# Repaying our debts.

To whom it may concern!

There is a wide-spread belief that, whenever a new area of human endeavour has been shown to be amenable to mathematical treatment, the rest of mathematics got eventually enriched by that development. It seems likely, and today I don't feel inclined to challenge the belief. But it raises a question.

Over the last 25 years, and at an increasing pace during the last 10 or so, programming has become "an area of human endeavour amenable to mathematical treatment": the art —or black magic!— of programming has evolved into a scientific discipline. In its relation to the rest of mathematics, computing science has so far been largely at the receiving end. The question I would like to raise is: "Has the time come to repay our debts?".

Is programming as a mathematical activity sufficiently novel? In other words, do we have any currency in which to repay our debts? Though aware of the danger of being accused of myopia, I venture an affirmative answer. I could give that affirmative answer more or less ex cathedra: the programming challenge being like nothing mathematicians have ever faced before, the mathematics that grew to meet it must be sufficiently unusual. But such an answer is, of course, only convincing when we preach to the already converted. Slightly more convincing is perhaps a public confession: when I saw Hoare's first correctness proof of a program, I declared most emphatically that such a ballet of symbols was <u>not</u> my cup of tea! I have rarely been so wrong; several years later I was dancing the ballet myself, and I liked it.

If I needed a catch-phrase to capture what seems novel, I would come up with something like "problems of scale in a formal environment". Coping mathematically with the programming problem obviously implies that we regard the programming language we use as some sort of formal system, and each program we consider as some sort of formal object. As long as we ignore problems of scale, there is nothing novel in that approach, for it would be rather similar to what all sorts of logicians do. But programs are often big! Admittedly, most formal experiments have been carried out with rather small programs, but sizes are increasing, and how to push the barrier still further is becoming an explicitly stated research topic.

I am perfectly willing to endorse the opinion that most vigorous mathematics results from a subtle balance between applications of formal techniques and of common sense. It is that view of mathematics that may explain some of programming's mathematical "novelty". In the previous paragraph I have explained the intrinsically formal nature of the programming activity; on the other hand, the programmer's environment —the modern computer— truly deserves the predicate "general purpose". The tension between the two might explain my feeling that the balance between formal techniques and common sense is more critical in programming than in many other areas of mathematics.

Over the last ten years, the predicate calculus became our tool for daily reasoning, its use gradually replacing more and more of the usual verbal arguments. This development was quite clearly forced upon us by the nature of the subject matter we had to deal with; the limited use the usual mathematician makes of the predicate calculus seems an indication of the "novelty" of the programmer's task.

A more wide-spread adoption of the predicate calculus in the rest of mathematics would be a very natural candidate for computing science's influence. I would consider that influence as very wholesome. Firstly, it seems a very effective way of cleaning up the presentation of many an argument. (We did experiments with a few examples, more or less chosen by accident, and each time the improvement was considerable.) Secondly, the discipline is purifying: formalization works as "an early-warning system" when things are getting contorted. Thirdly, it clarifies, since it reveals all sorts of mathematical distinctions - such as "proof by infinite regress" - as artefacts of an in retrospect unfortunate formulation of the argument.

In the long run its influence could be very profound indeed. With sufficient familiarity and sufficient manipulative agility the formal argument becomes the most concise way of capturing our "understanding": it becomes the way in which we prefer to "understand". Eventually it may invalidate the widely-held dogma of the unteachability of thinking.

In the above we need not be deterred by the predictable objection from mathematicians of the more conservative persuasion, who will point out that in most of existing mathematics the logical structure of the arguments is not intricate enough to warrant their formalization, since, even if their observation is correct, there remains the question of the chicken and the egg.

* * *

Another channel through which computing science could exert a potentially profound influence on the rest of mathematics is the computing scientist's attitude towards notational conventions. In the early days, when large sections of the computing community considered the constraints of the limited character set of the punched card sacrosanct, we have committed the most terrible sins, and —I am sorry to say— limited printing facilities of modern terminals seem nowadays to invite similar sinful behaviour.

