The Knowledge Engineering Review, Vol. 19:2, 181–186.
Printed in the United Kingdom

Book Reviews

Digital Typography by Donald E. Knuth, CSLI Publications, 685 pp., US $89.95, ISBN 1-57586-010-4 and **Pioneers of Modern Typography**, revised paperback edition by Herbert Spencer, MIT Press, 158 pp., US $29.95, ISBN 0-262-69303-8
doi:10.1017/S0269888904210207

In 1977, unhappy with the appearance of the galley proofs of the revised edition of Volume 2 of his *The Art of Computer Programming*, Donald Knuth began the development of the typesetting program TEX and its close associate, the font generation program METAFONT. At the time he started out, Knuth was beginning a year's sabbatical, and apparently thought that he could complete the necessary work within that time. He was wrong. The task ended up taking 'a substantial portion of my life' (Knuth, 1999a) and put back work on the complete *The Art of Computer Programming* by around 10 years.[1]

Much of the story of those years is told in *Digital Typography*, which collects many of Knuth's previously published essays on aspects of TEX and METAFONT, as well as a few previously unpublished fragments, including the draft report that Knuth wrote on the night of the 12–13 May 1977, laying out his initial ideas for the development of TEX.

These are fascinating essays, even if one does not have a deep interest in typography *per se*. One reason for this is that all of us now have to do a good deal of typesetting, in a way that would have been unheard of only a few years ago. Throughout my academic career I have taken it for granted that I would typeset my own work, but then I was still an undergraduate when TEX became widely available. By the time I became a graduate student, it was already common practice to write directly on a computer rather than writing by hand and typing up a finished draft, though I had much older colleagues in that period who still gave handwritten manuscripts to their secretary for typing. This change is partly because we all now have easy access to computers, of course, but I believe in part it is also because:

> If you let others do your typesetting, then there will be errors beyond your control; if you do your own, then you only have yourself to blame. (Aho *et al*., 1975, p. 740) (quoted in Knuth (1999c)).

Anyhow, the fact that we are all regularly engaged in typesetting means that reflections on a typesetting system that many of us use, by the author of that system, are almost bound to be interesting.

Another reason that these essays are so intriguing is that Knuth is such an engaging writer. This is clear from any of his work, but the fact that *Digital Typography* is such a hodge-podge of incunabula (including, naturally, *TEX Incunabula*) greatly magnifies the proportion of the kind of random trivia that makes his writing so engaging (the recipes in *Macros for Jill* for example, the photograph from the class outing in *A Course on* METAFONT *Programming*, or the list of early books typeset in TEX in *TEX Incunabula*).

However, the main reason why the essays in *Digital Typography* are engaging for me is the close link between the aims of digital typography as embodied in TEX and METAFONT, and the aims of the field of Artificial Intelligence.

At first glance that may seem like a wild statement, but I believe it holds up to closer examination. Underlying Knuth's whole program was a belief that mathematics has something to

[1] Though, as Knuth (1999b) says, the 10 years are an investment in a system which should save 6 or 7 years work during the writing of the rest of *The Art of Computer Programming*.

contribute to typography, that it is possible to capture the essence of the design of letters mathematically—not just in a way that describes a single letter, but in a way that describes families of letters in groups of fonts. Indeed, the very concept of a meta-font, which is at the heart of Knuth's approach, is that whole fonts are just particular parameterizations of a meta-font that itself describes all the fonts in a given family. In other words, a meta-font captures the very essence of a particular family of fonts. In fact, the concept of a meta-font goes even deeper. It is a paramaterization of the very notion of a font, in the sense that every font produced by METAFONT is a parametrization of a meta-font. To create a meta-font, then, is to isolate that part of the intelligence of a type-designer which identifies the fonthood of specific designs—a non-trivial piece of pattern recognition of the 'without-thinking' kind that has proved elusive in Artificial Intelligence.

It is not clear that Knuth would agree with this analysis, though his remark that the art of letter design will not be fully understood until it can be explained to a computer (Knuth, 1999d, p. 291) shows he has gone some way down this path. However, the concept of a meta-font is very close to Hofstadter's (1998) work on fluid concepts, especially the attempt to characterize what it is that defines a specific letter (Hofstadter & McGraw, 1998). Hofstadter's argument is that identifying such concepts is fundamental to human intelligence, and I agree. Furthermore, I believe that meta-fonts are exactly such concepts. (Note, though, that Hofstadter's work is an attempt to do this identification by machine—as he points out in (Hofstadter & McGraw, 1998) it is the fact that METAFONT has to be told exactly what to do that distinguishes his work from Knuth's.)

