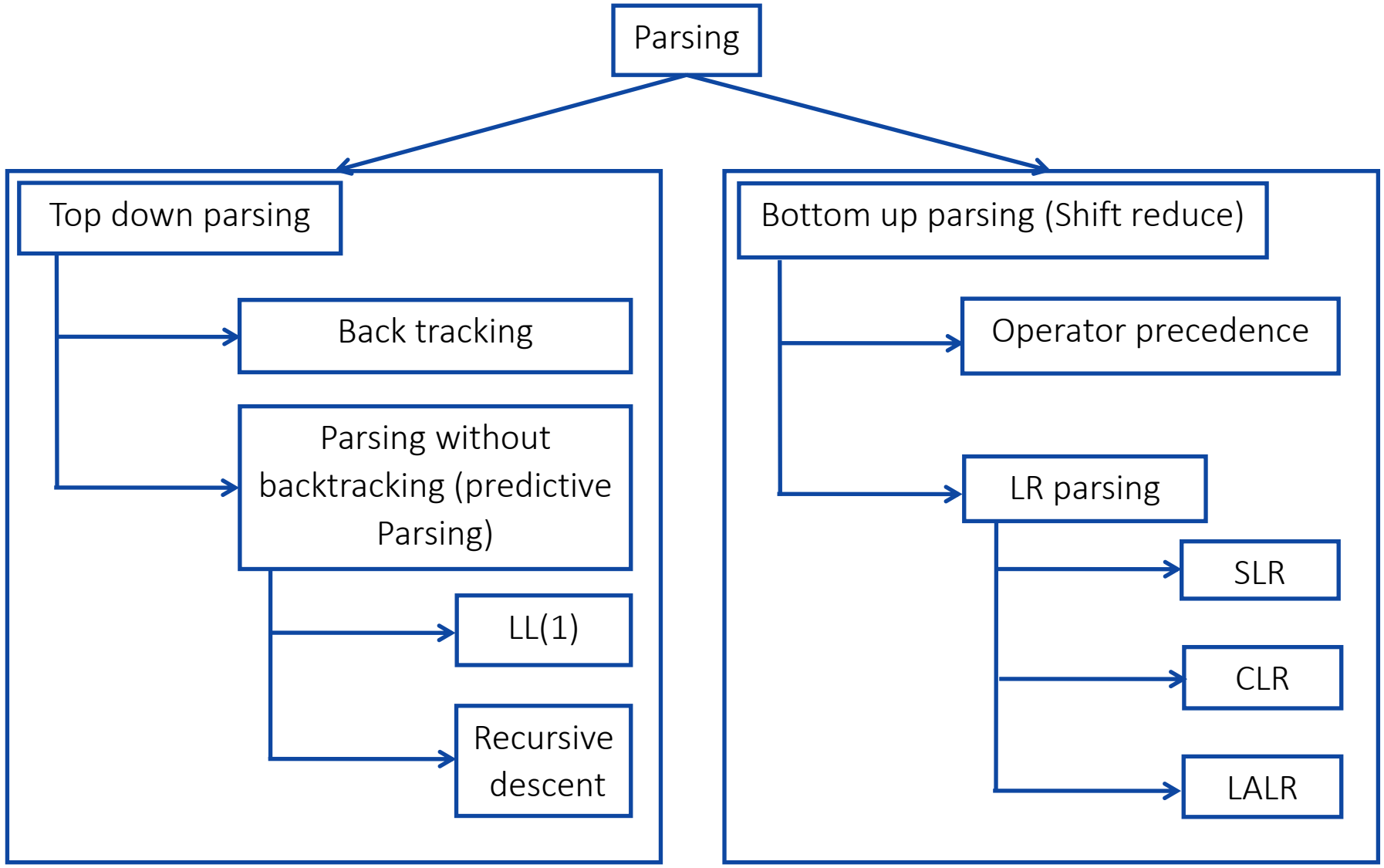
Module 2 – Syntax Analysis
Bottom Up Parsing

Parsing Methods



Handle & Handle pruning

- **Handle**: A "handle" of a string is a substring of the string that matches the right side of a production, and whose reduction to the non terminal of the production is one step along the reverse of rightmost derivation.
- **Handle pruning:** The process of discovering a handle and reducing it to appropriate left hand side non terminal is known as handle pruning.

E→E+E
E→E*E String: id1+id2*id3
E→id

Rightmost Derivation

E
E+E
E+E*E
E+E*id3
E+id2*id3
id1+id2*id3

Right sentential form	Handle	Production
id1+id2*id3		

Shift reduce parser

- The shift reduce parser performs following basic operations:

