LR(1) Analysis

Drawbacks of SLR(1). Look-ahead symbols

- As explained before, concerning SLR(1), it is possible to consult the **next** set to determine, in the reduction states, for which symbols it would be possible to perform reductions.
- The key concept is the ***look-ahead symbol***. Using this symbol will allow us to bear in mind, together with the input symbol that is being studied, the next one that appears in the input.
  - Because this symbol will also be used in a later step, it receives the name of *look-ahead symbol*.
- In SLR(1), when we reduced a rule *R* in a state *s* only for the elements in the **next** set, that action is like considering all the look-ahead symbols for R in all the possible states.

LR(1) Analysis

Drawbacks of SLR(1): example

- We have seen an example grammar which is not SLR(1):
  - **(1) S→A**
  - **(2) S→xb**
  - **(3) A→aAb**
  - **(4) A→B**
  - **(5) B→x**
  - this grammar generates the following language:

$$\{xb\} \cup \{a^n x b^n \mid n \geq 0\}$$

LR(1) Analysis

Drawbacks of SLR(1): example

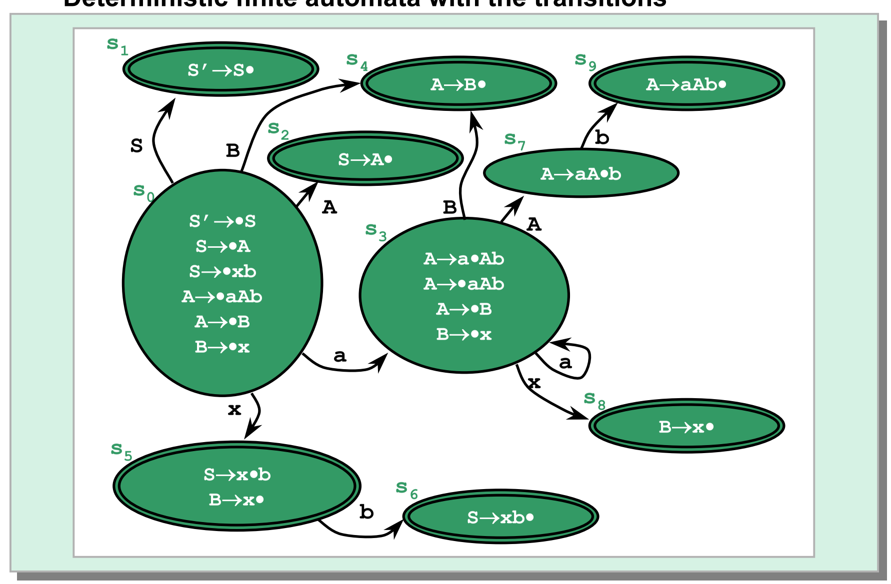
- Remember the augmented grammar:

(0) S'→S$
(1) S→A
(2) S→xb
(3) A→aAb
(4) A→B
(5) B→x

LR(1) Analysis

Drawbacks of SLR(1): example

- **Deterministic finite automata with the transitions**



LR(1) Analysis

Drawbacks of SLR(1): example

- And this was the analysis table for the SLR(1) grammar, with the conflict

(0) S′→S$
(1) S→A
(2) S→xb
(3) A→aAb
(4) A→B
(5) B→x

	Σ_T				Σ_N		
E	a	b	x	$	S	A	B
0	s3		s5		1	2	4
1				acc			
2				r1			
3	s3		s8			7	4
4		r4		r4			
5		r5/s6		r5			
6				r2			
7							
8		r5		r5			
9		r3		r3			
	Action				Go-to		

LR(1) Analysis

Drawbacks of SLR(1): example

- The last example seen was an example of an ambiguous grammar which was not SLR(1).

(1) E→E+E
(2) E→E*E
(3) E→i

LR(1) Analysis

Drawbacks of SLR(1): example

- The following was the augmented grammar:

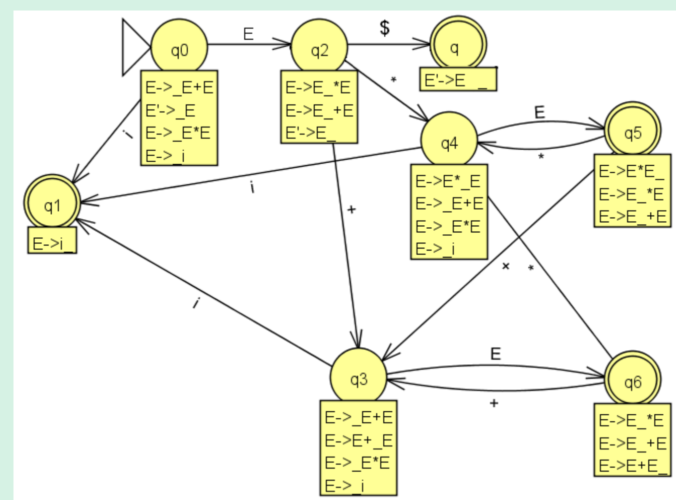
(0) E′→E$
(1) E→E+E
(2) E→E*E
(3) E→i

LR(1) Analysis

Drawbacks of SLR(1): example

- Remember the transition diagram in the SLR(1) automata

(0) E′→E$
(1) E→E+E
(2) E→E*E
(3) E→i



LR(1) Analysis

Drawbacks of SLR(1): example

- And the SLR(1) matrix showing the conflicts

(0) $E' \to E\$$
(1) $E \to E+E$
(2) $E \to E*E$
(3) $E \to i$

E	*	+	i	$	B
	Σ_T (Action)				
0			s1		2
1	r3	r3		r3	
2	s4	s3		acc	
3			s1		6
4			s1		5
5	r2/s4	r2/s3		r2	
6	r1/s4	r1/s3		r1	

- The grammar is not SLR(1)

LR(1) Analysis

Introduction to LR(k) with k=1

- In the analysis, at each step, we shall take into consideration:
  - The current symbol that is being treated
  - The k symbols that can follow it (k look-ahead symbols).
- Each state in the automata will have several copies, as many as possible sequences of k symbols that can follow the current state.

LR(1) Analysis

States of the automata in an LR(1) analyser: introductory example

- The example used to illustrate LR(1) and LALR(1) grammars will be the following:

$$\{xb,\ a^n x b^n \mid n \geq 0\}$$

(1) $S \to A$
(2) $S \to xb$
(3) $A \to aAb$
(4) $A \to B$
(5) $B \to x$

- This is not the simplest grammar we can build for this language, but it is a good example, because:
  - It is not SLR(1)
  - It is LR(1)
  - It is LALR, but the automata obtained with this method will be different than the one obtained for an LR(1) parser
- We shall see, in this introductory example, the initial state s0 and the transition from s0 to s5.