That physical austerity has not been totally unwholesome. With $B$ a predicate on the natural numbers, I denote quantification by

$$(\underline{E}\, x: 0 \leq x < X: B\, x) \quad \text{and} \quad (\underline{A}\, x: 0 \leq x < X: B\, x) \quad .$$

Historically, the use of underlining for the creation of new characters has been inspired by the constraints of the typewriter (and of the Flexowriter in particular). But even in handwriting I prefer the $\underline{E}$ and the $\underline{A}$ over the $\exists$ and the $\forall$ respectively. It can immediately be generalized to

$$(\underline{N}\, x: 0 \leq x < X: B\, x) \qquad (1)$$

for the number of distinct values $x$ in the range $0 \leq x < X$, such that $B\, x$ holds. (This is just to illustrate that the "upside-down convention" is an unfortunate one.)

The recursive definition of (1) is left as an exercise for the reader. The underlined "$\underline{N}$" in (1) is, however, only a minor issue: it has been included because
a) I could not work without it myself, and
b) I never encountered it in the mathematical literature.

I did encounter in the literature —even several times— something else, viz. the remark that it is very clumsy to express formally that the integer sequence $A[0..N-1]$ is a permutation of the integer sequence $B[0..N-1]$. But, using (1), one can do so very easily! I could not escape the impression that, thanks to computing science, a somewhat different tradition of using symbolism has emerged.

*   *   *

Here is another observation. I think the average mathematician is not as keen as he could be to avoid avoidable case analyses. Let me give you one simple example from Ross Honsberger's "Mathematical Morsels." The example is interesting, since the copied page has been taken from a book in a series, explicitly devoted to "the purpose of furthering the ideal of excellence in mathematical exposition."

Compare the alternative exposition. Consider a rectangular array of 7 rows of 3 points each. In each row one of the colours dominates. Select _four_ rows in which the same colour dominates and select in each of those two points of the dominant colour. At least two of such latter selections occurred in the same pair of columns, since a pair of columns can only be selected in _three_ different ways. The conclusion follows.

Honsberger's "say red" and his final three-case analysis have completely disappeared. (So have his P's, Q's, and R's; and all the subscripts.) In a comparable degree of verbosity the alternative exposition is only half as long. (The example has been chosen by coincidence: I encountered the problem when I had already started on this text.)

## COLORING THE PLANE*

Suppose each point of the plane is colored red or blue. Show that some rectangle has its vertices all the same color.

**Solution:**

Any set of 7 points must contain at least 4 which are the same color. Of 7 points on a line, then, we must have 4 collinear points $P_1, P_2, P_3, P_4$ which are all the same color, say red. If these points are projected onto two other lines parallel to the first, two collinear quadruples of points $(Q_1, Q_2, Q_3, Q_4)$,$(R_1, R_2, R_3, R_4)$ are obtain-

| | | |
|---|---|---|
| $P_1\cdot$ | $Q_1\cdot$ | $R_1\cdot$ |
| $P_2\cdot$ | $Q_2\cdot$ | $R_2\cdot$ |
| $P_3\cdot$ | $Q_3\cdot$ | $R_3\cdot$ |
| $P_4\cdot$ | $Q_4\cdot$ | $R_4\cdot$ |

ed which determine several rectangles among themselves and others with the $P_i$. Now, if any 2 of the $Q$'s are red, an all red rectangle $P_iP_jQ_jQ_i$ results. Similarly for 2 red $R$'s. If neither of these cases hold, then some 3 (or more) of the $Q$'s and some 3 (or more) of the $R$'s must be blue. But these trios of blue points cannot avoid being lined up so that a pair of each trio face each other to yield an all blue rectangle. The conclusion follows.

Observe that the result is valid for any region of the plane which contains all the points inside some circle, no matter how small. Essentially this question occurred on the 5th U.S. Mathematics Olympiad, Spring 1976.

*Pi Mu Epsilon, Vol. 3, 1959-64, p. 474, Problem 138, proposed by David Silverman, Beverly Hills, California, solved by John E. Ferguson, Oregon State University.

