Part I

Grammar

1 Lexer

```
COMMENT:     '-' '-' ~('\n'| '-' )* ('-''-'|'\n')  {skip();};
WS:          ('\t'|'\r'|'\n'|' ')                  {skip();};
```

First of all, tokens for the comments and whitespaces are created; as can be seen a comment is everything between two sets of "– –" or a "– –" and a new line character, and the whitespaces are tabulation characters, new lines or simply spaces. Both these tokens are skipped during the parsing operations.

```
fragment A:        'A'|'a';
fragment B:        'B'|'b';
fragment C:        'C'|'c';
fragment D:        'D'|'d';
fragment E:        'E'|'e';
fragment F:        'F'|'f';
fragment G:        'G'|'g';
fragment I:        'I'|'i';
fragment J:        'J'|'j';
fragment K:        'K'|'k';
fragment L:        'L'|'l';
fragment M:        'M'|'m';
fragment N:        'N'|'n';
fragment O:        'O'|'o';
fragment P:        'P'|'p';
fragment R:        'R'|'r';
fragment S:        'S'|'s';
fragment T:        'T'|'t';
fragment U:        'U'|'u';
fragment V:        'V'|'v';
fragment NUMBER: (('0' '.')|(('1'..'9') ('0'..'9')* '.'?) )('0'..'9')*;
```

WaSabi is a non-case sensitive language, and because of this characteristic (and also to create a tidier grammar file), fragments containing both the upper-case and lower case are defined for some letters; a fragment for all the numbers but the zero is also defined. The state machine used to parse the NUMBER fragment is shown in figure 1.

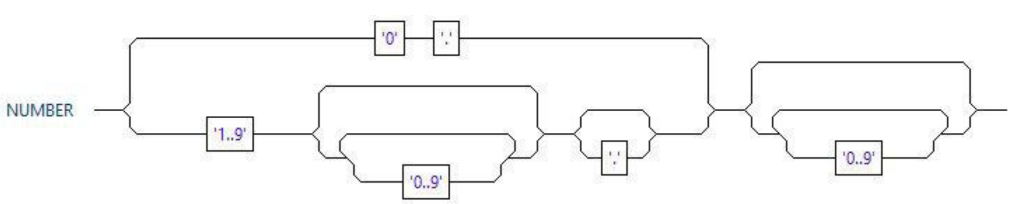

Figure 1: Numbers state machine

```
OPENBRACKET:     '(' | '[' | '{';
CLOSEBRACKET:    ')' | ']' | '}';
SEPARATOR:       ',' | ';';
```

Following, there are the brackets and separator tokens: all the types of brackets can be used alternatively, and both the comma and semicolon can be used as separators.

```
CIRCUIT:      C I R C U I T             ':'?;
LIBRARY:      L I B (R A R I E S)?      ':'?;
CONST:        C O N S T (A N T S)?      ':'?;
```

```
COMPONENT:   C O M P (O N E N T S)?  ':'?;
SIMULATE:    S I M (U L A T E )?     ':'?;
```

Next, tokens definitions for the basic WaSabi program sections[I], with a cuple variations each and an optional colon.

```
AT:      A T;
GND:     G N D;
DC:      D C;
AC:      A C;
MODEL:   M O D (E L)?;
RES:     R (E S (I S T A N C E)?)?;
CAP:     C (A P (A C I T A N C E )?)?;
IND:     L | (I N D (U C T A N C E)?);
VOL:     V (O L (T A G E)?)?;
CUR:     I | (C U R (R E N T)?);
DIO:     D (I O (D E)?)?;
BJT:     B (J T )?;
MOS:     M (O S (F E T)?)?;
JFET:    J (F E T)?;
```

The tokens used for some basic keywords such as "at" or "gnd" and the tokens for the components type are pretty straightforward and use a combination of letter fragments (sometimes with optional parts to create alternative keywords).

```
EQUAL: '=';
```

The equal token is also pretty straightforward.

```
PI:  NUMBER? P I;
```

The "*PI*" token is used to indicate the number $i \cdot \pi$ with $i > 0$. Note that the "*NUMBER*" token has been used so that the zero is not an option when declaring a multiple of π.

```
UNIT:   F |
        P |
        N |
        U |
        M I L L |
        K |
        M E G |
        G |
        T;
```

The next token, the "*UNIT*" one, is used to indicate an SI prefix used in numbers scientific notation; a range of prefixes varying from 10^{-15} to 10^{12} can be used in the WaSabi language. This range has been chosen to be the same as the one SPICE uses.

```
fragment FILENAME: (~('/'|'\\'|':'|'*'|'?'|'<'|'>'|'|'|'\r'|'\t'|'\n')+);

         PATH:     (('A'..'Z' ':')?)
                   (('\\')? FILENAME )+
                   '.' ('A'..'Z'|'a'..'z')+;
```