LR(1) Analysis

Look-ahead symbols in an LR(1) analyser: introductory example

- For each state, we shall need to calculate the set of possible ***look-ahead symbols***.
- This process can be incorporated to the construction of the automata.
- It will suffice with:
  - Specifying a procedure for calculating all the look-ahead symbols in all the configurations in each state.
- The new states will add explicitly, to each configuration, the set of look-ahead symbols that are valid, using the following notation:

$$s_i=$$
$$\{\ \langle configuration\rangle_0\{\langle look\text{-}ahead\rangle_{j0}\},$$
$$\ldots$$
$$\langle configuration\rangle_i\{\langle look\text{-}ahead\rangle_{ji}\},$$
$$\ldots$$
$$\langle configuration\rangle_n\{\langle look\text{-}ahead\rangle_{jn}\}\ \}$$

LR(1) Analysis

Look-ahead symbols in an LR(1) analyser: introductory example

- Considering the initial state,
  - As in LR(0) and SLR(1), it was calculated as the closure of the following configuration:

    S'→•S$

- **S** is the old axiom, the non-terminal symbol to which we should derive the complete derivation tree after we finish the analysis. Therefore, the only look-ahead symbol that we should expect at this position is the end-of-program symbol (**$**)
- Therefore, that configuration in the initial state will have the set of look-ahead symbols

  {$}

- The next diagram shows this example.

LR(1) Analysis

Look-ahead symbols in an LR(1) analyser: introductory example

- Furthermore, the initial state will contain the closure of the configuration **S'→•S**
- For calculating closures, we need to calculate the look-ahead symbols for each of the new configurations added:
  - This case will be solved with the help of the ***first*** and ***next*** sets.
  - We shall use the parts of the right-hand sides of the non-terminal symbol that follows the dot.
  - For instance, in the initial state s_0

    s_0=
    {S'→•S{$},
    S→•A{$},
    S→•xb{$}}

  - the two new configurations added can have, as look-ahead, anything that could appear after the S in **S'→•S**
  - Because the S was at the end of the rule, and the look-ahead symbol for that rule was $, the look-ahead symbols for the new configurations are the same.

LR(1) Analysis

Look-ahead symbols in an LR(1) analyser: introductory example

- We can continue the closure with the rules for **A**:

  s_0={S'→•S{$},
  S→•A{$},
  S→•xb{$},
  A→•aAb{$},
  A→•B{$}}

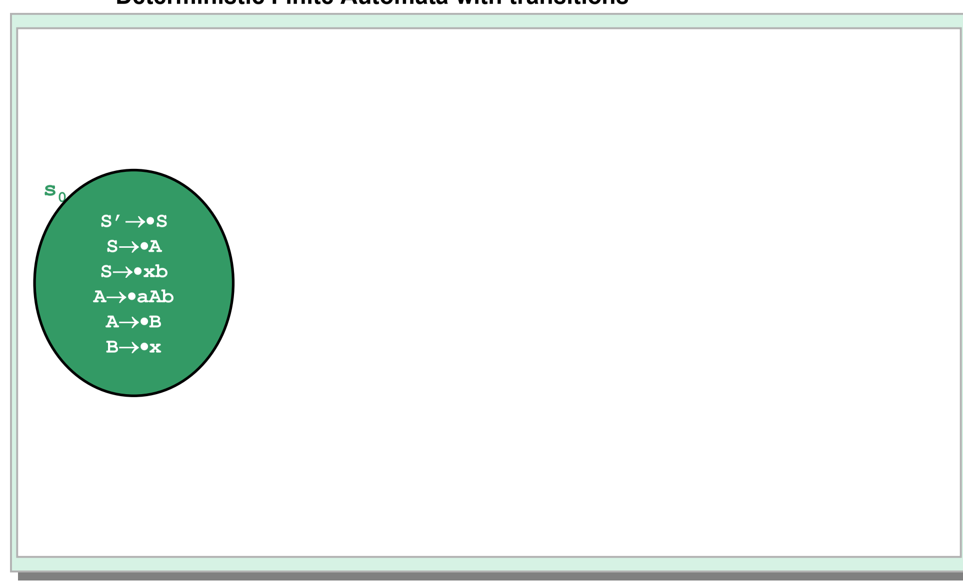
  - As before, because A was at the end of the rule **S→•A**, after the A we may find anything that we could find after the S (the look-ahead symbols for that configuration): {$}
  - The same argumentation can be followed to show that the look-ahead symbols for the last configuration to be added to this state is also {$}:

    s_0={S'→•S{$},
    S→•A{$},
    S→•xb{$},
    A→•aAb{$},
    A→•B{$},
    B→•x{$}}

LR(1) Analysis

Look-ahead symbols in an LR(1) analyser: introductory example

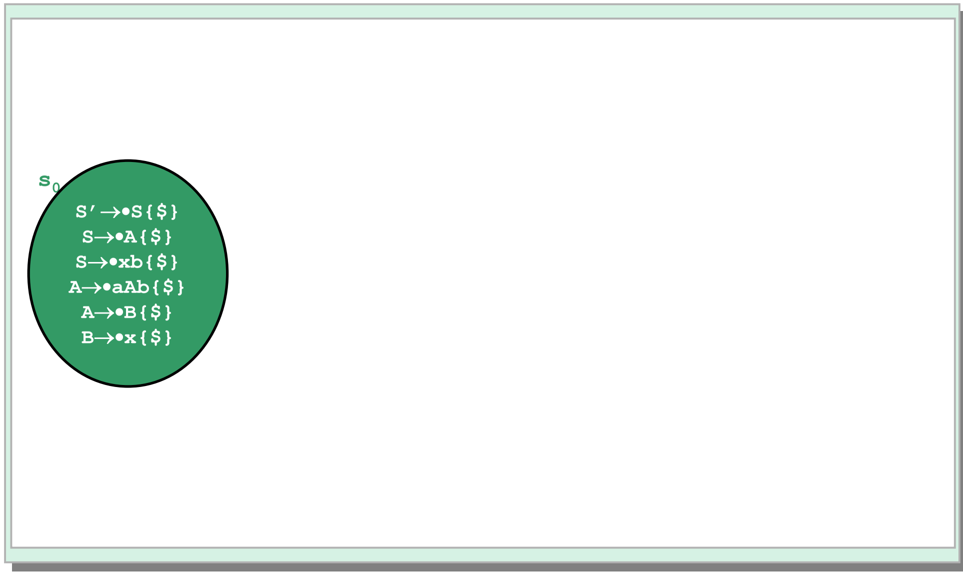
- **Deterministic Finite Automata with transitions**



