Developing British Hyphenation Patterns

Dominik Wujastyk

April 5, 1993

Contents

1 The problem

> Hyphens are regrettable necessities, and to be done without when they reasonably may.
>
> Fowler, *The King's English*, p. 284

There is a substantial difference between the hyphenation practices in British and American English. We are all used to feeling a distinct twinge on reading words like 'color, favor, honor' and so on. But sometimes one feels a less tangible pain: the other day I stalled at a line ending 'de-'. The next line began 'stroy', but for a brief moment I couldn't make sense of it. Later, I had a chance to check the word in British and American hyphenation dictionaries, and I felt satisfyingly vindicated when I found that against the American

$$de - stroy$$

there is the British

$$des - troy$$

It turns out that approximately one third of any list of words will hyphenate differently in British and American English.[1] This high percentage surprised me when I first calculated it, and it certainly justifies the necessity for British hypenation patterns for TEX.→ Show OHP's

There has been a certain amount of discussion of this issue on the networks, usually based on the example-word 'helicopter'. The Americans hyphenate this the way they say it:

$$heli - copter$$

The British should, the argument goes, hyphenate it according to etymology:

$$heli - co - pter$$

(thus displaying that the correspondent knows the interesting and amusing etymology of the word: a pterodactyl which flies in a helix). However, what the British actually write is

$$he - li - copter$$

[1] $diff.a = 171$, $uk/us.a = 394; 171/394 = 43\%$.
$diff.s = 57$, $uk/us.s = 256; 57/256 = 22\%$.
Totals: $228/650 = 35\%$.

the same as the Americans, but with one added degree of freedom. So much for etymology.

What then are the principles of British hyphenation? I can't answer better than by quoting Hugh Williamson:

> The customs of word-division derive partly from etymology, partly from meaning, partly from pronunciation, and partly from tradition. Effective communication depends upon conventions, in word-division as elsewhere, and the best conventions are those the reader is likely to expect. The first part of a divided word should not mislead the reader about the pronunciation or meaning of the second part.
>
> Word-division for the benefit of the reader, however, is best determined by a reader's perceptions; different customs apply to different words, and a few simple rules are not enough to find the right place.
>
> *Methods of Book Design*, pp. 48, 89.

We are all probably familiar with Knuth's fun examples of bad computer hyphenation: 'the-rapists who pre-ached on wee-knights' (TB, 449). Williamson adds some other beauties:

$$decent - ralization$$

$$propheth - ood$$

$$pro - ud$$

$$photo - compo - sition$$

$$thermonuc - lear$$

and I'm sure we all have our favourites.

2 TEX's Hyphenation System

TEX hyphenates by looking up every word in a dictionary to see where a hyphen is allowed.

This isn't exactly true of TEX82 (although I believe it was how TEX78 worked) because under Knuth's supervision, Franklin Liang worked out a strikingly clever way of compressing the dictionary into a tiny space. Essentially, what Liang discovered was that it is possible to arrange the words of the dictionary in a data structure called a packed *trie*. Don't ask me about this: I'm not qualified to answer. All I can say is that Liang's algorithm reads a hyphenation dictionary repeatedly, each time extracting more detailed information about how sub-patterns in words which do or don't have hyphens between them. Finally, it constructs a list of word-fragments, with numerical weights showing the possibilities for breaking at particular places. The word list is run through the algorithm several times, each time upping the ante, as it were, by reading in the results of the previous run, and looking for patterns not already covered.

The program embodying Liang's algorithm is `PATGEN`, and it forms a basic part of the TEX system. `PATGEN` is written in `WEB`, and was published as an appendix to Liang's published thesis in August 1983.

When I first started thinking about generating British hyphenation patterns, in late 1990, I thought that it would be a simple matter to whack a list of words through `PATGEN`, rather like using a sausage machine. I was wrong.

3 Sources of hyphenation information

There is a major difference between British and American dictionaries: in America, it is normal practice for dictionaries to include hyphenation points. This is never done in British dictionaries. So for information on how *we* should hyphenate, we are thrown back onto specialist publications.

