

# The Scenario — in Three Versions

hhparmrk does it

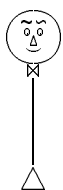
**Herman Haverkort & Frans Goddijn**

herman@fgbbs.iaf.nl & goddijn@fgbbs.iaf.nl

April 1995

## Abstract

During work towards a flexible document as a continuous report on a wide variety of contacts for the Meridian Arts Ensemble in New York, Frans Goddijn felt the need to tag and mark certain paragraphs for specific groups of readers. Herman Haverkort wrote a package for L<sup>A</sup>T<sub>E</sub>X<sub>2 $\epsilon$</sub> , hhparmrk, which facilitates this by offering the possibility to set various signs next to paragraphs. This article presents hhparmrk, gives examples of its use and a short manual. For the hackers among us some of the T<sub>E</sub>Xnical tricks involved behind the scenes are glanced at.



**Keywords:** paragraph, mark, distribution, select

During the process of organizing concerts for a delightful brass quintet from New York called the Meridian Arts Ensemble I noticed that there is at least *one* aspect about playing all over the world which causes anxiety. Namely, the fact that in many places, many people (are supposed to) look after your interests and it's very hard to keep track of who is doing what.

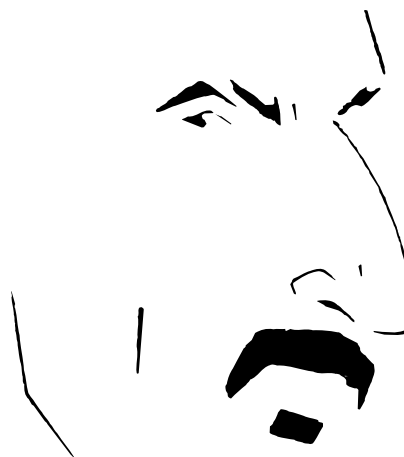
I used my knowledge of L<sup>A</sup>T<sub>E</sub>X to generate reports on all my Meridian activities. Instead of building up a heap of separate emails, notes, letters and memos I wrote or compiled them all as `\sections` and `\subsections` into the same master document, which I entitled THE SCENARIO. With *CorelDraw!* I designed a title page, L<sup>A</sup>T<sub>E</sub>X took care of the table of contents and an elaborate index, and gradually I was generating a tight mass of information which I could print out at will and send to the Meridians.

This L<sup>A</sup>T<sub>E</sub>X product impressed them, as they'd never before seen such a 'roadmap' of all that was done or not yet done in their interest. This helped them in making the decision to leave a professional booking agent and let me coordinate all further activities in this part of the world! THE SCENARIO quickly grew in size to over fifty pages filled with valuable information about how to initiate contacts with promoters and presenters in concert halls and other venues. The advantage of using L<sup>A</sup>T<sub>E</sub>X over some other typesetting or word processing tools is that it's fast and simple to copy plain ASCII emails into it. For instance, the tuba player of the group uses the internet to send me updates for the `tabular` listing their concert dates. The same goes for newspaper reviews and incoming faxes, which I run through a scanner with OCR to transfer it into electronic text. To brighten up the text page, I sometimes create a

graphic scan of a hand written quote of musical score or a concert program.

Over the weeks, the book even got its own 'gossip' section, and soon it turned into a kind of family scrapbook, besides building up the more formal data.

By this time, the new print looked so handsome that it was a shame not to use it for a wider range of readers. Other groups could benefit from the many addresses listed in THE SCENARIO, and some fans had heard of its existence and were keen to get a copy of the book with all its *inside information*.



A stylized portrait of Frank Zappa. The Meridian Arts Ensemble is especially renowned for their interpretations of Zappa's compositions, arranged by Jon Nelson.

How to go about this? I would now need three versions of the document. One full version with all info for me and the

⊙ Just a demonstration of hhparmrk's way of bracing and footnoting a paragraph.

Meridians, one slimmed down version where any explicitly confidential text would be left out and one minimal version especially geared towards concert hall programmers with only the basic material (introduction, biographical info, reviews and the like).

