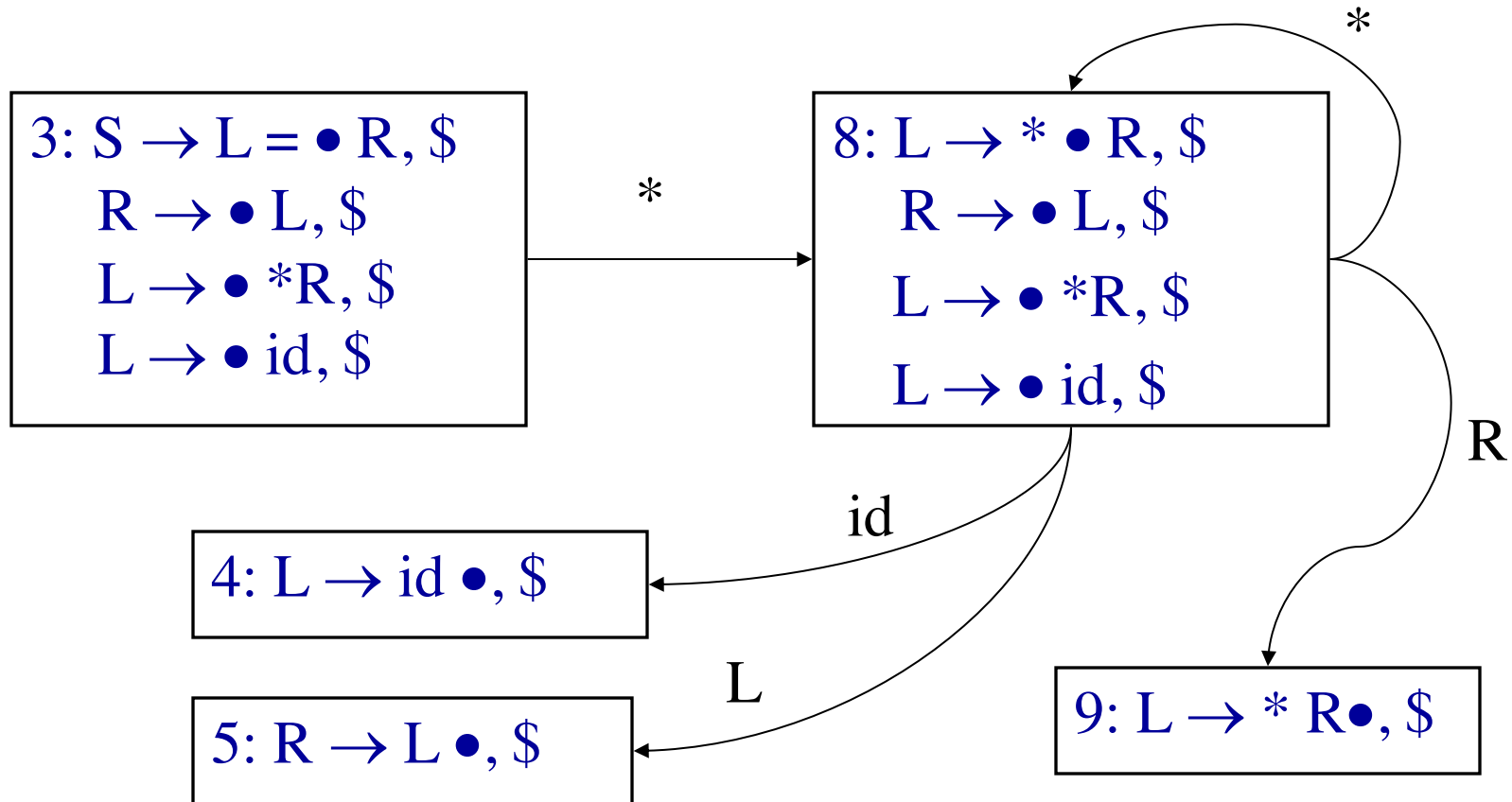
LR(1) Successor(C, X)

- Let $I = [A \rightarrow \alpha \bullet B \beta, a]$ **or** $[A \rightarrow \alpha \bullet b \beta, a]$
- $\text{Successor}(I, B)$
= $\text{closure}([A \rightarrow \alpha B \bullet \beta, a])$
- $\text{Successor}(I, b)$
= $\text{closure}([A \rightarrow \alpha b \bullet \beta, a])$