LR(1) Analysis

Look-ahead symbols in an LR(1) analyser: introductory example

- **Deterministic Finite Automata with transitions**



LR(1) Analysis

Look-ahead symbols in an LR(1) analyser: introductory example

- Let us continue the example with a transition between two states:
- Again, the closures will be solved with the help of the ***first*** and ***next*** sets.
- For instance, from the initial state s_0

  s_0=
  {S'→•S{$},
  S→•A{$},
  S→•xb{$},
  ***A→•aAb{$}*,**
  A→•B{$},
  B→•x{$}}

- When we calculate **s_0** as the resulting state of *go_to(s_0,a),* the underlined configuration can be shifted to ***A→a•Ab{$}***

LR(1) Analysis

Look-ahead symbols in an LR(1) analyser: introductory example

- The closure of ***A→a•Ab{$}*** implies the inclusion of two new configurations:

  A→•aAb
  A→•B
  B→•x

- As this is the case, we probably have to change the sets of look-ahead symbols, as follows:
  - The closure of ***A→a•Ab{$}*** forces us to keep the hypothesis that we might now find any right-hand side of **A.**
  - On the other hand, after processing **A** we shall also need to shift **b** and, only after we have found the **b**, we shall be prepared to reduce **A→aAb** only in the presence of the look-ahead symbols in the set **{$}.**
  - This means that the three configurations added to the closure of ***A→a•Ab{$}***, that is, **A→•aAb**, **A→•B** or **B→•x**, will need to keep **b** as look-ahead symbol, because that **b** appears right after the **A** (***A→a•Ab{$}***).

LR(1) Analysis

Look-ahead symbols in an LR(1) analyser: introductory example

- In summary, with respect to the state **s_5**, because of the symbol **A,**

  A→•aAb
  A→•B

- both have the look-ahead symbol **b**:

  A→•aAb{b}
  A→•B{b}

- The presence of **A→•B{b}** in **s_5** forces us to extend the closure with all the right-hand sides of **B.**
- The situation in this case is different, because there is no symbol after the **B** in the rule (**A→•B(*){b}**)
- In this case, as in s_0 before, the closure does not modifies the set of look-ahead symbols, and the element which will be look-ahead symbol of the configuration **B→•x** will be **{b}**

LR(1) Analysis

Look-ahead symbols in an LR(1) analyser: introductory example

- So we can conclude that

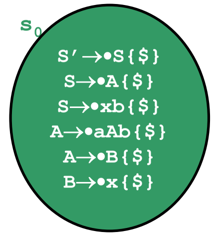
s_5=
{A→a•Ab{$},
A→•aAb{b},
A→•B{b},
B→•x{b} }

- as we can see in the following diagram:

LR(1) Analysis

Look-ahead symbols in an LR(1) analyser: introductory example

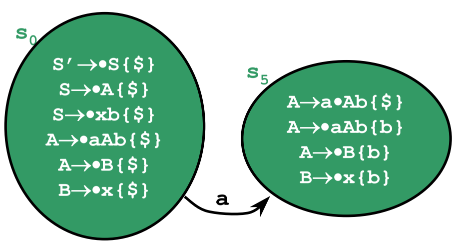
- **Deterministic Finite Automata with transitions**



LR(1) Analysis

Look-ahead symbols in an LR(1) analyser: introductory example

- **Deterministic Finite Automata with transitions**



LR(1) Analysis

Look-ahead symbols in an LR(1) analyser: formalisation

- Formally,
- to **calculate the look-ahead symbols** for the states transitions diagram,
  - The initial configuration in the initial state have the following look-ahead symbols:

    {$}
  - For calculating s_j=closure(s_i') for the elements s_i' of the form

    $P \to \alpha \bullet N\beta\{\sigma_1, \ldots, \sigma_n\}$,, $N \in \Sigma_N$
  - we need to add next(N), calculated using the previous rule, as look-ahead symbol for all the new configurations

    N→• . . . .
  - This next(N) set will be
    - first($\beta\{\sigma_1, \ldots, \sigma_n\}$)
    - next(P) if $\beta\{\sigma_1, \ldots, \sigma_n\} \to^* \lambda$

Remember, from the calculation of the next set, that, if β can derive the empty word, we have to add not only first(β) to the set next(N), but also first(σ_1), where σ_1 is the symbol that follows β in the right-hand side of the rule, and next(P), if β is the last symbol in the right-hand side!!!!

LR(1) Analysis

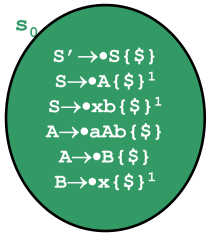
Introductory exercise

- Build the LR(1) analysis table for the following grammar that generates language L=**{xb, $a^n x b^n$ | n≥0}**
  - **(1) S→A**
  - **(2) S→xb**
  - **(3) A→aAb**
  - **(4) A→B**
  - **(5) B→x**
- Use them to analyse the following two strings:
  - **aaxbb**
  - **ax**
- The first step is to obtain the augmented grammar:
  - **(0) S′→S$**
  - **(1) S→A**
  - **(2) S→xb**
  - **(3) A→aAb**
  - **(4) A→B**
  - **(5) B→x**