Luckily, I had just introduced a young man to L<sup>A</sup>T<sub>E</sub>X. I think that every T<sub>E</sub>X user has made attempts, with more or less success, to convince others of the beauty and pleasures of T<sub>E</sub>X and I am no exception to this rule. Most of the time, people have no clue what I'm talking about. Sometimes, one buys the 4allT<sub>E</sub>X CD-ROM and installs it but rarely an avid new user is born. This Herman Haverkort was different: my letter telling him about L<sup>A</sup>T<sub>E</sub>X happened to reach him on a friday when he was most bitterly sick of his 'WordPerfect' software and he immediately took action. He got the software from me, installed it, read the T<sub>E</sub>Xbook from screen ignoring all *dangerous bend*-signs and the next week he devoured the printed book itself.

Herman recognized my problem and as he had a similar project at hand, he created a new style file for us, called hhparmrk.

Now I could brand some paragraphs for *exclusive* readership, others for a *circle* of interested readers and the rest was for *wide* distribution. In the 'Wide' version, paragraphs of the other two categories should vanish automatically (actually there is a fourth version, 'Concise', which is a 'Circle' version limited to the very basic information).

First, I took macros for the *disappearing acts* from the comment package but a little later I figured out an easier way, by using a macro with a parameter and never using that parameter! Here is the disappearing act:

```
\newcommand{\LeaveOut}[1]{}

```

`\LeaveOut{Sh*! Wish I hadn't said that}` enables me to put in some lengthy paragraphs that I don't (yet) intend to really use in print, but want to have there in the source file as my own private comments, or I can have T<sub>E</sub>X ignore portions of text I might want to use later.

In the 'Circle' version, only the *exclusive* texts must disappear and some more or less confidential paragraphs must be marked accordingly. In the 'Exclusive' version, everything is visible but the reader must be able to see what parts will be occluded for others.

Now look at the following new commands. At first they are without any use, later on they get their tasks assigned. Look at them, bland and expressionless like babies, with only their names to distinguish them from other creations...

```
\newcommand{\ForWhom}{}
\newcommand{\Circle}[1]{}
\newcommand{\Exclusive}[1]{}
\newcommand{\EndConcise}{}

```

This is what they get to do in life:

`\ForWhom` will remember for whom the current version is made.

`\Circle` will be a macro with one argument at a time,

namely a paragraph that must be left out in the 'Wide' version and marked as 'Circle' in the other versions.

`\Exclusive` will also be a macro with one argument at a time, this time a paragraph that must be left out in the 'Wide' and in the 'Circle' versions and marked as 'Exclusive' in the 'Exclusive' version.

`\EndConcise` will normally mean nothing, but the command is placed at a point in the text where it must end with a new page and an index if I want to create a 'Concise' version.

Then the definitions of the different standard versions:

```
\newcommand{\ForExclusive}{
  \renewcommand{\ForWhom}{\Exclusive}
  \renewcommand{\Exclusive}[1]{%
    \MarkThisExclusive{##1}}
  \renewcommand{\Circle}[1]{%
    \MarkThisCircle{##1}}
}

```

What happened above is that `ForWhom` will now remember it's for 'Exclusive' use, and both 'Exclusive' and 'Circle' paragraphs are classified as such in the margin.

```
\newcommand{\ForCircle}{
  \renewcommand{\ForWhom}{\Circle}
  \renewcommand{\Exclusive}[1]{(\ldots)}
  \renewcommand{\Circle}[1]{%
    \MarkThisCircle{##1}}
}

```

What happened above is that `ForWhom` will now remember it's for 'Circle' use. 'Exclusive' paragraphs are ignored and '(...)' is printed in their place, while 'Circle' paragraphs are classified as such in the margin.

```
\newcommand{\Concise}{
  \renewcommand{\EndConcise}{%
    \newpage \printindex \end{document}}
  \renewcommand{\ForWhom}{concise \Circle}
  \renewcommand{\Exclusive}[1]{(\ldots)}
  \renewcommand{\Circle}[1]{%
    \MarkThisCircle{##1}}
}

```

What happened above is that `ForWhom` will now remember it's for 'concise Circle' use. 'Exclusive' paragraphs are ignored and '(...)' is printed in their place, while 'Circle' paragraphs are classified as such in the margin. Furthermore, at the place where the 'dummy' macro `\EndConcise` was loitering, it is now told to end the document neatly with a new page and an index.

```
\newcommand{\ForWide}{
  \renewcommand{\ForWhom}{\Wide}
  \renewcommand{\Exclusive}[1]{(\ldots)}
  \renewcommand{\Circle}[1]{(\ldots)}
}

```

The long dull version. Only harmless material is printed, and lots of it, even beyond the `\EndConcise` macro.

```
\newcommand{\MarkThisExclusive}[1]{%
  \begin{trafficsigned*}{\trapbox:}{\small Ex}
  #1\end{trafficsigned*}}
\newcommand{\MarkThisCircle}[1]{%
  \begin{trafficsigned*}{\ringbox:}{\small Ci}
  #1\end{trafficsigned*}}

```

The above two lines control all markings in the text, using new macros and environments which are defined in hhparmrk.