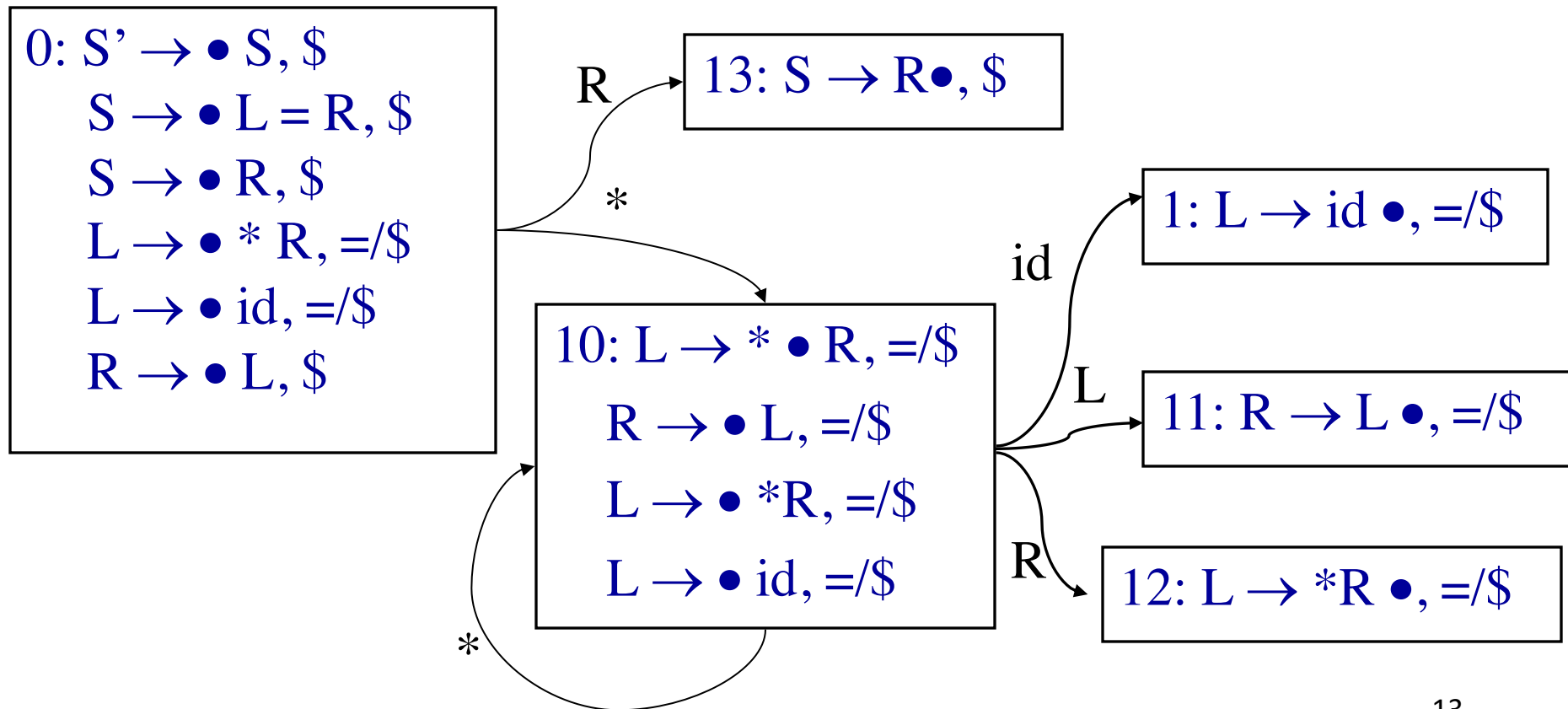
LR(1) Example



LR(1) Example (contd)



LR(1) Example (contd)



Productions	
1	$S \rightarrow L = R$
2	$S \rightarrow R$
3	$L \rightarrow * R$
4	$L \rightarrow id$
5	$R \rightarrow L$

	id	=	*	\$	S	L	R
0	S1		S10		7	2	13
1		R4		R4			
2		S3		R5			
3	S4		S8			5	6
4				R4			
5				R5			
6				R1			
7				Acc			
8	S4		S8			5	9
9				R3			
10	S1		S10			11	12
11		R5		R5			
12		R3		R3			
13				R2			

