

The formal-grammar package*

Martin Vassor
bromind+ctan@gresille.org

November 15, 2021

Contents

1	Introduction	1
2	Usage	1
2.1	Basic usage	1
2.2	Advanced capabilities	2
3	Implementation	4

1 Introduction

This package provides a new environment (**grammar**) and associated commands to typeset BNF grammars. It allows to easily write formal grammars. For instance, the syntax of the λ calculus is given in Grammar 1.

\mathcal{T}	$::=$	\mathcal{V}	<i>Variable</i>
		$(\mathcal{T} \mathcal{T})$	<i>Application</i>
		$\lambda \mathcal{V} \cdot \mathcal{T}$	<i>Abstraction</i>
\mathcal{V}	$::=$	x, y, \dots	<i>Variables</i>

Grammar 1: λ calculus syntax

2 Usage

2.1 Basic usage

Creating a grammar. We first start creating a grammar using the **grammar** environment.

grammar This is the main environment to write your grammar. **grammar** accepts 3

*This document corresponds to **formal-grammar** v1.1, dated 2021/11/15.

$$\begin{array}{lll} \mathcal{A} & ::= & () \quad \textit{Parenthesis} \\ & | & \{\} \quad \textit{Curly brackets} \end{array}$$

Grammar 2: A simple grammar

optional arguments: the first one is a possible caption; the second is a positioning option; and the third is a label.

If none of the optional arguments are provided, the grammar is inlined (i.e. not in a float environment). If the first argument is set (the optional caption), the grammar is typeset in a float, captionned with the provided caption. The second optional argument is a positioning option (one of `t`, `b`, `p`, `h`, etc.). The default is `p`. The last argument is a label, used to reference the grammar elsewhere in the document.

The grammar can then be populated using two basic constructs: `firstcase` and `otherform`.

`\firstcase` The `firstcase` command creates a new non-terminal of the grammar. It takes 3 mandatory arguments: the letter(s) of the non-terminal, the definition, and an explanation. On the other hand, `otherform` create an alternative for the preceding non-terminal, on a new line. It takes two arguments: the definition of the alternative, and an explanation. For instance, the following grammar typesets as the grammar in Grammar 2.

```
\begin{grammar}[A simple grammar][t][gr:simple_grammar]
\firstcase{A}{()}{Parenthesis}
\otherform{\{\}}{Curly brackets}
\end{grammar}
```

`\nonterm` **Referencing non-terminals.** This allows you to typeset a symbol as a non-terminal. In the current version, the default typesetting is to wrap in a `\mathcal` command. This allow to reference those non-terminals, both in grammar rules and elsewhere in the document. Notice that, since the typesetting is just a wrapper over `\mathcal`, it should be used in a math environment. For instance, the only non-terminal of Grammar 2 is \mathcal{A} (`\(\nonterm{A}\)`).

2.2 Advanced capabilities

In this subsection, we will explain the more advanced capabilities of the package. These would allow to typeset more complex grammars such as the one displayed in Grammar 3.

`\gralt` **Variants on the same line.** When variants are short and simple, it is possible to display multiple of them on the same line using `\gralt`. For instance, the first line of Grammar 3 is typeset with the following command:

```
\firstcase{B}{(\nonterm{B})\gralt \{\nonterm{B}\}}{Nested parenthesis or brackets}
```

\mathcal{B}	::=	$(\mathcal{B}) \mid \{\mathcal{B}\}$	<i>Nested parenthesis or brackets</i>
\mathcal{C}_1	::=	\mathcal{B}	<i>Example of subtil non-terminal</i>
\mathcal{D}	::=	\mathcal{B}	<i>An interesting line</i>
		\mathcal{D}	<i>An uninteresting line</i>
		$\mathcal{D} + \mathcal{D} \mid \mathcal{A}$	<i>Important item</i>

Grammar 3: A more advanced grammar

Subtle typesetting of non-terminals. Since nonterminals are, by default, typeset using `\mathcal`, it can lead to the usual issues of `\mathcal` (typically, for lowercases). Therefore, we provide *subtle* variants of `\firstcase` and `\nonterm`, in which the non-terminal symbol is not typeset (i.e. as the user, you have to typeset it manually).

`\nontermsubtil` This is equivalent to `\nonterm`, but where typesetting is left to the user. In the current implementation does nothing. However users are encouraged to use it for future modifications of the package. For instance, it is possible to typeset a non-terminal with a number index as follow \mathcal{C}_1 with the following command:
`\(\nontermsubtil{\nonterm{C}_1}\)`

`\firstcasesubtil` The subtle variant of `\firstcase`. It works similarly, except that the non-terminal (i.e. the first argument) is not embedded in a `\mathcal` macro. For instance, the \mathcal{C}_1 in Grammar 3 is typeset with the following command:

`\firstcasesubtil{\(\nonterm{C}_1\)}{\nonterm{B}}{Example of subtil non-terminal}`

Highlighting and downplaying variants. Three commands are provided to highlight or downplay some parts of a grammar. `\highlight` highlights a whole line, `\loghighlight` highlights a part of a line, while `\downplay` downplays a line.