People with a programming background are more alert at trying to avoid case analyses than the other mathematicians, who don't seem to mind so much. I made this observation in the mid-seventies - during my, at the time numerous, travels - and it has never amazed me: as a programmer you know only too well that you have to "pay" for each case analysis as reflected by an alternative construct "$\underline{\text{if}}$ ...... $[]$ ...... $\underline{\text{fi}}$" in your program, and the analogy between program structure and proof structure is too obvious to be missed. The tendency to avoid avoidable nomenclature (such as Honsberger's P's, Q's, and R's) is in all probability spin-off of the same influence.

* * *

The discrete nature of much of computing's subject matter has quite naturally increased the rôle of mathematical induction. First an observation. In order to prove for some predicate $P$,

$$(\underline{A}\, n: n \geq 0 : P\, n) \qquad (2)$$

a proof by mathematical induction proves

$$(\underline{A}\, n: n \geq 0 : P'\, n) \qquad (3)$$

with $P'$ given by

$$(P'\, n) = (P\, n) \vee (\underline{E}\, i: 0 \leq i < n : \neg(P\, i)) \qquad (4)$$

Since (3) is of the same structure as (2), why not prove it by mathematical induction? I.e. why not prove (3) by proving

$$(\underline{A}\, n : n \geq 0 : \mathcal{P}''\, n)$$

with — as in (4) — $\mathcal{P}''$ defined by

$$(\mathcal{P}''\, n) = (\mathcal{P}'\, n) \vee (\underline{E}\, j : 0 \leq j < n : \neg(\mathcal{P}'\, j)) \qquad ?$$

By substituting (4) into the right-hand side of the definition of $\mathcal{P}''$, however, we deduce $\mathcal{P}'' = \mathcal{P}'$. In other words, we can apply the trick only once, and the decision to conduct a proof by means of mathematical induction is idempotent. Is this observation important? I don't know, but from a methodological point of view I think it is illuminating; it is in any case very fundamental. Admittedly my sample was small: I showed this observation to about a dozen mature mathematicians, and learned that it was new — even a surprise! — for <u>all</u> of them. In my more optimistic moments such an experience makes me believe that a greater methodological awareness could be one of the currencies in which computing science could repay its debts.

More in particular, I believe that mathematics in general could benefit from more explicit applications of mathematical induction. By way of example

I refer the reader to my article in Acta Informatica (Jan. 1980), which derives by means of mathematical induction over n the multiplicity of a prime factor p in n! ; this inductive proof is much simpler than the traditional argument - see, for instance, D.E.Knuth's "The Art of Computer Programming", Vol. I - which relies on the shrewd rearrangement of factors. (Eventually, it even needs the summation formula for the geometric series!)

Note. That traditional argument worries me very much since I met an otherwise gifted mathematician that particularly liked it precisely for its shrewdness. (End of Note.)

* * *

Assuming, for the sake of the argument, that the usual style of mathematical reasoning leaves considerable room for improvement, we may raise the question "Does it matter?". I think it does.

I remember vividly how I showed to one of my colleagues in the Department how we had learned to deal formally and intellectually in a satisfactory manner with all sorts of little algorithms. My colleague wasn't impressed at all and only asked "But does this enable you to find new algorithms?" I was shocked by that reaction.

Firstly, because I would have answered in full confidence "Of course it will.", hadn't it been for the fact that I could answer truthfully "Yes, it has already done so.".

Secondly, because he allowed the "quod" to prevail so strongly over the "quo modo". This utilitarian view of science is very much en vogue today. I am afraid that it is the prevailing view among today's mathematicians, but I have a lurking suspicion that exactly that attitude is mainly responsible for the fact that mathematics is no longer regarded as an integral part of our culture and —to quote Morris Kline— "Indeed, ignorance of mathematics has attained the status of a social grace."

I am daily faced with the havoc created by the wide-spread profound ignorance of what mathematics is about. By the cultural isolation of mathematics we all lose, and mathematicians had better remain aware that Elegance is the very core of their business! (And, please, remember that this remark is made by a pragmatic industrial mathematician.)

Plataanstraat 5
5671 AL NUENEN
The Netherlands

1 December 1980
prof.dr. Edsger W. Dijkstra
Burroughs Research Fellow.