Digital Typography describes work that was revolutionary in the sense that it changed the nature of type. In contrast, *Pioneers of Modern Typography*, a new paperback edition of a design classic, is largely about revolutionary ways to *use* type. The bulk of the work it surveys uses fairly conventional type, but lays it out in revolutionary ways.[2] These revolutionary layouts range from the poetry of Apollinaire (in which the words of 'Il Pleut' rain down the page) through Werkman's commemoration of Lenin's death (in which the word **LENIN** is dwarfed by high-rise towers of giant stacked Ms), to Rodchenko's constructivist photomontages. These are all landmark pieces of graphic design, but they can also be viewed as a natural extension of the kind of typography that Knuth has spent so long building tools for.[3]

Knuth (1999d, pp. 303–305) develops an analogy between font design and musical composition (and, equally, between languages that allow the description of fonts and musical notation). He uses this to discuss the idea of meta-music (in which setting parameters could change the mood from pathos to excitement in the same meta-composition), before arguing that there is a fundamental difference between font design and musical composition because:

> alphabets are not symphonies; an alphabet is a 'medium' while a symphony is a 'message'.

This is true—the typographic analogy to symphonies are artistic works executed using alphabets. In other words, typographic symphonies are precisely the kinds of design in *Pioneers of Modern Typography*. The arrangement of the type on the page, which could be done using $\TeX$ and METAFONT, is what creates the artwork, in the same way that it is the arrangement of the notes, using musical notation, that creates a symphony. What closes the loop so neatly is the fact that $\TeX$ and METAFONT have such a flexible notion of 'font' and 'character' that they would have been the perfect tools for the artists celebrated by *Pioneers of Modern Typography*—just compare the photo-montage by Klustis (Spencer, 1983, p. 51) and the example using the `lisa300` font (in which characters are fragments of an image of the Mona Lisa) (Knuth, 1999e, p. 429).

[2] There is, for example, far more variation of type in Figure 1 on p. 299 of *Digital Typography* than in most of the works in *Pioneers of Modern Typography*.

[3] Though they were created well before—much of the material in *Pioneers of Modern Typography* appeared 50 years before Knuth started work on $\TeX$.

Of course, the successors to the artists celebrated in *Pioneers of Modern Typography* don't use T_EX and METAFONT—they use tools like Photoshop. But that is a shame. T_EX and METAFONT are certainly harder to get the hang of to begin with, and might even be more arduous to use in the long term, but the possibilities are larger as well. Indeed, I think there is a whole new field—'computational graphic design' maybe—to be explored at the intersection of computer science and graphic design, which is concerned with designs that are driven both by artistic intuition and what is computationally and mathematically possible. Knuth asks, rhetorically:

> . . . does anybody really need a $6\frac{1}{7}$th-point font that is one fourth of the way between Baskerville and Helvetica?

Well, they might not *need* it, but it certainly might be part of an interesting piece of art when combined with some of the other things that T_EX and METAFONT are capable of.

To my mind, this is where METAFONT connects with work on 'artistic intelligence'. In Cohen's work on AARON, a computer program that creates art, his role, as McCorduck (1990) argues, is that of a *meta-artist*. Cohen provides a degree of control of the creative process, delineating the scope of the program, while allowing free creative reign. One part of AARON's knowledge is concerned with how to draw, and that is the part that corresponds to METAFONT. Another part of AARON's knowledge is concerned with the composition of pictures, and that is the kind of knowledge that one would need to add to METAFONT in order to create the kind of work we see in *Pioneers of Modern Typography*. With the addition of this knowledge, one might be able to produce a virtual visual artist to challenge artistic creativity of AARON.

Reviewed by Simon Parsons
Brooklyn College, City University of New York, USA

References

Aho, AV, Johnson, SC and Ullman, JD, 1975, ''Typesetting by ACM considered harmful'' *Communications of the ACM* **18** 740.

Hofstadter, D, 1998, *Fluid Concepts and Creative Analogies: Computer Models of the Fundamental Mechanisms of Thought*, London: Penguin Books.

Hofstadter, D and McGraw, G, 1998, ''Letter spirit: Esthetic perception and creative play in the rich microcosm of the Roman alphabet'', in *Fluid Concepts and Creative Analogies: Computer Models of the Fundamental Mechanisms of Thought*, London: Penguin Books, ch. 10, pp. 407–466.