The "*PATH*" token indicates a file system path: the fragment "*FILENAME*" represent all the possible names that a folder or a file can have, and is basically a sequence of allowed characters, whereas the "*PATH*" token is a "*FILENAME*" optionally preceded by a disk name (such as C:\\) and followed by the file extension. The state machine for the PATH terminal is shown in figure

```
VALUE:  NUMBER | '0';
```

The "*VALUE*" token is simply a "*NUMBER*" token, or the zero.

```
MODTYPE:  '<' ('A'..'Z'|'a'..'z'|'0'..'9'|'-')+ '>';
```

[I] See the user manual for more in-depth explaination about program sections.

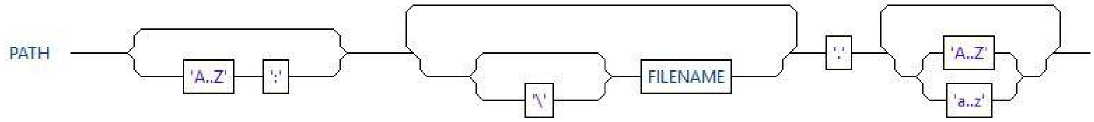

Figure 2: Path terminal state machine

Next, the "*MODTYPE*" token is used to indicate a custom model: usually a model name is a combination of numbers, letters and dashes. Per language specification, a model name is enclosed in pointy brackets: this choice has been made mainly in order to make it easily identifiable by the lexer.

```
ID:  ('A'..'Z'|'a'..'z'|'_') ('0'..'9'|'a'..'z'|'A'..'Z'|'_')*;
```

The "*ID*" token; this represents the name of a constant during declaration: as usual it's created combining a letter with a sequence of numbers, letters and underscores.

```
DIRECTIVE:  '.' (~('\n'|'\r'))+;
```

Finally, the last token is the "*DIRECTIVE*" token: it is used to indicate a SPICE directive, and the correctness check is delegated to SPICE itself. The WaSabi language enforces only that a directive starts with a period.

2 Parser

The WaSabi language uses the following parsing rules:

```
axiom:  title? pr;
```

The grammar axiom is composed of the "*title*" non-terminal (which is optional) followed by the "*pr*" (short for program) non-terminal.

```
title:  CIRCUIT ID;
pr:     ( ( library | constants | components | sim ) pr)?;
```

The "*title*" non-terminal is pretty simple, and it's composed of the "*CIRCUIT*" and "*ID*" terminals, which have already been covered in section 1; the "*pr*" non-terminal is right-recursive and also optional: this format allows the user to define the four program sections in any order, repeating them any number of times.

```
library:  LIBRARY listlib;
listlib:  (newlib listlib)?;
newlib:   PATH;
```

The "*library*" non-terminal starts with a "*LIBRARY*" token, which has already been presented, followed by a list of paths, generated by the non-terminal "*listlib*". The "*listlib*" non-terminal is optional, which means that the list can also be empty (a circuit that imports no libraries is permitted); the list structure is realized using, again, a right-recursive format.

```
constants:  CONST listconst;
listconst:  (newconst listconst)?;
newconst:   ID EQUAL newconst2;
newconst2:  newvalue|newwave|newmod;
```

The "*constants*" non-terminal is very similar to the "*library*" one: the list generates from the non-terminal in the same way, but the difference lies in the elements of the list; the one generated by "*listconst*" are composed of an "*ID*" terminal (the name of the constant), followed by equals and the "*newconst2*" non-terminal, which can be a value, a waveform or a model, generated respectively from a "*newvalue*", "*newwave*" and "*newmod*" non-terminals.

```
newvalue:  VALUE units | PI;
units:     UNIT?

newmod:    MODTYPE;

newwave:   newwaveDC|newwaveAC;
```

```
newwaveDC:  DC OPENBRACKET valueID CLOSEBRACKET;
newwaveAC:  AC OPENBRACKET valueID SEPARATOR valueID CLOSEBRACKET;
```

The "*newvalue*" and "*newmod*" non-terminals are simple, just a combination of their respective terminals; the "*newwave*" non-terminal is a little bit more complex, given that it must take into account that two possible waveform declaration are permitted: DC and AC. The two types of waveform are similar, the only difference being that the AC one requires one more parameter that its DC counterpart; both these non-terminal generate a terminal followed by the list of parameters (one for DC, two for AC) enclosed in brackets.

```
components:  COMPONENT listcomp;
listcomp:    (newcomp listcomp)?;
newcomp:     resistance    |
             capacitance   |
             inductance    |
             voltage       |
             current       |
             diode         |
             bjt           |
             mosfet        |
             jfet          |
             model;
```

