Frank Liang

Frank Liang worked with Don Knuth on the hyphenation algorithm for TEX78, and Frank's PhD thesis presented a better hyphenation algorithm that was used in the TEX we know today and has been used by many other typesetting systems.

[Profile completed 28 July 2010.]

[*Editor's note: Frank Liang was unable to participate in a traditional interview but helped us as we pulled together these quotes and notes.*]

Frank has told us:[1]

> I grew up on the east coast (Delaware) and later Southern California. My parents came from China in the early 1950s as graduate students in physics. My father worked in industry and was later a professor at San Fernando Valley State College (now California State University Northridge). I attended California Institute of Technology in Pasadena, where I majored in math. While there I took a course on "Analysis of Algorithms" which I believe was initiated by Prof. Knuth when he was there and working on the material that became *The Art of Computer Programming.* So I applied to Stanford in Computer Science and became one of Don's students (fall 1976).

By 1977 Franklin Liang was involved with TEX, working on hyphenation and other TEX activities.

In Frank's *TUGboat* paper on hyphenation, he states that the hyphenation algorithm described in that paper "was developed by Prof. Knuth and myself in the summer of 1977."[2]

The original implementation of hyphenation in TEX was changed for TEX82, using a pattern-based method which was the subject of Frank's 1983 PhD thesis, *Word Hy-phen-a-tion by Com-put-er.*[3] This method is also described in Appendix H of *The TEXbook.*

As part of the panel discussion at the TUG2010 annual conference,[4] Frank answered the question, "How did you discover your hyphenation algorithm?"

> I was assigned this problem as a thesis in 1978, I believe. As I have mentioned in my thesis, there was an initial suggestion to use this kind

[1] E-mail of July 20, 2010.

[2] Frank M. Liang, "TEX and Hyphenation," *TUGboat*, Volume 2, No. 2, July 1981, pp. 19–20, `http://tug.org/TUGboat/Articles/tb02-2/tb03liang.pdf`

[3] http://www.tug.org/docs/liang/

[4] `http://river-valley.tv/tug-2010-panel`

> of statistical algorithm which looked at two letters — well actually four letters — surrounding a potential break point, and then you were supposed to make tables using two letters before, the middle two letters, and the last two letters and then combine these tables somehow to do hyphenation. And I quickly found that that wasn't sufficient; you needed more context. One example I mentioned in my thesis is that sometimes a letter seven letters away from the hyphen point can alter the breakpoint. Anyway, I'm playing with word lists a lot and came upon the idea that just patterns of letters was a very simple way. Because I had started with just these two letter digrams, it was natural just to extend that to longer letter sequences. And then through a long process of evolution I then came up with patterns and then with the rules and exceptions. Don actually came up with the idea of assigning the numbers and having the [inaudible] at one of my thesis review meetings. I sort of had the idea of having rules and exceptions, and he said, "Oh, you could assign numbers to them." So that's part of the answer.

Frank continues his explanation.[5]

> One of the hardest parts of the whole thing was acquiring a suitable word list. I didn't do a very exhaustive search for dictionary databases, there weren't that many at the time, and I suspected that most were proprietary anyway. So since the Merriam-Webster one was available at the Stanford AI Lab, and more or less suitable for my purposes, I ended up working with that. It had hyphenation points, but upon looking at it, there were many errors and lots of typos and things that were not quite right. So I had to hand edit the dictionary, and that took about three months. So that is where most of the work was actually.

In response to a follow-up question relating to packed tries, Frank answered:

> The tries came up separately after the pattern idea. I was playing around with the word lists all the time, and I needed some kind of relatively fast algorithm to quickly collect information about the patterns in the word lists with hyphenation points and then to test out various theories. Because these were just simple strings, it was natural to look at variations of standard data structures like tries and to read related papers like Don's on pattern matching. Using tries is very fast because it is based on indexing, but they tend to be very sparse so you then have to use various tricks to speed up things. The idea for doing the weird packing came from paper by a another professor at Stanford at the time, Andrew Yao. His paper talking about storing a sparse table. His was a somewhat different application, but he had the idea for sparse things of sort of interleaving them all in one array and thereby saving space while maintaining speed.

Frank's pattern-based approach to hyphenation essentially solved the hyphenation problem for TEX and has been used in many other free software document processors, such as OpenOffice and Apache FOP. A 2007 document on *troff*[6] credits Frank's

[5] This "quote" is a combination of another four sentences from Frank's answer during the panel discussion and a follow-up clarification from Frank in an e-mail of July 28, 2010.

[6] `http://heirloom.sourceforge.net/doctools/troff.pdf`

approach to hyphenation. The method has also been used in commercial systems. By now, sets of patterns have been produced for essentially all languages that use hyphenation.[7]

Frank also worked with Michael Plass on the prototype implementation of TEX. In the TUG2010 panel session, Don Knuth said, "Can I ask a question of Michael Plass to describe his experiences in 1978 when I went to China and asked him to implement the prototype of TEX."[4]

In response Michael noted that Knuth left "a few pages of his implementation ideas,"[8] Frank handled the hyphenation and output, and Michael himself worked on storage allocation and the macro process. Michael noted that because Frank was "working on the output — the printer driver end of this — we were able to make some prints by the end of that summer," when Don returned from China.

Frank elaborated,

> Well, what I remember in addition to the hyphenation, which actually I did while Don was here, was that after looking at his notes, Mike decided (he was sort of in change) — we decided to split it up and I would do the output and he would do the rest. And I said this didn't sound like much. At the time he thought output sounded pretty difficult, because maybe he didn't know how to do it right off the bat. And of course I had already been playing around with the XGP so I knew how to do it. So for me actually it wasn't that much work, and he ended up with with much more than he thought. What he gave me was a list of boxes, graphics boxes, and I just put them on the printer, so that wasn't much work for me. Obviously we way underestimated how much work it was going to be and obviously it took two more years, or several more years, of Don's work later.

In 1979 Frank was a teaching assistant for Knuth's Concrete Mathematics course at Stanford, which was given that year by Ron Graham. Also during his time at Stanford, Frank did other mathematical work.[9]

Regarding life after Stanford, Frank says:[1]

> After finishing up my thesis on hyphenation for TEX, I moved to Seattle, where I was the second person hired to work on Microsoft Word (late 1982).[10] Windows was still in early development, so we produced several versions for the PC, as well as the first versions for Mac. I worked mostly on printing and page layout, and we supported a lot of dot matrix and daisy wheel printers, and then the HP LaserJet when that came out. I also put the TEX hyphenation algorithm into PC-Word. Later I did some technical management and also worked on proofing tools. But since the mid 1990s I have been retired from the tech industry.

[7] `http://ctan.org/tex-archive/language/hyph-utf8`

[8] Also, in the summer of 1977 Don Knuth has revised `TEXDR.AFT` (*Digital Typography*, pp. 481–504.) to be `TEX.ONE` (Idid., pp. 505–532) providing an initial spec for TEX.

[9] For instance: "A Lower Bound for On-Line Bin Packing," *Information Processing Letters 10*(2), pp. 76–79, 1980; *The Dinner Table Problem*, with Bengt Aspvall, `ftp://db.stanford.edu/pub/cstr.old/reports/cs/tr/80/829/CS-TR-80-829.pdf`

[10] In the panel discussion at the TUG 2010 annual conference Frank noted that his office was "right next to Bill Gates's for about a year. But I hardly ever saw him except he would walk by in the morning and would walk out in the evening."