Knuth, DE, 1999a, ''The future of T_EX and METAFONT'' in *Digital Typography (CSLI Lecture Notes, no. 78), Stanford, CA: CSLI Publications, ch. 30, pp. 571–572.*

Knuth, DE, 1999b, ''Questions and answers I'' in *Digital Typography (CSLI Lecture Notes, no. 78)*, Stanford, CA: CSLI Publications, ch. 31, pp. 573–600.

Knuth, DE, 1999c, ''Mathematical typography'', in *Digital Typography (CSLI Lecture Notes, no. 78)*, Stanford, CA: CSLI Publications, ch. 2, pp. 19–65.

Knuth, DE, 1999d, ''The concept of a meta-font'' in *Digital Typography (CSLI Lecture Notes, no. 78)*, Stanford, CA: CSLI Publications, ch. 15, pp. 289–313.

Knuth, DE, 1999e, *Digital Typography (CSLI Lecture Notes, no. 78)*, Stanford, CA: CSLI Publications.

McCorduck, P, 1990, *Aaron's Code: Meta-Art, Artificial Intelligence and the Work of Harold Cohen*, WH Freeman & Co.

Spencer, H, *Pioneers of Modern Typography*, revised paperback edition. Cambridge, MA: MIT Press (1st edn 1969, 2nd edn 1982, 1st paperback edn 1983).

Principles of Data Mining by David J. Hand, Heikki Mannila and Padhraic Smyth, MIT Press, 546 pp., £34.50, ISBN 0-262-08290-X
doi:10.1017/S0269888904220203

Data mining is a topic of growing importance. While scientists have always been gathering and analysing data—this is, after all, how science proceeds—new kinds of scientific endeavour, like the

human genome project, are now providing us with much more data to gather and analyse than we have ever had before. In addition, the trend to ever greater computerization of our lives, combined with the falling cost of electronic storage media,[4] means that many more organizations, like supermarket and hospital operators, are generating and recording huge volumes of data which they have an interest in analysing. Supermarket operators, for example, would like to identify purchase patterns that they can exploit to sell us more groceries. Hospital operators, for example, would like to identify patterns that identify medical errors, so that they can improve treatment. Data mining is the mechanism by which these patterns can be spotted, and the importance of the field is growing alongside the volume of stored data.

While the term 'data mining' is relatively recent, techniques for identifying patterns in data are much older. Indeed, one can argue that the entire field of statistics is concerned with exactly that. Whether or not one could win the argument, what is undeniable is that there are many techniques from statistics and other fields that can be used by data miners, and clearly it would be a great help to data miners to have a book that collects many of these techniques, explains the principles behind them, and identifies what kinds of task they may be used for. *Principles of Data Mining* is such a book, and if providing this collection was the only thing it did, it would be a book well worth buying. However, *Principles of Data Mining* does even more—as the authors say in the introduction (stating it much better than I could):

> Rather than discuss specific data mining applications ... we have instead focused on the underlying theory and algorithms that provide the 'glue' for such applications.

The glue is a combination of statistical theory and algorithms for searching and optimizing. In my opinion, this makes *Principles of Data Mining* close to essential for anyone who is entering the field of data mining (one can obviously get by without it, but life will be much harder).

The structure of the book can be neatly illustrated by its handling of the EM algorithm. Broadly speaking, this is an algorithm that is useful to data miners when they have missing data—that is there are some variables in the dataset whose values are unobserved in some cases. Such missing data can make it difficult to apply techniques that optimize some scoring function of interest to the data miner (for example, the likelihood of two products being bought by the same shopper, in the case of a supermarket operator, or the likelihood of a patient with a given condition being the subject of a medical error, in the case of a hospital operator). EM is a mechanism for getting around this problem, providing the expected best score.

EM is first introduced towards the end of the second part of the book (on p. 260 to be precise), the second part of the book being concerned with the fundamental statistical approaches to data mining (the first part of the book gives a general introduction to the business of data mining, measurement, visualization and inference). This material on EM provides the statistical theory, but stops short of showing what precisely EM can be used for, or providing an algorithmic interpretation of the statistical theory. In order to find out more, one has to read on into the third part of the book, where the techniques introduced in the second part are applied to specific data mining tasks. Here (on p. 281) one can find a description of how to apply EM to building mixture models, and EM crops up again (on p. 316) as a technique for clustering.

