Scientific Disproof of Computer Program Verification

Steven Meyer
Pragmatic C Software Corp., Minneapolis, MN, USA
smeyer@tdl.com

The method of scientific natural philosophy developed by the founders of modern physics and elucidated by the Lakatos-Feyerabend-Kuhn philosophical research programme is used to scientifically disprove computer program verification. Since the author's 1970s paper that used science to criticize the Hoare Dijkstra method was rejected for publication, program verification has been the subject of heated academic debate. DeMillo, Lipton, and Perlis offered a sociological criticism of program verification by claiming that mathematical social processes could not be applied to computer program verification. Fetzer criticized the very idea of program verification using philosophical analysis.

This paper returns to scientific testing of program verification by first showing that all three program verification research programs are based on incorrect assumptions, or at least assumptions that must be scientifically tested. The assumptions include assuming Tarski instead of Finsler definitions of truth and proof, accepting irrational sociology as a valid method of comparing research programs, assuming that algorithms and programs are different, and assuming that probability can determine mathematical truth.

Next two correctness of computation experiments are analyzed: 1) the still open question of the correctness of Heisenberg's 1920 laminar flow analysis versus a mathematical disproof and the later computer simulation support, 2) the inability of the Hoare Dijkstra method to see three value sorting as both two passes of two value sorting and three value distribution at the same time. The paper concludes by advocating establishment of CS departments that study basic questions of computational natural philosophy independent from engineering and business.

Scientific Disproof of Computer Program Verification - Extended Abstract

1. Introduction

Twentieth century physics not only revolutionized science, it also changed scientific method for all time. It replaced both Newtonian physical reality and previous pre-scientific natural philosophy with a new generalized scientific natural philosophy based on testing philosophical concepts using objective scientific experimentation and research programme comparison. It tamed and simplified Kantian phenomenology.[1] Modern scientific natural philosophy offers methods for testing computational as well as physical theories.[2]

Although scientific natural philosophy (sometimes referred to as 'science' below) was well understood by the founders of modern physics, it was not explicitly defined in philosophical terms so that it could be used as a prescriptive theory (meta-theory). The Lakatos-Feyerabend-Kuhn methodology of scientific research programmes (LFK programme) explained and made explicit scientific natural philosophy so that it could be applied outside modern physics. The LFK programme also connected natural philosophy to its philosophical and historical roots.[3]

The author presented a general argument against formalist computer science (CS) at last year's ECAP 2005 conference using arguments from modern scientific natural philosophy.[4] Science is applied in this paper to criticize the three current computer program verification research programmes: Dijkstra Hoare program verification by formal proof,[5] DeMillo, Lipton and Perlis verification using the psychology of social processes,[6] and Fetzer impossibility from inductive/deductive epistemology.[7] The paper argues that scientific testing disproves formal program verification without needing

---

1. Kant, E. *A Critique of Pure Reason*, 2nd edition, 1787. Trans. Guyer, P. and Wood, A. Cambridge Press, 1998.

2. The founders of modern physics explicitly studied method. For example see: Bohr, N. (Sanders J. Ed.) Mathematics and natural philosophy - 1954 NYU lecture, *Essays and Papers*, Vol. 2, 550. Planck, M. *Where is Science Going*, Trans. J. Murphy (New York 1932). Heisenberg, H. *Physics and Philosophy*, Prometheus books, 1958. Einstein, A. What is the theory of relativity?", *Ideas and Opinions*, reprint of 1919 London Times article, New York, Crowne, 1954, 227-232.

3. Meyer, S. "Proposal to teach Lakatos-Feyerabend-Kuhn Philosophy of Science", unpublished, URL: www.tdl.com/~smeyer. The paper uses documents from the Lakatos archive at LSE to show the three philosophers were building one theory to attempt to save rationalism, albeit in a weakened form, from the failure of logical positivism.

4. Meyer, S. "Toward Anti-Formalist Computer Science", E-CAP'05 Abstracts, Computing and Philosophy, p. 27 (Extended abstract and presentation slides at URL www.tdl.com/~smeyer).

5. Dijkstra, E. *A Discipline of Programming*, Prentice Hall, 1976. Also, Hoare, C. Mathematics of programming, *BYTE (August 1986)*, 115-149.

6. DeMillo, R., Lipton, R. and Perlis, A. Social Processes and proofs of theorems and programs. *Comm. ACM 22, 5 (May 1979)*, 271-280.

mathematical axioms, or sociological or philosophical assumptions.

The paper concludes by arguing CS is not mere engineering of information technology, but needs to be a separate discipline protected from engineering and business that uses the methods of natural philosophy to study computation (computational natural philosophy).

2. Program Verification Assumptions Require Scientific Testing

The methodological assumptions of all three program verification research programmes are probably incorrect. This paper considers the assumptions of each research programme in detail. Natural philosophy correctly treats assumption and methodology testing as part of scientific research. Here is a brief list of some of the problematic assumptions:

1. Hoare Dijkstra mathematical verification requires definitions of truth and proof, but NP completeness and Polya's heuristics use computation to question Tarski style definitions of truth.[8]
2. Finsler set theory with its Platonism and rejection of meta-mathematics shows that determining acceptable rules for verification of computation requires scientific testing.[9] This is Lakatos' concept of quasi-empirical mathematics.[10]
3. Obviously, social processes are not a reliable way to verify anything. See MacKenzie's sociological analysis of the program verification wars including the attempt to prevent publication of Fetzer's paper for a reflexive example.[11] Also, see Mirowski's recent discussion of sociology of Tarski's meta-mathematics.[12]
4. Fetzer's distinction between an algorithm and the program that implements it is a question for scientific testing. Imagine how difficult the development of modern physics would have been if philosophers had required physicists to conceptually separate light as waves from light as particles.
5. Fetzer's reliance on the epistemology of inductive arguments (*Ibid.* Fetzer p. 1051) is contradicted by Lakatos criticism of inductive logic.[13]