This text may be seen by everybody.

```
\Circle{This text may be seen, marked with a
        circled 'Ci', by a certain circle.}
```

```
\Exclusive{This text may only be seen, marked
            with a 'Ex' in a trapezoid box, by a
            small group of readers exclusively.}
```

I am now able to change the look and size of the document by activating one of the following commands (while commenting out the others):

```
%\ForExclusive
%\ForCircle
%\ForWide
\Concise
```

Herman Haverkort will, later on in this article, explain the new and hitherto unknown commands here. I'm glad he does. They fill me with wonder. I began creating the macros only after looking hard and intensely into the manual material... and immediately after I'd finished writing them, I stacked them away in a separate style file. That way I don't have to see them so often and I can try to forget they're there at all!

Now I will first show you one of the first pages, where I explain to the readers what versions there are and what version they're holding. Next, I will try my luck at displaying the result for you...

```
\ForExclusive
```

```
\noindent{This \textsc{scenario} comes in
three different prints, stemming from the same
source file: an abridged 'Wide' version for
presenters and other people who are involved
with the Meridian Arts Ensemble, a more
complete 'Circle' version for some people who
work together with the Ensemble and a full
'Exclusive' version with some private details
that are only useful for communication
between the members of the Ensemble and the
Meridian Foundation.
```

```
% reserve sign for HH's address, using macros
% defined in hhmuf.sty:
\mufhire emailhh:{herman@fgbbs.iaf.nl}
```

```
\MarkThisExclusive{This is an example of the
'roadsign' used in the margin of text parts
which are only visible in the 'Exclusive'
version. It was designed for us by Herman
Haverkort\muf emailhh:{}, a grand \textsc{mae}
fan and \TeX\ wizard. Texts marked in this
manner are represented as '(\ldots)' in the
'Circle' and 'Wide' printings.}
```

```
\MarkThisCircle{This is an example of the
'roadsign' used in the margin of text parts
which are only visible in the 'Circle' and
'Exclusive' version. It was designed for us
by Herman Haverkort\muf emailhh:{}, a grand
\textsc{mae} fan and \TeX\ wizard. Texts
marked in this manner are represented as
'(\ldots)' in the 'Wide' printing.}
```

Furthermore, a concise version of this scenario consists of only the first sections,

---

∇ herman@fgbbs.iaf.nl


as an introduction to presenters.


```
\textbf{NOTE: this is a print of the \ForWhom\
version.}}
```

which results in:

---

This SCENARIO comes in three different prints, stemming from the same source file: an abridged 'Wide' version for presenters and other people who are involved with the Meridian Arts Ensemble, a more complete 'Circle' version for some people who work together with the Ensemble and a full 'Exclusive' version with some private details that are only useful for communication between the members of the Ensemble and the Meridian Foundation.

 This is an example of the 'roadsign' used in the margin of text parts which are only visible in the 'Exclusive' version. It was designed for us by Herman Haverkort,<sup>∇</sup> a grand MAE fan and T<sub>E</sub>X wizard. Texts marked in this manner are represented as '(...)' in the 'Circle' and 'Wide' printings.

 This is an example of the 'roadsign' used in the margin of text parts which are only visible in the 'Circle' and 'Exclusive' version. It was designed for us by Herman Haverkort,<sup>∇</sup> a grand MAE fan and T<sub>E</sub>X wizard. Texts marked in this manner are represented as '(...)' in the 'Wide' printing.

Furthermore, a concise version of this scenario consists of only the first sections, as an introduction to presenters.

**NOTE: this is a print of the 'Exclusive' version.**

---



The `hhparmrk` kernel consists of the environments `bracespanned` and `markspanned`, which I will present now, starting with `bracespanned`.

**bracespanned**

The environment bracespanned can be used to set paragraphs braced like this one. This paragraph is done with:

```
\begin{bracespanned}%
({\{\}}:-{FG}()\{\}):-{HH})
  The environment
```

... concluded by:

```
demonstrated here.
\end{bracespanned}
```

The nasty details which determine the way of bracing are all specified just after `\begin{bracespanned}`; the concluding `\end{bracespanned}` is always as straightforward as demonstrated here.

