# Pretty Printing Context Free Grammars

Hugh Osborne
Department of Informatics,
Faculty of Mathematics and Informatics,
University of Nijmegen,
Toernooiveld,
6525 ED Nijmegen,
The Netherlands.
tel. +31.(0)80.652599
email `hugh@cs.kun.nl`

August 26, 2006

## 1 The grammar Environment

The `grammar` environment will take a context free grammar, written using the notation used in LP81[1] and pretty print the grammar. The `grammar` environment has up to five optional parameters, which specify the notation to be used to print the grammar. Note that whichever notation is chosen for the output, the `grammar` environment only recognises the LP81 notation in the input.

A short summary of the notation used in LP81 is presented here for those readers not familiar with it. It is assumed that the reader is familiar with context free grammars.

- There is a ":" between the left hand side of a rule and its right hand side
- Two alternatives within a rule are separated by a ";"
- A "," is placed between members of the same alternative
- Each rule ends with a "."

The optional parameters of the `grammar` environment are all of the form `[(`SYMBOL NAME`)`REPLACEMENT`]`, where SYMBOL NAME is one of "`colon`", "`semicolon`", "`period`", "`comma`" and "`quote`". REPLACEMENT is then the

[1] Harry R. Lewis and Christos H. Papadimitriou, *Elements of the Theory of Computation,* Prentice-Hall, 1981

text[2] that will replace the symbol specified by SYMBOL NAME. Note that a new line (\\) can be included in the replacement text, in which case the output will automatically start on a new line after each occurrence of the symbol specified. The default values are

- `[(colon){:\\}]`
- `[(semicolon){;\\}]`
- `[(period){.\\}]`
- `[(comma){,}]`
- `[(quote){‘‘}{’’}]`

Strings in the grammar environment are indicated by double-quote symbols (`"`). The colon, semicolon, period and comma have no special significance within a string, but are the same simple characters they are outside the grammar environment. A double-quote symbol can be produced by using the `\quotesymbol` command.

All of these features are demonstrated below. The text:

```
\begin{grammar}
  [(colon){::$\Rightarrow$}]
  [(semicolon){ $|$}]
  [(period){ \rule{1ex}{1ex}\\}]
  [(quote){‘}{’}]
grammar environment:\\
  "\verb!\begin{grammar}!",\\
optional parameters,\\
 context free grammar,\\
  "\verb!\end{grammar}!".
optional parameters:\\
  "[",optional parameter,"]",\\
   optional parameters;.
optional parameter:\\
  "(", specification, ")",\\
   \LaTeX\ strings.
\LaTeX\ strings:\\
   \LaTeX\ string;\\
   "\{",\LaTeX\ string,"\}",\\
   \LaTeX\ strings.
specification:\\
  "{\tt colon}";\\
```

[2] In the case of “`quote`”, the texts — the first text is the replacement for an opening quote, the second for a closing quote.

```
  "{\tt semicolon}";\\
  "{\tt period}";\\
  "{\tt comma}";\\
  "{\tt quote}".
context free grammar:\\
  rule,\\
  context free grammar;.
rule:\\
  nonterminal,":",\\
  alternatives,".".
alternatives:\\
  alternative,";",\\
   alternatives;.
alternative:
  members;.
members:\\
  member,",",members;\\
  member.
member:\LaTeX\ string,string.
string:"{\tt \quotesymbol}",
  \LaTeX\ string ,
  "{\tt \quotesymbol}".
\end{grammar}
```

produces the following output:

grammar environment::⇒
‘`\begin{grammar}`’,
optional parameters,
context free grammar,
‘`\end{grammar}`’ ■
optional parameters::⇒
‘[’, optional parameter, ‘]’,
optional parameters | ■
optional parameter::⇒
‘(’, specification, ‘)’,
LaTeX strings ■
LaTeX strings::⇒
LaTeX string |
‘{’, LaTeX string, ‘}’,
LaTeX strings ■
specification::⇒
‘`colon`’ |
‘`semicolon`’ |
‘`period`’ |

‘comma’ |
‘quote’ ■
context free grammar::⇒
rule,
context free grammar | ■
rule::⇒
nonterminal, ‘:’,
alternatives, ‘.’ ■
alternatives::⇒
alternative, ‘;’,
alternatives | ■
alternative::⇒ members | ■
members::⇒
member, ‘,’, members |
member ■
member::⇒LaTeX string, string ■
string::⇒‘"’, LaTeX string , ‘"’ ■

## 2 Update

The `grammar` environment has been extended to allow the use of the ‘<’ and ‘>’ symbols to delimit nonterminals in the grammar (in a manner reminiscent of Backus-Naur form). These symbols can then be redefined in the same way as the ‘"’ symbol by means of an optional paramter at the start of the environment. The SYMBOL NAME part is then “`nonterminal`”. The default value is to use the symbols themselves. If the symbols have been redefined, the ‘<’ and ‘>’ symbols can be produced using `\lessthan` and `\greaterthan` respectively.

```
\begin{grammar}
      [(colon){$\rightarrow$}]
      [(semicolon)$|$]
      [(comma){}]
      [(period){\\}]
      [(quote){\begin{bf}}{\end{bf}}]
      [(nonterminal){$\langle$}{$\rangle$}]
<expression>:<number>;\\
             <number>, <relational operator>, <number>.
<number>:<digit>;<digit>,<number>.
<digit>:"0";"1";"2";"3";"4";"5";"6";"7";"8";"9".
<relational operator>:"$=$";"$\lessthan\greaterthan$";
                      "$\lessthan$";"$\greaterthan$";
                      "$\lessthan=$";"$\greaterthan=$";"in".
\end{grammar}
```

gives

⟨expression⟩→⟨number⟩|
⟨number⟩ ⟨relational operator⟩ ⟨number⟩
⟨number⟩→⟨digit⟩|⟨digit⟩⟨number⟩
⟨digit⟩→**0**|**1**|**2**|**3**|**4**|**5**|**6**|**7**|**8**|**9**
⟨relational operator⟩→=|<>| <|>| <=|>=|**in**

There is one feature of the `grammar` environment which could be considered a bug. The normal line breaking routine is still in force inside a `grammar` environment but a new line produced by LaTeX will *not* use the `grammar` environment's layout. *Only newlines given explicitly with a* `\\` *or* `\newline` *command, or implicitly by means of one the redefined symbols (‘: ’, ‘; ’ etc.) will use the* `grammar` *layout!* It is therefore recommended that you avoid this situation by providing the layout explicitly.

# 3 Second Update

Some bugs have been reported. Most of these have been corrected. See the `bnf.bugs` file accompanying this package. Two additions have been made to the grammar environment.

Firstly the (font type) command `\escapegrammar` has been provided. This declares a non-grammar "font" to be in effect, so that the "`:;.,"<>`" characters all have their normal meaning. Be careful to enclose uses of `\escapegrammar` in braces, as you would an ordinary font command, otherwise the rest of your grammar will not be pretty printed.

The second addition is to the optional parameter declarations. For the sake of consisitency with the `[(nonterminal)...]` declaration, a `[(terminal)...]` declaration has been provided. In order to maintain compatability, the old `[(quote)...]` declaration remains possible.

A short example using both of these innovations is

```
\begin{grammar}[(terminal){(}{)}]
{\escapegrammar This grammar only uses the ‘{\tt :}’, ‘{\tt "}’
and ‘{\tt .}’ ‘‘grammar symbols’’.}

S:"a".
\end{grammar}
```

produces

This grammar only uses the ‘:’, ‘"’ and ‘.’ "grammar symbols".

S:
(a).