Again, the "*component*" non-terminal is similar to both the "*library*" and "*constants*" ones: it just creates a list using the right-recursive non-terminal "*listcomp*"; the list can be made up of various components, where each unique component has its own non-terminal, which will be illustrated next.

```
resistance:   RES   valueID   AT OPENBRACKET node SEPARATOR node CLOSEBRACKET;
capacitance:  CAP   valueID   AT OPENBRACKET node SEPARATOR node CLOSEBRACKET;
inductance:   IND   valueID   AT OPENBRACKET node SEPARATOR node CLOSEBRACKET;
voltage:      VOL   waveID    AT OPENBRACKET node SEPARATOR node CLOSEBRACKET;
current:      CUR   waveID    AT OPENBRACKET node SEPARATOR node CLOSEBRACKET;
diode:        DIO   modtypeID AT OPENBRACKET node SEPARATOR node CLOSEBRACKET;

bjt:          BJT   modtypeID AT OPENBRACKET node SEPARATOR node SEPARATOR node CLOSEBRACKET;
mosfet:       MOS   modtypeID AT OPENBRACKET node SEPARATOR node SEPARATOR node CLOSEBRACKET;
jfet:         JFET  modtypeID AT OPENBRACKET node SEPARATOR node SEPARATOR node CLOSEBRACKET;

model:        MODEL modtypeID AT OPENBRACKET listnodes CLOSEBRACKET;
listnodes:    node listnodes2;
listnodes2:   (SEPARATOR node listnodes2)?;
```

The first classification of components are the bipoles: the resistance, capacitance, inductance, diode and voltage/current sources fall in this category. All the non-terminals representing these components follow the same pattern:

- A terminal indicating the type of component;
- The "*valueID*", "*waveID*" or "*modtypeID*" non-terminal ; these non-terminal are used instead of "*newvalue*", "*newwave*" and "*newmodtype*" respectively to allow on-the-go constants declaration inside components declarations;
- The terminal "*AT*";
- An opening bracket;
- The "*node*" non-terminal, representing the first note to which the bipole is connected;
- A separator;
- The second "*node*" non-terminal;
- A closing bracket;

The second classification of components are the tripoles, such as the BJT, MOSFET and JFET transistors. These are very similar to the bipoles, but have one more "*node*" non-terminal between the brackets.

Lastly, the third classification of components are the n-poles, created from the "model" non-terminal, and can have an indefinite number of "*node*" non-terminals between the brackets; this makes necessary the use of a list, that is created using two more non-terminals ("*listnodes*" and "*listnodes*2") one of which is right-recursive.

```
node:       GND      | ID;
valueID:    newvalue | ID;
waveID:     newwave  | ID;
modtypeID:  newmod   | ID;
```

The "node" and IDs are straightforward: the "*node*" can be a ground or a user-defined name, and the IDs can be a user defined name (if the user has already defined the constant) or a on-the-go constant definition.

```
sim:     SIMULATE dirsim;
dirsim:  (newdir dirsim)?;
newdir:  DIRECTIVE;
```

To conclude the explaination of the parser, the "*sim*" non-terminal is the last piece of the "*pr*" non-terminal symbol, which creates a list of "$DIRECTIVE$" terminals.

2.1 Analysis

For the grammar to be $LL(1)$, the following must be true:

$$P(A \to \alpha_1) \cap P(A \to \alpha_2) = \varnothing$$

for each pair of productions $A \to \alpha_1$,$A \to \alpha_2$ generated by the same non-terminal A, and where $P(A \to \alpha)$ is the predict set of $A \to \alpha$.

To compute the predict sets, the first sets F and follow sets L are needed.

2.1.1 Follow sets

In a grammar with alphabet Σ, the follow set of a non-terminal A is:

$$L(A) = \{a \in \Sigma | s \to \alpha A \beta\}$$

thus the follow sets of the WaSabi grammar are as follows:

Non-terminal symbol	Follow set
axiom	$\swarrow$
pr	$\swarrow$
title	LIBRARY, CONST, COMPONENT, SIMULATE, $\swarrow$
library	LIBRARY, CONST, COMPONENT, SIMULATE, $\swarrow$
listlib	LIBRARY, CONST, COMPONENT, SIMULATE, $\swarrow$
newlib	PATH, LIBRARY, CONST, COMPONENT, SIMULATE, $\swarrow$
constants	LIBRARY, CONST, COMPONENT, SIMULATE, $\swarrow$
listconst	LIBRARY, CONST, COMPONENT, SIMULATE, $\swarrow$
newconst	ID, LIBRARY, CONST, COMPONENT, SIMULATE, $\swarrow$
newconst2	ID, LIBRARY, CONST, COMPONENT, SIMULATE, $\swarrow$
newvalue	ID, LIBRARY, CONST, COMPONENT, SIMULATE, $\swarrow$
newwave	ID, LIBRARY, CONST, COMPONENT, SIMULATE, $\swarrow$
newwaveDC	ID, LIBRARY, CONST, COMPONENT, SIMULATE, $\swarrow$
newwaveAC	ID, LIBRARY, CONST, COMPONENT, SIMULATE, $\swarrow$
newmod	ID, LIBRARY, CONST, COMPONENT, SIMULATE, $\swarrow$
components	LIBRARY, CONST, COMPONENT, SIMULATE, $\swarrow$
listcomp	LIBRARY, CONST, COMPONENT, SIMULATE, $\swarrow$
newcomp	RES, CAP, IND, VOL, CUR, DIO, BT, MOS, JFET, MODEL, LIBRARY, CONST, COMPONENT, SIMULATE, $\swarrow$
resistance	RES, CAP, IND, VOL, CUR, DIO, BT, MOS, JFET, MODEL, LIBRARY, CONST, COMPONENT, SIMULATE, $\swarrow$
capacitance	RES, CAP, IND, VOL, CUR, DIO, BT, MOS, JFET, MODEL, LIBRARY, CONST, COMPONENT, SIMULATE, $\swarrow$
inductance	RES, CAP, IND, VOL, CUR, DIO, BT, MOS, JFET, MODEL, LIBRARY, CONST, COMPONENT, SIMULATE, $\swarrow$
voltage	RES, CAP, IND, VOL, CUR, DIO, BT, MOS, JFET, MODEL, LIBRARY, CONST, COMPONENT, SIMULATE, $\swarrow$
current	RES, CAP, IND, VOL, CUR, DIO, BT, MOS, JFET, MODEL, LIBRARY, CONST, COMPONENT, SIMULATE, $\swarrow$
diode	RES, CAP, IND, VOL, CUR, DIO, BT, MOS, JFET, MODEL, LIBRARY, CONST, COMPONENT, SIMULATE, $\swarrow$
bjt	RES, CAP, IND, VOL, CUR, DIO, BT, MOS, JFET, MODEL, LIBRARY, CONST, COMPONENT, SIMULATE, $\swarrow$
mosfet	RES, CAP, IND, VOL, CUR, DIO, BT, MOS, JFET, MODEL, LIBRARY, CONST, COMPONENT, SIMULATE, $\swarrow$
jfet	RES, CAP, IND, VOL, CUR, DIO, BT, MOS, JFET, MODEL, LIBRARY, CONST, COMPONENT, SIMULATE, $\swarrow$
model	RES, CAP, IND, VOL, CUR, DIO, BT, MOS, JFET, MODEL, LIBRARY, CONST, COMPONENT, SIMULATE, $\swarrow$
node	SEPARATOR, CLOSEBRACKET
valueID	AT, SEPARATOR, CLOSEBRACKET
waveID	AT
modtypeID	AT
listnodes	CLOSEBRACKET
listnodes2	CLOSEBRACKET
sim	LIBRARY, CONST, COMPONENT, SIMULATE, $\swarrow$
dirsim	LIBRARY, CONST, COMPONENT, SIMULATE, $\swarrow$
newdir	DIRECTIVE, LIBRARY, CONST, COMPONENT, SIMULATE, $\swarrow$

First and predict sets In a grammar with alphabet Σ, the first set of a string stringα is

$$\begin{aligned} F(\alpha) &= \{a \in \Sigma \mid \alpha \to a\beta\} \\ F(\varepsilon) &= \varnothing \end{aligned}$$

and the predict set of a production $A \to \alpha$ is

$$\begin{array}{rl} F(\alpha) \cup L(A) & \text{if } \alpha \text{ can be null} \\ F(\alpha) & \text{otherwise} \end{array}$$

Axiom The axiom non-terminal has the following production:

axiom → title? pr

which is equivalent to

(1) axiom →title pr

(2) axiom → pr

Production	First set	Predict set
1	CIRCUIT	CIRCUIT
2	LIBRARY, CONST, COMPONENT, SIMULATE	LIBRARY, CONST, COMPONENT, SIMULATE, $\surd$

Every intersection between each pair of predict set is empty, therefore this production is $LL(1)$.

Pr The pr non-terminal has the following production:

pr → ((library | constants | components | sim) pr)?

which is equivalent to

(1) pr → library pr

(2) pr → constants pr

(3) pr → components pr

(4) pr → sim pr

(5) pr → ε

Production	First set	Predict set
1	LIBRARY	LIBRARY
2	CONST	CONST
3	COMPONENT	COMPONENT
4	SIM	SIM
5	$\varnothing$	$\surd$

Every intersection between each pair of predict set is empty, therefore this production is $LL(1)$.