You might suspect that the left brace and comment ('FG' in the above example) are specified between left parentheses, while the right brace and comment are specified between right parentheses. Well, that is right. You do not have to specify both left and right stuff: you may leave one of them out, as in some of the examples below. The following paragraphs will all start with a box containing its bracing specification, that is: all that appears between `\begin{bracespanned}` and the text of the paragraph.

`(){\}\}:-{\muf:{Just an example}})` Instead of the comments 'FG' en 'HH' in the above example, you can of course specify whatever you want for a comment, for example a footnote. This paragraph provides an example using the `\muf` footnote macro, which is defined in the `hhmuf` package! If you want to use standard footnotes, note that all that is spanned by `bracespanned` and the comments are so-called forbidden environments. To set a footnote you would have to use `\footnotemark` and `\footnotetext`; just using `\footnote` would not work.

`(){\{\}}:-{\})` This paragraph illustrates that any extendable mathematical delimiter symbols can be used instead of braces, even symbols which are pointing the 'wrong' way. Just replace the `\{\}` or `\}\}` in the example above by `{(}`, as in this example, or whatever symbol you like.

`(){\}\}:-{65pt}{This may...})`  
 The `:-` in the examples above specifies the width of the spanning symbol plus comment. `:-` stands for the natural width of the symbol with comment, which usually satisfies. Another possible width specification is a colon followed by a braced dimension, like `{65pt}`. Such a specification fixes the width of the symbol plus comment, thus enabling multi-line comments, like demonstrated here.

`(){\}] {ExampleId} : {ex. i})` Sometimes it may be desirable to have the comments of several spanned paragraphs set all to the same width, thus leaving equal line widths for the spanned paragraphs. This can be accomplished by giving a width specification which consists of some braced identifier followed by a colon. The identifier may be chosen freely.

`(){\}] {ExampleId} : {ex. ii})` The previous paragraph and this one get the same width identifier (`ExampleId`) so that their comments are set to the same width: the natural width of the widest. As a result, the text bodies of both paragraphs are equally wide. However, in general you have to compile your document twice to get this result. If a second run may be necessary, the `hhunits` package issues a warning 'Unit values may have changed. Rerun to get them right.'

`(({\}\}:-[50pt]{Gosh!})` In the examples above paragraphs were indented on the sides to make room for the spanning symbols and comments. The amount of indentation was automatically determined by the `hhparmrk` macros. This automatic determination can be overruled by specifying the amount of indentation in a bracketed optional argument, given between the width specification and the comment. This paragraph provides an example: it is indented exactly 50pt. Specifying a 0pt indentation would cause the spanning symbol and the comment to be set in the margin.

`([:-( )]:-)` T<sub>E</sub>X hackers who know when braces can be omitted are able to specify the way of spanning a paragraph quite elegantly — I think — as demonstrated by this paragraph.

⊕ Just an example

⌋ See the article about HH getting carried away, also published in this MAPS

**markspanned**

START The environment `markspanned` can be used to set three-part marks next to paragraphs. Such a mark consists of an upper part, a lower part, and a fill part in between. The upper and lower part have fixed size, but the fill part can be stretched so that the assembled mark spans the entire paragraph. This paragraph provides a simple example.

FINISH

The above paragraph is typeset with:

```
\begin{markspanned}%
({\sc start}[\msprule]{\sc finish}{10pt}{
  The environment \envirname{markspanned} can
  :           :           :           :
  This paragraph provides a simple example.
}\end{markspanned}
```

In the above example an upper part, a fill, a lower part and the mark separation are successively specified. The fill is the `hhparmrk` macro `\msprule`, which connects the upper and the lower part by a rule. The 10pt mark separation determines the smallest distance between the text and the three-part mark.

The nasty details which determine the way of marking are all specified just after `\begin{markspanned}`; the concluding `\end{markspanned}` is always as straightforward as demonstrated above. The following paragraphs will all start with a box containing its marking specification, that is: all that appears between `\begin{markspanned}` and the text of the paragraph.

ST ({\sc st}[\msprule]{\sc fi}[r]{10pt}) In the example above the mark parts are centered with respect to each other. Instead of centering one can force left or right alignment by means of `[l]` or `[r]` just after the definition of the lower part. This paragraph gives an example of right alignment.