Some of the sources I have found are:

1. Ronald McIntosh, *Hyphenation* (Bradford: Computer Hyphenation, 1990). [Available from Computer Hyphenation Ltd., Bradford Science Park, 1 Campus Road, Bradford BD7 1HR. Tel.: 0274 733317. Fax: 0274 736553.] ISBN 1-872757-00-6 (hardback), 1-872757-01-4 (paperback).

2. R. E. Allen (compiler), *The Oxford Minidictionary of Spelling and Word-Division* (Oxford: Clarendon Press, 1986. Reprinted 1990.) ISBN 0-19-869133-5.

3. *Hart's Rules* 39th edition (Oxford: OUP, 1983. Reprinted 1986.) ISBN 0-19-212983-X.

4. Market House Books, *The Penguin Spelling Dictionary* (Harmondsworth: Penguin, 1990). ISBN 0-14-051230-6.

5. John O. E. Clark, *Harrap's English Punctuation & Hyphenation* (Bromley: Harrap, 1990). ISBN 0-245-60020-5.

6. Susie B. Marshall, *Collins Gem Dictionary: Spelling & Word Division* (Glasgow: Harper Collins, 1968, reprinted 1991). ISBN 0-00-458749-9.

These are of widely varying usefulness. The most amusing is McIntosh's delightful book; perhaps the most downright useful is Clark's compilation for Harrap. This has an interesting and helpful discussion about hyphenation contributed by Fred Gill, as well as a list of 12,000 carefully hyphenated words. It emerges clearly that Gill is a craftsman in this area, and actually knows what he is talking about.

In contrast to this, the Penguin dictionary is positively misleading, being a thinly disguised dictionary of American hyphenations.

4 OUP's tape of words

Before describing my wrestling match with `PATGEN`, however, let me talk about the actual word-list that I used.

I was put on the track of the word list by a small book I bought in late 1990, called *The Oxford Minidictionary of Spelling and Word-Division*, compiled by R. E. Allen, and offering 60,000 hyphenated words. I wrote to Mr. Allen at the OUP in November 1990, in the following terms:

> I recently acquired a copy of your *Minidictionary of Spelling and Word-Division*, because I badly needed a guide to British English hyphenation.
>
> As you will know, the American Webster's range of dictionaries give hyphenation points as a matter of course, as does the *American Heritage Dictionary*. But guides to the – different – British rules of hyphenation are harder to come by. So I was delighted to find your *Minidictionary*.

I should now like to incorporate these British hyphenation rules in the software that I use for formatting my writings. I use the public domain program TEX, written by Prof. Donald Knuth at Stanford, and now widely used in academic circles.

The TEX program has an excellent hyphenation system built into it. This system needs to be set up initially by being fed a "hyphenation table". This table, in turn, is derived from a list of words hyphenated at allowed points, a list that looks almost exactly like your *Minidictionary*. Thereafter, TEX's hyphenation is automatic and very rapid. However, TEX has so far only been fed an American [English] hyphenation table.

In the *TEXbook* by D. E. Knuth (Addison Wesley, 1984, appendix H) Prof. Knuth describes how his student, Dr. Frank Liang, went about creating the American hyphenation table. He typed out a version of *Webster's Pocket Dictionary* (Merriam, 1966) of about 50,000 hyphenated words. Then he obtained an up-to-date machine-readable hyphenation dictionary of about 115,000 words from a publisher, ran comparisons and produced a refined and corrected final list. The table based on this list has been the basis of TeX's hyphenation ever since 1982. The hyphenation table (not the actual word list) is distributed free of charge and, like TEX, it is in the public domain.

I was wondering if it would be possible to acquire from the Oxford University Press a machine-readable copy of your *Minidictionary*, and permission from the OUP to use it to build a British hyphenation table for use with TEX?

The granting of such permission would be greatly to the credit of the OUP, since its help in this matter could be publicly acknowledged and would be recognised wherever TEX is used (there are over a million installations worldwide).