Title The title non-terminal has the following production:

title → CIRCUIT ID

which is equivalent to

(1) title → CIRCUIT ID

Production	First set	Predict set
1	CIRCUIT	CIRCUIT

There are no possible intersection between predict sets, therefore this production is automatically $LL(1)$.

Library The library non-terminal has the following production:

$$\text{library} \rightarrow \text{LIBRARY listlib}$$

which is equivalent to

$$(1) \qquad \text{library} \rightarrow \text{LIBRARY listlib}$$

Production	First set	Predict set
1	LIBRARY	LIBRARY

There are no possible intersection between predict sets, therefore this production is automatically $LL(1)$.

Listlib The listlib non-terminal has the following production:

$$\text{listlib} \rightarrow (\text{ newlib listlib })?$$

which is equivalent to

$$(1) \qquad \text{listlib} \rightarrow \text{newlib listlib}$$
$$(2) \qquad \text{listlib} \rightarrow \varepsilon$$

Production	First set	Predict set
1	PATH	PATH
2	∅	LIBRARY, CONST, COMPONENT, SIMULATE, $\surd$

Every intersection between each pair of predict set is empty, therefore this production is $LL(1)$.

Newlib The newlib non-terminal has the following production:

$$\text{newlib} \rightarrow \text{PATH}$$

which is equivalent to

$$(1) \qquad \text{newlib} \rightarrow \text{PATH}$$

Production	First set	Predict set
1	PATH	PATH

There are no possible intersection between predict sets, therefore this production is automatically $LL(1)$.

Constants The newlib non-terminal has the following production:

$$\text{constants} \rightarrow \text{CONST listconst}$$

which is equivalent to

$$(1) \qquad \text{constants} \rightarrow \text{CONST listconst}$$

Production	First set	Predict set
1	CONST	CONST

There are no possible intersection between predict sets, therefore this production is automatically $LL(1)$.

Listconst The listconst non-terminal has the following production:

$$\text{listconst} \rightarrow (\text{ newconst listconst })?$$

which is equivalent to

$$(1) \qquad \text{listconst} \rightarrow \text{newconst listconst}$$
$$(2) \qquad \text{listconst} \rightarrow \varepsilon$$

Production	First set	Predict set
1	ID	ID
2	∅	LIBRARY, CONST, COMPONENT, SIMULATE, $\surd$

Every intersection between each pair of predict set is empty, therefore this production is $LL(1)$.

Newconst The newconst non-terminal has the following production:

$$\text{newconst} \rightarrow \text{ID EQUAL newconst2}$$

which is equivalent to

$$(1) \qquad \text{newconst} \rightarrow \text{ID EQUAL newconst2}$$

Production	First set	Predict set
1	ID	ID

There are no possible intersection between predict sets, therefore this production is automatically $LL(1)$.

Newconst2 The newconst2 non-terminal has the following production:

$$\text{newconst2} \rightarrow \text{newvalue} \mid \text{newwave} \mid \text{newmod}$$

which is equivalent to

$$(1) \qquad \text{newconst2} \rightarrow \text{newvalue}$$
$$(2) \qquad \text{newconst2} \rightarrow \text{newwave}$$
$$(3) \qquad \text{newconst2} \rightarrow \text{newmod}$$

Production	First set	Predict set
1	VALUE	VALUE
2	DC, AC	DC, AC
3	MODTYPE	MODTYPE

Every intersection between each pair of predict set is empty, therefore this production is $LL(1)$.

Newvalue The newvalue non-terminal has the following production:

$$\text{newvalue} \rightarrow \text{VALUE units} \mid \text{PI}$$

which is equivalent to

$$(1) \qquad \text{newvalue} \rightarrow \text{VALUE units}$$
$$(2) \qquad \text{newvalue} \rightarrow \text{PI}$$

Production	First set	Predict set
1	VALUE	VALUE
2	PI	PI

Every intersection between each pair of predict set is empty, therefore this production is $LL(1)$.

Units The units non-terminal has the following production:

$$\text{units} \rightarrow \text{UNIT?}$$

which is equivalent to

$$(1) \qquad \text{units} \rightarrow \text{UNIT}$$
$$(2) \qquad \text{units} \rightarrow \varepsilon$$

Production	First set	Predict set
1	UNIT	UNIT
2	$\varnothing$	ID, LIBRARY, CONST, COMPONENT, SIMULATE, $\surd$

Every intersection between each pair of predict set is empty, therefore this production is $LL(1)$.

