How did Dennis Ritchie Produce his PhD Thesis? A Typographical Mystery

David F. Brailsford
School of Computer Science
University of Nottingham
Nottingham NG8 1BB, UK
dfb@cs.nott.ac.uk

Brian W. Kernighan
Department of Computer Science
Princeton University
Princeton, NJ 08540, USA
bwk@cs.princeton.edu

William A. Ritchie
Thinkfun Inc.
1725 Jamieson Ave
Alexandria, VA 22314
bill.ritchie@thinkfun.com

ABSTRACT

Dennis Ritchie, the creator of the C programming language and, with Ken Thompson, the co-creator of the Unix operating system, completed his Harvard PhD thesis on recursive function theory in early 1968. But for unknown reasons, he never officially received his degree, and the thesis itself disappeared for nearly 50 years. This strange set of circumstances raises at least three broad questions:

- What was the technical contribution of the thesis?
- Why wasn't the degree granted?
- How was the thesis prepared?

This paper investigates the third question: how was a long and typographically complicated mathematical thesis produced at a very early stage in the history of computerized document preparation?

CCS CONCEPTS

I.7 [**Document and Text Processing**]: I.7.1: Document and Text Editing, I.7.2: Document Preparation.

KEYWORDS

mathematical typesetting, electric typewriter, digital restoration, archiving, troff, Postscript fonts, IBM 2741

ACM Reference Format

David F. Brailsford, Brian W. Kernighan and William A. Ritchie. 2022. How did Dennis Ritchie Produce his PhD Thesis? A Typographical Mystery. In Proceedings of The 22nd ACM Symposium on Document Engineering (DocEng2022). ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3558100.3563839

1. Introduction

In June 2020, David Brock, a historian of technology and director of the Computer History Museum's Software History Center, published *Discovering Dennis Ritchie's Lost Dissertation*, an article about Dennis's long-lost PhD thesis, *Program Structure and Computational Complexity.*

Brock's article [1] makes for fascinating reading. Much of it is focused on the thesis's contributions to recursive function theory and early theoretical computer science. To over-simplify, the thesis showed that a class of programs expressed as assignments, increments, and nested loops was capable of performing arbitrary computations. Quoting Brock, "In loop programs, one can set a variable to zero, add 1 to a variable, or move the value of one variable to another. That's it. The only control available in loop programs is ... a simple loop, in which an instruction sequence is repeated a certain number of times. Importantly, loops can be "nested," that is, loops within loops." In more modern terms, these *loop programs* are a Turing-complete computational model, equivalent to Turing machines and Church's lambda calculus.

The first section of Brock's article, "Everything but Bound Copy," explores an intriguing open question. Although the thesis was essentially finished, lacking only a handful of trivial typographical corrections and presumably a pro forma final public oral exam, the thesis was never submitted to Harvard (or so it is believed), it definitely was never accepted by Harvard, and thus Dennis never actually received his PhD.

Why wasn't the thesis accepted by Harvard? Why didn't Dennis ever get his PhD? Indeed, why did he never explicitly acknowledge the unusual situation? And how did the thesis simply disappear for nearly 50 years, coming to light only after Dennis's sister Lynn tracked down a copy after his death in October 2011?

One oft-told story was that Harvard wanted a fee for processing the thesis and Dennis thought that he shouldn't have to pay it. If thesis rejection was as simple as a library fee dispute, however, we should expect that Dennis would have recounted the story, embraced it in his typically self-deprecating way, and turned it into a life lesson similar to the way he described how he wasn't smart enough to become a physicist so he turned to computing.

Instead, the thesis disappeared for 50 years and was never mentioned by Dennis. More significant is how he allowed the uncorrected story that he had a doctorate to spread

throughout his lifetime. This includes citations for the Turing Award and the Japan Prize, both of which mention his Harvard doctorate. Dennis was so guileless his entire life that this must have weighed on him, but it is unlikely that we will ever know for sure. This part of the story, still a work in progress, is told by Bill Ritchie, Dennis's youngest brother, at dmrthesis.net, a web site devoted to the thesis [2].

This paper explores a different set of questions that are particularly appropriate for a conference on document engineering. How did Dennis manage to produce multiple versions of a long and complicated document with exceptionally high quality and accuracy, at a time when computer preparation of mathematical documents was not even in its infancy? What hardware and software might he have used, and how did it relate to the state of the art at the time?

We also describe our attempt to recreate the thesis using standard Unix document preparation tools like *troff* and *eqn* that became widely available only a few years later [3]. A machine-readable version of the thesis would enable a number of studies that are too hard with only imperfect scans of typewriter-like printouts, and thus might shed some light on the early history of computational complexity.

The results of our experiments and supporting documents for this paper are available at www.cs.princeton.edu/~bwk/dmr.

2. Background: 1960s Typing Technology

The 1960s was the decade of the electric typewriter, but not yet the electronic typesetter. IBM, the leader in the field, was focused on the burgeoning business community rather than scientific research; their primary goal was office integration and efficiency rather than the elaborate scientific notation that was required for academic research papers.

2.1. IBM Electronic Typewriters

In 1959 IBM introduced the Model C basket-and-typebar typewriter, then in 1967 the Model D of the same design. Other manufacturers including Royal, Olympia, Smith Corona and Olivetti offered typewriters with similar design and features. For the most part these machines used a fixed width system, either 10 or 12 characters per inch, with 6 lines per inch vertical spacing.

In 1961 IBM introduced the revolutionary new Selectric typewriter with its spinning typeball or 'golf-ball' design, which kept the paper stationary while the typeball moved its way across the platen. Inserting a special purpose character such as a mathematical symbol or Greek letter was much easier with a Selectric: the typist could simply swap in a new typeball rather than having to insert a supplementary type stick into the basket, which was the method with traditional machines.