LR(1) Analysis

Constructing LR(1) analysis tables

- **Deterministic Finite Automata with transitions**

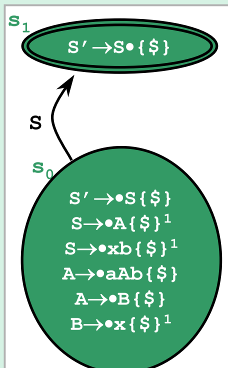


[1] first($)={$}

LR(1) Analysis

Constructing LR(1) analysis tables

- **Deterministic Finite Automata with transitions**

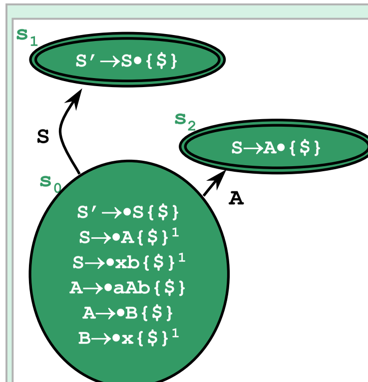


[1] first($)={$}

LR(1) Analysis

Constructing LR(1) analysis tables

- **Deterministic Finite Automata with transitions**

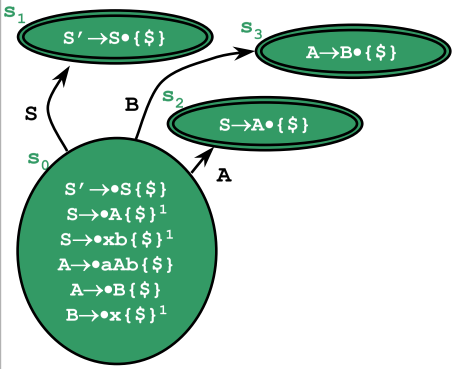


[1] first($)={$}

LR(1) Analysis

Constructing LR(1) analysis tables

- **Deterministic Finite Automata with transitions**

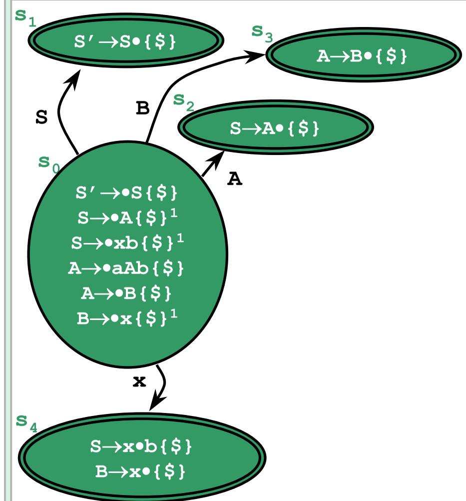


[1]first($)={$}

LR(1) Analysis

Constructing LR(1) analysis tables

- **Deterministic Finite Automata with transitions**

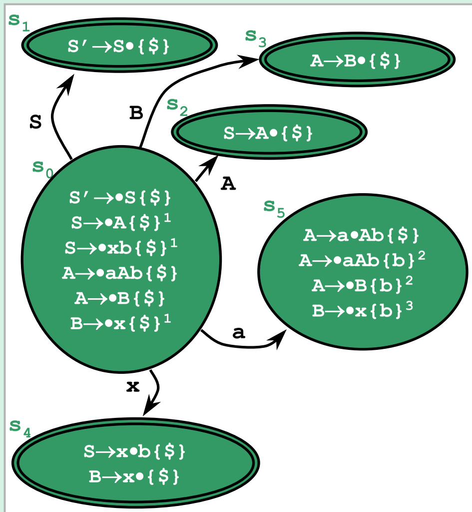


[1]first($)={$}

LR(1) Analysis

Constructing LR(1) analysis tables

- **Deterministic Finite Automata with transitions**



[1]first($)={$}

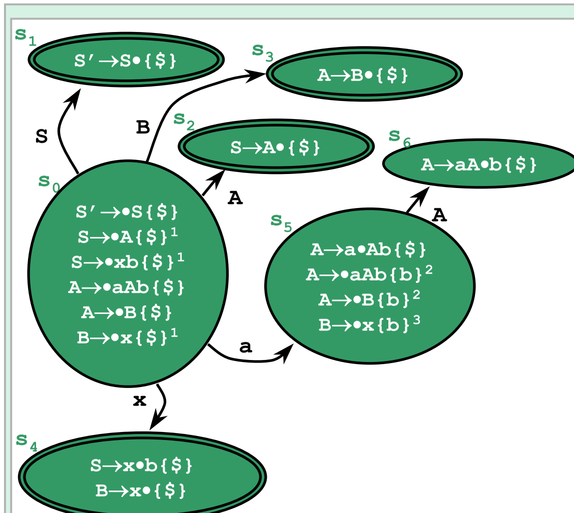
[2]first(b$)={b}

[3]first(b)={b}

LR(1) Analysis

Constructing LR(1) analysis tables

- **Deterministic Finite Automata with transitions**



