A political pamphlet from the Middle Ages.

This note concerns a very ugly paper [1]. Its authors seem to claim that trying to prove the correctness of programs is a futile effort and, therefore, a bad idea. To quote from the opening sentence:

> "program verification [...] is bound to fail in its primary purpose: to dramatically increase one's confidence in the correct functioning of a particular piece of software".

As rendered above, this statement is obviously wrong: we all know of cases in which our confidence in the correct functioning of a particular piece of software has been increased dramatically by a proof of its correctness.

The style of the paper is revealed by what I omitted: in the place indicated by "[...]" they had written "as perceived by a large segment of the computer science community". They give a complete caricature of program verification --suggesting, for instance, that on the average one line of program requires 66 (sic!) lines of formal correctness proof-- , accuse without substantiation that "a large segment of the computer science community" accepts this nonsense as a fact of life, and then try to sell the great message that the computer science community has been misguided. That is what I call the style of a political pamphlet.

They argue (rightly) that communication between mathematicians is an essential ingredient of our mathematical culture; they conclude that proofs of program correctness, not being communicated among colleagues, are therefore no good. How do they know that these proofs are not communicated and subjected to the judgement of others? Simply by painting them as so long, ugly, and boring that they are , now almost by definition, not fit for communication. They just ignore that how to prove --not in the silly ways they depict, but more elegantly-- "the correct functioning of particular pieces of software" is the subject of a lively interchange of experiences between scientists active in the field. In short: again the unsubstantiated accusation that is characteristic for political pamphlets.

They argue (rightly) that long formal proofs are unconvincing, and subsequently discredit formal techniques, with an appeal to the work of Albert Meyer, by remarking that "For even the most trivial mathematical theories, there are simple statements whose finite proofs are impossibly long." As if that matters! In most mathematical theories there are even simple statements that cannot be proved at all, but who cares? It is the mathematician's task to arrange his arguments in such a fashion that avoidable formal manipulations are, indeed, avoided, and to discover those theorems that do admit a concise proof. They fail to argue why these "simple statements whose finite proofs are impossibly long" should be of any interest, neither do they substantiate their suggestion that the theorems needed for proving program correctness must be so uncomfortably long. (They would have a hard time if they tried, for it is exactly in the area of proving program correctness that I have found formal techniques both indispensable and, when applied with good taste, eminently suitable.) Also such omissions are characteristic for political pamphlets.

They suggest an antagonism between "formal" and "understandable" which is misleading. Eventually a nice formal treatment is always the most concise way of capturing our understanding and the most effective way of conveying the argument with all its convincing power to someone else. (By suggesting this antagonism, they seem to have fallen into the same trap as the author who wrote in his preface "the standard symbols for the logical connectives have been avoided for the sake of clarity", unaware as he was that it was precisely for the sake of clarity that these symbols have been invented!) Such misrepresentations are characteristic for political pamphlets.

They don't distinguish between love of perfection and claim of perfection, and blame people for the first by accusing them of the latter.

Besides political, the paper is pre-scientific in the sense that even the shallowest analogy is accepted as justification. Referring to Rabin's algorithm for the probable primality of a large integer, they suggest that most theorems mathematicians work on are as unprovable as the primality of very large integers is untestable, but the only support they provide is obtained by ignoring the difference between Rabin's (mathematical notion of) "probable" and their own (woolly notion of) "believable". The first one, however, has nothing to do with human fallibility, the second one (see their "filters") everything.

They display the same pre-scientific attitude when they argue as if a bridge and a software system were significantly similar objects: reading the text one can only conclude that this opinion has been induced by the verbal similarity between the terms "Mechanical Engineering" and "Software Engineering".

Unaware that the "problems of the *real* world" are those you are left with when you refuse to apply their effective solutions, they confirm the impression of anti-intellectualistic reactionaries by sentences such as "*real* programs deal with *real* human activity and are thus detailed and messy" (their italics and their conclusion!). They boldy postulate that "...the transition between specification and program *must* be left unformalized". It is as if they remembered examples of how programs were formally derived from their specification, for several lines further down, they quote in gratitude from a private communication that "the input assertions for many numerical algorithms are not even formulatable". If the essential properties of these numerical algorithms cannot be formulated, we are left wondering how their usage can ever be justified (but presumably this is explained in the private communication quoted).

* * *

By this time the reader may feel that by not being more specific in my complaints I am committing the same sin as I have accused the authors of. The trouble is that it is very hard to be more specific; their text is slippery --they disagree with "a large segment of the computer science community", but accept the average implementor's attitude when it suits their argument-- ; the text is written in sometimes very poor English --"It is exactly those processes which mediate (sic) proofs of theorems in mathematics that *require* (sic) that..."-- and their arguments are rambling. Supposing that they had something sensible to say we can only regret that they have buried it under so much insinuating verbiage. As it stands it leaves the reader wondering why they have put so much venom in their text, because they seem to have gone much farther than the usual practioner's backlash.

None of the many papers about program verification and derivation that I have written or seen uses APL as a programming vehicle. This could be an accident, it could also be a consequence of the rich expression structure of APL . (They refer to proofs as "...substitutions to be checked with the aid of simple algebraic identities" which, in the case of APL-expressions are perhaps not so simple....) If the latter conjecture is correct, it would explain why APL-addicts might feel unhappy about (or threatened by) modern achievements in proving the correctnes of (non-APL) programs. Does it help the understanding of this paper and its venom to know of the heavy involvement with APL among its authors? We can only guess and have our private opinions.

[1] DeMillo, Richard A., Lipton, Richard J., Perlis, Alan J., "Social Processes and Proofs of Theorems and Programs" , Conference Record of the 4th Symposium on Principles of Programming Languages, Los Angeles, January 17-19, 1977

Plataanstraat 5
5671 AL NUENEN
The Netherlands

prof.dr.Edsger W.Dijkstra
Burroughs Research Fellow