Drawing syntactic proofs in logic with synproof

Paul Isambert
zappathustra@free.fr
http://paulisambert.free.fr
June 9, 2007

0 Introduction

This simple package, based on PSTricks, allows you to draw logical syntactic proofs designed after the Gamut books\(^1\). A pair of commands is enough, but fine tuning can be achieved easily through optional arguments of the environment.

1 Installation

Declare synproof.sty with usepackage as usual. An option may be loaded, named symbols, which defines a set of commands for logical operators without math mode:

\begin{synproof} \[(Optional specifications)\] \{⟨Length of the derivation⟩\}\end{synproof}

This environment works as usual. The optional specifications are explained in the following section. The length of the derivation is not computed automatically, so you have to specify it. A derivation of 10 lines is roughly of length 6. Note that you can print a grid that will help you measure the right length of your derivation by typing \psgrid after \begin{synproof}\{n\}.

Each new derivation, that is, each new synproof environment sets the line numbers back to 1. If, for some reason, you want a derivation to start at \(n\), just type \LineNum\{n\}. You can do that in the course of a derivation too.

\begin{itemize}
\item \step\{⟨Optional line number⟩\}⟨Expression⟩\{⟨Rule⟩\}⟨Optional label⟩
\end{itemize}

The first mandatory argument is the expression processed at the current step, e.g. \(p, p\land q, (\exists x(Fx)\rightarrow\exists x(Gx))\rightarrow\exists x(Fx\rightarrow Gx)\), and so on. The second mandatory argument takes the rule used to derive that expression, with reference to the previous line(s) to which that rule applies, like \(E\land, 13\), for instance.

The line number is optional and is automatically incremented. Thus, the first optional argument may be specified at will. If not, the line number will be handled exactly. Note that if you enter a wrong line number, the counter won’t follow you and will display the right ones if you let it take over the following lines. For instance, if you type \texttt{\textbackslash step[12](p)\{Assumption\}} at line 11 and \texttt{\textbackslash step(q)\{Assumption\}} at line 12, then both lines will have number 12. If you need automatic numbering starting where you want to, use \texttt{\textbackslash LineNum{}} as above.

Finally, the last optional argument is a label that may be used to refer to the current line in the rest of the derivation. When specifying the label, write its name with no backslash (e.g. \texttt{\textbackslash step(p\&\textbackslash And q)\{Assumption\}[\textbackslash foo\}]), and when you call it, use a backslash followed by the label’s name like a usual command (e.g. \texttt{\textbackslash step(p)\{E\&\textbackslash And, \textbackslash foo\}}). That label gives the number of the line where it was defined. If the number has been specified with the first optional argument, then the label will return that number, even if it’s not the ‘real’ one. Note that you must create a new label every time. This is an obvious shortcoming, so when you’re not completely lost in your derivation, you may write yourself the number of the line you want to refer to².

We can already draw a simple derivation. For instance:

1. \(p \land q\) Assumption
2. \(p\) E\& 1
3. \(q\) E\& 1
4. Same line number as above 4

\begin{synproof}{2.5}
\step{p\&q}{Assumption}[\textbackslash foo\} \step{p}{E\& } \step{q}{E\& \textbackslash foo}\}
\step{Same line number as above}{\textbackslash foo\} \end{synproof}

\assumption
\assumend

Those two commands delimit the beginning and the end of an assumption. \texttt{\assumption} precedes the step which is assumed, and \texttt{\assumend} precedes the conclusion. For instance, we can prove \(\vdash p \rightarrow (q \rightarrow p)\):

\begin{synproof}{3}
\assumption
\step{p}{Assumption}[\textbackslash p\}
\assumption
\step{q}{Assumption}
\step{p}{Rep, \textbackslash p}\}
\assumend
\step{q\Implies p}{I\Implies}
\assumend

\end{synproof}

Moreover, your label shouldn’t conflict with an existing command. For instance, creating a label named par is a very bad idea, since \texttt{\textbackslash bar} already exists. Writing your labels’ names in uppercase is a simple way to avoid this.

2
3 Fine tuning

This section describes how the various dimensions of a derivation may be modified. The optional specifications of \begin{synproof} take the form of a list of key=value pairs. Default value is 0 for each. Negative values are allowed, and they don’t need to be integers. Here are the parameters:

1. **NumToEx** sets the distance between the line number and the expression.
2. **ExToRule** sets the distance between the expression and the rule used to derive it.
3. **OutLine** sets the position of the outer (vertical) line relative to the line numbers.
4. **LineSpace** sets the distance between the (vertical) lines of embedded assumptions. You shouldn’t increase **LineSpace** without decreasing **OutLine** (or increasing the latter without decreasing the former). Indeed, each embedded line has a position relative to the previous one. So everything depends upon the ‘depth’ of the assumption.
5. **AssumeLine** sets the length of the horizontal line which ends an assumption (more exactly, it sets the position of its ending point).