While not fully modern, the Selectric was more suitable for electronic control than were traditional typewriters. In 1964 IBM introduced a standalone word processing machine, the MT/ST (magnetic tape, Selectric typewriter), which was marketed primarily to large businesses along with dictation equipment with the intention to better integrate secretaries and their bosses in the executive suite.

The Selectric also formed the guts of the IBM 1050 printer terminal, which was introduced in 1963 for use with the System/360 and other mainframe computers, and the more streamlined 2741 terminal which launched in 1965. Compared with other printing devices of the time, Selectric-based terminals were faster, had better print quality, were quieter and had more easily interchangeable special characters and type fonts. But IBM always saw these machines as proprietary and never made any attempt to use ASCII standards or otherwise embrace the coming electronic revolution.

2.2. Early Formatting Programs

In 1964 Jerry Saltzer, while working for Project MAC at MIT, wrote the text formatting program RUNOFF to help type his thesis proposal and subsequently his thesis. (Conveniently he had an IBM 1050 terminal in his home that was connected through CTSS to the 7094 mainframe at Project MAC.) As much of a historical breakthrough as this was, RUNOFF did not have any facility to help change golf-balls in the middle of printing a manuscript; likewise it had no commands for superscripts or subscripts as these were not supported by the Selectric.

According to Saltzer, "If you look skeptically at its list of features you will discover that it includes just enough to allow me to prepare my own PhD thesis, nothing more. For example, my thesis had no equations, so RUNOFF had no facility for them. Development of RUNOFF features ended in 1966 when I turned in my thesis."

So what were the conditions that Dennis faced as he was completing work on his doctoral thesis in late 1967 and early 1968? Mike Fischer (a fellow Harvard grad student and brother of Dennis's first thesis advisor, Patrick Fischer) says "Computerized typesetting was in its infancy in the 1960s. Of course people had the idea of having the computer print their paper instead of a typist, but printers were limited in both their quality and their range of allowed fonts. For the time that Dennis was at Harvard, it was electric typewriters."

It may be hard for readers today to appreciate just how labor-intensive it was to prepare documents before the creation of word processing programs, when there were only mechanical typewriters—better than clay tablets or quill pens, to be sure, but any change of more than a few words in a document could require a complete retype. Thus most documents went through only one or two revisions, with handwritten changes on a manuscript that had to be laboriously retyped to make a clean copy.

As Jerry Saltzer said "The standard procedure for preparing a PhD thesis in the 1960s was to assemble a rough draft either by typing or in longhand and then hire a professional thesis typist."

Mike Fischer echoes this: “With my own thesis, I began by typing a rough draft myself that I then edited using scissors and tape and pencil and white-out. Once a page became too patched up, I made a Xerox copy to create a clean new version that I could then edit and chop further as needed. Once my draft was finalized, I paid $500 to a professional typist ($4,300 today) to retype it into a final dissertation that I could bind and submit to Widener Library. I specifically recall that my typist used typebar special characters, which would have meant that she used an IBM Model C or D typebar typewriter.”

Brian Kernighan created a stripped-down version of RUNOFF that he called Roff to print his Princeton PhD in 1968. Roff was written in Fortran and ran on an IBM 7094. Input was on punch cards, so making a revision required replacing some cards with new ones, then submitting 3 boxes of cards (about 20 pounds or 10 kg) to an operator.

Kernighan’s thesis also avoided mathematical expressions as much as possible because his output device, an IBM 1403 line printer, couldn’t handle them. Subscripts and superscripts were eliminated, and special characters were inserted by hand after the fact. Roff’s one novelty was that it did automatic capitalization of the first letter of each sentence, since punch cards really only supported upper case.

In short, typing a mathematical paper in the 1960s was hard work: time consuming, detailed, and laborious.

So in practical terms what were Dennis’s options for typing his thesis? As indicated, many doctoral students used a department secretary or professional typist to finalize their theses. This is certainly a possibility, though in this event Dennis still must have exercised close oversight over all character placements.

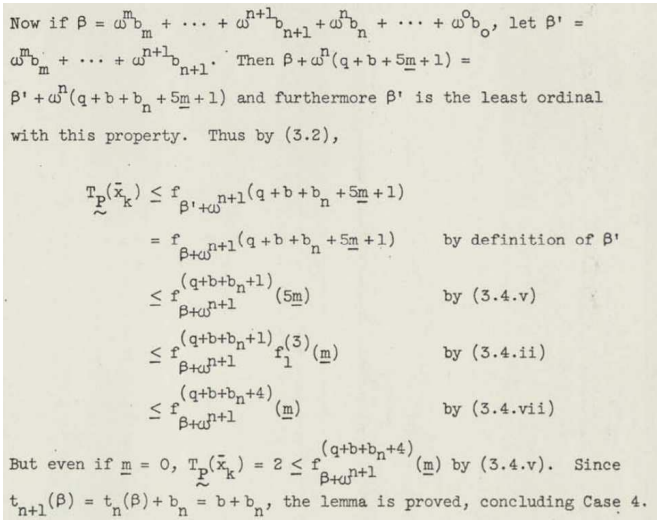
He could have typed it himself, or perhaps he had a Bell Labs technical typist help him with final drafts. In late 1967 when he was finalizing his dissertation he was working at the Labs at Murray Hill NJ and living at his family home in Summit. However, neither family members nor Bell Labs colleagues ever heard him discuss anything like this.

Dennis worked on Project MAC during graduate school and had a CTSS account which he kept active while moving to the Labs and starting work on Multics. Bill Ritchie recalls that at some point shortly after starting work, Bell Labs installed a toll-free phone line and a 2741 terminal in his home basement office, though he is not certain exactly when. If there was a way to use these to help type his thesis, one would think that Dennis would have figured it out, especially with RUNOFF as a model. But there is no evidence that he did this, or even that he could have done so given the technical limitations described above.

Dennis never talked about his thesis, to colleagues or to family, so no one knows how the document was actually produced. For the purposes of this paper it really doesn’t matter. What is important is the actual work product, which from a typographic perspective is vastly superior to most other math dissertations of the period.