[1]first($)={$}

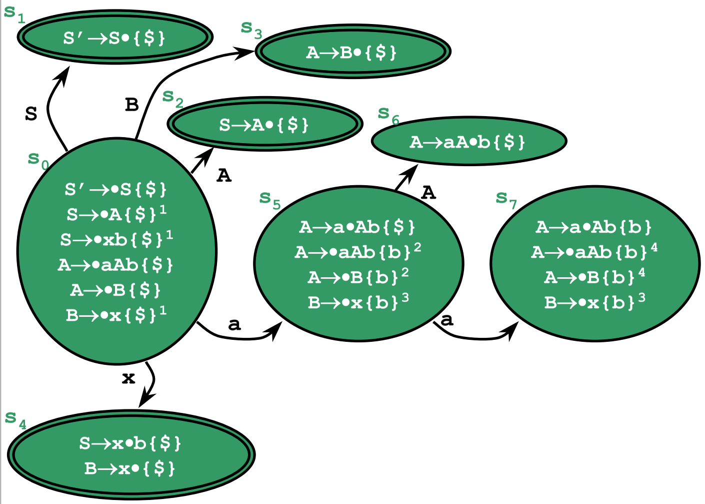
[2]first(b$)={b}

[3]first(b)={b}

LR(1) Analysis

Constructing LR(1) analysis tables

- **Deterministic Finite Automata with transitions**



[1]first($)={$}

[2]first(b$)={b}

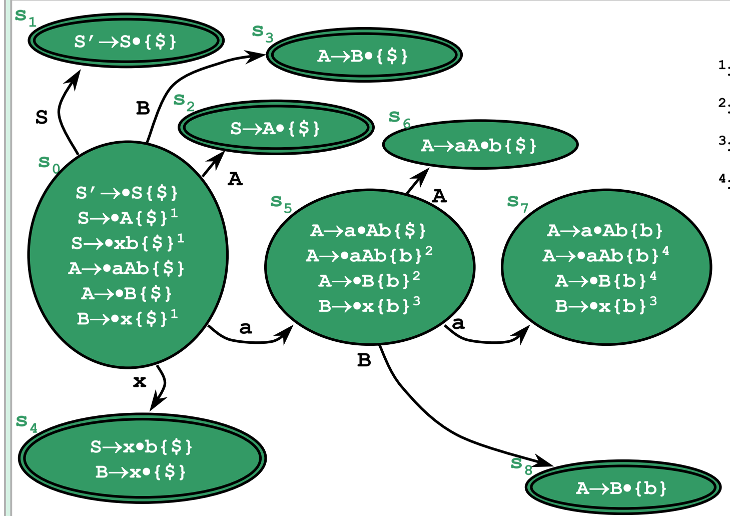
[3]first(b)={b}

[4]first(bb)={b}

LR(1) Analysis

Constructing LR(1) analysis tables

- **Deterministic Finite Automata with transitions**



[1]first($)={$}

[2]first(b$)={b}

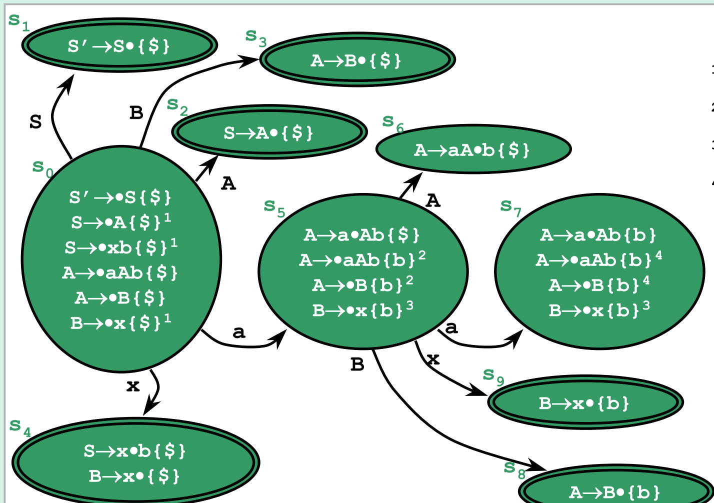
[3]first(b)={b}

[4]first(bb)={b}

LR(1) Analysis

Constructing LR(1) analysis tables

- **Deterministic Finite Automata with transitions**



[1]first($)={$}

[2]first(b$)={b}

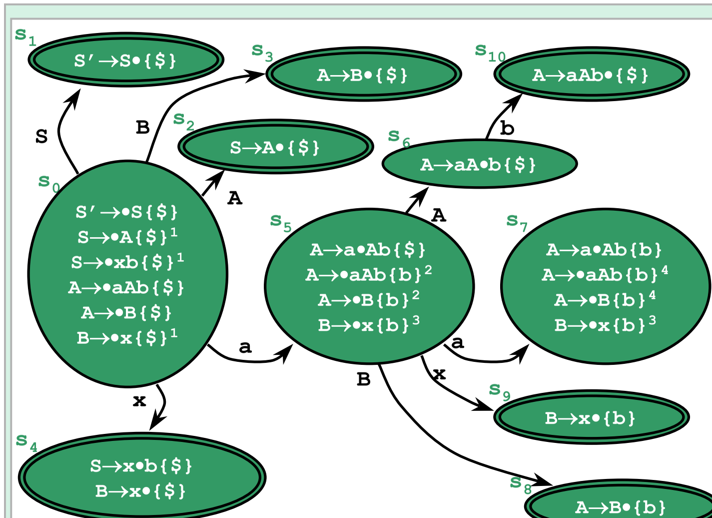
[3]first(b)={b}

[4]first(bb)={b}

LR(1) Analysis

Constructing LR(1) analysis tables

- **Deterministic Finite Automata with transitions**



[1]first($)={$}

[2]first(b$)={b}

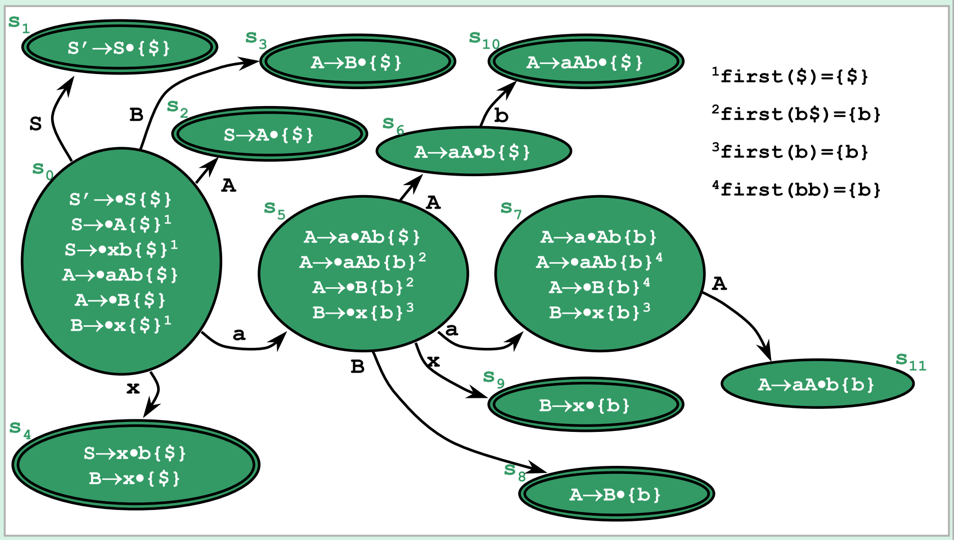
[3]first(b)={b}

[4]first(bb)={b}

LR(1) Analysis

Constructing LR(1) analysis tables

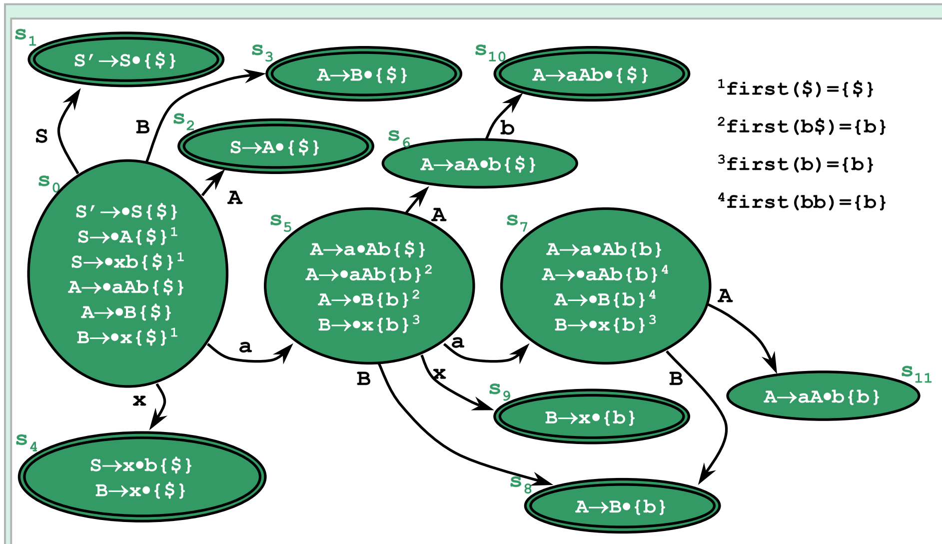
- **Deterministic Finite Automata with transitions**



