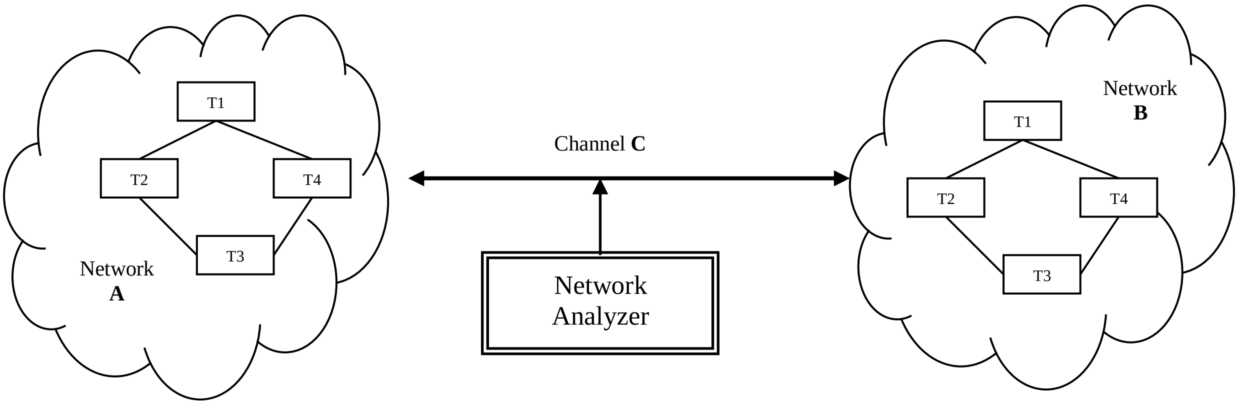
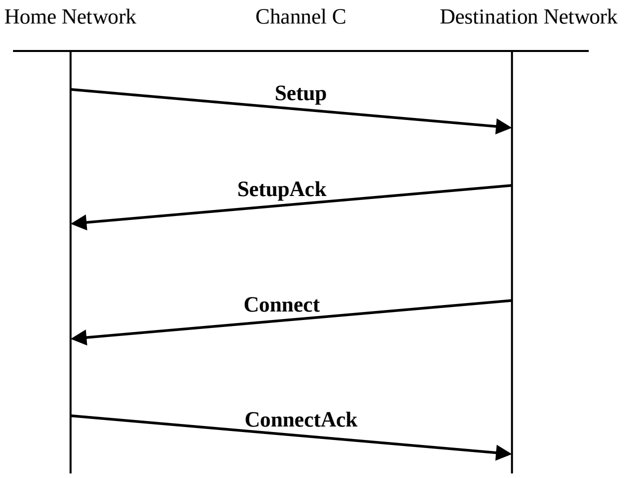
UNIT 6: BOTTOM-UP PARSING TECHNIQUES

There are two terminal networks (A and B) connected by a half-duplex C transmission channel (transmitted in both directions, but not at the same time). It also has a network analyzer that reads the messages that circulate through the channel



When one terminal of a network wants to establish a communication with a terminal of the other network, four messages are exchanged through channel C described by the following protocol:



From the home network a **Setup message** is sent to the destination network indicating that it wants to establish a communication with a specific machine of that network. The destination network must respond with a **SetupAck** message as acknowledgment of the received **Setup** message. Once the destination network terminal accepts the call request, it must send a **Connect message** to the terminal of the source network. The source network must respond with a **ConnectAck** as an acknowledgment of the received Connect message.

Each message consists of a series of Information Elements (**EI**). Each **EI**, in turn, consists of two fields. The first field always has a size of 1 byte and its value corresponds to the identifier of that **EI**. The second field will be the **EI** argument and its size depends on the **EI**.

David Griol Barres, Antonio Berlanga de Jesús, Jesús García Herrero, Juan Manuel Alonso Weber

The messages with the **EI**s of which they are composed are described below. These **EI** must appear in the order shown in the following table. The **EI** of a message may be mandatory or optional. Mandatory **EI** must always be included in the message. The optional **EI**s may or may not appear.

MESSAGE	**Information Elements**	**Mandatory / Optional**
Setup	EISetup	Mandatory
	EIAddress	Mandatory
	EIOrigin	Optional
	EIDestination	Mandatory
SetupAck	EISetupAck	Mandatory
	EIDirection	Mandatory
Connect	EIConnect	Mandatory
	EIDirection	Mandatory
ConnectAck	EIConnectAck	Mandatory
	EIDirection	Mandatory

The following table describes the defined **EI** with the description of the fields of which it is composed. For each field its size in bytes and its value are indicated. The first field always has a fixed value since it is the identifier of the **EI** and the second field has a variable value that corresponds to the argument of described **EI**:

		FIELDS					
		Field 1 (Identifier EI)			**Campo 2 (Identifier EI)**		
		Name	**Size**	**Value**	**Name**	**Size**	**Value**
EI	EISetup	IDSetup	1 byte	‘A’	CREF	1 byte	[A-Z,a-z,0-9] Any alphabetic or numeric character
	EISetupAck	IDSetupAck	1 byte	‘B’	CREF	1 byte	Same that Setup
	EIConnect	IDConnect	1 byte	‘C’	CREF	1 byte	Same that SetupAck
	EIConnectAck	IDConnectAck	1 byte	‘D’	CREF	1 byte	Same that Connect
	EIDirection	IDDirection	1 byte	‘E’	Direction	1 byte	A→B = ‘0’ B→A = ‘1’
	EIOrigin	IDOrigin	1 byte	‘F’	Origin	3 bytes	$[0-9]^3$. Three digits
	EIDestination	IDDestination	1 byte	‘G’	Destination	3 bytes	$[0-9]^3$. Three digits

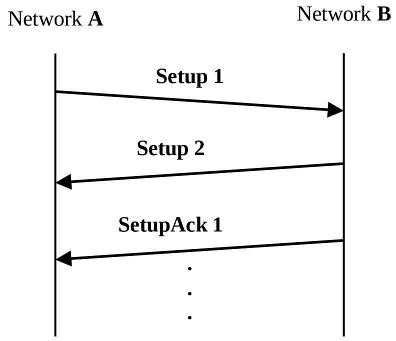
Since messages of several different communications can be circulated through the same channel, it is possible that a message of another communication appears between the four messages corresponding to the same communication. In order to identify the communication to which a message belongs, the CREF field of **EISetup, EISetupAck, EIConnect** and **EIConnectAck** is used. Therefore, the four messages of which a communication is composed must contain the same value in their CREF argument of said Information Elements.

On the other hand, the **EI** EIDirection indicates whether the message has been transmitted in the A→B or B→A direction. Its possible values are '0' or '1' respectively.

EIOrigin and EIDestination contain in their argument the address of the origin and destination terminals respectively.

Example:

Imagine that a **Setup** is sent from A to B, but before receiving the **SetupAck** from B, a **Setup** from B to A corresponding to another communication has been sent



The network analyzer would collect the following data:

A	4	E	0	F	3	1	2	G	4	7	5	A	8	E	1	G	1	3	4	B	4	E	1

EISetup (A 4) · EIDirection (E 0) · EIOrigin (F 3 1 2) · EIDestination (G 4 7 5) → Message Setup 1

EISetup (A 8) · EIDirection (E 1) · EIDestination (G 1 3 4) → Message Setup 2

EISetupAck (B 4) · EIDirection (E 1) → Message SetupAck 1

It is required:

1. Describe formally the automaton that the network analyzer uses as a lexical analyzer, whose tokens are the Elements of Information (**EI**).

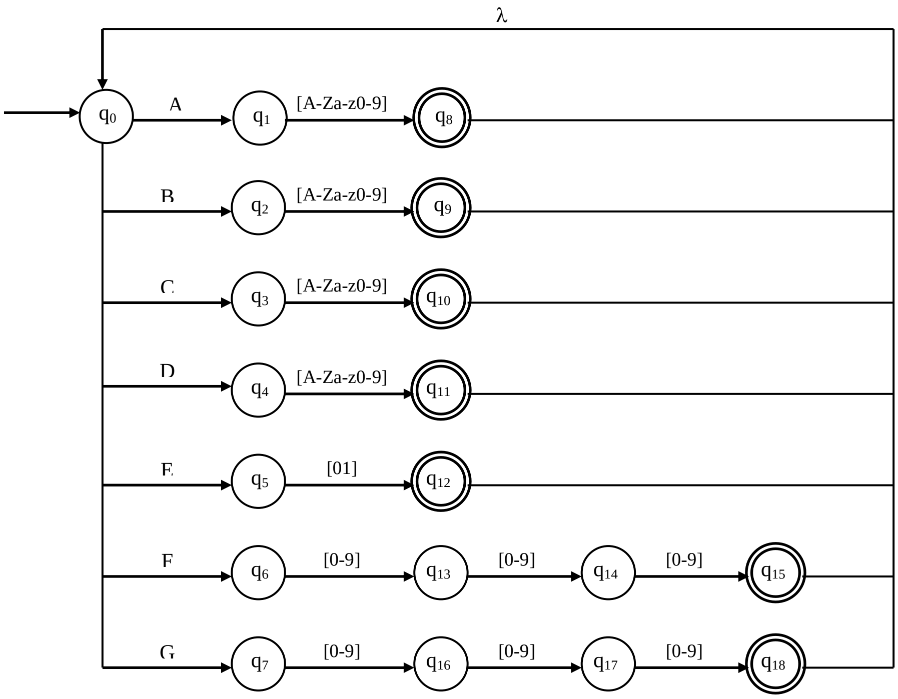
2. Define a grammar that generates messages that can be read on channel C. Is it LL(1)?. Otherwise, make the necessary modifications to make it.

3. Can the grammar of the second exercise be used to perform an LR(1) analysis? If not, modify it to make it so. Generate the first 8 states of the LR(1) analyzer for that grammar.