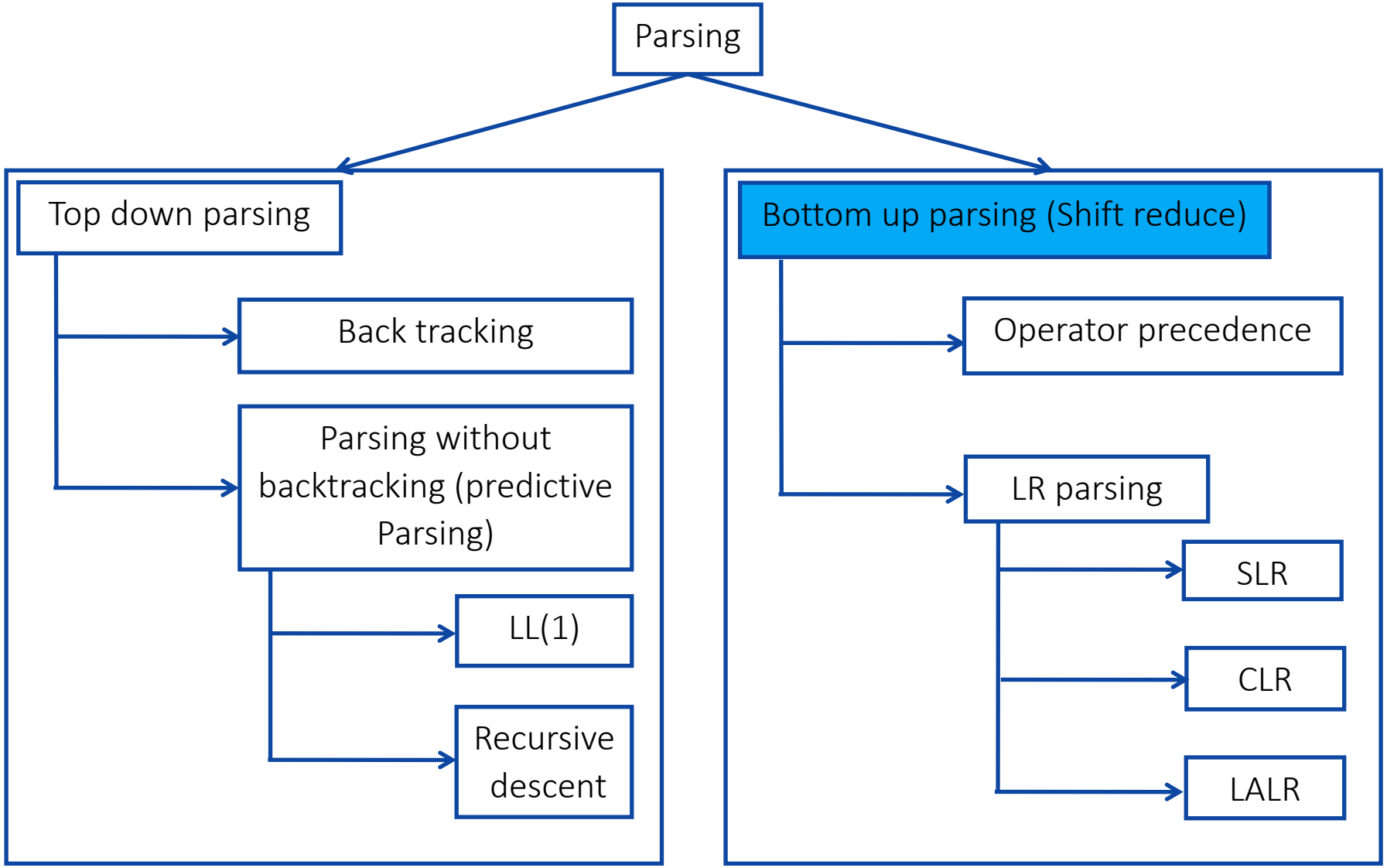
1. **Shift**: Moving of the symbols from input buffer onto the stack, this action is called shift.
2. **Reduce**: If handle appears on the top of the stack then reduction of it by appropriate rule is done. This action is called reduce action.
3. **Accept**: If stack contains start symbol only and input buffer is empty at the same time then that action is called accept.
4. **Error**: A situation in which parser cannot either shift or reduce the symbols, it cannot even perform accept action then it is called error action.

