BIJLAGE S

Unusual paragraph shapes

Victor Eijkhout

Introduction

Although the TeX book [1] states that TeX's paragraph mechanism 'can be harnessed to a surprising variety of tasks', the strangest paragraph shapes that I have implement use no feature of the line-breaking algorithm. Instead, I have found that the control sequences `\everypar` and `\lastbox` are extremely powerful tools. I give three examples of this.

How many TeXers does it take to typeset a lightbulb? [2]

A while back I found somewhere in Netland a largish collection of jokes – several hundreds of them – consisting of a question of the above form, and an answer to it. It seemed to me that this document would make a nice gift, if properly typeset and bound. The following layout was suggested to me [3]:

How many evolutionists does it take to screw in a light bulb?

Only one, but it takes eight million years.

How many folk singers does it take to screw in a light bulb?

Two. One to change the bulb, and one to write a song about how good the old light bulb was.

As all jokes[1] had a question mark concluding the first line, and the file had blank lines seperating the jokes, the following macro should do the job when placed in front of each joke:

```
\def\lbj#1? #2\par{{\it
    \hangfrom{\vrule width 1in height 1ex
              depth 0in \kern .5em}#1\par}
    #2\par}
```

Only I didn't feel like putting some command in front of a couple of hundreds of jokes. Then I remembered about `\everypar`, the token list that is inserted in front of every paragraph. If I would set

```
\everypar={\lbj}
```

the jokes would automatically be scooped up as the arguments of `\lbj`.

Well, *almost.* The snag is that `\lbj` itself produces two paragraphs, in front of each of which `\everypar` is inserted. With infinite recursion as a result. Repair may take the following form:

```
\everypar={\bgroup\everypar={}\lbj}
\def\lbj#1? #2\par{ ...  %% the same
    ...                  %% as above
    #2\par\egroup\bigskip}
```

Now `\everypar` opens a group inside which it is zero, and `\lbj` inserts the closing brace of the group, so after the joke `\everypar` has its old value again.

[1] I guess the answer to the above question is: they are still waiting for the `dvi2lightbulb` program

A Maserati, a Mack Truck! [4]

In the previous example, the `\everypar` takes the whole paragraph as its argument. This is not a wise approach, however, as it places memory demands on TEX in case of long paragraphs, and it means that the text is scanned twice. One may then define

```
\everypar{\catchthepar\everypar={}}
\def\par{\endgraf\egroup}
```

with for instance

```
\def\catchthepar{\vbox\bgroup}
```

I took this approach when I implemented 'vario setting' of text, a paragraph layout where lines are right adjusted, except when they have to stretch beyond some point. In that case they
remain 'ragged right'. This way of setting text has in recent years become very fashionable in advertisements, especially for automobiles. Provided the paragraphs are suitably long, say five lines or more, and the right proportion of lines is left unadjusted, the visual effect of a block with an
occasional gap in the right margin can be, well, interesting. Some typographers, however, loathe this concoction [3].

When I first tried to implement this, I fiddled around with the usual parameters, but that led to nothing. Also I didn't want to take the approach from [5], as this would preclude hyphenated
words. The only other way that I could think of, was to let TEX set the paragraph right adjusted, and afterwards to unset the glue in some lines. Getting the lines from the paragraph would be
possible, I thought, using `\lastbox`.

As it turned out there were two difficulties. The first one was that lines that hang on the main vertical list can't be picked with `\lastbox`. I solved this by a variation of the above construct:

```
\everypar={\vbox\bgroup
    \everypar={}
    \def\par{\endgraf\pickthelines
             \egroup}}
```

Implementing `\pickthelines` brought to light the second difficulty: successive calls to `\lastbox` yield void boxes after the first call. At least, this was how it looked to me at first.

After a while I realised that the real reason was that a paragraph consists of an alternation of a box, a penalty, and a glue item, and so on. So I arrived at

```
\def\pickthelines{
    \setbox\investigation=\lastbox
    \ifvoid\investigation \else
    \unskip \savepenalty=\lastpenalty
    \unpenalty
       {\pickthelines}
    \investigatethebox
    \penalty\savepenalty \fi}
\def\investigatethebox
   {\setbox\tester=\hbox
           {\unhcopy\investigation}
    \ifdim\wd\tester<.94\hsize
          \box\tester
    \else \box\investigation \fi}
```

The recursion in `\pickthelines` uses TEX’s save stack, so this macro is not suited for very long paragraphs. But that’s not what it is intended for.