I would agree formally, of course, to use the OUP's word list for no other purpose, and to share it with no one else. It would be used for the single task of generating a hyphenation table of British English, in a format suitable for use by TEX. The word list itself would never be distributed; the hyphenation table would be the only item distributed in the public domain. I should like to note that it would be impossible to turn this hyphena-

> tion table into a commercial product: the international academic community has a record of great activity in the creation of utilities for TEX and a British hyphenation table is already under development at another centre (that team has agreed to suspend development pending the results of this enquiry). But it is a real opportunity for the name of the OUP to be linked with the use of TEX.

Robert Allen passed the letter on to Andrew Rosenheim in OUP's Electronic Publishing division. Imagine my delight when in December he wrote to me shortly before Christmas, saying:

> ... I am writing you now to say that you may have our permission to use this dictionary to build a British hyphenation table for use with TEX. We would like our help publicly acknowledged. ... We share the almost universal admiration for TEX, and are, therefore, grateful to be associated with it.

→ OHP of letter

True to their word, the OUP subsequently supplied not just the 60,000 hyphenated words from the *Minidictionary*, but a tape with 4 megabytes of data giving a list of almost 115,000 hyphenated words. This is more than twice as many word as Liang used for the American patterns, although he did compare and correct his results against a list of about this length.

4.1 Toal's filter

In February 1991 I received the tape from the OUP, and began putting the data into a format that `PATGEN` could swallow. First the file had to be transferred from tape, via floppy diskettes, to my hard disk. At that point, the file looked like this.→ OHP

Luckily, OUP had provided a complete key to interpreting the data!→ OHP

As I mentioned, someone else had already shown interest in developing British hyphenation patterns for TEX, and this was Graham Toal (and Graham Asher?). When the tape arrived, I began fooling around with AWK in an attempt to convert the data into a usable format. But DOS's memory limitations prevented the DOS ports of AWK from handling a file as big as our word-list. I turned to Graham for help in working with it, and in a

matter of hours he had written a memory-efficient utility to filter the data into a cleaner format for PATGEN.→ OHP

Because of OUP's understandable restrictions about sharing the data, Graham and I worked on the machine in my office. This was a handicap to Graham, since he did his work blind, so to say, on his own machine. Also, since he tends to work at night and sleep all day, it was sometimes hard to find a time to meet. Usually we would have sessions in the late afternoon, just after his breakfast and before my supper.

After filtration, the word-list was down to a much more manageable 1.5 megabytes. By March, and thanks entirely to Graham, the word-list was ready to be fed to PATGEN.

5 The struggle to compile a BigPatgen

The next hurdle was to compile PATGEN. I was reasonably familiar with WEB, and I had Breitenlöhner's excellent port of TANGLE and WEAVE. So getting a PASCAL version of PATGEN wasn't a problem. But of course it wouldn't compile under Borland's Turbo Pascal.

At about this time, Peter produced a change file for PATGEN, enabling it to compile under TP. (He also began modifying PATGEN in important ways to work with 8-bit character sets, but this aspect of his work didn't affect our project.) Peter's PATGEN compiled and ran, which was good news, but in spite of special code in the program to manage memory dynamically, it couldn't handle a word-list the size of ours.

I had a copy of – don't laugh – Microsoft's QuickC. Graham swung into action again, and produced several C versions of PATGEN using the public domain translator P2C. Although the code compiled, it wouldn't run.

At this time, I contacted Eberhard Mattes. I knew from discussions with Eberhard that he had used a Pascal-to-C translator to convert the TEX WEB distribution into C code for compilation. Eberhard was kind enough to send me his program, but I was not able to use it. I didn't understand enough of how it worked, the documentation was in German, and I didn't at that time have the emx/gcc development system that Eberhard had produced. It would have been a big job to download it, it would have meant creating more space on my creaking hard disk, and I still wasn't sure it would work.