3. The Thesis Document

The final draft of “Program Structure and Computational Complexity” is 180 double-spaced pages and includes nearly 40 different math symbols and Greek letters scattered across many pages. Nearly every page after the introduction contains multiple sub- and superscript expressions, often three layers deep; many pages contain dozens of sub- and superscripted characters. Figure 1 shows half of one comparatively easy page.



Now if $\beta = \omega^{m}b_{m} + \cdots + \omega^{n+1}b_{n+1} + \omega^{n}b_{n} + \cdots + \omega^{o}b_{o}$, let $\beta' = \omega^{m}b_{m} + \cdots + \omega^{n+1}b_{n+1}$. Then $\beta + \omega^{n}(q + b + 5\underline{m} + 1) = \beta' + \omega^{n}(q + b + b_{n} + 5\underline{m} + 1)$ and furthermore β' is the least ordinal with this property. Thus by (3.2),

$$
\begin{aligned}
T_{\underset{\sim}{P}}(\bar{x}_k) &\le f_{\beta'+\omega^{n+1}}(q + b + b_n + 5\underline{m} + 1) \\
&= f_{\beta+\omega^{n+1}}(q + b + b_n + 5\underline{m} + 1) && \text{by definition of } \beta' \\
&\le f^{(q+b+b_n+1)}_{\beta+\omega^{n+1}}(5\underline{m}) && \text{by (3.4.v)} \\
&\le f^{(q+b+b_n+1)}_{\beta+\omega^{n+1}} f_1^{(3)}(\underline{m}) && \text{by (3.4.ii)} \\
&\le f^{(q+b+b_n+4)}_{\beta+\omega^{n+1}}(\underline{m}) && \text{by (3.4.vii)}
\end{aligned}
$$

But even if $\underline{m} = 0$, $T_{\underset{\sim}{P}}(\bar{x}_k) = 2 \le f^{(q+b+b_n+4)}_{\beta+\omega^{n+1}}(\underline{m})$ by (3.4.v). Since $t_{n+1}(\beta) = t_n(\beta) + b_n = b + b_n$, the lemma is proved, concluding Case 4.

Figure 1: Half of page 46

3.1. The Basic Grid

Clearly the document was typed on a fixed width typewriter, 12 characters per horizontal inch and double spaced at 6 lines per vertical inch. Most typewriters at the time had a manual platen adjustment that allowed 1/12" line height half-steps, which in principle allowed a typist to organize the character placement of any paper onto a 1/12" by 1/12" grid.

We applied a 12x12 grid over every tenth page to evaluate character placement. The document is a scan of a copy so not everything lines up perfectly but the precision of positioning is evident in the excerpt in Figure 2 and other figures; a full set of sample pages is available at dmrthesis.net.

The equations line up perfectly on the overlaid 1/12" by 1/12" lattice. Every character is placed exactly inside its appropriate cell, with no errors or out of place symbols. For the most part this precision is evident throughout the entire document.

Evaluating the thesis after it surfaced, Mike Fischer explained “The fact that it was typed on a 1/12" x 1/12" grid is no surprise to me. 12 characters/inch was the standard "elite" typewriter. 6 lines/inch was also standard, but the

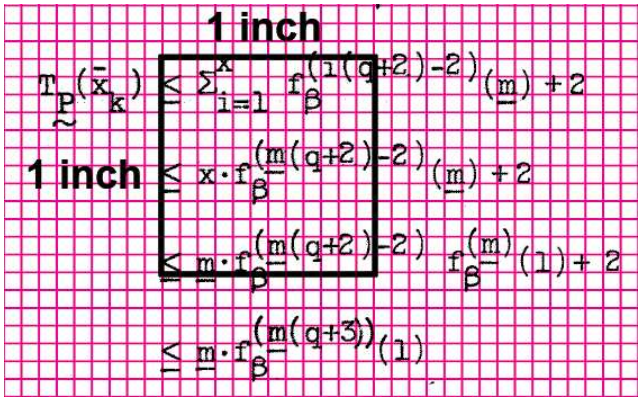


Figure 2: Character positioning

IBM typewriters had a platen that would allow half-spacing, i.e., 12 half-spaces per inch. So there was no need to do manual continuous positioning in order to type a page where all letters ended up on the grid. Rather, in order to type a superscript "2", one would roll the paper down one click, type the "2", and roll it back up one click to continue."

In our research we found only a few math doctoral theses from the 1960s where sub/superscripts consistently follow a 1/12" by 1/12" grid, and none of these were nearly as complex as Dennis's thesis. One possible explanation is that it was just too hard to keep track while typing the individual characters of a complicated expression onto the page one character at a time.

3.2. Vertical Positioning

But there are fascinating departures from the grid as well. We start with vertical control, the up and down spacings that create the sub and superscripts. There are many examples through the thesis where, even with multiple layers and extensive undulations, the characters lie directly in the grid.

There are several dozen cases sprinkled throughout the document where characters seem to be wedged into place outside the grid system. In some instances this could be a relaxation, where the typist took a moment off from the rigors of the grid and opened the escape mechanism to allow a single character to be inserted free form. Other times the placement seems intentional, as if an individual character belonged in a half-step between lines.

The argument for intentional placement comes into more clear focus when we examine some of the more complex mathematical expressions contained within the document. For a relatively small but not insignificant number of instances, a superscript is itself an exponent expression containing its own sub or superscripts. In these cases Dennis appears to have doubled the precision of the vertical grid, centering the exponent on a 1/24" grid so that its own sub/superscripts can center onto the main 1/12" grid.

3.3. Horizontal Positioning and the + Symbol

Dennis was also able to split the horizontal grid of 12 characters per inch. We have identified two specific situations where he manipulated the document copy onto a 1/24" center in order to reposition one or more characters. Typewriters of the day were not set up to handle this kind of adjustment; repositioning the paper or adjusting the typing mechanism by this small precise amount would be difficult, exacting work. And yet in each of these cases it appears that he made adjustments to widen or justify horizontal spacing purely for aesthetic reasons.