LR(1) Analysis

Constructing LR(1) analysis tables

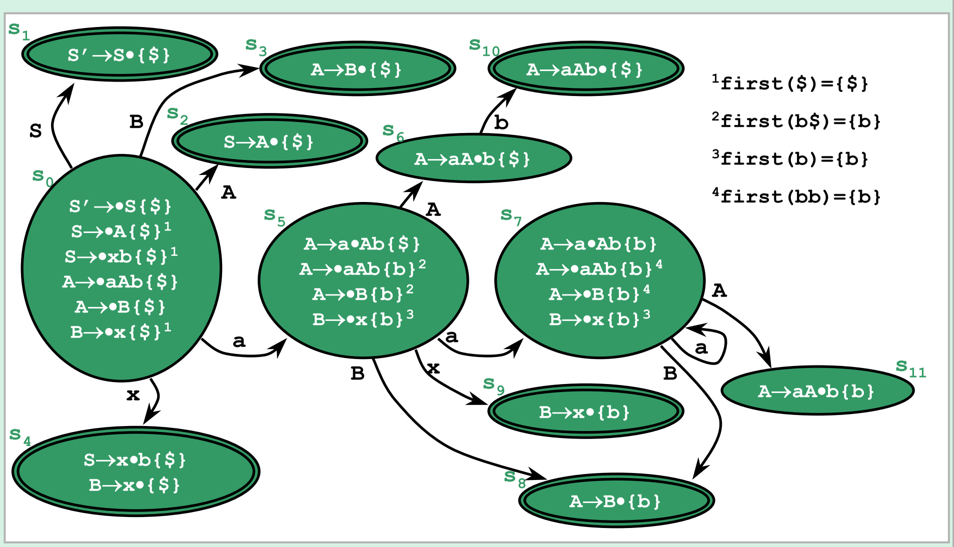
- **Deterministic Finite Automata with transitions**



LR(1) Analysis

Constructing LR(1) analysis tables

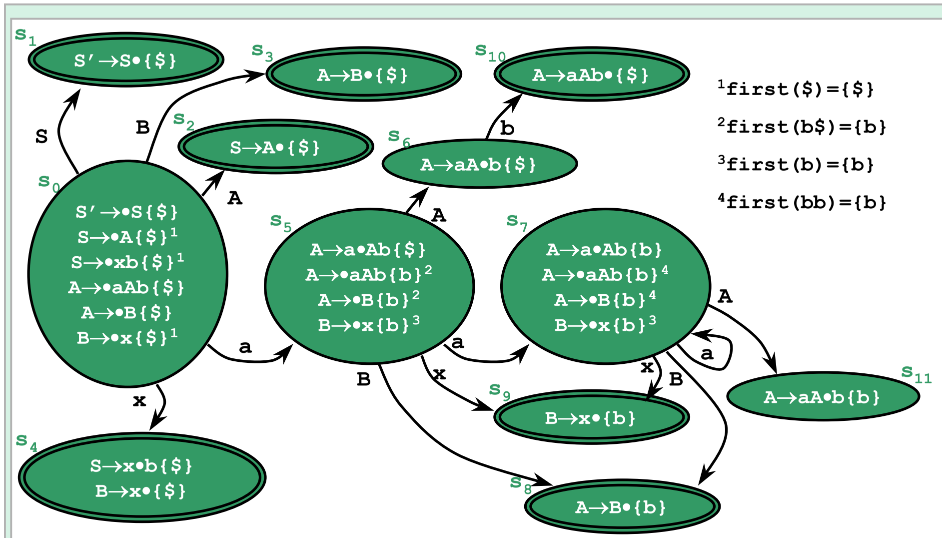
- **Deterministic Finite Automata with transitions**



LR(1) Analysis

Constructing LR(1) analysis tables

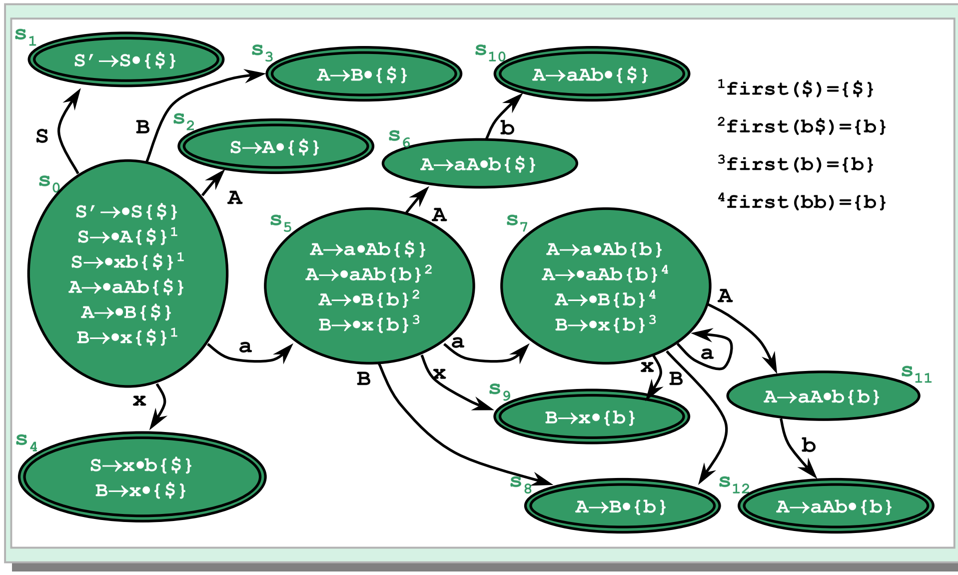
- **Deterministic Finite Automata with transitions**



LR(1) Analysis

Constructing LR(1) analysis tables

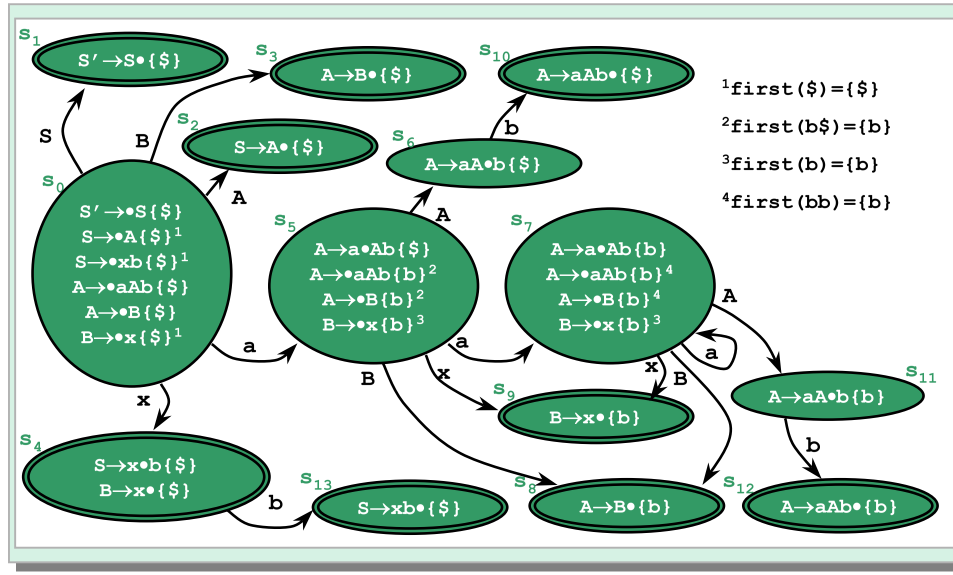
- **Deterministic Finite Automata with transitions**