5.1 djgpp

Now the saga moved forward as I learned about the DOS port of the GNU compiler, `gcc`, by D. J. Delorie of New Hampshire. This is an extraordiarily generous piece of work: a full port of the Free Software Foundation's `gcc`, `C++`, an 80387 sofware emulation and `go32`, an memory extender for 80386 and higher processors. This was the key to progress. Up to this point we had been completely bogged down by the memory limitations of DOS when dealing with a large file like ours. Luckily, I was working on an 80386 machine, so at last I had an industrial-strength compiler.

5.2 Karl Berry's WEBtoC

The next step forward was identifying the C translation of `PATGEN` which existed in the `WEB`toC distribution maintained by Karl Berry. Here was a Unix C version of `PATGEN`, complete with configuration files. As a Unix port, it treated memory as a single, flat address space. And now, at last, I had a compiler which could manage this arrangement.

When these pieces were finally assembled, they worked. I finally had a Big`PATGEN`, capable of processing our word-list.

6 Running `PATGEN`

The next hurdle was working `PATGEN`. As I have said, my understanding of the innards of `PATGEN` is minimal. This isn't entirely my fault: Liang's report on `PATGEN`, kindly photocopied for me by Graham, is written at a doctoral computer science level, and assumes a working knowledge of all sorts of data types, hashing, dictionary searching, and so on. And the crucial chapter 4, where Liang talks about how he generated the US patterns, is maddeningly vague at crucial points. I believe this is not an accident, and that there are aspects of the application of `PATGEN` which remain heuristic.

After kind advice from Peter Breitenlöhner (Feb–Nov 1991), Norbert Schwartz (4 March 1991), and others, I was not much clearer about what figures to feed to `PATGEN`. When you give it a word-list, it asks for the cryptic quantities *good_wt, bad_wt, threshold*. The `WEB` file for `PATGEN` includes an example of an interactive session with the program, but this didn't help much.

After several re-readings of Liang's description, including the crucial passage where he says (p. 36):

> The above somewhat vague considerations do not specify the exact pattern selection parameters that should be used for each pass, especially the first three passes. These were only chosen after much trial and error, which would take too long to describe here. We do not have any theoretical justification for these parameters; they just seem to work well.→ UHP

I decided that I wasn't likely to do better than to copy his own choices. So I simply used the identical figures he had used for the American patterns, as given on p. 37 of his report.→ OHP

Even this wasn't entirely without problems, however, because in the final pass through `PATGEN` it is important to set the *bad_wt* to an infinite value. This ensures the the patterns do not include any disallowed hyphenation points. Liang prints an infinity sign, but there wasn't one of these on my keyboard! After carefully perusing the `PATGEN` source code, and getting some more helpful advice from Peter Breitenlöhner I decided that 32,000 was infinite enough.

7 The results

Once the above pieces were in place, the actual generation of the patterns was rather quick, taking only a couple of afternoons. I arrived at a set of patterns 8526 lines long (as against 4449 for the US patterns). The UK patterns require a *trie_size* of about 14,000 (about twice the US requirements).

The final run of `PATGEN` on the British word-list produced 168,882 good hyphenation points, 284 bad ones, and missed 19,574. In percentage terms, this makes 89.61% good hyphens, with 0.15% bad and 10.39% missed. These figures are almost exactly equal to Liang's (89.3% and 0.0% with the 2nd decimal place omitted.).

8 The lessons

Get an operating system!

9 The future

You can help by providing genuine exceptions. We need to catch those remaining 284 words. But please check the *Minidictionary*, and perhaps the other sources, first. These hyphenations are not 'true', they are the OUP's. That is authoritative enough for me, and I don't wish to enter into arguments about improving these patterns.

10 The Good Guys

Graham Toal, Wayne Sullivan, Peter Breitenlöhner (Feb–Nov 1991), Norbert Schwartz (4 March 1991), Karl Berry.

11 The End

Finally, I leave you with the following contrast. I rest my case.

British hyphenation *American hyphenation*

Eng-lish En-glish