When it appears in a subscript or superscript position, the "+" symbol is treated like other character, with no blank space either side. But when a " + " appears in the baseline of an exponent, Dennis almost always inserted one or more blank characters before and after the symbol to improve visual spacing.

His primary spacing rule consisted of positioning the "+" symbol between two adjacent blank cells, then striking the "+" directly between these cells on a 1/24" center. The "+" symbol thus becomes a 1/12" wide character surrounded by two 1/24" blank half-spaces: a single character centered within two cells, as shown in Figure 3, marked with blue arrows.

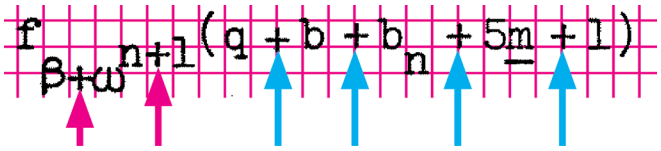


Figure 3: Plus sign example

Dennis' treatment of spacing around the "+" sign stands out as among the least disciplined parts of the document. He kept to his standard "2 cell spacing" about 80% of the time, but there are plenty of instances when other spacings are used as well.

We have no idea how it would even be possible to make these horizontal adjustments with the typing machines of the time. Why did Dennis not simply put a full character space left and right and avoid splitting the grid? He did this a few times; nothing stood in the way of this decision. The "+" sign leaves us with a basket of unresolved issues.

3.4. Roman Numeral Lists

Roman numeral lists were a frequent component of the thesis; Dennis used such lists twenty three times. With the first five lists, all typed characters stayed in their respective 1/12" by 1/12" cells.

Starting with the Roman numeral list on page 39, the spacing changed to create a new typographic effect. Dennis accomplished this by shifting right the characters of Roman number (i) by 1/12", then shifting the characters on line (ii) right by 1/24", centering both numerals on numeral (iii) below.

Then every even number in each list after this follows this new typography rule, adjusting to a 1/24" center alignment; see Figure 4, where the center line is marked.

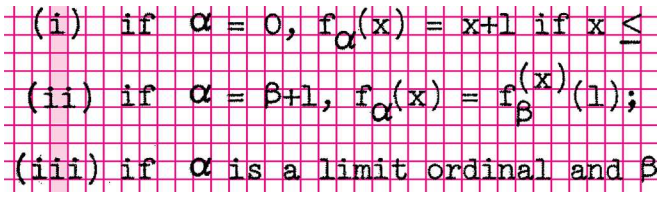


Figure 4: Centered roman numeral list

It is as if for this Roman numeral list Dennis found a new skill which he then employed on the remaining 13 lists. We have no idea how he did this. Moreover, the visual distinctions between the center-justified and non-justified lists are so subtle as to seem almost gratuitous. Thus we also have no idea why he did this. And yet this treatment of the Roman numeral lists is what Dennis elected, including a collection of 1/24" half steps, seemingly for the pure fun of it.

3.5. Version Control

Since Dennis's death in 2011, two versions of the thesis have surfaced. In January 1968 he had sent copies of a late draft, nearly complete but containing approximately 60 minor typos with pencil corrections noted, to his student colleague Albert Meyer and to his initial thesis advisor Patrick Fischer. These are identical except for a few of his pencil comments in margins.

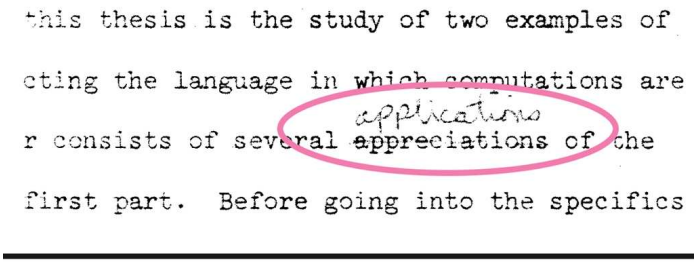
Sometime around 2015, David Brock discovered a thesis manuscript in Dennis's personal papers, which had been donated to CHM by the Ritchie family. This version was nearly identical to the January draft except that all 60 typos had been corrected. This undated draft was likely made in late January or February 1968 and was extremely close to finished; only 6 minor corrections were still needed.

For the most part the revisions from January to February version were simple typos, replacing a few characters on a single line of copy, as in Figure 5, which shows a handwritten note from page 8 of Albert Meyer's copy, and the correction on the final version.

It appears that Dennis was able to replace incorrect copy with new text in the surrounding original copy, but note the extra space in the corrected version. Other than corrections, the January and February versions are identical in every way: February is not a new version, but a copy of the January draft except for corrections.

To test whether this could really be true, we chose six sample pages at random, scanned January and February versions of each page, then overlaid them to see where differences might emerge. The pages are available at dmrthesis.net.

So this leaves us with our final mystery: How could Dennis have done this? The natural explanation is that he used white-out and a Xerox copy machine. Except that there have



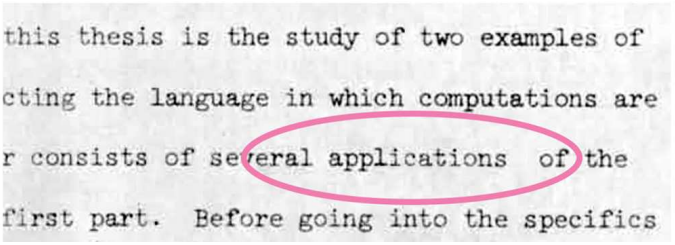


Figure 5: Handwritten correction noted and made

been no reports from anyone from this era that white-out / replace / Xerox editing treatment was possible at the precision and reproducibility seen here.

