MATHEMATICS AND COMPUTING - THE GREAT DIVORCE

Timothy Murphy

Software is a branch of logic, and therefore of mathematics.

A computer program is a sequence of propositions, precisely analogous to a proof in geometry or the differential calculus. It is true that these propositions are metaphors for changes in the internal state of a real or imaginary machine; but equally, a theorem in geometry (say) carries a metaphorical allusion to a real or imagined world.

Computing is the child of mathematics. The fore-runners of computing - Pascal, Babbage, von Neumann - were all drawn from a mathematical background; as were the founders of theoretical computing - Gödel, Turing, Church.

(How amused, incidentally, these patriarchs would have been to hear themselves described as "computer scientists". It would have seemed as absurd to them as speaking of Gauss as a "number scientist".

The term "science", as has often been remarked, is only attached to a subject if its proponents entertain secret doubts about its scientific respectability. Books are written on "Astrological Science"; no-one speaks of "Astronomical Science" - the word "Astronomy" being, so to speak, its own guarantee. Sociologists are scientists; social scientists, in general, are not. *Lux quod non lucendo.)*

What went wrong? What caused the Great Divorce? And is it too late for a reconciliation?

Certainly the linking of software and hardware, under the name of computer science, cannot be regarded as more than a temporary convenience. The connection between, say, the complexity of algorithms and the technique for depositing ions in a semiconductor is tenuous, to say the least; and it is just as absurd to require the student of one to study the other as it would be to link algebraic geometry and surveying, or to expect a student of fluid mechanics to be *au fait* with the latest advances in plumbing.

There is no doubt that the fault lay, historically, with Mathematics. Most mathematicians affected - and many still do affect - to despise computers and all their works. Only a Turing could see past the feeble machines of his day to the profound theoretical concepts beyond.

Any yet. When all that is said, it remains puzzling why the concept of computability, for example, has not been absorbed into the main-stream of mathematics. After all, it affects every branch of the subject. Most mathematicians are aware, however dimly, that the problems they are sweating over may in fact be insoluble. The ghost of Gödel haunts us all, however far our interests may lie from the computer.

There may be a purely technical explanation (at least in part) why theoretical computing remains so much a "foreign "body" in the corpus of mathematics.

Computing is necessarily concerned with "partial" functions

$$f:X \to Y,$$

ie functions which are defined everywhere on their domain X - and which are therefore not *functions* in the strict mathematical sense. For the computer program may never finish. And what is more, Turing has shown that one cannot simply set, say, $f(x) = 0$ in this case without destroying the computable character of f; for there is no way (in general) to determine

whether or not the computation will end in a particular case, other than by carrying it out.

Although this use of partial functions may seem a small matter, it did nevertheless set computing on a course away from the mathematical consensus, which was striking a more and more puritanical attitude towards the singularities of functions. If Cantor's insights were to be exploited to their full potential, it seemed essential that the concepts of set and map (or function) should be rigorously defined, and restricted to those definitions. The careless schoolboy notion of function which regarded cos(x) and log(x) as siblings, simply caused too many headaches when complex constructs were at issue.

Scott's Data Types

If that is so - and it may be an oversimplification, or at any rate of less significance than is suggested here - then Scott's concept of a Data Type may mark an important step towards reconciliation between Mathematics and Computing [1].

For Scott has, in effect, supplied computing with its own category - the category of Data Types - which is, one might say, the ultimate accolade of mathematical respectability.

We can take Scott's concept in two steps.

Firstly, he sidesteps the partiality of $f:X \to Y$ by passing to the extended function

$$F:2^X \to 2^Y$$

where 2^X denotes the set of subsets of X, and we extend f to F by setting

$$F(S) = \{f(x):x \in S\}$$

for each subset $S \in X$. If f(x) is undefined for a particular $x \in S$ then no value for f(x) is included on the right-hand side; and in particular $F(\{x\}) = \phi$.