Example: Shift reduce parser

Grammar:
E→E+T | T
T→T*F | F
F→id
String: id+id*id

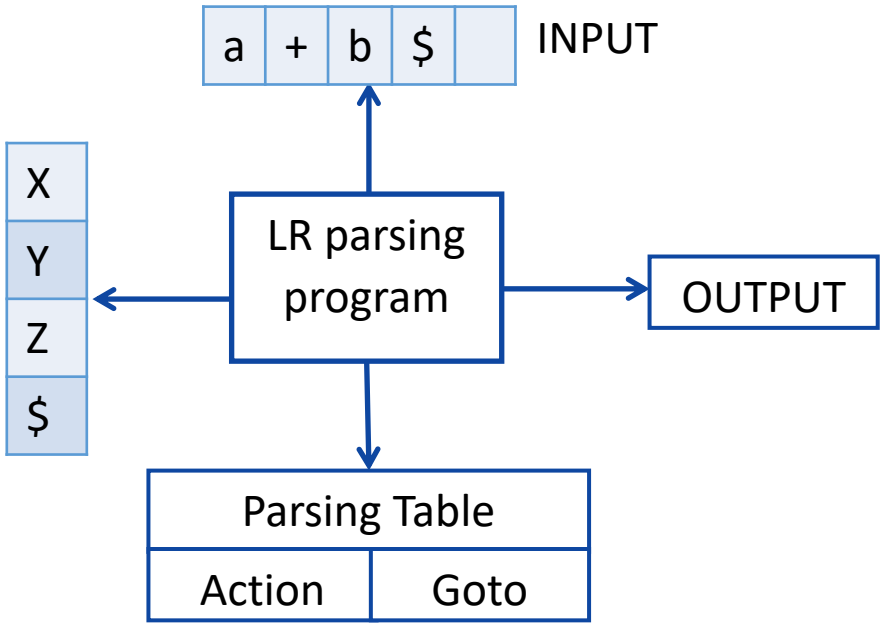
Stack	Input Buffer	Action

Parsing Methods

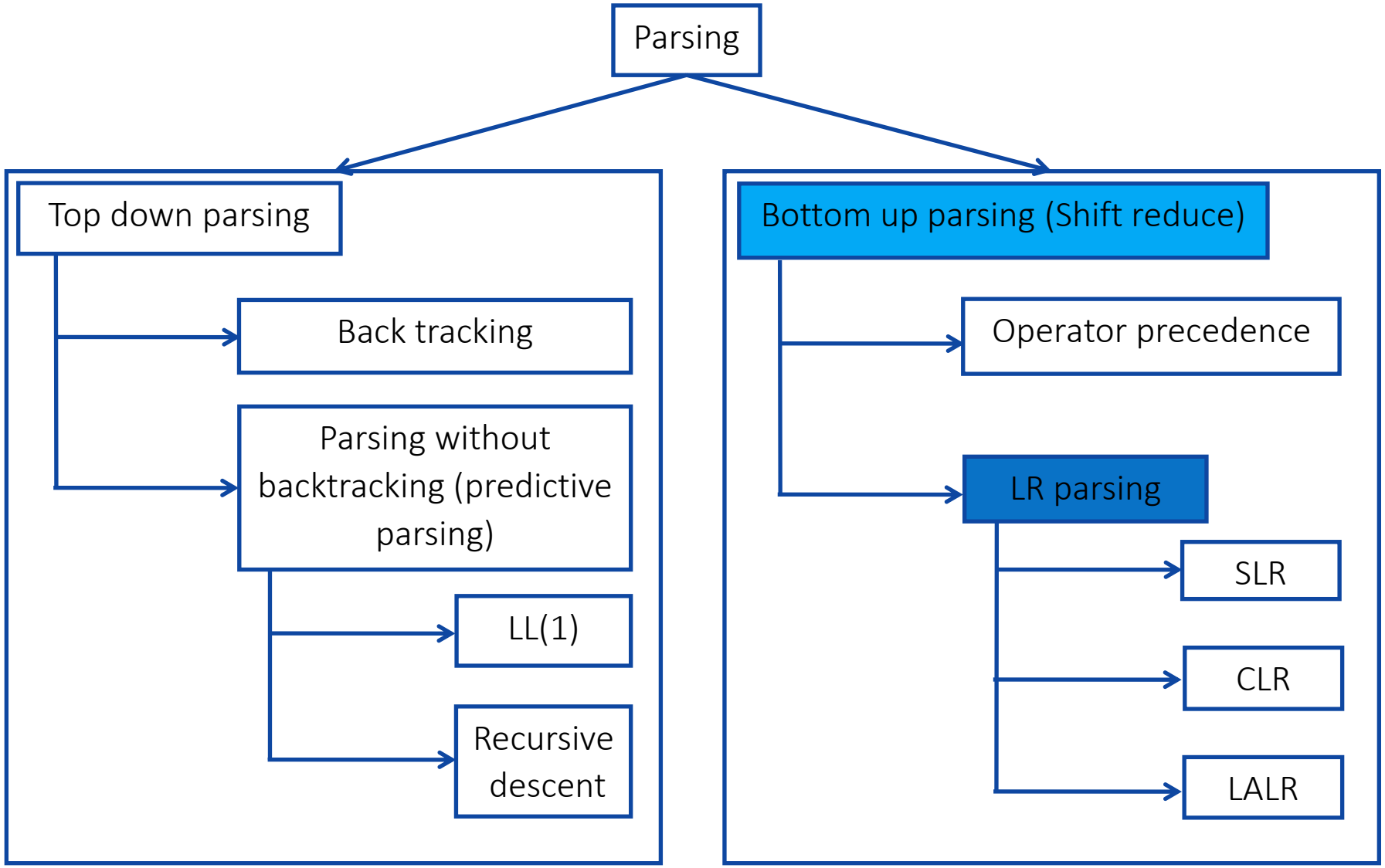


LR parser

- LR parsing is most efficient method of bottom up parsing which can be used to parse large class of context free grammar.
- The technique is called LR(k) parsing:
  1. The "L" is for left to right scanning of input symbol,
  2. The "R" for constructing right most derivation in reverse,
  3. The "k" for the number of input symbols of look ahead that are used in making parsing decision.



Parsing Methods



Computation of closure & go to function

X→ Xb

Closure(I):

X→ .X b

Goto(I,X)