FI

Until now marked paragraphs were automatically indented just enough to make room for the marks so that they did not stick out into the margins. Like with `bracespanned` one can control the amount of indentation ‘manually’ by specifying an optional argument, just after the mark separation.

⊂ ({\sc cap}{\sc cup}{5pt}[20pt]) This paragraph provides an example. It is indented exactly 20pt. This paragraph also shows that the fill part of a mark is optional and may be left out.

/ ({\\$/\}{\bs}{5pt}()\bs}{\\$/\}{5pt}) \ Of course three-part marks could be set on the right by using right parentheses instead of left ones, just like with `bracespanned`. Three-part marks on both sides are possible too, like demonstrated here.

**More about the Fill Part**

As shown in the above examples, the second argument of a three-part mark specification determines the fill part of the mark. You may omit this specification: in that case an

empty fill is used. Besides `\msprule` and the empty fill one could use any desired self-made fill as long as the following is regarded:

- the fill should be a macro that takes one argument: the required size. For example `\msprule` is defined by `\newcommand\msprule[1]{\vrule height #1 width \fboxrule}`.
- the width of the fill is not taken in account when determining the positioning of the mark. Therefore the width of the fill should not be greater than both the width of the upper part and the width of the lower part of the three-part mark.

**Traffic Signs**

`hhparmrk` contains the following definition (shown here in syntactically simplified version):

```
\newenvironment{trafficsigned*}[2]{%
  \begin{markspanned}(%
    {#1\separbox{2pt}{\large\bf #2}}%
    [\msprule]%
    {\sepbox{0pt,1pt,0pt,0pt}{%
      \large\ensuremath{\bigtriangleup}}}%
    {1em})%
  }%
  \ifhmode\strut\fi
  \end{markspanned}%
  \scopecorrection
}
```

**A** \begin{trafficsigned\*}{\trapbox:}{A}  
The environment `trafficsigned*` produces a three-part mark on the left which forces the signed text to indent. Its upper part is the second argument, boxed by the `hhflxbox` macro `\separbox`<sup>1</sup> and the tokens specified by the first argument. These are typically framing macros like `\trapbox:` (defined in `hhparmrk`; sets a trapezium frame), `\ringbox:` (defined in `hhflxbox`; sets a circle frame), or `\setlength\fbboxsep{0pt}\fbox` (sets a rectangular frame). These paragraphs show some possible results. The  $\TeX$  code used to start each paragraph is shown in the boxes at the beginnings. Each paragraph is ended in the source file by `\end{trafficsigned*}`.

**B** \begin{trafficsigned\*}{\ringbox:}{B}  
Before the `\end{markspanned}` in the definition of `trafficsigned*` a conditional `\strut` is added. This is to prevent the foot of the sign from ostensibly floating according to the depth of the last line of a signed paragraph. The `\scopecorrection` is not really needed in most cases but it guards against some rare mysterious errors. See the section about  *$\TeX$ nical Details*, subsection *hhparmrk: Parallel Marks?* for explanation.

<sup>1</sup> See the article about HH getting carried away, also published in this MAPS

**C** `... *}{\setlength\fbboxsep{0pt}\fbbox}{C}` that I could put the displays and the `\vtoped` text together. Besides `trafficsigned*` there exists a similar environment `trafficsigned` which sets the traffic sign in the left margin. To avoid letter-traffic collisions it is not demonstrated here.

## T<sub>E</sub>Xnical Details

I will not present the definition of `bracespanned` here: it is too long and complicated. There is lots of fuss in it, caused by the need or wish to parse a lot of obligatory and optional arguments which determine the exact way of bracing. However, I would like to lift a corner of the veil which covers `bracespanned`, to give hackers some idea of what is going on behind the scenes. Maybe, if I am lucky, there is some hacker out there who will be highly amazed by unnecessary complexity in my approach, and will offer me a simpler approach instead.

## Fooling T<sub>E</sub>X's Gluing

My first try to build a useful `bracespanned` environment or macro consisted of straightforward use of a mathematical display. It is not difficult to set a brace spanning a box with multiple lines of text in a mathematical display environment. Alas that did not work out properly in all cases. When `bracespanned` text was surrounded immediately by normal text, the interline skip between the top spanned line and the first unspanned line above was too small, as was the distance between the bottom spanned line and the first unspanned line below. T<sub>E</sub>X considered the whole mathematical display to be one unusually high line of text. Therefore T<sub>E</sub>X did its very best to squeeze the display in at the place of one normal text line, although the display actually contained several lines.