This has the incidental but important advantage of allowing *non-deterministic* computations. For if we allow that

$$f(x) = F(\{x\})$$

may be empty, then we may equally well go in the opposite direction, by allowing f(x) to be multi-valued, i.e. by allowing

$$|F(\{x\})| > 1,$$

where $|S|$ denotes the cardinal number of S.

This fits in very neatly with the recent preoccupation of both theoretical and practical computing with parallel processing, and the interaction of different computers linked in some way, e.g. across a network. For in that case it is impractical to suppose that the behaviour of a particular computer is entirely predictable. Even if we knew what message would reach the computer, we cannot be certain of the exact instant at which the message will arrive; and so an element of indeterminacy is necessarily injected into our calculations.

However, Scott noted - and this was the second step towards his concept of a data type - that it would go too far to allow *any* function

$$F:2^X \to 2^Y.$$

There are simply too many of them. He observed that the functions F arising from (possibly partial) functions $f:X \to Y$ are order-preserving, i.e.

$$S \subset T \Rightarrow F(S) \subset F(T).$$

In fact they possess a stronger property:

$$S = \cup S_i \Rightarrow F(S) = \cup F(S_i).$$

The neatest way of expressing this condition - and one that leads naturally towards generalization - is to impose a topology on 2^X, and to restrict F to *continuous* functions

$$F:2^X \to 2^Y.$$

The appropriate topology turns out to be the Tychonoff product topology on 2^X, where however we take the so-called Sierpinski topology on the factor spaces `2'. To be precise, if we take the set B = {⊥,T} in place of `2' then the topology is that defined by taking as open sets the 3 subsets

$$\emptyset,\{T\},B.$$

A Data Type D, then, is a topological space, albeit of a rather special kind. It turns out that "special kind" can be most simply defined in the starkest of mathematical terms: a data type D is an *injective object* in the category of T-spaces.

That is a convenient point to break the story. Scott has presented computing with its own category, the category of Data Types (and continuous maps). This gives a new unity and discipline to what has tended to be a fragmentary and disparate subject. Such diverse subjects as Computability, Coding Theory, Algorithmic Complexity and Semantics can at least be discussed in a common language; and that language is one familiar to mathematicians at least.

Computing has come home; it is once again in the mainstream of mathematical thought.

References

1. SCOTT, Dana.
"Data Types as Lattices", *SIAM J. Computing*, 5 (1976) 522-587.

School of Mathematics, Trinity College, Dublin 2.

"CURRENT MATHEMATICAL PUBLICATIONS" AS A RESEARCH TOOL IN THE MATHEMATICAL SCIENCES

Anthony Karel Seda

By the end of this calendar year (1984), it is expected that some 46,000 books and papers will have been reviewed in Zentralblatt für Mathematik. The corresponding number for Mathematical Reviews is about 40,000. Put in a more accessible way, these figures mean that a research worker in Partial Differential Equations (Mathematical Review Code 35), Numerical Analysis (65), Statistics (62) or Global Analysis (58) is faced with a flow of papers and books of about, or in excess of, 50* per week. A worker in Computer Science (68), Functional Analysis (46), Operator Theory (47), Mathematical Logic and Foundations (03) or Combinatorics (05) is faced with a flow of about 30* papers and books per week. Necessarily, therefore, time being limited, such a worker will be highly selective with regard to his or her choice of material for in-depth study. But making such a choice requires an awareness of the current literature, or "current awareness" in library jargon. Now there are several sources of current awareness, but only one offering anything significantly beyond current awareness, and this is Current Mathematical Publications (CMP) published by the American Mathematical Society.

Conversations with several of my colleagues and others have, surprisingly, led me to believe that the usefulness of CMP is not as widely appreciated as it might be. Certainly its value purely from the current awareness point of view is clearly recognised, but it has certain other merits which do not seem to be common knowledge. In this note I want to comment, as a regular and fairly longstanding user, on the effect-

* These figures are based on a simple count of entries appearing in this year's issues of CMP, taking into account cross-referencing and secondary classification.