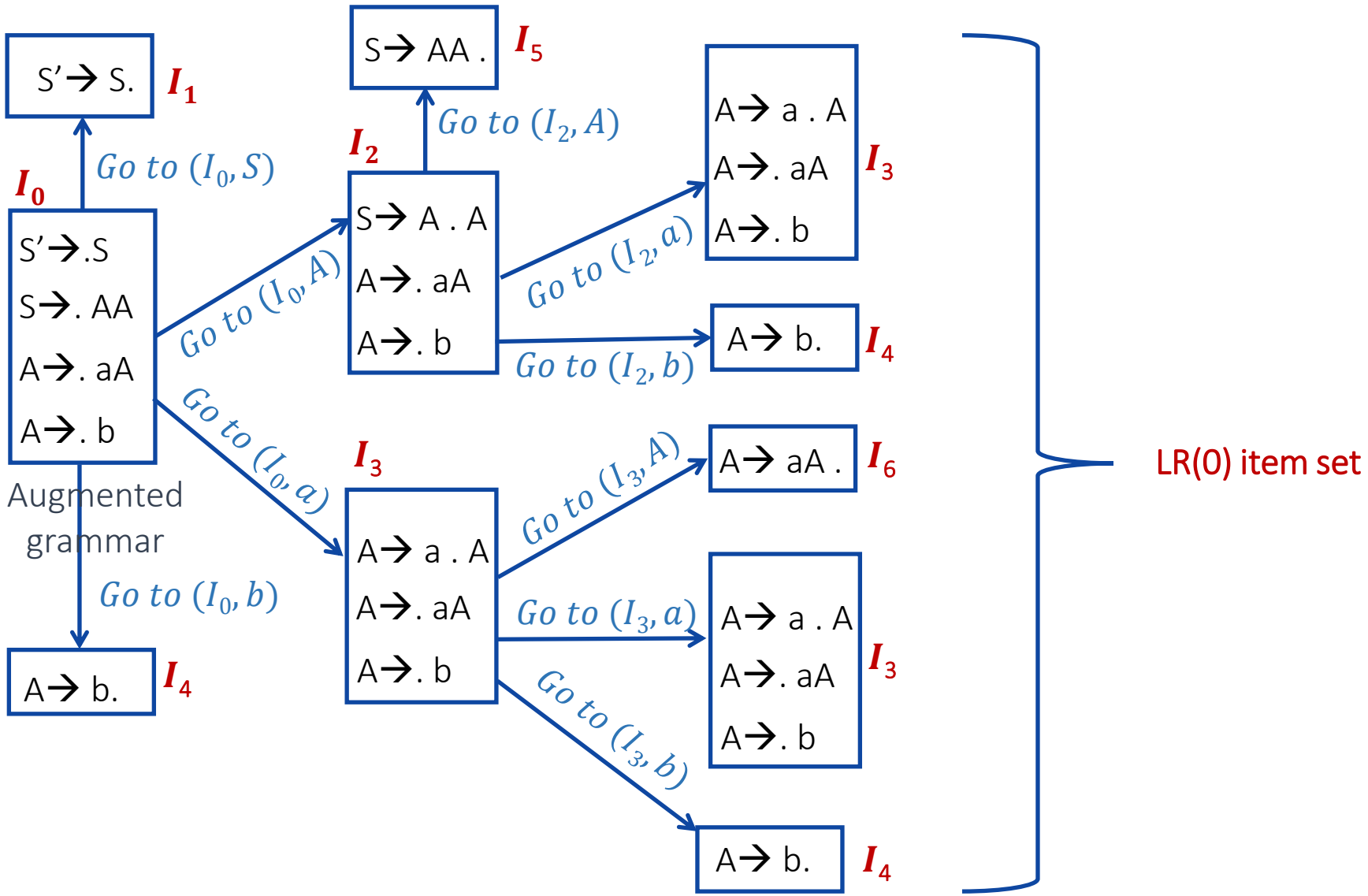
X→.X b

Steps to construct SLR parser

1. Construct Canonical set of LR(0) items
2. Construct SLR parsing table
3. Parse the input string

Example: SLR(1)- simple LR

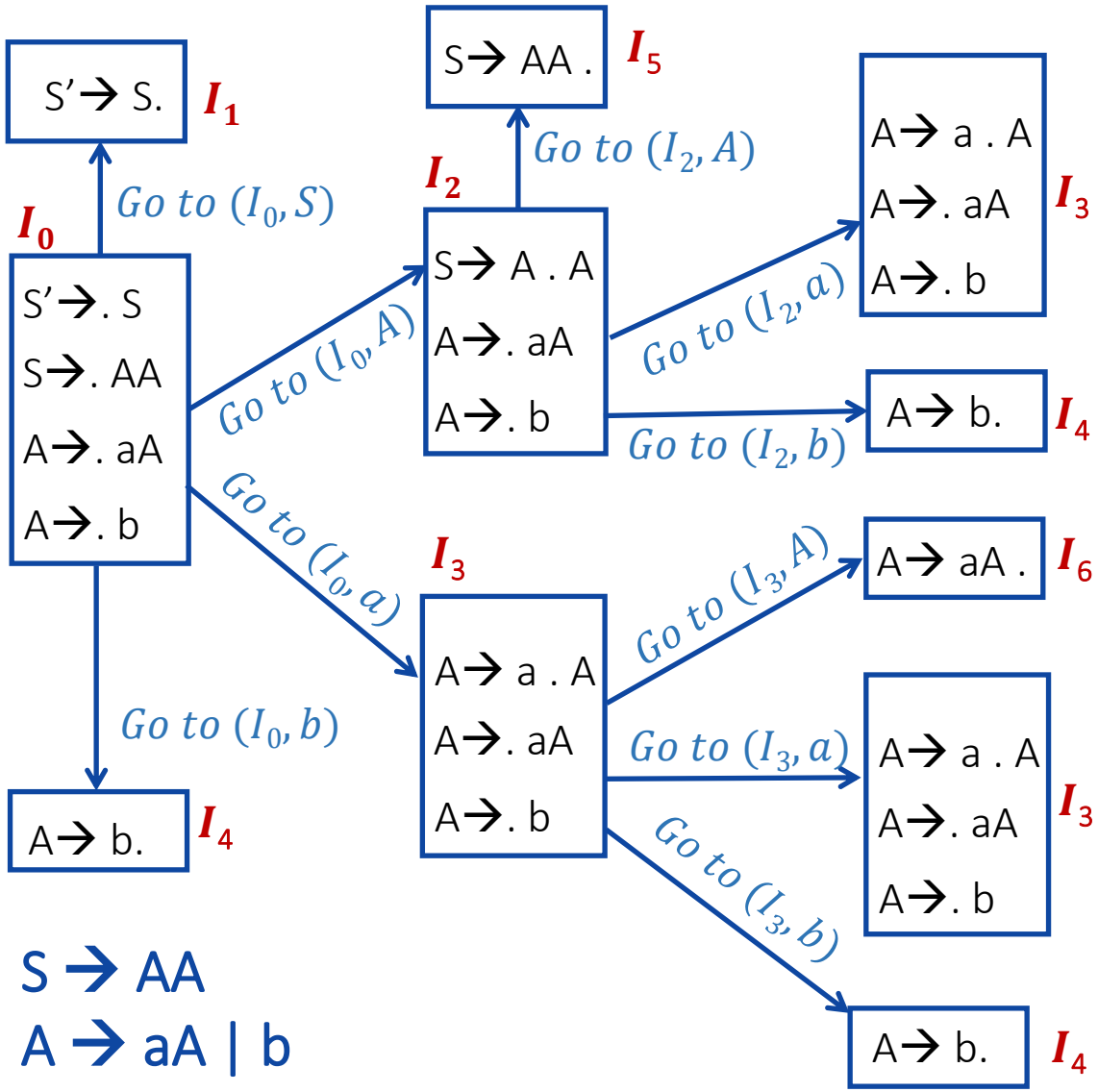
S → AA
A → aA | b



Rules to construct SLR parsing table

1. Construct $C = \{ I_0, I_1, \ldots \ldots In \}$, the collection of sets of LR(0) items for G'.
2. State i is constructed from I_i. The parsing actions for state i are determined as follow :
   a) If $[A \rightarrow \alpha . a\beta]$ is in I_i and GOTO $(Ii , a) = I_j$, then set $ACTION[i, a]$ to "shift j". Here a must be terminal.