LR(1) Construction

1. Construct $F = \{I_0, I_1, \dots, I_n\}$
2. a) if $[A \rightarrow \alpha \bullet, a] \in I_i$ and $A \neq S'$
then $\text{action}[i, a] := \text{reduce } A \rightarrow \alpha$
b) if $[S' \rightarrow S \bullet, \$] \in I_i$
then $\text{action}[i, \$] := \text{accept}$
c) if $[A \rightarrow \alpha \bullet a \beta, b] \in I_i$ and $\text{Successor}(I_i, a) = I_j$
then $\text{action}[i, a] := \text{shift } j$
3. if $\text{Successor}(I_i, A) = I_j$ then $\text{goto}[i, A] := j$

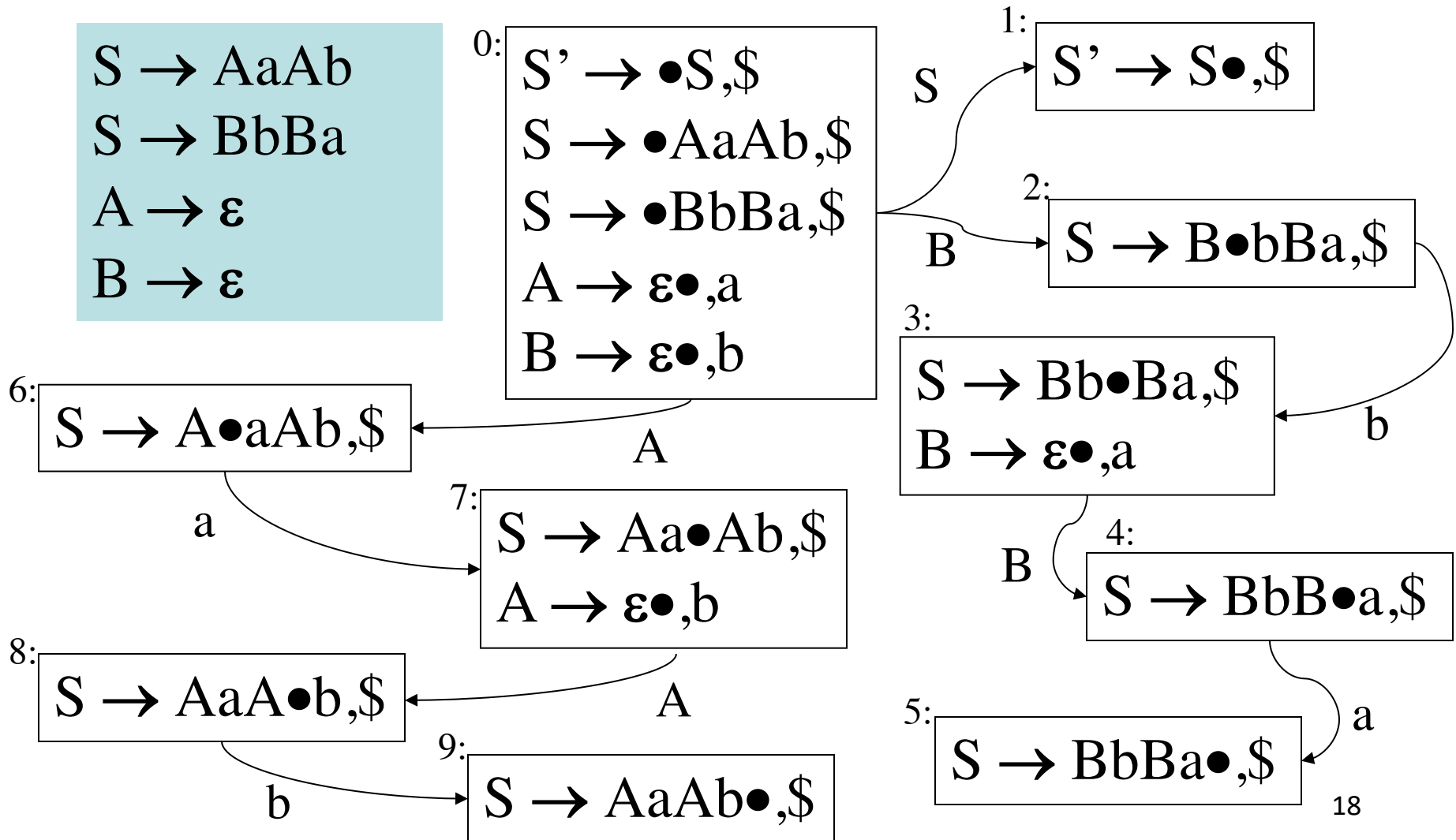
LR(1) Construction (cont'd)

