# A misguided educational effort.

The other day I received a nice problem, which is a generalization of a problem I once invented. (See EWD570 and EWD578.) For $n \geq 1$ the function $f(n)$ is given by

$$\begin{array}{lll} f(1) = 1 & & (0) \\ f(2k) = f(k) & \text{for } k \geq 1 & (1) \\ f(2k+1) = h(f(k), f(k+1)) & \text{"} & (2) \end{array}$$

The problem is to design an efficient algorithm for the computation of $f(N)$ when no special properties of the integer function $h$ —such as symmetry or associativity— are given.

From (1) and (2) we observe that the $f$-values for the consecutive argument pair $(2k, 2k+1)$ are expressed in those of the consecutive argument pair $(k, k+1)$. Because

$$f(2k+2) = f(k+1) \qquad , \qquad (1')$$

also the $f$-values for the consecutive argument pair $(2k+1, 2k+2)$ are expressed in those of $(k, k+1)$. (Note that the above observations are not "surprising": they are about the only thing inspection of (1) and (2) can teach us.)

The above observations suggest to carry out the computation using pairs of $f$-values corresponding to pairs of consecutive arguments.

More precisely, with $(x,y)$ standing for a pair of consecutive arguments, i.e.

$$x+1 = y \qquad (4)$$

and with $(p,q)$ standing for the pair of corresponding $f$-values, i.e.

$$p = f(x) \wedge q = f(y) \qquad (5)$$

it is suggested to use $(4) \wedge (5)$ as invariant relation. It can be established for the smallest values of the arguments — $(x,y) = (1,2)$ — ; when we can hit the argument $N$ by repeatedly "doubling" the pair of consecutive arguments $(x,y)$ under invariance of (5), we have solved the problem. Because the initial value of $y$ is already greater than $N$'s minimum value, we shall try to hit $N$ by $x$, i.e. to establish $x = N$.

The repeated "doubling" of the argument pair $(x,y)$ suggests the introduction of a variable, $r$ say, whose value is restricted to powers of 2 ; the guiding of the "doubling" process requires two-sided control, which we can express by enclosing $N$ in a shrinking interval. In short, we propose

$$(\underline{E}\, i: i \geq 0: r = 2^i) \wedge r*x \leq N < r*y \qquad (6)$$

as our next invariant.

The above suggestions lead <u>directly</u> to the following program:

```
x, y, p, q := 1, 2, 1, 1 ;
r := 1 ; do N ≥ r*y → r := 2*r od {(4) ∧ (5) ∧ (6)} ;
do r ≠ 1 →
     r := r/2 ;
     if r*(x+y) ≤ N →
            x, y, p := x+y, 2*y, h(p,q)
     ▯ N < r*(x+y) →
            x, y, q := 2*x, x+y, h(p,q)
     fi {(4) ∧ (5) ∧ (6)}
od {(4) ∧ (5) ∧ (6) ∧ r=1, hence p=f(N)}
```

Storing r.x and r.y as a and b respectively, we can code the same algorithm alternatively

```
a, b, p, q := 1, 2, 1, 1 ;
do N ≥ b → a, b := 2*a, 2*b od {a ≤ N < b} ;
do a+1 ≠ b →
     k := (a+b)/2 ;
     if k ≤ N → a, p := k, h(p,q)
     ▯ N < k → b, q := k, h(p,q)
     fi
od {p = f(N)}
```

and we cannot help recognizing our dear old friend, the Binary Search.

In the above, the heuristics, though standard, have been fully spelled out for pedagogic reasons. Finding the above algorithm should not take a well-trained programmer very much time, nor much more paper than the proverbial backside of an envelope.

* *
*

There exists a school (I wouldn't call it "a school of thought") that believes in "teaching by example" and in "discovery by example". I don't. I concluded EWD376, which describes in detail the actual steps in which I had solved a problem from graph theory, with:

> "Finally we draw attention to the fact that we did not need a single example to explain what we were talking about or (even worse!) to discover what the program should do. And this, of course, is as it should be."

From a colleague, who had written to me in 1970 "Somehow you seem to ignore (or attempt to ignore) what is known about the psychology of invention.", the quoted paragraph evoked in 1974 the comment "The final remark looks to me like rubbish. But highly indicative of the author's attitude!". (Curiously enough -for he is a conscientious fellow- he did not take the trouble to reconcile his characterization "rubbish" with the fact that his own solution of the same problem was vastly inferior to mine.)

When we have to deal with a large set, inspection of individual members is usually a waste of time; when the inspected subset is not "representative", the inspection of individual members can even be profoundly misleading. Since, eventually, we have to deal with the whole set, thereby appealing to its definition, it is as a rule much more effective to analyse that definition directly. For the sake of effectiveness, one tries —as always!— to avoid exploding case analyses like the plague; inspecting individual members is almost the extreme of what one should learn to avoid. (Whenever you hear the traditional programmer's excuse for a bug: "Oh, but that was a very special case!", someone reveals he hasn't been trained to think effectively.)

* * *

The little problem mentioned and solved above occurs in a paper, whose main topic is apparently programming methodology. I haven't seen the rest of the paper, but it must be terrible.

Drawn as trees, $f(4)$ through $f(16)$ are enumerated. The text continues with "We shall not use now our standard generalization procedure since it asks for many examples and would become a little cumbersome.". Then the trees are re-drawn, with left and right leaves labelled differently; comparing $f(n)$ with $f(2^3+n)$ for $4 \leq n < 7$, the authors formulate a hypothesis about $f(x)$ and $f(x-2^i)$ when "$2^i$ is the greatest power of 2 contained in $x$."

After the confession: "The reader may feel that this induction hypothesis comes from too little information and would be right (we actually used traces for $1 \leq x \leq 128$)", they inspect a few more labelled trees and come up with yet another "induction hypothesis".

Finally they produce a very clumsy formulation of the solution given above (expressed in terms of such delightful functions as the length of the train of leading ones in the binary representation of x). That final clumsiness of their solution is, of course, no accident: it faithfully reflects the twisted route via which the solution had been reached.

This last observation gives me a second reason for objecting to such primitive heuristics: besides making problem solving a laborious and not very effective process, they lead —if they lead to anything at all— to inferior solutions. The observation invalidates the rather Jesuitical standpoint that, in heuristics, any approach is as good as any other as long as it leads you to an idea how to solve a problem.

Almost needless to say, the authors present (in 1980!) their solution without even a shadow of a correctness proof; "poor man's induction" is deemed sufficient.

* * *

That even at major universities such primitive heuristics are taught and recommended is depressing. I have no reason to suspect that the authors don't teach their methodology with the best of intentions and in good faith. But the faith is wrong, despite its perpetuation by the psychologists and the educationists. I know that they try to observe and describe how "people" solve and learn, but what is the value of such experiments? Even the vast majority of mathematicians should be regarded as amateur thinkers; in view of the "human material" used for such experiments, we cannot expect the psychologists and educationists to observe anything other than bad habits. It is our responsibility not to raise them to standards. I think we had, indeed, better "attempt to ignore what is known about the psychology of invention". Most of it "looks to me like rubbish".

Plataanstraat 5
5671 AL NUENEN
the Netherlands

31 October 1980
prof.dr. Edsger W. Dijkstra
Burroughs Research Fellow.