**HorAlign** allows you to move the entire derivation to the left or to the right.

| 1. p | Assumption |
| 2. ¬p | Assumption |
| 3. ⊥ | E¬, 1, 2 |
| 4. ¬¬p | I¬, 5 |
| 5. p | ¬¬, 4 |

6. \( p \rightarrow p \) \hspace{2cm} I→

Those specifications are local: they work in the current synproof environment, not beyond. If you want them to work for all subsequent environments, use \SetDim{} with the same key=value pairs, outside of any synproof environment of course (otherwise it will have a local effect again). Note that \SetDim{} is ‘additive’: if you write \SetDim{LineSpace=2} and then later \SetDim{LineSpace=1}, all subsequent derivations will have a space of length 3 between the assumption lines. On the other hand, if you write \SetDim{LineSpace=2} and then \SetDim{NumToEx=2}, **LineSpace** will keep its value 2. This works with the local specifications set with \begin{synproof} too. Thus, \SetDim{LineSpace=2} and then \begin{synproof}\{LineSpace=1\}{n} will create a derivation with 3 units between its assumption lines, although this additional unit won’t affect subsequent derivations (since it has been added locally). Finally, you can reset all global specifications back to default with \ResetDim.

4 Example

The next two pages show the derivation of \( \vdash (\exists x(Fx) \rightarrow \exists x(Gx)) \rightarrow \exists x(Fx \rightarrow Gx) \) and its code.

\( \frac {p \rightarrow \neg p} {E\neg, 1, 2} \)

\( p \rightarrow p \) \hspace{2cm} I→
1. $\exists x(Fx) \rightarrow \exists (Gx)$ Assumption
2. $\neg \exists x(Fx \rightarrow Gx)$ Assumption
3. $\neg(Fa \land \neg Ga)$ Assumption
4. $Fa$ Assumption
5. $\neg Ga$ Assumption
6. $Fa \land \neg Ga$ I∧, 4, 5, 8
7. $\bot$ E¬, 3, 6
8. $\neg Ga$ I¬, 8
9. $Ga$ ¬¬, 8
10. $Fa \rightarrow Ga$ I¬, 13
11. $\exists x(Fx \rightarrow Gx)$ I∃, 10
12. $\bot$ E⊥, 3, 11
13. $\neg \neg(Fa \land \neg Ga)$ I¬, 13
14. $(Fa \land \neg Ga)$ ¬¬, 13
15. $\neg Ga$ E∧, 14
16. $\forall x(\neg Gx)$ I∀, 15
17. $Fa$ E∧, 14
18. $\exists x(Fx)$ I∃, 17
19. $\exists x(Gx)$ E→, 1, 18
20. $Ga$ Assumption
21. $\neg Ga$ E∀, 16
22. $\bot$ E⊥, 20, 21
23. $Ga \rightarrow \bot$ I→
24. $\bot$ E∃, 19, 23
25. $\neg \exists x(Fx \rightarrow Gx)$ I¬, 25
26. $\exists x(Fx \rightarrow Gx)$ ¬¬, 25
27. $(\exists x(Fx) \rightarrow \exists x(Gx)) \rightarrow \exists x(Fx \rightarrow Gx)$ I→
\begin{synproof}[HorAlign=1,OutLine=2,LineSpace=.5]{17}
  \assumption
  \step{\Exists x(Fx) \Implies \Exists x(Gx)} \{Assumption\}

  \assumption
  \step{\Neg \Exists x(Fx) \Implies \Exists x(Gx)} \{Assumption\}

  \assumption
  \step{\Neg \Neg (Fa \And \Neg Ga)} \{Assumption\}
  \step{Fa \And \Neg Ga} \{I \And, 4, 5\}
  \step{\Falsum} \{E \Neg, 3, 6\}
  \assumend

  \step{\Neg \Neg Ga} \{I \Neg\}
  \step{Ga} \{\Neg \Neg, 8\}
  \assumend

  \step{\Neg \Neg (Fa \And \Neg Ga)} \{I \Neg\}
  \step{(Fa \And \Neg Ga)} \{\Neg \Neg, 13\}
  \step[15]{\Neg Ga} \{E \And, 14\}
  \step[16]{\Forall x(\Neg Gx)} \{I \Forall, 15\}
  \step[17]{Fa} \{E \And, 14\}
  \step{\Exists x(Fx) \Implies \Exists x(Gx)} \{I \Exists, 10\}
  \step{\Falsum} \{E \Falsum, 3, 11\}
  \assumend

  \step[18]{\exists x(Fx) \Implies \exists x(Gx)} \{I \Exists, 17\}
  \step[19]{\exists x(Fx)} \{E \Implies, 1, 18\}

  \assumption
  \step[20]{Ga} \{Assumption\}
  \step[21]{\Neg Ga} \{E \Forall, 16\}
  \step[22]{\Falsum} \{E \Neg, 20, 21\}
  \assumend

  \step[23]{Ga \Implies \Falsum} \{I \Implies\}
  \step[24]{\Falsum} \{E \Exists, 19, 23\}
  \assumend

  \step[25]{\Neg \Neg \exists x(Fx) \Implies \exists x(Gx)} \{I \Neg\}
  \step[26]{\exists x(Fx) \Implies \exists x(Gx)} \{\Neg \Neg, 25\}
  \assumend

  \step{\exists x(Fx) \Implies \exists x(Gx)} \{I \Implies\}
\end{synproof}