`\downplay` The two commands `\highlight` and `\downplay` work similarly: when used before a `\firstcase`, `\firstcasesubtil`, or `\otherform`, the next line is highlighted in blue, or printed in light grey. For instance, in Grammar 3, the rule for non-terminal \mathcal{D} is typeset with:

`\highlight`
`\firstcase{D}{\nonterm{B}}{An interesting line}`
`\downplay`
`\otherform{\nonterm{D}}{An uninteresting line}`

`\lochighlight` For more local highlighting, it is possible to use `\lochighlight`, which prints some part of a rule in red. The last line of Grammar 3, which contains such local highlight, is typeset with the following command:

`\otherform{\lochighlight{\nonterm{D} + \nonterm{D}}`
`\gralt \nonterm{A}}{Important item}`

Customizing the $::=$ symbol. At the end of the preamble (i.e. before the `\begin{document}`), the package checks if a command `\Coloneqq` is defined. If that is the case, it is used instead of $::=$. Typically, packages `mathtools`, `txfonts` and `pxfonts` define this command, but you can also define it manually if you use the symbol elsewhere in the document.

3 Implementation

`floatgrammar` This is a new float that contains floating grammars. This is needed so that they are labeled with 'Grammar'.

```

1 \DeclareFloatingEnvironment[
2 name=Grammar,
3 listname={List of Grammars},
4 placement=tbhp,
5 ]{floatgrammar}

```

`\formal@rowstyle` The default `rowstyle` is empty.

```

6 \newcommand*{\formal@rowstyle}{}

```

`\rowstyle` An command used to set the style of a row. In addition, we add column types to reset the style (=) and to keep the style from one column to the other (+). As of today, it is not advised for the user to use `\rowstyle` to define their own style (i.e. I have not tested it), although I hope it will someday be possible.

```

7 \newcommand*{\rowstyle}[1]{% sets the style of the next row
8   \gdef\formal@rowstyle{#1}%
9   \formal@rowstyle\ignorespaces%
10 }
11 \newcolumntype{\formal@reset}{% resets the row style
12   >{\gdef\formal@rowstyle{}}%
13 }
14
15 \newcolumntype{\formal@add}{% adds the current row style to the next column
16   >{\formal@rowstyle}%
17 }

```

`grammar` This is the implementation of the `grammar` environment. The main difficulty is to check whether optional arguments are provided. If the first is provided, we embed the grammar into a `floatgrammar`; then if the second argument is provided, we use it as the position, (otherwise, we use `p`). Finally, if the third argument is provided, we use it as a label. Notice that, if the grammar is not a float (is inline), we do *not* break line before and after the grammar.

```

18 \ExplSyntaxOn
19 %% 1st argument: caption (makes it float)
20 %% 2nd argument: positionning option ('p' by default)
21 %% 3rd argument: label
22 \NewDocumentEnvironment{grammar} {o O{p} o}

```

```

23 {
24 \IfNoValueF{#1}{
25 \begin{floatgrammar}[#2]
26 \centering
27 }
28
29 \begin{tabular}{\formal@reset l \formal@add r \formal@add l \formal@add l}
30 }{
31 \end{tabular}
32
33 \IfNoValueF{#1}{
34 \caption{#1}
35 \IfNoValueF{#3}{
36 \label{#3}
37 }
38 \end{floatgrammar}
39 }
40 }
41 \ExplSyntaxOff

```

\firstcase The `\firstcase` is typeset as a new line in the array, which first cell is the symbol of the non-terminal, the second cell is just $::=$, the third cell is the rule (it is directly printed, without any modification), and the last cell is the description of the rule, in greyish color.

```

42 \newcommand{\firstcase}[3]{\(\mathcal{#1}\) & \(\formal@Coloneqq\) & \(\#2\) & {\itshape \color{

```

\firstcasesubtil The `\firstcasesubtil` is implemented similarly to `\firstcase`, except that the first argument is not surrounded by `\mathcal`.

```

43 \newcommand{\firstcasesubtil}[3]{#1 & \(\formal@Coloneqq\) & \(\#2\) & {\itshape \color{gray!90!

```

\otherform Adds a line with an empty first cell, and which second cell is just a pipe. The third and fourth cells are similar to `\firstcase`.

```

44 \newcommand{\otherform}[2]{& \(\|\) & \(\#1\) & {\itshape \color{gray!90!black} \#2}\}

```

\nonterm Typesets in `\mathcal`.

```

45 \newcommand{\nonterm}[1]{\mathcal{#1}}

```

\nontermsubtil Does nothing right now.

```

46 \newcommand{\nontermsubtil}[1]{#1}

```

\gralt `\gralt` is simply a pipe surrounded by large spaces.

```

47 \newcommand{\gralt}[0]{\quad |\quad }

```

\highlight We simply set the row color to LightCyan.

```

48 \newcommand{\highlight}[0]{\rowcolor{LightCyan}}

```

\lohighlight We simply surround the argument with red.

```

49 \newcommand{\lohighlight}[1]{\color{red} #1}

```

`\downplay` We simply apply a style that write in light grey for the row.

```
50 \newcommand{\downplay}[0]{\rowstyle{\color{white!80!black}}}
```

Finally, we check, at the end of the preamble, if there already exist a `::=` symbol. We search for a command called `Coloneqq`, e.g. defined in the `mathtools`.

```
51 \AtBeginDocument{%
52 \ifdefined\Coloneqq
53 \let\formal@Coloneqq\Coloneqq
54 \else
55 \newcommand{\formal@Coloneqq}{::=}
56 \fi
57 }
```