LR(1) Analysis

Constructing LR(1) analysis tables

- **Deterministic Finite Automata with transitions**



LR(1) Analysis

Constructing LR(1) analysis tables: method

- **Shifts in the table:**
  - **It is the same as in LR(0)**
  - They can be obtained by following the transitions in the table.
  - If the automata can go from s_i to s_j by means of symbol **X**, then we shall add the following action:
    - `Syntactic_table[i,X]=` $\begin{cases} \texttt{sj} & \texttt{Si } X \in \Sigma_T \\ \texttt{j} & \texttt{Si } X \in \Sigma_N \end{cases}$
- **Reductions in the table:**
  - In the cells for the states which contain reduction configurations, of the form $A \to \gamma \bullet \{\sigma_1, \ldots, \sigma_n\}$ we have to add the reduction of the rule $A \to \gamma$ only in the columns for their look-ahead non-terminal symbols, i.e., $\{\sigma_1, \ldots, \sigma_n\}$.
  - Therefore, this step is **different to that in LR(0)**

LR(1) Analysis

Constructing LR(1) analysis tables: method

- **Acceptation:**
  - **It is the same as in LR(0) analysers**
  - If a state s_i has a transition with the terminal symbol **$** to the final state with the configuration `axioma'→axioma$•`, we have to add the accept action to `Syntactic_table[i,$]`.
  - Is it possible to find alternative techniques for acceptation in LR(1) parsers.
- **Error:**
  - **It is the same as in LR(0)**
  - All the empty cells have associated the `error` action.

LR(1) Analysis

Constructing LR(1) analysis tables: method

- **Analysis table**
  The following is an example of analysis with two strings:

  aaxbb
  ax

E	Σ_T				Σ_N			
	a	b	x	$	S'	S	A	B
0	s5		s4			1	2	3
1				acc				
2				r1				
3				r4				
4		s13		r5				
5	s7		s9				6	8
6		s10						
7	s7		s9				11	8
8		r4						
9		r5						
10				r3				
11		s12						
12		r3						
13				r2				
	Action				Go-to			

LR(1) Analysis: exercise

E	Σ_T				Σ_N			
	a	b	x	$	S'	S	A	B
0	s5		s4			1	2	3
1				acc				
2				r1				
3				r4				
4		s13		r5				
5	s7		s9				6	8
6		s10						
7	s7		s9				11	8
8		r4						
9		r5						
10				r3				
11		s12						
12		r3						
13				r2				
	Action				Go-to			

{xb, $a^n x b^n$ | $n \geq 0$}

a	a	x	b	b	$

0

(0) S'→S$
(1) S→A
(2) S→xb
(3) A→aAb
(4) A→B
(5) B→x

LR(1) Analysis: exercise

E	Σ_T				Σ_N			
	a	b	x	$	S'	S	A	B
0	**s5**		s4			1	2	3
1				acc				
2				r1				
3				r4				
4		s13		r5				
5	s7		s9				6	8
6		s10						
7	s7		s9				11	8
8		r4						
9		r5						
10				r3				
11		s12						
12		r3						
13				r2				
	Action				Go-to			

{xb, $a^n x b^n$ | $n \geq 0$}

a	a	x	b	b	$

5	a	0

(0) S'→S$
(1) S→A
(2) S→xb
(3) A→aAb
(4) A→B
(5) B→x

LR(1) Analysis: exercise

E	Σ_T				Σ_N			
	a	b	x	$	S'	S	A	B
0	s5		s4			1	2	3
1				acc				
2				r1				
3				r4				
4		s13		r5				
5	**s7**		s9				6	8
6		s10						
7	s7		s9				11	8
8		r4						
9		r5						
10				r3				
11		s12						
12		r3						
13				r2				
	Action				Go-to			

{xb, $a^n x b^n$ | $n \geq 0$}

a	a	x	b	b	$

7	a	5	a	0

(0) S'→S$
(1) S→A
(2) S→xb
(3) A→aAb
(4) A→B
(5) B→x

LR(1) Analysis: exercise

E	a	b	x	$	S'	S	A	B
	Σ_T				Σ_N			
0	s5		s4			1	2	3
1				acc				
2				r1				
3				r4				
4		s13		r5				
5	s7		s9				6	8
6		s10						
7	s7		**s9**				11	8
8		r4						
9		r5						
10				r3				
11		s12						
12		r3						
13				r2				
	Action				Go-to			

{xb, $a^n x b^n$ | $n \geq 0$}

a	a	x	b	b	$

9	x	7	a	5	a	0

(0) S′→S$
(1) S→A
(2) S→xb
(3) A→aAb
(4) A→B
(5) B→x

LR(1) Analysis: exercise

E	a	b	x	$	S'	S	A	B
	Σ_T				Σ_N			
0	s5		s4			1	2	3
1				acc				
2				r1				
3				r4				
4		s13		r5				
5	s7		s9				6	8
6		s10						
7	s7		s9				11	**8**
8		r4						
9		**r5**						
10				r3				
11		s12						
12		r3						
13				r2				
	Action				Go-to			

{xb, $a^n x b^n$ | $n \geq 0$}

a	a	x	b	b	$

8	B	7	a	5	a	0

(0) S′→S$
(1) S→A
(2) S→xb
(3) A→aAb
(4) A→B
(5) B→x

LR(1) Analysis: exercise

E	a	b	x	$	S'	S	A	B
	Σ_T				Σ_N			
0	s5		s4			1	2	3
1				acc				
2				r1				
3				r4				
4		s13		r5				
5	s7		s9				6	8
6		s10						
7	s7		s9				**11**	8
8		**r4**						
9		r5						
10				r3				
11		s12						
12		r3						
13				r2				
	Action				Go-to			

{xb, $a^n x b^n$ | $n \geq 0$}

a	a	x	b	b	$

11	A	7	a	5	a	0

(0) S′→S$
(1) S→A
(2) S→xb
(3) A→aAb
(4) A→B
(5) B→x

LR(1) Analysis: exercise

E	a	b	x	$	S'	S	A	B
	Σ_T				Σ_N			
0	s5		s4			1	2	3
1				acc				
2				r1				
3				r4				
4		s13		r5				
5	s7		s9				6	8
6		s10						
7	s7		s9				11	8
8		r4						
9		r5						
10				r3				
11		**s12**						
12		r3						
13				r2				
	Action				Go-to			