This kind of approach, which I think is exactly the right way to go about dealing with a subject, means that *Principles of Data Mining* is not a book that one can pick up, dip into, and quickly find the solution to whatever data mining problem one has. Instead it is a book that needs to be carefully worked through—sections are too tightly bound together and build on their predecessors too carefully for any other approach to be possible. If you want a quick guide to data mining techniques, *Principles of Data Mining* is not the book for you. But if you want to really understand the different approaches and the relationship between them, then this is the book to read.

Reviewed by Simon Parsons
Brooklyn College, City University of New York, USA

Genetic Algorithms—Principles and Perspectives, A Guide to GA Theory by Colin R. Reeves and Jonathan E. Rowe, Kluwer Academic Publishers, 332 pp., $120.00, ISBN 1-4020-7240-6
doi:10.1017/S026988890423020X

Unlike some specialist books, Genetic Algorithms—Principles and Perspectives (GAPP), does not fall into the common trap of expounding at length the authors' own work. While the authors certainly do include mention of their own research, GAPP is a very balanced read. This is nicely conveyed by the subtitle, *A Guide to GA Theory*. All the theoretical topics one would expect are included and are well described in a non-emotional style. We start with topics such as selection intensity. Then relate this to the speed at which every bit string in the population becomes identical (or nearly so). Various diversity maintenance schemes, which prevent or slow such convergence, are then described. Later chapters explain, without undue mathematics, schema theorems, Markov models, Walsh analysis, deception and the ever popular gray versus binary coding, No Free Lunch (NFL) and building block controversies. In addition many specialist GAs are summarized. It is nice to have them and their associated theory gathered together in one place. In addition to the references (and index) at the back of the book, each chapter finishes with a small *Bibliographic Notes* section, which criticizes the literature covered by the chapter and (for the unsatiated) indicates further reading. Many of us may delay this on first reading. Instead these notes may be more helpful when using GAPP as a reference work. To this end, it is a book which every self-respecting computer science library should have.

The emphasis is entirely on using genetic algorithms for optimization. Related evolutionary computation techniques (evolution strategies, genetic programming, etc.) get little mention. (There are already specialists books on the theory of these areas (Bäck, 1996; Langdon & Poli, 2002).) Similarly GAPP does not cover non-optimization uses of evolutionary computation such as artificial life, evolutionary art, design, invention or co-evolution.

Later in the book the authors also analyse population-based search both as a dynamical system and by using statistical mechanics and other approximation schemes. Essentially, like experimental design, these hope to capture the bulk behaviour of the population with mathematical models that use low-order cumulants (such as the mean and variance) and ignore higher-order interactions. This is similar to approximations successfully used in other fields, cf. Fourier analysis and Principle Component Analysis (PCA). However, while ignoring high frequencies or 'higher-order effects' may make the maths feasible, it is still difficult to predict even the average path evolution will take. Chapters 6 and 7 are not for the faint hearted.

In keeping with GAPP's tradition of completeness, an appendix is included to define many of the toy problems so beloved by the GA community. Also each of the main chapters (except Chapter 4 on NFL) includes a page or so of exercises. The answers are not included, either in the back of the book or, as far as I could see, on the Internet.

Drs Reeve and Rowe are academics with complementary but highly mathematical skills. The down side of this is that their book is mathematical. While the prose explains the topics well, it is reinforced with a liberal sprinkling of mathematical symbols which are not defined. For example, are you happy with $f{:}X \mapsto Y \subset \mathbb{R}$? If the sight of it and similar hieroglyphics offends your eye, then this is not the book for you.

This is definitely a book for the specialist. While including exercises might suggest it was aimed at students, it is too expensive for most students. However, teachers of evolutionary computing courses will find the exercises useful.

GAs have been widely used for some time. It's good that theory is catching up. But as Jon says 'You can think up seven new algorithms before breakfast. And then spend a life time analysing just one of them'. Section 10.4 *The Impact of Theory on Practice* explains why theory matters, including giving practical guidance to cover cases where people have previously misused GAs. However, even today GA theory does not say how to build a GA but *Genetic Algorithms—Principles and Perspectives* holds what is known.

Reviewed by W. B. Langdon
Computer Science, University College, Gower Street, London, WC1E 6BT, UK

References

Bäck, T, 1996, *Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms*, New York: Oxford University Press.
Langdon, WB and Poli, R, 2002, *Foundations of Genetic Programming*, Springer-Verlag.