Newwave The newwave non-terminal has the following production:

$$\text{newwave} \rightarrow \text{newwaveDC} \mid \text{newwaveAC}$$

which is equivalent to

$$\begin{array}{ll} (1) & \text{newwave} \rightarrow \text{newwaveDC} \\ (2) & \text{newwave} \rightarrow \text{newwaveAC} \end{array}$$

Production	First set	Predict set
1	DC	DC
2	AC	AC

Every intersection between each pair of predict set is empty, therefore this production is $LL(1)$.

NewwaveDC The newwaveDC non-terminal has the following production:

$$\text{newwaveDC} \rightarrow \text{DC OPENBRACKET valueID CLOSEBRACKET}$$

which is equivalent to

$$(1) \quad \text{newwaveDC} \rightarrow \text{DC OPENBRACKET valueID CLOSEBRACKET}$$

Production	First set	Predict set
1	DC	DC

There are no possible intersection between predict sets, therefore this production is automatically $LL(1)$.

NewwaveAC The newwaveAC non-terminal has the following production:

$$\text{newwaveAC} \rightarrow \text{AC OPENBRACKET valueID SEPARATOR valueID CLOSEBRACKET}$$

which is equivalent to

$$(1) \quad \text{newwaveAC} \rightarrow \text{AC OPENBRACKET valueID SEPARATOR valueID CLOSEBRACKET}$$

Production	First set	Predict set
1	AC	AC

There are no possible intersection between predict sets, therefore this production is automatically $LL(1)$.

Newmod The newmod non-terminal has the following production:

$$\text{newmod} \rightarrow \text{MODTYPE}$$

which is equivalent to

$$(1) \quad \text{newmod} \rightarrow \text{MODTYPE}$$

Production	First set	Predict set
1	MODTYPE	MODTYPE

There are no possible intersection between predict sets, therefore this production is automatically $LL(1)$.

Components The components non-terminal has the following production:

$$\text{components} \rightarrow \text{COMPONENT listcomp}$$

which is equivalent to

$$(1) \quad \text{components} \rightarrow \text{COMPONENT listcomp}$$

Production	First set	Predict set
1	COMPONENT	COMPONENT

There are no possible intersection between predict sets, therefore this production is automatically $LL(1)$.

Listcomp The listcomp non-terminal has the following production:

$$\text{listcomp} \rightarrow (\text{ newcomp listcomp })?$$

which is equivalent to

$$\begin{array}{ll} (1) & \text{listcomp} \rightarrow \text{newcomp listcomp} \\ (2) & \text{listcomp} \rightarrow \varepsilon \end{array}$$

Production	First set	Predict set
1	RES, CAP, IND, VOL, CUR, DIO, BJT, MOSFET, JFET, MODEL	RES, CAP, IND, VOL, CUR, DIO, BJT, MOSFET, JFET, MODEL
2	∅	LIBRARY, CONST, COMPONENT, SIMULATE, ↲

Every intersection between each pair of predict set is empty, therefore this production is $LL(1)$.

Newcomp The newcomp non-terminal has the following production:

$$\text{newcomp} \rightarrow \text{resistance} \mid \text{capaticance} \mid \text{inductance} \mid \text{voltage} \mid \text{current} \mid \text{diode} \mid \text{bjt} \mid \text{mosfet} \mid \text{jfet} \mid \text{model}$$

which is equivalent to

$$\begin{array}{ll} (1) & \text{newcomp} \rightarrow \text{resistance} \\ (2) & \text{newcomp} \rightarrow \text{capaticance} \\ (3) & \text{newcomp} \rightarrow \text{inductance} \\ (4) & \text{newcomp} \rightarrow \text{voltage} \\ (5) & \text{newcomp} \rightarrow \text{current} \\ (6) & \text{newcomp} \rightarrow \text{diode} \\ (7) & \text{newcomp} \rightarrow \text{bjt} \\ (8) & \text{newcomp} \rightarrow \text{mosfet} \\ (9) & \text{newcomp} \rightarrow \text{jfet} \\ (10) & \text{newcomp} \rightarrow \text{model} \end{array}$$

Production	First set	Predict set
1	RES	RES
2	CAP	CAP
3	IND	IND
4	VOL	VOL
5	CUR	CUR
6	DIO	DIO
7	BJT	BJT
8	MOSFET	MOSFET
9	JFET	JFET
10	MODEL	MODEL

Every intersection between each pair of predict set is empty, therefore this production is $LL(1)$.

Resistance The resistance non-terminal has the following production:

$$\text{resistance} \rightarrow \text{RES valueID AT OPENBRACKET node SEPARATOR node CLOSEBRACKET}$$

which is equivalent to

$$\begin{array}{ll} (1) & \text{resistance} \rightarrow \text{RES valueID AT OPENBRACKET node SEPARATOR node CLOSEBRACKET} \end{array}$$

Production	First set	Predict set
1	RES	RES

There are no possible intersection between predict sets, therefore this production is automatically $LL(1)$.