4. Recreating the Thesis Document

The previous sections have highlighted any number of mysteries about how the thesis was produced. When we began this work, the thesis was available only as a bitmap PDF from a scanned original. It seemed that the thesis would be more accessible for further study if it could be recreated as a searchable text document. Thus we have recreated the thesis by converting it to *troff* format, with extensive use of *eqn* for the mathematical parts. In a sense, this is anachronistic, since *nroff*, the typewriter-only precursor of *troff*, dates from about 1972, and *eqn* was not available until 1974, six years after Dennis's thesis was completed early in 1968. But it seems certain that Dennis would have used *nroff*, *eqn* and related tools if they had been available to him.

4.1. Font Issues

In a previous restoration project [4] we discovered the importance of finding that a set of fonts created for the Linotron 202 typesetter had been migrated into the modern era in Adobe Type 1 PostScript format. All the mechanical limitations of the 202 could be sidestepped when its fonts could be used on any PostScript-capable typesetting machine.

For our present work, we have assumed that Dennis did use an IBM 2741 terminal to print his thesis; this seems the most likely possibility. If only someone had made the 2741 character set available as a font on a typesetter or laser printer—rather than as a set of metal characters on a golf-ball for a near-obsolete device—then we could make Dennis's thesis machine-readable and amenable to further research.

Our font consultant, Chuck Bigelow, determined that the 2741 character set had indeed been converted to a fixed-pitch Bitstream Inc. font called 'Pica 10' with the same metrics as the original golf-ball element. Unfortunately this font only includes the standard character set. Despite intensive efforts we have been unable to trace any modern form of the fixed-pitch mathematical symbol and script characters that Dennis so clearly had available on golf-ball elements in early 1968.

Fortunately the Adobe Symbol font (for Greek characters and mathematical symbols) and as used in *troff* and *eqn*, worked well when supplemented by YandY's MTMS script collection. Occasionally using the *eqn* 'define' capability, we were able to confect missing symbols such as 'monus', which appears at the right-hand end of the first lines of Figures 6 and 9.

$$\text{for } 2 \le i \le n+1,\ \pi_i^{n+1}(z) = (\pi_{i-1}^{n}((z)_2) \div (z)_1) \div (z)_0$$

So π_i^{n+1} is also in $\mathfrak{L}_2$.

Figure 6: Constructing a weird symbol

4.2. Retyping the Thesis

Once the Pica 10 font had been tested we realized that a 'thesis-rebuild' would be possible, though it would need a major effort to bring it about. We have set up a repository web site [5] devoted to creating a canonical bitmap-PDF file of what the original 180-page thesis was intended to contain, together with source files for a step-by-step rebuild of the entire thesis, using *tbl*, *eqn* and *troff*, leading to a final high-quality PDF with full text and graphics.

The starting point for the recreation was a set of scanned-page bitmaps, principally created from two of the original 1968 hard copies. More recently, dating from the release of PDF in 1989, it has been possible to enclose these scans in a PDF wrapper.

For material of this sort the Adobe Acrobat PDF viewer provides an ingenious device for making the acquired pages seem to be, at least partly, searchable. An OCR engine analyses each page and calculates point-size string widths and bounding box coordinates for all the words it identifies.

It then re-typesets the page, often using a generic font such as Helvetica, in what is called Text Mode 3. This mode is a sort of electronic 'invisible ink' which is fully compliant with all of PostScript's capabilities for sizing and placing of text, except that it omits the final commands that render material and make it visible.

If this correctly-sized invisible layer is laid down in exact registration with the underlying bitmap original, a search request by the user, to highlight the word 'scanner' (say), delivers the bounding box of that string in the invisible layer, little realizing that the exact registration properties result in the highlight illuminating the bitmap version of the word in the overlaid bitmap layer.

Regrettably this impressive OCR capability is only of limited use to us in the present project. Less than 5% of the pages in the original thesis are in what might be called "plain English." For this small subset, the good performance of the Pica 10 font under OCR led to a set of pages that simply needed a few proofing corrections and the insertion of some *troff* markup.

The rest of the pages were all to some degree highly mathematical and the OCR engine, lacking any AI expertise in Dennis-authored material, could not cope with dense strings of what it saw as anomalously long non-existent words.

Accordingly, one of us (DFB) spent many hours with successive pages of the bitmap original displayed in a left-hand workstation window, while he worked out (and typed into a text-editor window) line-by-line, *eqn* coding to mimic what was on display. BWK did a great deal of proofreading, and occasionally contributed advice on how to use *eqn* to create exotica like multi-line braces that appear on many pages, such as those in Figure 7. Figure 8 shows the same display created with *eqn*.

$$\left.\begin{array}{ll} I_1: & \text{LOOP(n)}\ \ X \\ I_2: & \text{LOOP(n)}\ \ X \\ \vdots & \vdots \\ I_x: & \text{LOOP(n)}\ \ X \end{array}\right\} x$$

$$\underset{\sim}{Q}$$

$$\left.\begin{array}{ll} I_{\hat{e}}: & \text{END} \\ \vdots & \vdots \\ I_{\hat{e}+x-2}: & \text{END} \\ I_{\hat{e}+x-1}: & \text{END} \end{array}\right\} x$$

Figure 7: Hand-drawn braces

5. Two Challenges

We now take a detailed look at two of the problems encountered in bringing Dennis's material into the modern era.

5.1. Center Justification Revisited

We have already pointed out that although font characters in Pica 10 were of fixed width, Dennis seemingly had the ability to center-justify a column of Roman numerals around the mid-point of the longest of the strings. At the very minimum this would seem to require a horizontal half-space capability. Figure 9 shows what we are up against. The longest string is `(viii)`, which has an even number of characters and so the center of the string lies between the `(vi` and `ii)` sub-strings.