4. All entries not defined are errors
 5. Make sure I_0 is the initial state
- Note: LR(1) only reduces using $A \rightarrow \alpha$ for $[A \rightarrow \alpha\bullet, a]$ if a follows
 - LR(1) states remember context by virtue of lookahead
 - Possibly many states!
 - LALR(1) combines some states

LR(1) Conditions

- A grammar is LR(1) if for each configuration set (itemset) the following holds:
 - For any item $[A \rightarrow \alpha \bullet x \beta, a]$ with $x \in T$ there is no $[B \rightarrow \gamma \bullet, x]$
 - For any two complete items $[A \rightarrow \gamma \bullet, a]$ and $[B \rightarrow \beta \bullet, b]$ then $a \neq b$.
- Grammars:
 - $LR(0) \subset SLR(1) \subset LR(1) \subset LR(k)$
- Languages expressible by grammars:
 - $LR(0) \subset SLR(1) \subset LR(1) = LR(k)$

Set-of-items with Epsilon rules

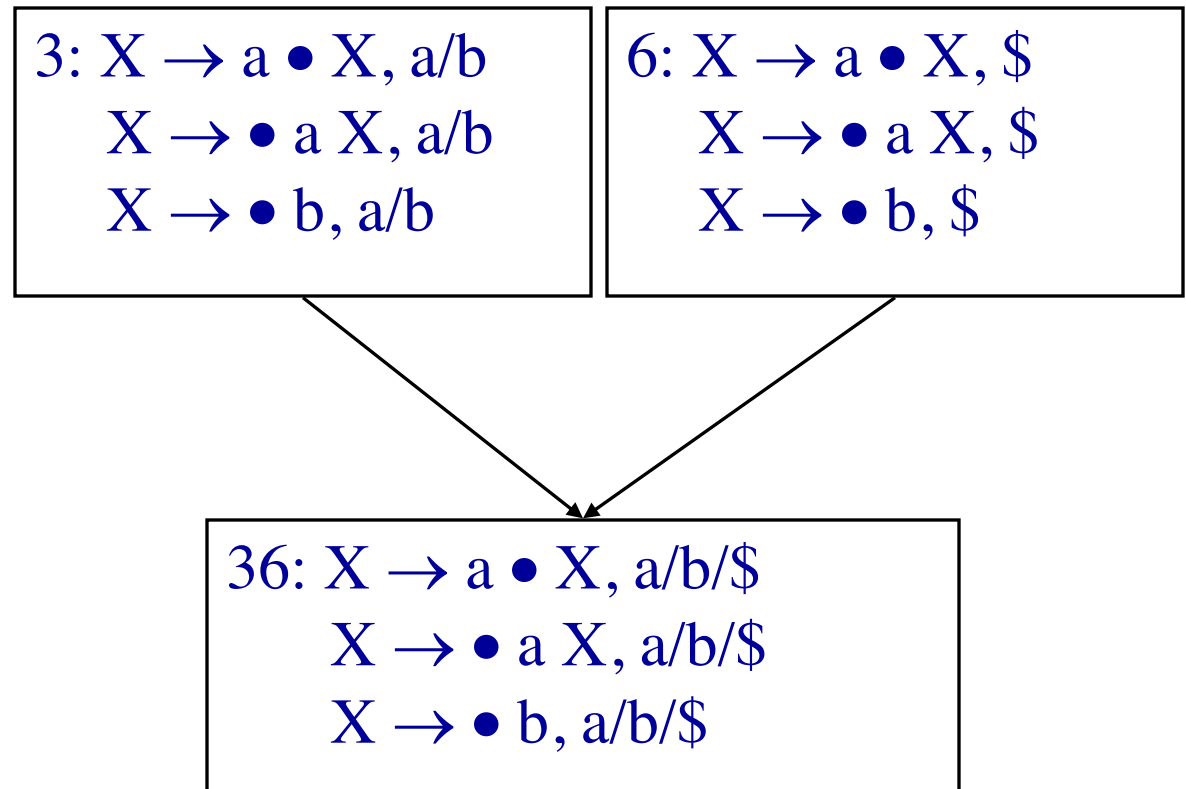


Canonical LR(1) Recap

- LR(1) uses left context, current handle and lookahead to decide when to reduce or shift
- Most powerful parser so far
- LALR(1) is practical simplification with fewer states