{xb, $a^n x b^n$ | $n \geq 0$}

a	a	x	b	b	$

12	b	11	A	7	a	5	a	0

(0) S′→S$
(1) S→A
(2) S→xb
(3) A→aAb
(4) A→B
(5) B→x

LR(1) Analysis: exercise

	Σ_T				Σ_N			
E	a	b	x	\$	S'	S	A	B
0	s5		s4			1	2	3
1				acc				
2				r1				
3				r4				
4		s13		r5				
5	s7		s9				6	8
6		s10						
7	s7		s9				11	8
8		r4						
9		r5						
10				r3				
11		s12						
12		r3						
13				r2				
	Action				Go-to			

{xb, $a^n x b^n$ | n≥0}

a	a	x	b	b	\$

6	A	5	a	0

(0) S′→S\$
(1) S→A
(2) S→xb
(3) A→aAb
(4) A→B
(5) B→x

LR(1) Analysis: exercise

	Σ_T				Σ_N			
E	a	b	x	\$	S'	S	A	B
0	s5		s4			1	2	3
1				acc				
2				r1				
3				r4				
4		s13		r5				
5	s7		s9				6	8
6		s10						
7	s7		s9				11	8
8		r4						
9		r5						
10				r3				
11		s12						
12		r3						
13				2				
	Action				Go-to			

{xb, $a^n x b^n$ | n≥0}

a	a	x	b	b	\$

10	b	6	A	5	a	0

(0) S′→S\$
(1) S→A
(2) S→xb
(3) A→aAb
(4) A→B
(5) B→x

LR(1) Analysis: exercise

	Σ_T				Σ_N			
E	a	b	x	\$	S'	S	A	B
0	s5		s4			1	2	3
1				acc				
2				r1				
3				r4				
4		s13		r5				
5	s7		s9				6	8
6		s10						
7	s7		s9				11	8
8		r4						
9		r5						
10				r3				
11		s12						
12		r3						
13				r2				
	Action				Go-to			

{xb, $a^n x b^n$ | n≥0}

a	a	x	b	b	\$

2	A	0

(0) S′→S\$
(1) S→A
(2) S→xb
(3) A→aAb
(4) A→B
(5) B→x

LR(1) Analysis: exercise

	Σ_T				Σ_N			
E	a	b	x	\$	S'	S	A	B
0	s5		s4			1	2	3
1				acc				
2				r1				
3				r4				
4		s13		r5				
5	s7		s9				6	8
6		s10						
7	s7		s9				11	8
8		r4						
9		r5						
10				r3				
11		s12						
12		r3						
13				r2				
	Action				Go-to			

{xb, $a^n x b^n$ | n≥0}

a	a	x	b	b	\$

1	S	0

(0) S′→S\$
(1) S→A
(2) S→xb
(3) A→aAb
(4) A→B
(5) B→x

LR(1) Analysis: exercise

	Σ_T				Σ_N			
E	a	b	x	$	S'	S	A	B
0	s5		s4			1	2	3
1				acc				
2				r1				
3				r4				
4		s13		r5				
5	s7		s9				6	8
6		s10						
7	s7		s9				11	8
8		r4						
9		r5						
10				r3				
11		s12						
12		r3						
13				r2				
	Action				Go-to			

{xb, a^nxb^n | n≥0}

Input: a a x b b $

Stack: 1 S 0

(0) S′→S$
(1) S→A
(2) S→xb
(3) A→aAb
(4) A→B
(5) B→x

LR(1) Analysis: exercise

	Σ_T				Σ_N			
E	a	b	x	$	S'	S	A	B
0	s5		s4			1	2	3
1				acc				
2				r1				
3				r4				
4		s13		r5				
5	s7		s9				6	8
6		s10						
7	s7		s9				11	8
8		r4						
9		r5						
10				r3				
11		s12						
12		r3						
13				r2				
	Action				Go-to			

{xb, a^nxb^n | n≥0}

Input: a x $

Stack: 0

(0) S′→S$
(1) S→A
(2) S→xb
(3) A→aAb
(4) A→B
(5) B→x

LR(1) Analysis: exercise

	Σ_T				Σ_N			
E	a	b	x	$	S'	S	A	B
0	s5		s4			1	2	3
1				acc				
2				r1				
3				r4				
4		s13		r5				
5	s7		s9				6	8
6		s10						
7	s7		s9				11	8
8		r4						
9		r5						
10				r3				
11		s12						
12		r3						
13				r2				
	Action				Go-to			

{xb, a^nxb^n | n≥0}

Input: a x $

Stack: 5 a 0

(0) S′→S$
(1) S→A
(2) S→xb
(3) A→aAb
(4) A→B
(5) B→x

LR(1) Analysis: exercise

	Σ_T				Σ_N			
E	a	b	x	$	S'	S	A	B
0	s5		s4			1	2	3
1				acc				
2				r1				
3				r4				
4		s13		r5				
5	s7		s9				6	8
6		s10						
7	s7		s9				11	8
8		r4						
9		r5						
10				r3				
11		s12						
12		r3						
13				r2				
	Action				Go-to			

{xb, a^nxb^n | n≥0}

Input: a x $

Stack: 9 x 5 a 0

(0) S′→S$
(1) S→A
(2) S→xb
(3) A→aAb
(4) A→B
(5) B→x

LR(1) Analysis: exercise

	Σ_T				Σ_N			
E	a	b	x	$	S'	S	A	B
0	s5		s4			1	2	3
1				acc				
2				r1				
3				r4				
4		s13		r5				
5	s7		s9				6	8
6		s10						
7	s7		s9				11	8
8		r4						
9		r5						
10				r3				
11		s12						
12		r3						
13				r2				
	Action				Go-to			

{xb, $a^n x b^n$ | $n \geq 0$}

a	x	$		
9	x	5	a	0

(0) S′→S$
(1) S→A
(2) S→xb
(3) A→aAb
(4) A→B
(5) B→x

LR(1) Analysis

Evaluation

- Power:
  - It can be shown that LR(1) is the most powerful analysis algorithm amongst those that analyse the string from left to right using just 1 look-ahead symbol.
- Efficiency:
  - As can be observed from the examples, there are considerably more states in LR(1) analysers than in LR(0) analysers.

LR(k) Analysis

Generalisation to k look-ahead symbols (k > 1)

- This same technique can be extended to consider any number of look-ahead symbols.
- This generalisation is out of the scope of this course.