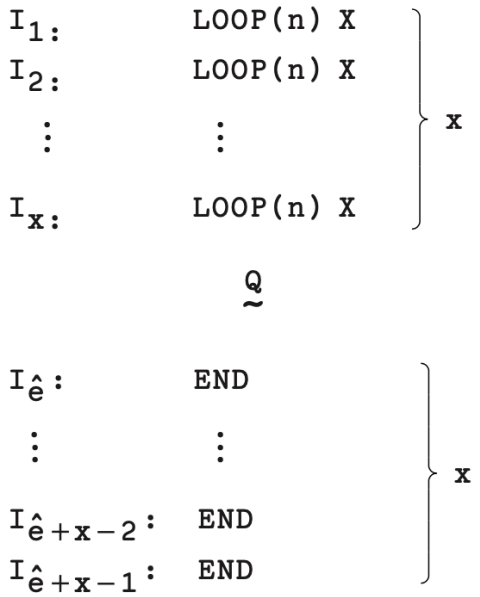


Figure 8: Braces produced with eqn

Now consider the shortest string, which is `(i)`. The mid-point of this string aligns with the center point of the letter `i`. Visual inspection shows that, somehow, Dennis has managed to get the alignment very close to perfect.

The task is to achieve the same effect using the *troff* typesetting programs. There is no macro in the standard *ms* macro package that can achieve this effect and *eqn* has no facility either. This may well be because any implementation of the effect requires look-ahead to find the longest string in the grouping followed by calculation of its mid-point.

Given that all the characters in the Pica 10 font are of fixed width it is possible to pad out the strings with thin spaces to make everything line up for center alignment. But ideally it would be better to have a more general facility that could cope with text strings in variable-width fonts.

Fortunately Michael Lesk's *tbl* program [6] for typesetting tables provides the necessary facility. Figure 9 may be thought of as a two-column table. The left-hand column contains the Roman numeral strings that are to be centered. The next column is a left-justified set of text strings which define the assumptions inherent in the overall Lemma.

When coping with the left column the *tbl* preprocessor uses the fact that low-level commands in *troff* itself enable text string lengths to be computed accurately, in machine units provided the individual character widths are known. So *tbl*'s task is to emit code that causes *troff* to measure the width of all strings and also to emit low-level interleaved spacing code for *troff* to obey, that will cause the entire left-hand column to be center-justified.

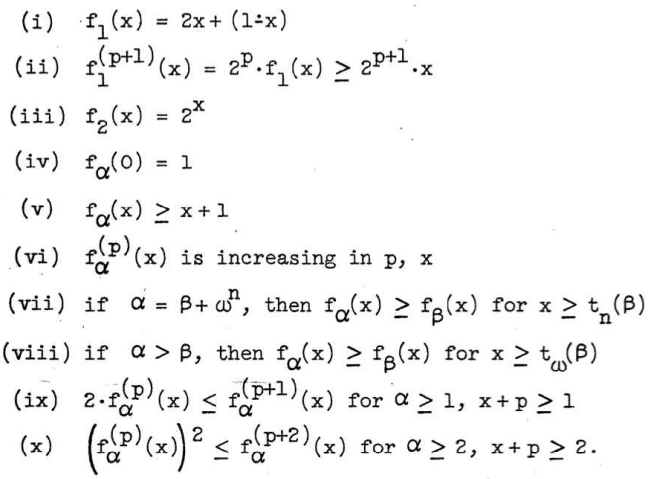


Figure 9: Original centered roman numerals

Figure 10 shows that center-justification of the entire left-hand column has indeed taken place.

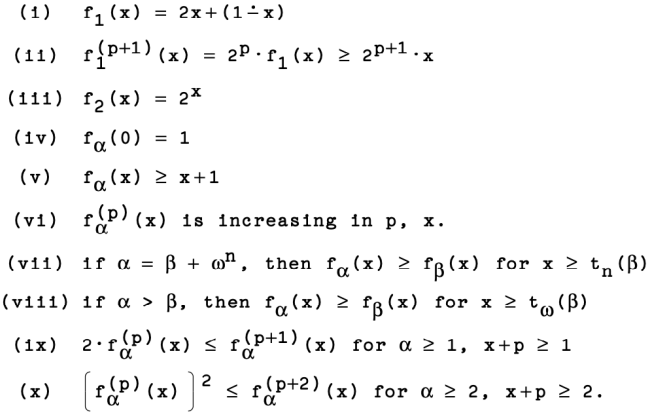


Figure 10: Figure 9 with Bitstream Pica 10 and tbl

5.2. Line Drawings

Our second example where we have allowed ourselves the luxury of 'anachronistic' improvements is in the diagram of Figure 12.5 in the thesis. Figure 12.5, shown here as Figure 12, is a complex box-and-line diagram showing the way that different varieties of loop programs are inter-related.

In the early 1970s devices such as the 2741 or a Teletype Model 37 could be driven from a computer using *nroff*. Dennis would have been well aware that such diagrams could only be drawn with hyphens or underscores for horizontal lines, together with sequences of bar | symbols for vertical lines. The results from these techniques were never elegant. For all these reasons we can conjecture that he thought it better to place his character symbols on the page by careful measurement and dead-reckoning, leaving the

boxes and lines (sometimes dotted or dashed) to be drawn in by hand.

Figure 12 shows the hand-boxed original page where the various box and line elements have clearly been hand-drawn, around carefully placed characters in a script font. Many of the script characters are not exactly elegant. Clearly Dennis had problems finding golf-balls for all the characters he needed.

We used *pic* [7] to create a version of the figure, Figure 13. It can place elements not just at absolute positions but also at positions *relative* to box and line boundaries, and other places within a diagram. So as not to stray too far from the constraints on the original material we did use absolute coordinates to place the original lines, boxes, etc.

6. Conclusion

It seems that the more we try to figure out how Dennis produced his thesis, the less certain we are in any conclusions we might draw. For example, the typed material is remarkably precisely laid out on the page, matching a grid almost perfectly. How did he (or some typist) manage to sustain such precision for 180 pages, with endless sequences of subscripts on superscripts? How did he manage the fractional spacing, especially horizontally, where so far as we know, the devices of the day did not provide a mechanism.

