A Pragmatic Approach to Paragraphs

Philip Taylor

Something that never ceases to amaze me is just how many TEX users (including some who are quite eminent!) are familiar with the most arcane areas of TEX, yet when faced with what should be the simplest task of all — that of persuading a given paragraph to set correctly, without either generating `underfull \hbox` messages or abusing `\spaceskip` and its ilk — seem unable to achieve anything approaching a satisfactory solution. Whenever I receive a TEX document from such a user, and process it to find a dozen or so warning messages which the author seems quite happy to ignore, or to find the whole document set with grossly exaggerated interword space and flexibility, I ask myself why this apparently simple task should seem so difficult to so many.

To be fair, part of the blame must be said to lie with Knuth, for while Boson SlowCoach and its rivals would seem happy to set a paragraph with t h e m o s t a p p a l l i n g l e t t e r s p a c i n g, or to set a line with just three words and the most enormous interword space, Don took the decision that TEX would have none of this: either a paragraph would set correctly, or it would not set at all! And of course, most if not all of us agree with Don; for why else would we eschew the WYSIWYG wonders of Boson for the cabalistic complexity of the TEX language (unless, of course, we are all intellectual masochists, which I sometimes suspect). But also, to continue to be fair, part of the blame must be said to lie with *The TEXbook*; for whilst it more than adequately describes the various parameters which govern TEX's setting of paragraphs, it is somewhat less forthcoming about the methods by which suitable values for those parameters may be established.

In this article, I hope to present what I term '*a pragmatic approach to paragraphs*', for until some mathematical genius comes up with a formula or an algorithm by which suitable values for these parameters may be determined for any given combination of font, measure,[1] indentation, hyphenation patterns, etc., etc., etc., lesser mortals such as I will continue to have the unenviable task of typesetting text to some apparently arbitrary combination of these variables, and of convincing those for whom we are typesetting that it will *not* go to bromide until it meets our own somewhat exacting standards of æsthetic excellence, as well as those of Don and TEX...

We should start by considering those parameters which (a) most closely affect whether or not a given paragraph will set correctly, and (b) may reasonably be deemed to be within the ægis of the typesetter, as opposed to those which affect the setting but which are strictly under the control of the designer. The following table, although not exhaustive, lists some of the more important of the parameters which come under these two headings:

Typesetter specified	Designer specified
`\emergencystretch`	`\font`
`\pretolerance`	`\hsize`
`\tolerance`	`\patterns`
`\hbadness`	`\parindent`
`\hfuzz`	`\fontdimen` n

Of course, the designer will probably want to set upper bounds on even the entries in the typesetter's column, whilst the typesetter would be well advised indeed not to meddle with the entries in the designer's column (if he or she ever wants to be employed again!).

As to the method, I believe it to be simplicity itself, albeit somewhat complex to explain:

1. Process the text with sensible default values for the five 'typesetter's parameters'. Suitable defaults might be:

   ```
   \emergencystretch = 0 pt
   \pretolerance = 150
   \tolerance = 250
   \hbadness = 150
   \hfuzz = 0 pt
   ```

   If no error message issues from TEX, all is well and the task is complete.

   If not, then the error messages must be classified into two sets: those representing `overfull \hboxes` (and therefore true errors), and those representing `underfull \hboxes` (and therefore warnings rather than errors). Of these, the true errors must be addressed first.
2. Re-process the text, but this time specify `\tolerance = 9999`; this is most easily done by removing the assignment to `\tolerance` from the preamble, and initialising it on the command-line itself, as in:

   ```
   TeX "\tolerance = 9999 \input text"
   ```

[1] I use 'measure' in the typesetters' sense, meaning the width of the printed page excluding margins; for multi-column work, it refers to the width of a single column excluding margins and gutters.

Again the error messages must be classified into true errors and warnings, on the same basis as before.

If there are no longer any true errors, go to step 7; if there are still true errors, then the text and format may reasonably be considered to be pathologically bad, and further action will be required.

3. Examine the magnitude of the `overfull \hboxes`; if it is not more than the designer's specified upper bound on `\hfuzz`, set `\hfuzz` to the maximum overrun and go to step 7.
4. There are *still* `overfull \hboxes`. Sigh deeply. Examine the log to ascertain the line(s) in which these occur, and determine whether the problem is one of inadequate hyphenation (which is usually soluble), or one of over-wide tables, unbreakable formulæ, etc. If the problem is caused by inadequate hyphenation, go to step 6.
5. The problem lies outside the scope of this paper! Consider setting the offending table in a smaller font, reducing `\tabskip`, etc. Ask the author if the unsplittable formula *could*, in fact, be split. Apply your own heuristics to the task, and re-join this procedure from step 7 when you have resolved the difficulty.
6. The problem is caused by inadequate hyphenation. Ascertain by inspection whether an addition is needed to the `\hyphenation` list, or whether the offending word needs explicit discretionary hyphens to be added. An addition to the hyphenation list would be in order if there were nothing unusual about the word, but insufficient or poorly placed hyphenation points were indicated; explicit discretionary hyphens would be required if the word contained some hyphenation-inhibiting character, such as an accent, or if it were not preceded by glue. Augment the hyphenation and repeat from step 2.
7. Success! There are no more true errors. Now all that remains is to optimise the document, such that the final version represents the 'best possible setting', in some vague sense.