Capacitance The capacitance non-terminal has the following production:

$$\text{capacitance} \rightarrow \text{CAP valueID AT OPENBRACKET node SEPARATOR node CLOSEBRACKET}$$

which is equivalent to

$$(1) \qquad \text{capacitance} \rightarrow \text{CAP valueID AT OPENBRACKET node SEPARATOR node CLOSEBRACKET}$$

Production	First set	Predict set
1	CAP	CAP

There are no possible intersection between predict sets, therefore this production is automatically $LL(1)$.

Inductance The inductance non-terminal has the following production:

$$\text{inductance} \rightarrow \text{IND valueID AT OPENBRACKET node SEPARATOR node CLOSEBRACKET}$$

which is equivalent to

$$(1) \qquad \text{inductance} \rightarrow \text{IND valueID AT OPENBRACKET node SEPARATOR node CLOSEBRACKET}$$

Production	First set	Predict set
1	IND	IND

There are no possible intersection between predict sets, therefore this production is automatically $LL(1)$.

Voltage The voltage non-terminal has the following production:

$$\text{voltage} \rightarrow \text{VOL waveID AT OPENBRACKET node SEPARATOR node CLOSEBRACKET}$$

which is equivalent to

$$(1) \qquad \text{voltage} \rightarrow \text{VOL waveID AT OPENBRACKET node SEPARATOR node CLOSEBRACKET}$$

Production	First set	Predict set
1	VOL	VOL

There are no possible intersection between predict sets, therefore this production is automatically $LL(1)$.

Current The current non-terminal has the following production:

$$\text{current} \rightarrow \text{CUR waveID AT OPENBRACKET node SEPARATOR node CLOSEBRACKET}$$

which is equivalent to

$$(1) \qquad \text{current} \rightarrow \text{CUR waveID AT OPENBRACKET node SEPARATOR node CLOSEBRACKET}$$

Production	First set	Predict set
1	CUR	CUR

There are no possible intersection between predict sets, therefore this production is automatically $LL(1)$.

Diode The diode non-terminal has the following production:

$$\text{diode} \rightarrow \text{DIO modtypeID AT OPENBRACKET node SEPARATOR node CLOSEBRACKET}$$

which is equivalent to

$$(1) \qquad \text{diode} \rightarrow \text{DIO modtypeID AT OPENBRACKET node SEPARATOR node CLOSEBRACKET}$$

Production	First set	Predict set
1	DIO	DIO

There are no possible intersection between predict sets, therefore this production is automatically $LL(1)$.

Bjt The bjt non-terminal has the following production:

$$\text{bjt} \rightarrow \text{BJT modtypeID AT OPENBRACKET node SEPARATOR node SEPARATOR node CLOSEBRACKET}$$

which is equivalent to

$$(1) \qquad \text{bjt} \rightarrow \text{BJT modtypeID AT OPENBRACKET node SEPARATOR node SEPARATOR node CLOSEBRACKET}$$

Production	First set	Predict set
1	BJT	BJT

There are no possible intersection between predict sets, therefore this production is automatically $LL(1)$.

Mosfet The mosfet non-terminal has the following production:

$$\text{mosfet} \rightarrow \text{MOS modtypeID AT OPENBRACKET node SEPARATOR node SEPARATOR node CLOSEBRACKET}$$

which is equivalent to

$$(1) \qquad \text{mosfet} \rightarrow \text{MOS modtypeID AT OPENBRACKET node SEPARATOR node SEPARATOR node CLOSEBRACKET}$$

Production	First set	Predict set
1	MOS	MOS

There are no possible intersection between predict sets, therefore this production is automatically $LL(1)$.

Jfet The jfet non-terminal has the following production:

$$\text{jfet} \rightarrow \text{JFET modtypeID AT OPENBRACKET node SEPARATOR node SEPARATOR node CLOSEBRACKET}$$

which is equivalent to

$$(1) \qquad \text{jfet} \rightarrow \text{JFET modtypeID AT OPENBRACKET node SEPARATOR node SEPARATOR node CLOSEBRACKET}$$

Production	First set	Predict set
1	JFET	JFET

There are no possible intersection between predict sets, therefore this production is automatically $LL(1)$.

Model The model non-terminal has the following production:

$$\text{model} \rightarrow \text{MODEL modtypeID AT OPENBRACKET listnodes CLOSEBRACKET}$$

which is equivalent to

$$(1) \qquad \text{model} \rightarrow \text{MODEL modtypeID AT OPENBRACKET listnodes CLOSEBRACKET}$$

Production	First set	Predict set
1	MODEL	MODEL

There are no possible intersection between predict sets, therefore this production is automatically $LL(1)$.