How did he deal with the multitude of characters—Greek, Fraktur, mathematical symbols, script letters—without losing track of their positions on the page. And as a weird aside, why are there two versions of the digit "4", one with a closed top and one with an open top, shown in Figure 11?

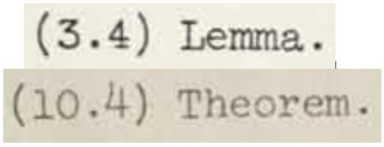


Figure 11: Two versions of the digit 4

How did he draw all the brackets around the vertical constructs of loop programs? The braces are not perfect—clearly they have been done by hand—but they are very precise. How did he manage the sole large diagram, Figure 12.5, with its intricate patterns of dashed and dotted lines? How did he manage hyphenation? How did he manage to keep pages all approximately the same height?

And so on: more questions than answers. We have greatly enjoyed working on this project and we feel that there is a lot of mileage left in it yet.

It is in many ways a blessing that Dennis's thesis did go missing for so long. Any attempt at early restoration, in the period from 1968 to 1985, would have had to use the newly developed tools such as *troff*, *eqn*, *tbl* and *pic* to confect a version of the document that was suitable for an electromechanical device such as a daisywheel printer or even a 2741. It was not until the mid-1980s that economical laser printers at decent resolution became widely available.

Most important was the development of PostScript by Adobe in 1984, followed by its more declarative form, PDF, in 1989. PDF has now become the *de facto* standard for interchanging visually complex material such as Dennis's thesis. And along with PostScript and PDF came font technologies such as Type1, TrueType and Open Type, which made possible our starting point of the Pica 10 font as a faithful representation of what was originally available on engraved golf-ball elements.

Of course Dennis worked at Bell Labs from 1967 through to his death in 2011 and he was certainly aware of (and a user of) all the Unix-based text processing tools that we have mentioned in this paper. Sadly, whatever happened in 1968 seems to have killed off any chance of him using these tools to have another go at rebuilding and submitting his thesis.

Despite the important result from his research that loop programs provide another Turing-complete computability model, it was still the case that working with Ken Thompson on the Unix operating system and the C compiler was much more in line with what he wanted to do in his professional career. Although Dennis did not get his Harvard doctorate, we should be thankful that this did not stop him from a lifetime of amazing contributions to practical computing.

Acknowledgements

Katie LaSeur of TheCreativeFold.com was enormously helpful with preparation of figures for this paper and for dmrthesis.net. We are also grateful to Steve Bagley, Chuck Bigelow, David Brock, Stu Feldman, Mike Fischer, Harry Lewis, Doug McIlroy, Sean Riley, John Ritchie, Jerry Saltzer and Tom Van Vleck for generously sharing their expertise, experience, and memories.

References

[1] David C. Brock, Discovering Dennis Ritchie's Lost Dissertation, https://computerhistory.org/blog/discovering-dennis-ritchies-lost-dissertation, June 2020.

[2] William A. Ritchie, Dennis Ritchie's "missing" PhD thesis. https://dmrthesis.net

[3] Brian W. Kernighan and Lorinda L. Cherry, "A System for Typesetting Mathematics," Communications of the ACM, vol. 18, no. 3, p. 151-157, 1975.

[4] Steven R. Bagley, David F. Brailsford, and Brian W. Kernighan, "Revisiting a Summer Vacation: Digital Restoration and Typesetter Forensics," in Proceedings of the ACM Symposium on Document Engineering (DocEng13), p. 3-12, ACM Press, 10-13 September 2013. DOI: 10.1145/2494266.2494275

[5] Repository of material for this paper: www.cs.princeton.edu/~bwk/dmr

[6] M. E. Lesk, Tbl—A Program to Format Tables, 1976. Bell Labs memorandum

[7] B. W. Kernighan, "PIC—A Language for Typesetting Graphics," Software--Practice and Experience, vol. 12, no. 1, p. 1-21, January, 1982.