   Note the worst instance of badness in the `underfull \hboxes`. Set `\tolerance` to this value, and `\hbadness` to one less, and re-process the document. There should be no true errors, and no `underfull \hboxes` whose badness exceeds `\tolerance`.

   Now comes the surprising part. It might reasonably be thought that we have determined a lower bound on `\tolerance`, yet for many documents this proves not to be the case. Set `\tolerance` to one less than the value set in the step above, and re-process the document. It *may* still set correctly! The reasons for this are subtle, but may easily be understood by realising that TEX's concept of an 'ideal' paragraph is one in which the *overall* badness is minimised; TEX is *not* interested in minimising the badness of any one line. Thus we may now have a paragraph in which two or more lines are 'bad' in some sense, whilst the badness of the worst line has been reduced. Overall the paragraph is 'worse', but æsthetically it may *appear* 'better' (a paragraph consisting entirely of loose lines may look better than one in which one line stands out as being extremely loose).

   If the paragraph set correctly with `\tolerance` one less than the apparent lower bound, it may well do so again! Every time that the paragraph sets without true error, set `\tolerance` to one less than the reported worst badness and repeat. Eventually no further reduction in `\tolerance` will be possible, and an `overfull \hbox` will occur; re-instate the previous value and stop. Global optimisation is now complete, and the optimal value for `\tolerance` has been determined. If is this less than or equal to the designer's specified upper bound, then our work is done; if not, we will need to invoke `\emergencystretch`, and then, perhaps, to proceed to local optimisation.
8. Reduce `\tolerance` to the designer's specified upper bound, and set `\emergencystretch` to a small positive dimension. The behaviour of `\emergencystretch`, and in particular its interaction with other related parameters, is poorly understood, and indeed is the subject of research for the LATEX3 project. However, a sound rule of thumb is to set it to `1 em` (based on the primary text font); this value, strange as it may seem, appears equally suitable for both wide and narrow measures.

   Set `\hbadness` to one less than the value of `\tolerance` and re-process the document. If `overfull` boxes are reported, then we have a problem: we *could* increase the value of `\emergencystretch`, but this rapidly leads to severely `underfull` boxes and appalling æsthetics; it is probably better to proceed to local optimisation in these circumstances, which is the next step anyway.

For each line reported as being either `overfull` or `underfull`, consider the associated paragraph and see if additional discretionary hyphens might enable TEX to pack a few extra or a few less glyphs on the line; consider also whether one or more of the parameters listed in Appendix A (e.g. `\doublehyphendemerits`, `\exhyphenpenalty`) might be forcing TEX to adopt a less-than-perfect setting for this paragraph. If necessary, consider modifying one or more of these parameters just for the duration of the paragraph, by enclosing the latter between a `\begingroup`/`\endgroup` pair, modifying the parameter(s) immediately after the `\begingroup`. Remember that the paragraph will need to terminate with a `\par` or blank line *before* the `\endgroup` if the parameter change(s) is/are to have any effect.

Continue local optimisation, with no adjustments to `\tolerance` or `\hbadness`, until no further improvement can be achieved; if `overfull` boxes remain, the best option is to invite the author to re-cast the paragraph(s); if only `underfull` boxes remain, discretion is called for: visually inspect the offending paragraph(s), and invite the author to re-cast if and only if æsthetic considerations warrant it.

Remember that `\tolerance` has been reduced from its 'optimal' value to the designer's upper bound; if the author is unwilling to recast an offending paragraph, then bracket that paragraph in a `\begingroup`/`\endgroup` pair, as above, and for the duration of that paragraph only, re-set `\tolerance` to its 'optimal' value (and advise the designer that this was necessary).

Local optimisation is now also complete. Modify the preamble to incorporate the experimentally determined values for `\tolerance` and `\hbadness`. The justification for setting `\hbadness` to one less than `\tolerance` was not given above: it is simply that by setting it to the recommended value, it is possible to check that the experimentally determined value for `\tolerance` was in fact necessary, whilst suppressing TEX's reporting of lesser warnings; if TEX fails to report an `underfull \hbox` of badness equal to `\tolerance`, some error has been made. Of course, for distribution, `\hbadness` should be set equal to `\tolerance` so as to eliminate spurious warning messages.

This procedure may appear complex, but it is in fact very straightforward, and is certainly intuitive once the subtlety of repeated reductions in `\tolerance` is fully appreciated. The steps involved require a bare minimum of editing, and maximum advantage is taken of the fact that TEX's behaviour can be influenced by command-line parameters. The procedure has been consistently applied as this (and several previous) articles were written, and is in regular day-to-day use at my College, where productivity is valued even more than æsthetic excellence!

◇ Philip Taylor
The Computer Centre, RHBNC,
University of London, U.K.
<P.Taylor@Vax.Rhbnc.Ac.Uk>

Appendix A: Summary of paragraph-related parameters

Integer parameters:

```
\adjdemerits
\doublehyphendemerits
\exhyphenpenalty
\finalhyphendemerits
\hbadness
\hyphenpenalty
\linepenalty
\looseness
\pretolerance
\tolerance
```

Dimension parameters:

```
\emergencystretch
\hangindent
\hfuzz
\hsize
\parindent
```

Glue parameters:

```
\leftskip
\parfillskip
\rightskip
\spaceskip
\xspaceskip
```

Miscellaneous parameters:

```
\fontdimen 2
\fontdimen 3
\fontdimen 4
\fontdimen 7
\parshape
```