So I decided that I had to fool T<sub>E</sub>X a bit. I constructed a mathematical display as before, but now I boxed it. Then I typeset the spanned text again behind the scenes, now using `\vtop`, to determine the height of the first line. I then shifted down the boxed display to make it have that same height. Finally I typeset the spanned text a third time behind the scenes, now using `\vbox`, to determine the depth of the last line of spanned text. Then I would insert the display, which had the height of its first line of text, and fool T<sub>E</sub>X by setting `\prevdepth` to the depth of the last line. T<sub>E</sub>X still considered the display to be a single high line, but I made T<sub>E</sub>X 'think', with respect to setting interline glue, that the display had the height of its top line and the depth of its bottom line, as if all lines in between were not there.

The approach described above had a major disadvantage: the spanned text was typeset three times. This was not only inefficient; it was also error-prone. For example: if a counter was stepped in the spanned text, then it was stepped three times. I solved this by boxing the spanned text once, using `\vtop`. Then I made a centered copy of the resulting box, a `\vphantom` of which I used to set the braces in mathematical displays. I shifted the displays down to make them have the same height as the `\vtoped` text, so

Finally I made a copy of the `\vtoped` text, which I unboxed to get its last line with `\lastbox` so that I could examine its depth. Then the remaining part of the procedure was like described in the previous paragraph.

All this resulted in the T<sub>E</sub>X code below (shown here abridged and simplified):

```
%Box what has to be spanned in \@tempboxa:
\setbox\@tempboxa\vtop{#1}%
%Make a centered copy of the result:
\sbox\@tempboxd{\ensuremath{\vcenter{
\copy\@tempboxa}}}%
%Determine how much the displays should be
%shifted down; store result in \@tempdima:
\setlength\@tempdima{\ht\@tempboxd}%
\addtolength\@tempdima{-\ht\@tempboxa}%
%Set the left brace in \@tempboxb:
\sbox\@tempboxb{%
\lower\@tempdima\hbox{%
%...
%in hhparrmk.sty one finds at this place
%the math display stuff which sets the left
%brace, using \vphantom{\copy\@tempboxd} to
%determine the height
%...}}%
%Set the right brace in \@tempboxc:
\sbox\@tempboxc{%
%...
%same story
%...}%
%Now determine the depth of the last line:
%Make a discardable copy of \@tempboxa in
%\@discabox:
\setbox\@discabox\copy\@tempboxa
\setbox\@discabox\vbox{%
%Get the last line:
\unvbox\@discabox
\setbox\@discabox\lastbox
%Save its depth in \h@virtualdepth:
\global\h@virtualdepth\dp\@discabox}%
%Finally put it all together:
\hbox{\llap{\box\@tempboxb}%
\box\@tempboxa\rlap{\box\@tempboxc}}%
%Fool TeX's gluing:
\prevdepth\h@virtualdepth
```

## Banishing Stubborn White Space

Some environments like to surround themselves by vertical white space. Section headings have the same tendency. But when complete (sub)sections are spanned by `bracespanned`, we do not want to get results like this:

**Heading**  
lots of blah...

It looks ugly. Vertical space added in the beginning of a spanned passage should be squeezed out: it should be set on top of the span, instead of *in* the span. This is implemented as follows. In the beginning of a passage being boxed `\prevdepth` is set to  $-4774\text{pt}$  (just some value smaller than  $-1000\text{pt}$ ). `\addvspace` is redefined to set vertical space only when `\prevdepth` is greater than  $-4774\text{pt}$ , that is: when we are not in the beginning of the