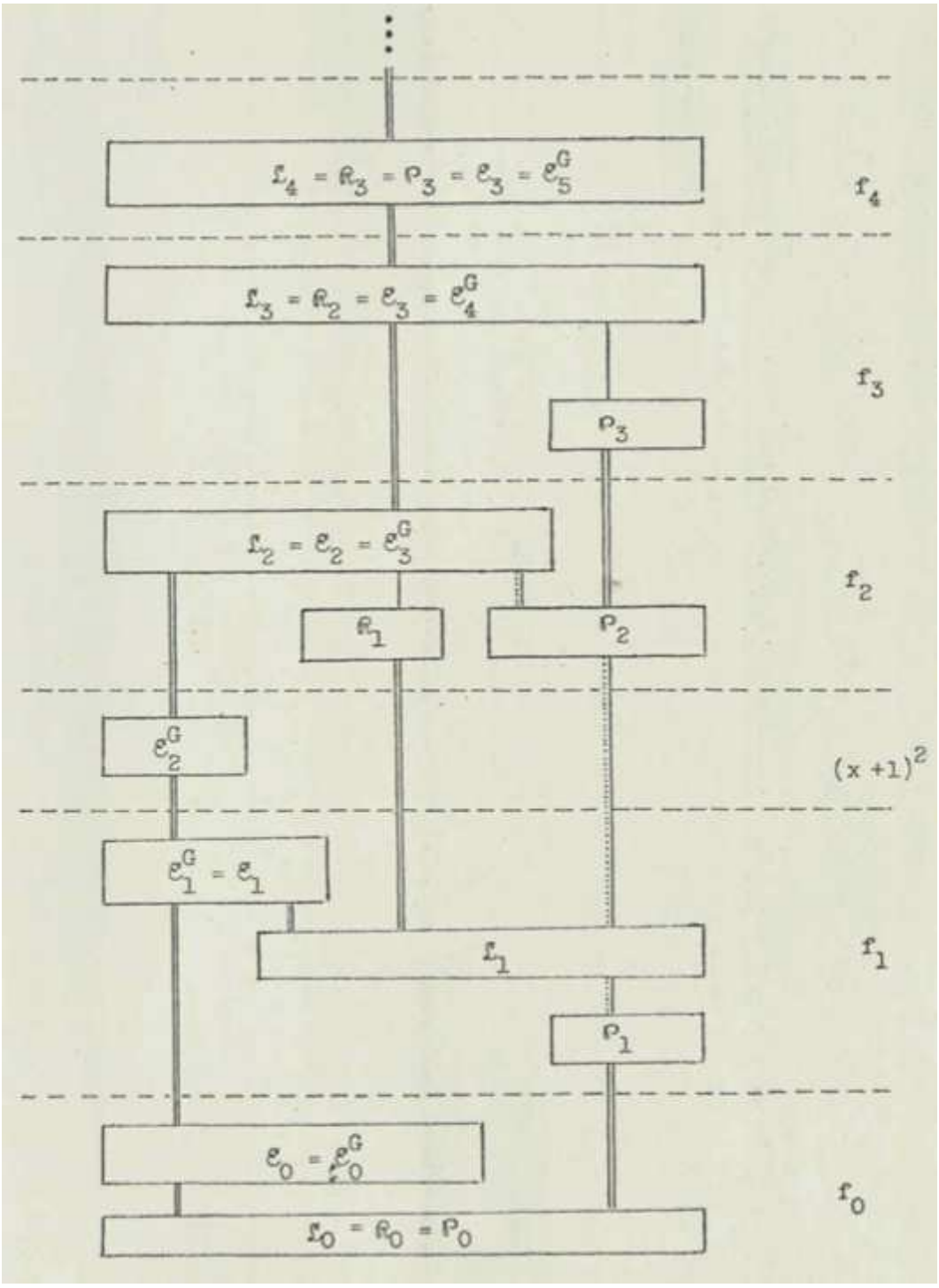


Figure 12: Original Figure 12.5

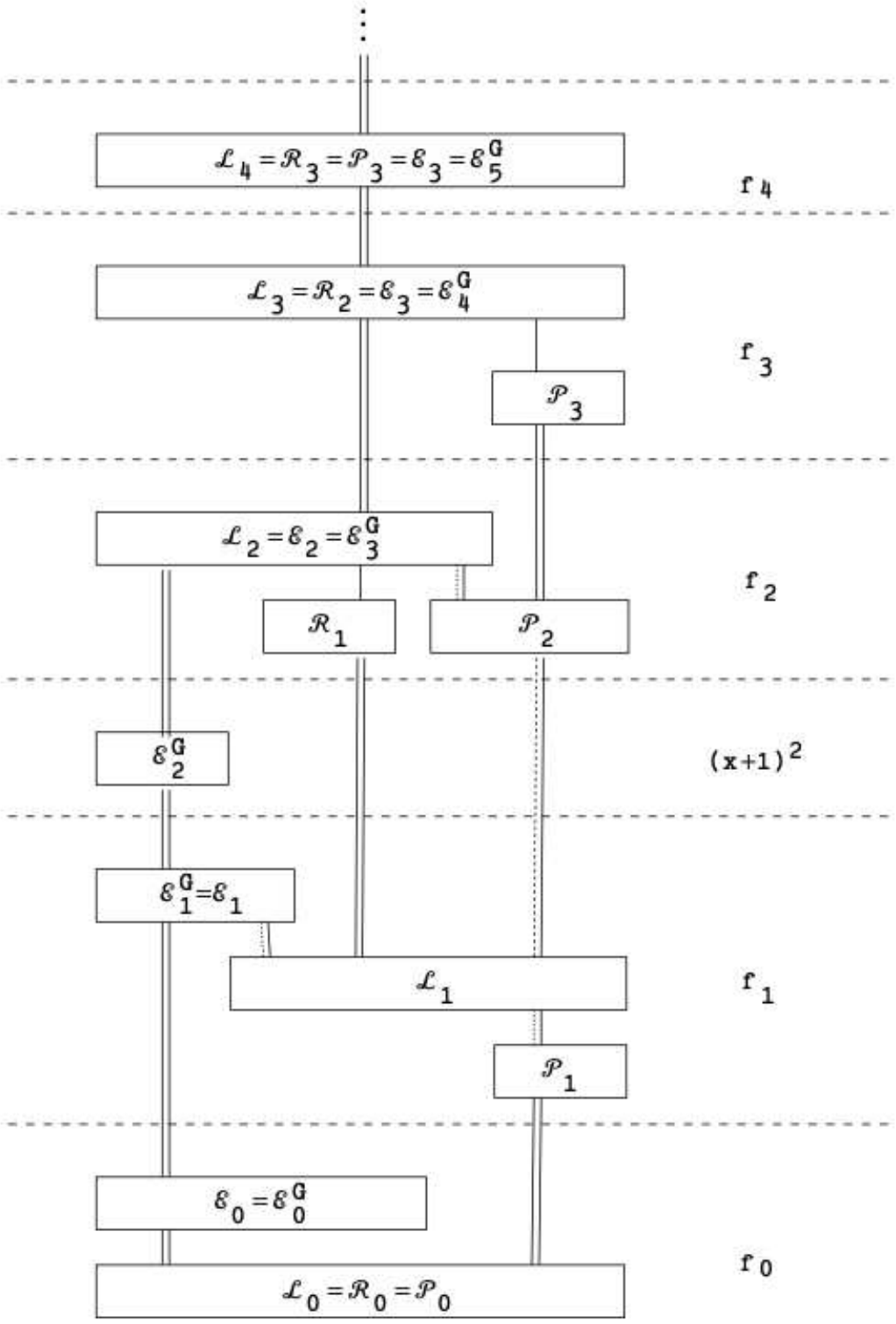


Figure 13: Pic version of Figure 12.5