   b) If $[A \rightarrow \alpha .]$ is in I_i, then set $ACTION[i, a]$ to "reduce A→ α" for all a in $FOLLOW(A)$; here A may not be S'.
   c) If $[S \rightarrow S.]$ is in I_i, then set action $[i, \$]$ to "accept".
3. The goto transitions for state i are constructed for all non terminals A using the$if\ GOTO(Ii , A) = I_j\ then\ GOTO\ [i, A] = j$.
4. All entries not defined by rules 2 and 3 are made error.

Example: SLR(1)- simple LR



S → AA

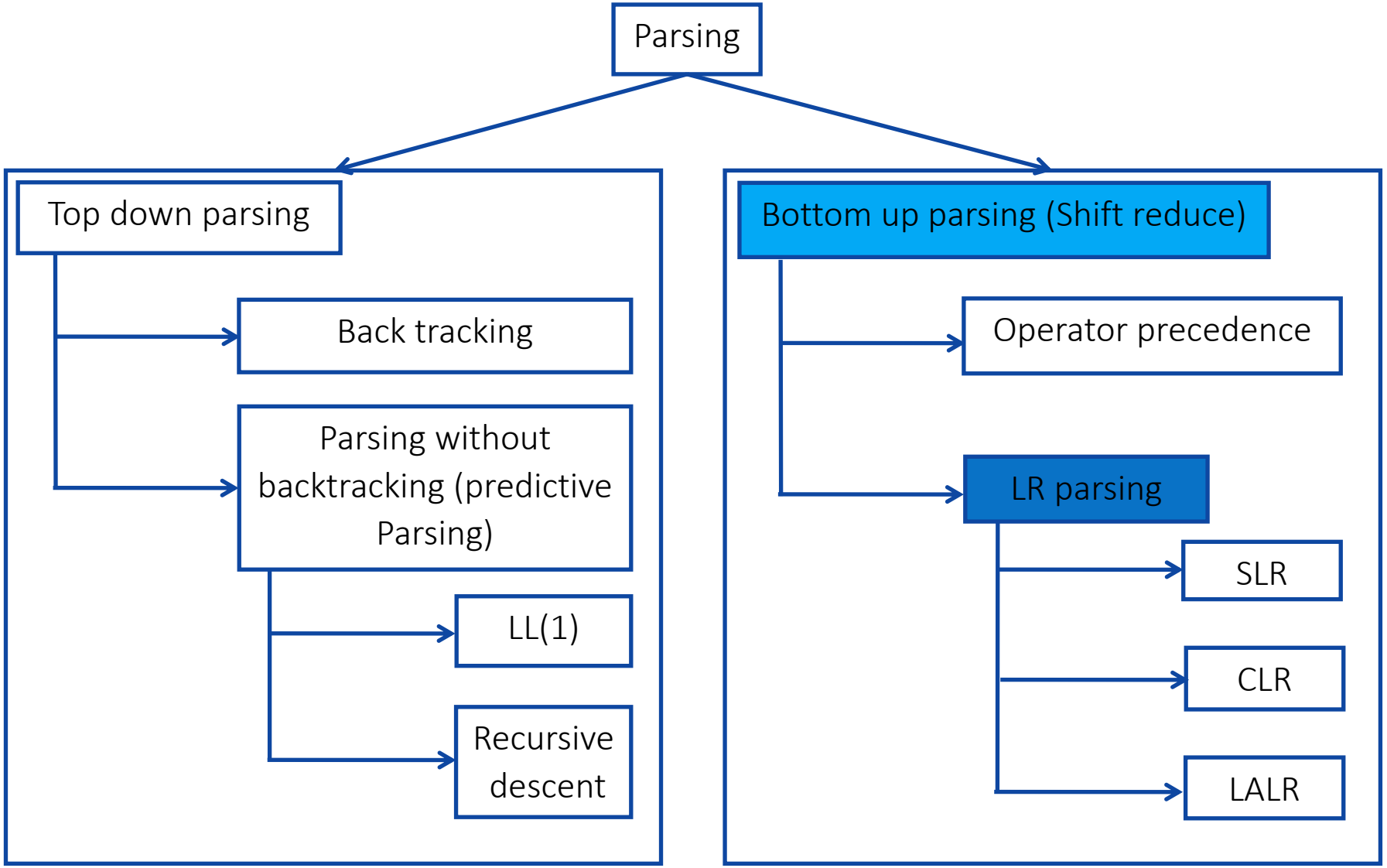
A → aA | b

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

$Follow(A) = \{a, b, \$\}$

	Action			Go to	
Item set	a	b	$	S	A
0					
1					
2					
3					
4					
5					
6					

Parsing Methods



How to calculate look ahead?

How to calculate look ahead?