spanned passage anymore. If `\prevdepth` still equals `-4774pt` then the vertical space is added to a box register which holds the squeezed out space. After the complete passage has been boxed, first the squeezed out space register is unboxed and added to the main vertical list, and then the boxed passage is spanned and set. This results in the following  $\TeX$  code for setting brace spans:

```
%Box what has to be spanned in \@tempboxa:
\setbox\@tempboxa\vtop{\topsqueezeout #1}%
%Make a centered copy etc. (see the previous
%listing)
%...
%Finally put it all together:
\topsqueezein
\hbox{\llap{\box\@tempboxb}%
\box\@tempboxa\rlap{\box\@tempboxc}}%
%Pool TeX's gluings:
\prevdepth\h@virtualdepth
where \topsqueezeout takes care of redefining
\addvspace, and \topsqueezein adds the accumu-
lated squeezed out space:
\newbox\h@tsqo@squeeze
```

```
\def\topsqueezeout{%
% Save original \addvspace:
\let\h@tsqo@addvspace=\addvspace
% Redefine \addvspace:
\def\addvspace##1{%
\ifdim\prevdepth>-4774pt\relax
% If we are not in the beginning of the
% box anymore, call original \addvspace:
\h@tsqo@addvspace{##1}%
\else
% else add space to box register which
% holds the squeezed out space:
\global\setbox\h@tsqo@squeeze\vbox{%
\unvbox\h@tsqo@squeeze
\h@tsqo@addvspace{##1}}
\fi}%
% Initialize box holding squeezed out space:
\global\setbox\h@tsqo@squeeze\vbox{}%
% Set \prevdepth to -4774pt to indicate the
% beginning of the box:
\setlength\prevdepth{-4774pt}}
```

```
\def\topsqueezein{\unvbox\h@tsqo@squeeze}
```

The real implementation in `hhparmrk` is more complex: it also redefines `\addpenalty`, and it contains more fuss to account for nested bracespanned environments.

At the bottom of the spanned passage vertical space should be squeezed out as well. This is also done using a box register to hold the squeezed out space. After the passage being boxed has been entirely added to the box in which it is set, the space at the end is examined with `\lastskip`. The space is removed with `\unskip` and added to the box holding squeezed out space. Because there may be multiple skips at the end of the passage this procedure is repeated until `\lastskip` returned zero three times. I could not find a way to distinguish zero skips and no skips: `\lastskip` returns zero in both cases. Since three consecutive zero skips seem to be unlikely, the algorithm terminates when `\lastskip` yielded zero three times consecutively.

## hhparmrk: Parallel Marks?

`hhparmrk` actually stands for: *Herman Haverkort's parallel marks*. The 'philosophy' behind this is that `hhparmrk`'s marks should not interfere with the hierarchical structure of the document. Ideally marked and unmarked passages are typeset and processed just like they normally are, except for the presence of the marks.

In practice this is not fully attainable. First it is probably inevitable to set each marked passage as a separate paragraph, and that is what is done indeed.

A second problem is that marked passages are set in internal vertical mode, which causes footnotes and marginal notes to disappear. For `hhmuf`'s style<sup>1</sup> footnotes this problem has been solved. For standard footnotes this problem can be solved, but I did not bother to do it yet.

A third problem is the grouping invoked by using environments and boxing commands. This grouping causes the scope of local assignments in marked passages to be reduced. Because that is exactly what is expected when  $\LaTeX$ 's environments are used I decided not to do much about it, to avoid confusion. However, I built in a small 'scope correction' which suppresses the scope reduction of assignments to `\everypar`, `\par`, `\@par` and `\@currentlabel`. The first three should not be really necessary, but the handling of `\@currentlabel` can be useful when section headings or the like are spanned for some reason. The scope correction can be activated by the macro `\scopecorrection`, which is defined as follows:

```
\def\spherecorrection{%
\h@savelocals
\h@restorelocals}

\def\h@savelocals{%
\global\h@sc@everypar=\everypar
\global\let\h@sc@par=\par
\global\let\h@sc@@par=\@par
\global\let\h@sc@@currentlabel=
\@currentlabel}

\def\h@restorelocals{%
\aftergroup\h@@restorelocals}
\def\h@@restorelocals{%
\everypar=\h@sc@everypar
\let\par=\h@sc@par
\let\@par=\h@sc@@par
\let\@currentlabel=\h@sc@@currentlabel}
```

## Where to Get what Files?

To be able to use `hhparmrk`, you should also have the packages `hhflxbox`, `hhunits`, `hhqueue` and `hhutils0` available. These packages are automatically loaded by `hhparmrk`. All files needed can be obtained from FGBBS<sup>∞</sup> by requesting the file `hh.arj`. I will try to submit the packages to CTAN as well. Note that  $\LaTeX$  2.09 versions are not available.

<sup>1</sup> See the article about HH getting carried away, also published in this MAPS

<sup>∞</sup> FGBBS — tel. (085) 21 70 41