Node The node non-terminal has the following production:

$$\text{node} \rightarrow \text{GND} \mid \text{ID}$$

which is equivalent to

$$(1) \qquad \text{node} \rightarrow \text{GND}$$
$$(2) \qquad \text{node} \rightarrow \text{ID}$$

Production	First set	Predict set
1	GND	GND
2	ID	ID

Every intersection between each pair of predict set is empty, therefore this production is $LL(1)$.

ValueID The valueID non-terminal has the following production:

$$\text{valueID} \rightarrow \text{newvalue} \mid \text{ID}$$

which is equivalent to

$$\begin{array}{ll} (1) & \text{valueID} \rightarrow \text{newvalue} \\ (2) & \text{valueID} \rightarrow \text{ID} \end{array}$$

Production	First set	Predict set
1	VALUE, PI	VALUE, PI
2	ID	ID

Every intersection between each pair of predict set is empty, therefore this production is $LL\,(1)$.

WaveID The waveID non-terminal has the following production:

$$\text{waveID} \rightarrow \text{newwave} \mid \text{ID}$$

which is equivalent to

$$\begin{array}{ll} (1) & \text{waveID} \rightarrow \text{newwave} \\ (2) & \text{waveID} \rightarrow \text{ID} \end{array}$$

Production	First set	Predict set
1	DC, AC	DC, AC
2	ID	ID

Every intersection between each pair of predict set is empty, therefore this production is $LL\,(1)$.

ModtypeID The modtypeID non-terminal has the following production:

$$\text{modtypeID} \rightarrow \text{newmod} \mid \text{ID}$$

which is equivalent to

$$\begin{array}{ll} (1) & \text{modtypeID} \rightarrow \text{newmod} \\ (2) & \text{modtypeID} \rightarrow \text{ID} \end{array}$$

Production	First set	Predict set
1	MODTYPE	MODTYPE
2	ID	ID

Every intersection between each pair of predict set is empty, therefore this production is $LL\,(1)$.

Listnodes The listnodes non-terminal has the following production:

$$\text{listnodes} \rightarrow \text{node listnodes2}$$

which is equivalent to

$$\begin{array}{ll} (1) & \text{listnodes} \rightarrow \text{node listnodes2} \end{array}$$

Production	First set	Predict set
1	GND, ID	GND, ID

There are no possible intersection between predict sets, therefore this production is automatically $LL\,(1)$.

Listnodes2 The listnodes2 non-terminal has the following production:

$$\text{listnodes2} \rightarrow (\ \text{SEPARATOR node listnodes2}\ )?$$

which is equivalent to

$$\begin{array}{ll} (1) & \text{listnodes2} \rightarrow \text{SEPARATOR node listnodes2} \\ (2) & \text{listnodes2} \rightarrow \varepsilon \end{array}$$

Production	First set	Predict set
1	SEPARATOR	SEPARATOR
2	∅	CLOSEBRACKET

Every intersection between each pair of predict set is empty, therefore this production is $LL\,(1)$.

Sim The listnodes non-terminal has the following production:

$$\text{sim} \rightarrow \text{SIMULATE dirsim}$$

which is equivalent to

$$\begin{array}{ll} (1) & \text{sim} \rightarrow \text{SIMULATE dirsim} \end{array}$$

Production	First set	Predict set
1	SIMULATE	SIMULATE

There are no possible intersection between predict sets, therefore this production is automatically $LL\,(1)$.

Dirsim The dirsim non-terminal has the following production:

$$\text{dirsim} \rightarrow (\ \text{newdir dirsim}\ )?$$

which is equivalent to

$$\begin{array}{ll} (1) & \text{dirsim} \rightarrow \text{newdir dirsim} \\ (2) & \text{dirsim} \rightarrow \varepsilon \end{array}$$

Production	First set	Predict set
1	DIRECTIVE	DIRECTIVE
2	∅	LIBRARY, CONST, COMPONENT, SIMULATE, $\swarrow$

Every intersection between each pair of predict set is empty, therefore this production is $LL\,(1)$.

Newdir The newdir non-terminal has the following production:

$$\text{newdir} \rightarrow \text{DIRECTIVE}$$

which is equivalent to

$$\begin{array}{ll} (1) & \text{newdir} \rightarrow \text{DIRECTIVE} \end{array}$$

Production	First set	Predict set
1	DIRECTIVE	DIRECTIVE

There are no possible intersection between predict sets, therefore this production is automatically $LL\,(1)$.

2.1.2 Conclusion

Given that each production of the WaSabi grammar is $LL\,(1)$, the grammar itself is $LL\,(1)$.