David Griol Barres, Antonio Berlanga de Jesús, Jesús García Herrero, Juan Manuel Alonso Weber

Solution

For the implementation of the lexical analysis, we are going to use a non-deterministic finite automaton as the one shown below:



The formal description is;

$$NFA = (\Sigma_T, \{q_0..q_{18}\}, q_0, f, F)$$

$$\Sigma_T = \{A\text{-}Z, a\text{-}z, 0\text{-}9\}$$
$$F = \{q_8, q_9, q_{10}, q_{11}, q_{12}, q_{15}, q_{18}\}$$

$f(q_0, A) = q_1$
$f(q_0, B) = q_2$
$f(q_0, C) = q_3$
$f(q_0, D) = q_4$
$f(q_0, E) = q_5$
$f(q_0, F) = q_6$
$f(q_0, F) = q_7$
$f(q_1, \{A\text{-}Z,a\text{-}z,0\text{-}9\}) = q_8$
$f(q_3, \{A\text{-}Z,a\text{-}z,0\text{-}9\}) = q_{10}$
$f(q_4, \{A\text{-}Z,a\text{-}z,0\text{-}9\}) = q_{11}$
$f(q_5, \{01\}) = q_{12}$
$f(q_6, \{0\text{-}9\}) = q_{13}$
$f(q_7, \{0\text{-}9\}) = q_{16}$

$f(q_8, \lambda) = q_0$
$f(q_9, \lambda) = q_0$
$f(q_{10}, \lambda) = q_0$
$f(q_{11}, \lambda) = q_0$
$f(q_{12}, \lambda) = q_0$
$f(q_{13}, \{0\text{-}9\}) = q_{14}$
$f(q_{14}, \{0\text{-}9\}) = q_{15}$
$f(q_{15}, \lambda) = q_0$
$f(q_{16}, \{0\text{-}9\}) = q_{17}$
$f(q_{17}, \{0\text{-}9\}) = q_{18}$
$f(q_{18}, \lambda) = q_0$

When a token is read, it returns to the initial state q0 to read the next one. In the table, f (qi, {c1, ..., cn}) = qj means that the transition is made with any of the characters that appear between braces.

The final states generate the following tokens:

sTATE	Token
q_8	(EIsetup, A)
q_9	(EIsetupack, B)
q_{10}	(EIconnect, C)
q_{11}	(EIconnectack, D)
q_{12}	(EIDirection, [01])
q_{15}	(EIOrigin, $[0-9]^3$)
q_{18}	(EIDestination, $[0-9]^3$)

A posible grammar is:

1 S ::= M S
2 ::= λ
3 M ::= eisetup EIDirection O EIDestination
4 ::= eisetupack EIDirection
5 ::= eiconnect EIDirection
6 ::= eiconnectack EIDirection
7 O :: EIOrigin
8 ::= λ

To create the LL(1) analyzer, we start with the FIRST and FOLLOW set:

Σ_N	FIRST	FOLLOW
S	eisetup, eisetupack, eiconnect, eiconnectack, λ	$
M	eisetup, eisetupack, eiconnect, eiconnectack	eisetup, eisetupack, eiconnect, eiconnectack, $
O	EIOrigin, λ	EIDestination

The LL table is:

Σ_T	S	M	O
eisetup	1	3	
eisetupack	1	4	
eiconnect	1	5	
eiconnectack	1	6	
EIDirection			
EIOrigin			7
EIDestination			8
$	2		

This table is LL(1).

To know if it is LR(1) it is necessary to generate the table LR(1) and check that there is no problem. The first eight states generated from the grammar of the previous exercise will be.

State 0		Action	Go To
S'	::= •S		(0, S) = 1
S	::= •M S		(0, M) = 2
	::= λ	(0, $) = R2	
M	::= •eisetup EIDirection O EIDestination	(0, eisetup) = D3	
	::= •eisetupack EIDirection	(0, eisetupack) = D4	
	::= •eiconnect EIDirection	(0, connect) = D5	
	::= •eiconnectack EIDirection	(0, connectack) = D6	

State 1	Action	Go To
S' ::= S •	(1, $) = Aceptar	

State 2	Action	Go To
S ::= M • S		(2, S) = 7
S ::= •M S		(2, M) = 2
::= λ	(2, $) = R2	
M ::= •eisetup EIDirection O EIDestination	(2, eisetup) = D3	
::= •eisetupack EIDirection	(2, eisetupack) = D4	
::= •eiconnect EIDirection	(2, connect) = D5	
::= •eiconnectack EIDirection	(2, connectack) = D6	

State 3	Action	Go To
M ::= eisetup • EIDirection O EIDestination		

State 4	Action	Go To
M ::= eisetupack • EIDirection		

State 5	Action	Go To
M ::= eiconnect • EIDirection		

State 6	Action	Go To
M ::= eiconnectack • EIDirection		

State 7	Action	Go To
S' ::= M S •	(7, $) = R1	