Note that TEX automatically reinserts the correct baselineskips, but that I have to do the appropriate penalties myself. The value beyond which a line should ‘snap back’ has to be determined experimentally; it highly depends on the `\hsize`, and the average word length of the text. Also, in order to prevent multi-line gaps, a test should be implemented if the previous line was set at natural width. Further fine tuning may involve the `\spaceskip` and the `\tolerance`.

Romeo, wherefore art thou indented? [6]

Browsing through Shakespeare’s complete works, I lighted upon a phenomenon that I thought would be pretty hard for TEX. Rather than explaining it at length, I’ll just show an example.

Jul. Romeo!
Rom. My dear!
Jul. At what o’clock to-morrow

shall I send to thee?
Rom. At the hour of nine.
Jul. I will not fail; ’tis twenty years till then.

The input for this example is:

```
\open
\Jul  Romeo!
\Rom[ My dear!
\Jul[ At what o’clock to-morrow
      shall I send to thee?
\Rom[ At the hour of nine.
\Jul  I will not fail; %
’tis twenty years till then.
\close
```

The reason for the strange indentation lies in a form principle of Shakespeare’s plays: all lines are in iambic meter, which implies that the number of syllables is somewhat predetermined. As this number is nowhere near reached after an exclamation like ‘Romeo!’, the addressed party will have to finish the line for Juliet. He falls far short of this ideal, by the way.

For this paragraph shape I didn’t use `\everypar`, but `\lastbox` would certainly be needed, namely to measure the last line of the previous player. In order to be able to use `\lastbox` all the text would have to be included in `\vbox`es. Each player should then do the following:

- measure the last line of the previous player,
- close the text of that player and hang it to the page, and
- then indent by the measured amount if a square bracket follows.

With some precautions for the first and last line of a scene, the basic commands look like:

```
\def\player#1 {\expandafter
    \def\csname#1\endcsname
       {\global\def\name{#1.}\playerline}}
\player Rom \player Jul
\newif\ifplaying % action started yet?
\def\open{\obeylines \playingfalse}
\def\close{\closepreviousplayer}
\def\playerline{\ifplaying
    \closepreviousplayer
    \else \playingtrue \fi
    \futurelet\next\maybeindentedline}
```

The job of ‘closing the previous player’ consists of taking the last box, measuring it, and hanging it back on the list. I leave the details to the reader.

```
\def\closepreviousplayer
   {\getpromptlength\egroup\unvbox\lines}
```

After the previous player has been closed, a new one can be opened, i.e., the box `\line` can be opened, with an indentation if necessary.

```
\def\maybeindentedline{\if\next[
    \expandafter\eatbracket
    \else \global\promptlength=0cm
          \nonindentedline \fi}
\def\eatbracket[{\nonindentedline}
\def\nonindentedline{\setbox\lines=
    \vbox\bgroup\leavevmode\strut
        {\setbox\tempbox=\hbox{\it\name\enspace}%
         \ifdim\promptlength>\wd\tempbox
               \hbox to \promptlength
                    {\it\name\hfil}%
         \else \box\tempbox \fi}}
```

For the complete layout used in the edition of Shakespeare’s plays I have these macros will have to be augmented by provisions for narrow columns: a too long line is split with the remainder set flush right.

Conclusion

Aberrant paragraph shapes can be automated in TEX to such an extent that no commands need to be inserted in the actual text. The token list `\everypar` is a handy tool for this, and for applications involving the length of lines `\lastbox` is a powerful instrument.

References

[1] Donald Knuth, The TEX book, 1986.

[2] Anonymous, The canonical collection of light bulb jokes.

[3] Inge Eijkhout, conversations, 1988/9.

[4] The Tubes, What do you want from life, 1975.

[5] Anne Brüggeman-Klein, Tugboat, 1988.

[6] William Shakespeare, Romeo and Juliet, 1594.