S→CC

C→ cC | d

Closure(I)

S'→.S,$

S→.CC, $

C→.cC, c|d

C→.d, c|d

S'	→		.	S		,	$
A	→	α	.	X	β	,	a

Lookahead = First(βa)

First($)

= $

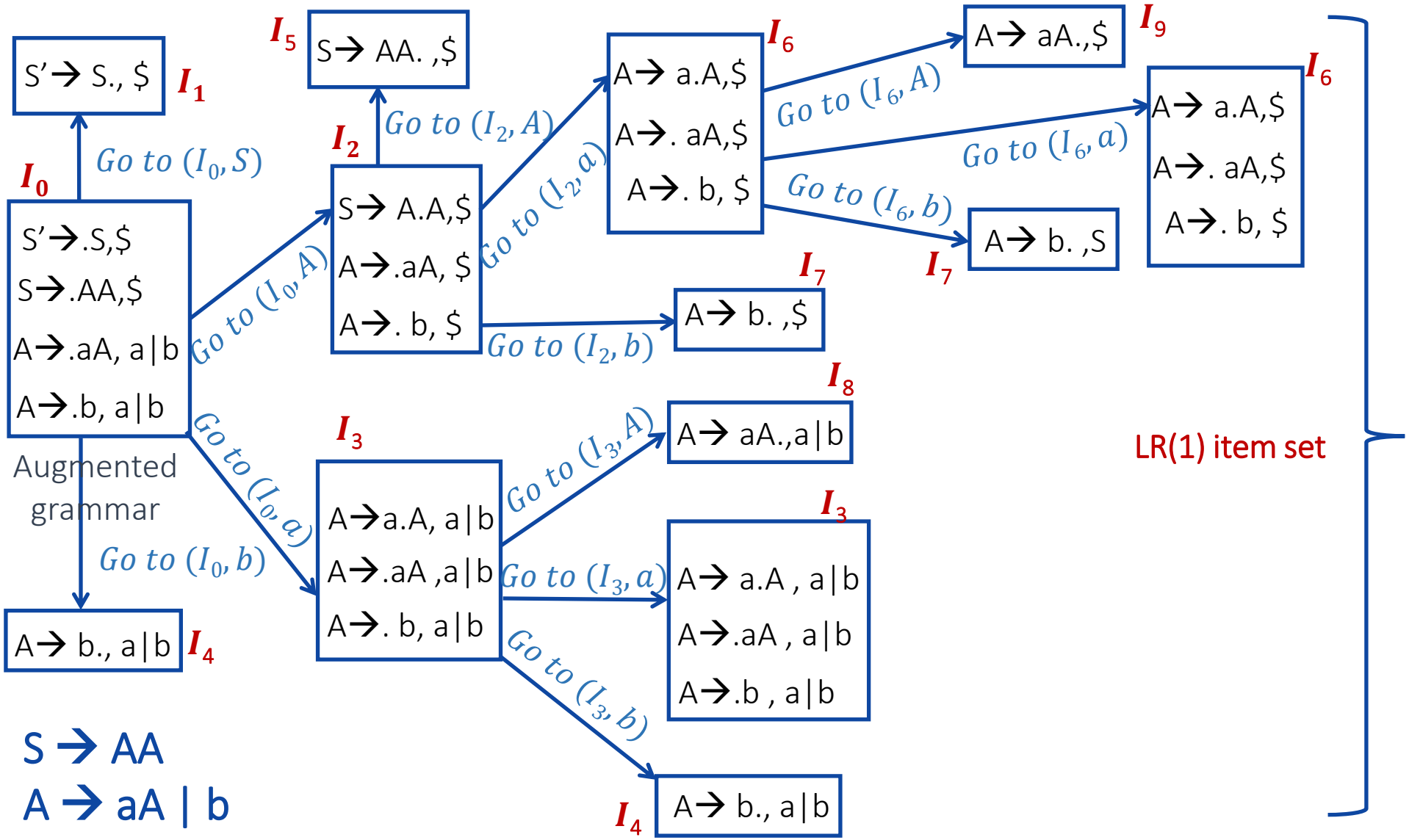
S	→		.	C	C	,	$
A	→	α	.	X	β	,	a

Lookahead = First(βa)

First(C)