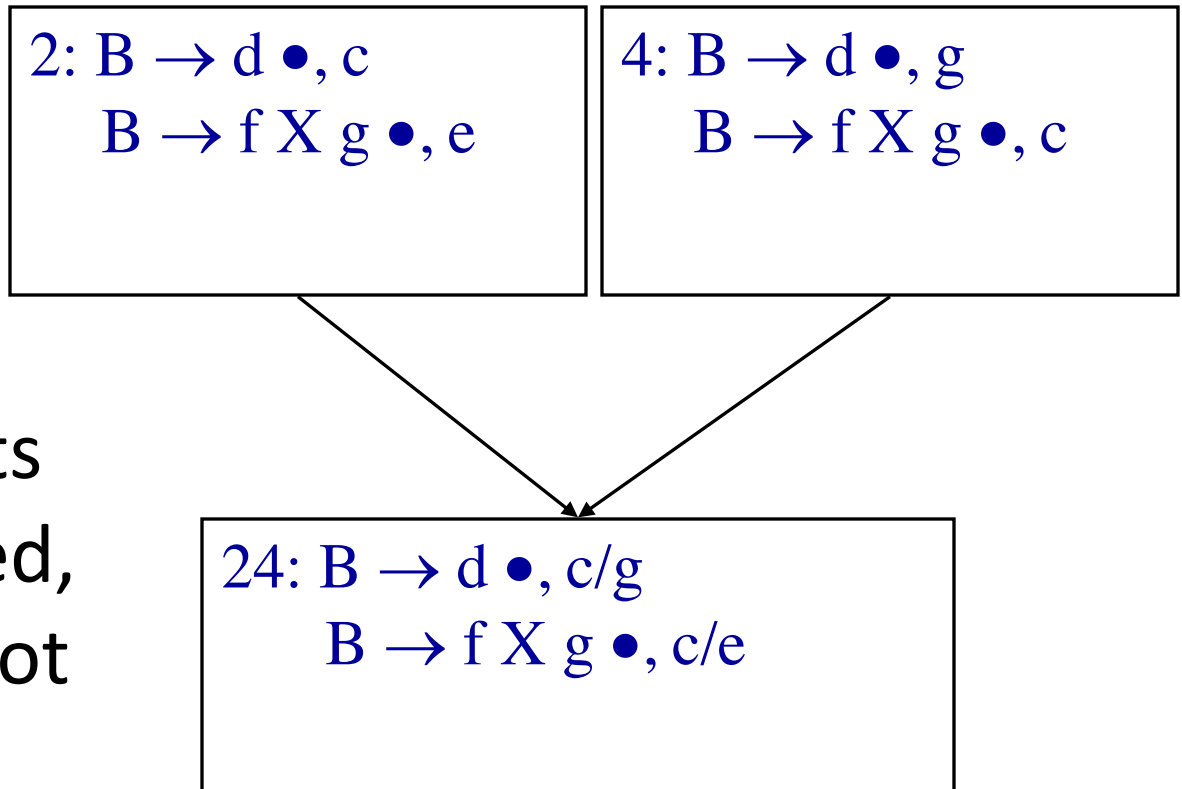
Merging States in LALR(1)

- $S' \rightarrow S$
 $S \rightarrow XX$
 $X \rightarrow aX$
 $X \rightarrow b$
- Same **Core Set**
- Different lookaheads



R/R conflicts when merging

- $B \rightarrow d$
 $B \rightarrow f X g$
 $X \rightarrow \dots$



- If R/R conflicts are introduced, grammar is not LALR(1)!

LALR(1)

- LALR(1) Condition:
 - Merging in this way does not introduce reduce/reduce conflicts
 - Shift/reduce can't be introduced
- Merging brute force or step-by-step
- More compact than canonical LR, like SLR(1)
- More powerful than SLR(1)
 - Not always merge to full Follow Set