$= c, d$

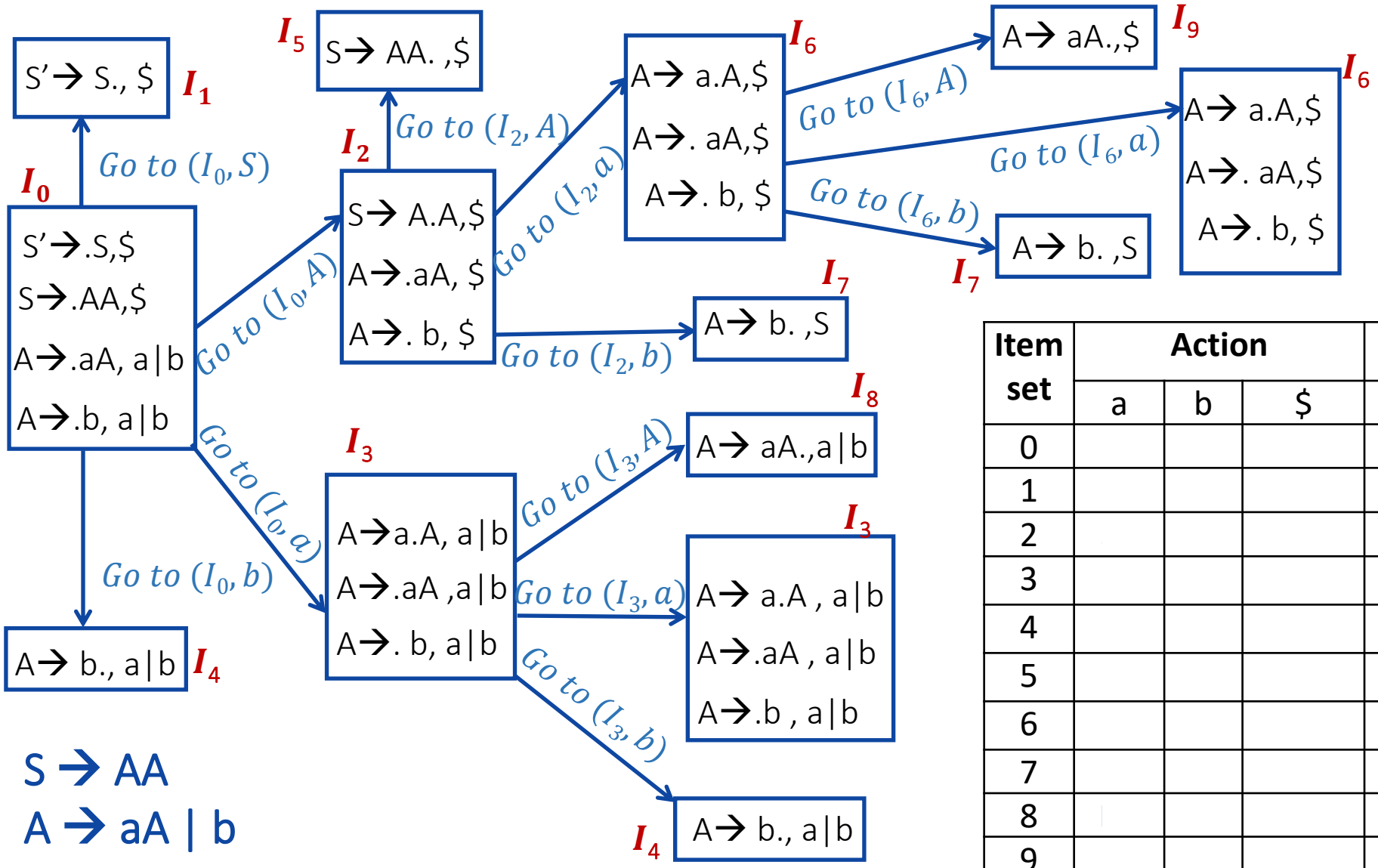
Example: CLR(1)- canonical LR



S → AA

A → aA | b

Example: CLR(1)- canonical LR

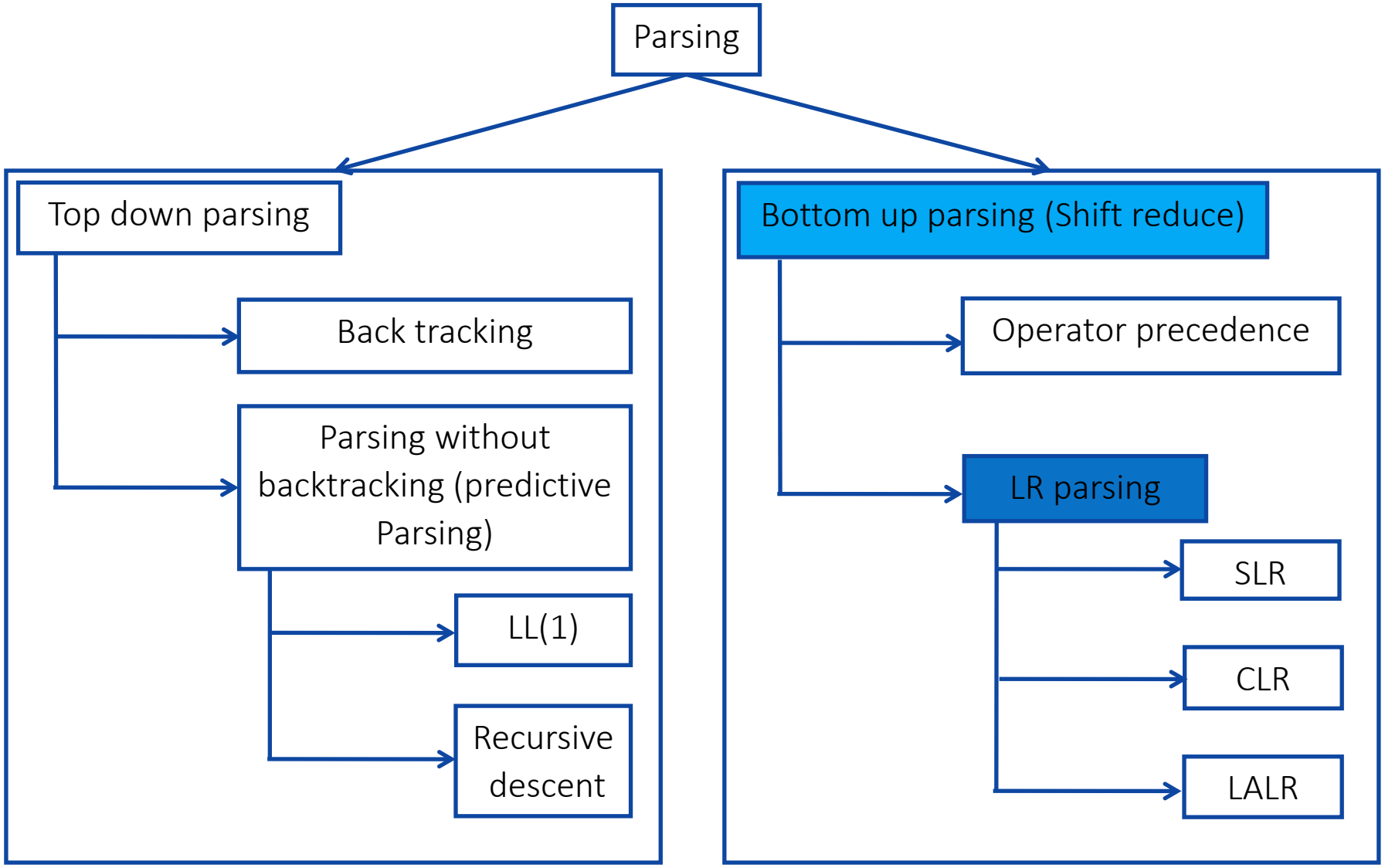


S → AA

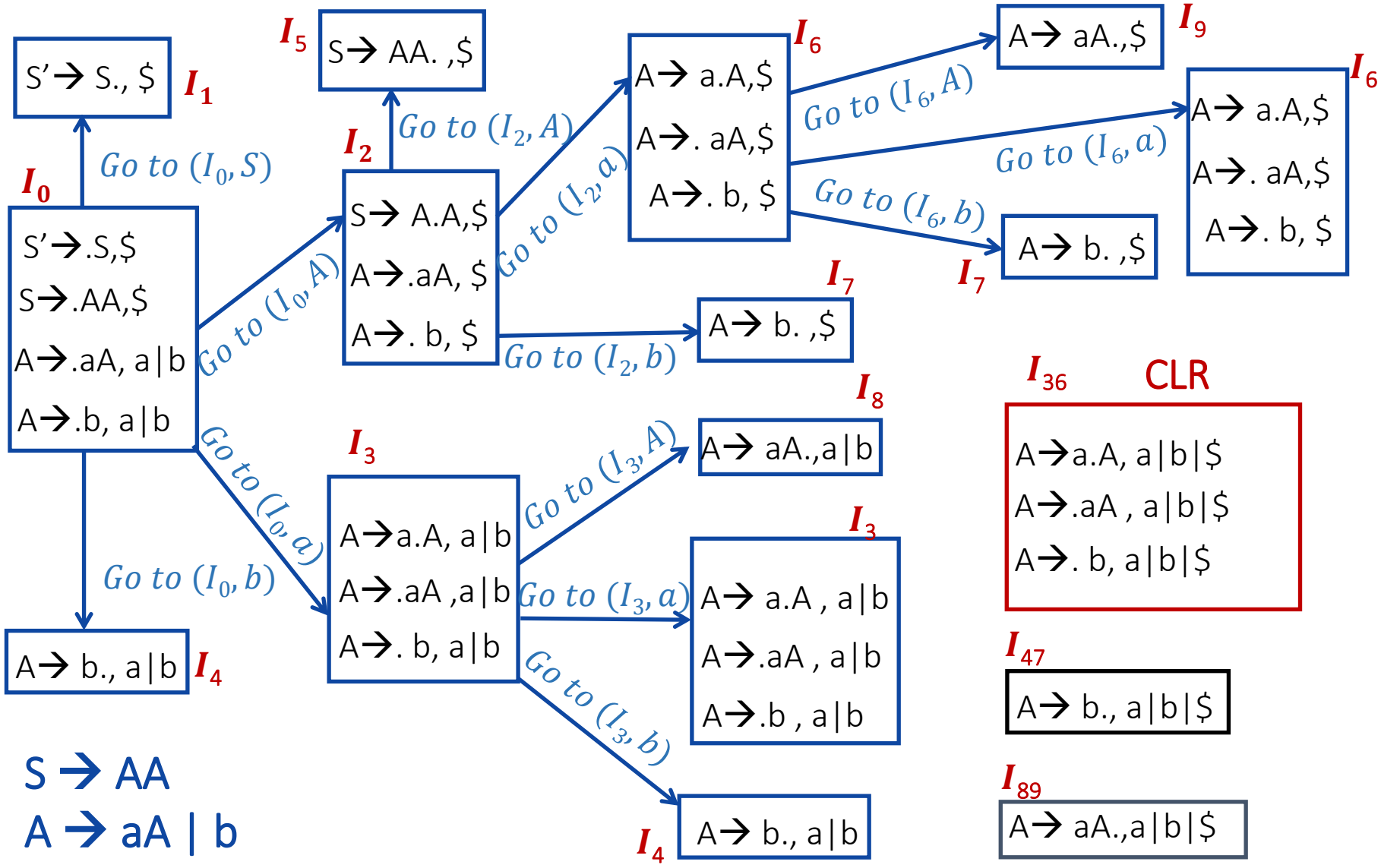
A → aA | b

Item set	Action			Go to	
	a	b	$	S	A
0				_	
1					
2					
3					
4					
5					
6					
7					
8					
9					

Parsing Methods



Example: LALR(1)- look ahead LR



I_{36} CLR

A→a.A, a|b|$
A→.aA , a|b|$
A→. b, a|b|$

I_{47}

A→ b., a|b|$

I_{89}

A→ aA.,a|b|$

Example: LALR(1)- look ahead LR

Item set	Action			Go to	
	a	b	$	S	A
0	S3	S4		1	2
1			Accept		
2	S6	S7			5
3	S3	S4			8
4	R3	R3			
5			R1		
~~6~~	~~S6~~	~~S7~~			~~9~~
~~7~~			~~R3~~		
8	R2	R2			
~~9~~			~~R2~~		

CLR Parsing Table

Item set	Action			Go to	
	a	b	$	S	A
0	S36	S47		1	2
1			Accept		
2	S36	S47			5
36	S36	S47			89
47	R3	R3	R3		
5			R1		
89	R2	R2	R2		

LALR Parsing Table