---

7. Fetzer, J. Program verification: the very idea, *Comm. ACM 31(1988)*, 1048-1063.
8. Amazingly, although Alfred Tarski and George Polya taught together for many years, there is no mention of Polya in a recent biography of Tarski. See Feferman, A. and Feferman, S. *Alfred Tarski: Life and Logic*, Cambridge, 2004. If there is no agreement on even the facts of intellectual history, how can mathematical foundations be anything but conventionalism?
9. Finsler, P. (Booth, D. and Ziegler, R. eds.) *Finsler set theory: Platonism and Circularity.* Birkhauser, 1996. Also see the discussion of Finsler in Breger, H. (Gillies, D. Ed.). *Revolutions in Mathematics.* Oxford, 1995, 249-264.
10. Lakatos, I. *Proofs and Refutations.* Cambridge, 1976.
11. MacKenzie, D. *Mechanizing Proof - Computing, Risk, and Trust.* MIT Press, 2001, 197-281.
12. Mirowski, P. Hoedown in the OK corral: more reflections on the 'social' in current philosophy of science. *Stud. Hist. Phil. Sci. 36 (2005)*, 791-799 especially 795.

6. Why should one assume that the sociology (social processes) of mathematics has anything to to with verification of computations. At minimum a scientific proof is required.[14]

7. Why should one assume certain types of computer program constructs result in better programs? Section 3 contains scientific evidence supporting the exact opposite.

8. Lakatos' argument that the probability of any theory is zero because there are an infinite number of theories without any formal ordering seems to describe computations (as opposed to axiomatized systems) exactly.[15]

3. Two Computational Verification Experiments

Testing the very idea of program verification or comparing various research programmes is easy. Just apply scientific natural philosophy as elaborated by the LFK research programme. Here are two examples:

1. **Heisenberg's analysis of laminar flow contradicts mathematical truth**
   The correctness of a hydro-dynamics computation made by Werner Heisenberg is still open although the physics is well understood.[16] Heisenberg starts the discussion 'I learned more from Bohr than anybody else the new type of theoretical physics which is almost more experimental than mathematics. That is you have to cover the experimental situation by means of concepts which fit.'

   In 1992 Heisenberg wrote a paper on the stability of laminar flow (small oscillations around laminar flows). A year later mathematician Fritz Noether applied a general mathematical theory to show the flow was stable. The proof looked good to everyone including Heisenberg, but Heisenberg believed from his physical intuition (application of a natural philosophical thought experiment) that he was correct.

   In 1950, a pupil of Von Neumann performed a digital computer simulation that produced results close to Heisenberg's. This example shows that formalism can not replace science and shows a need to move to physics based on symbolic calculations since if the assumptions of the numeric simulation were incorrect, or if untested probablistic assumptions were used, Heisenberg's result may still turn out to be wrong.

---

13. Lakatos I. Changes in the problem of inductive logic. *Philosophical papers Vol. 2*, Cambridge, 1978, 128-192.

14. Yandell describes a completely different kind of mathematical social processes in Yandell, B. *The Honors Class - Hilbert's Problems and Their Solvers.* A K Peters, 2002.

15. Lakatos I. *Philosophical papers Vol. 1*, Cambridge, 1978, 110.

16. See transcript of the AHQP (Sources for the History of Quantum Physics) interview of Werner Heisenberg by Thomas Kuhn, Film A 603.2 (2), 11-32.

2. **Dijkstra's verification of the three value sorting problem failed**
Professor Dijkstra took his formal mathematical beliefs so seriously that he described his application of the theory in *A Discipline of Programming.*[17] Dijkstra attempted to apply formal refinement to probably the easiest of the non trivial CS problems that requires sorting tertiary values (3 colored balls) into 3 separate regions. One needs a kind of scientific duality that sees the problem as a two pass binary division or as a process 3 balls at once distribution. Since such duality is not possible with axiomatic refinement, the published solution was either incorrect or extremely efficient. The incorrectness depends on how one parses Dikjstra's English and interprets Dijkstra's claims of efficiency (*Ibid.* Meyer 1983, pp. 6, 11).

4. Conclusion

The scientific bleakness of computer program verification shows the need to establish scientific academic departments to study computational natural philosophy. Unless CS is separated from information technology and business, the most important conceptual questions of our era will never be answered. It is time to return to Thomas Kuhn's 1960 historical analysis: 'historically, science and technology have been relatively independent enterprises', going back as far as classical Greece and Imperial Rome![18]

---

17. Dijkstra, E. *A Discipline of Programming.*, Prentice Hall, 111-116. My unpublished paper 'A Failure of Structured Programming' analyses the methodological anomaly in detail. See Meyer, S. *Pragmatic Versus Structured Computer Programming.*, Unpublished (URL www.tdl.com/~ smeyer), 1983, 4-9. An unscientific defense of the failure follows on pages 10-21. DeMillo, Lipton and Perlis were referees of my paper and were involved in the decision to prevent its publication. Not only did their action eliminate scientific testing of program verification, but they also neglected to include science as an element of their social processes.

18. Kuhn quoted from 1962 National Bureau of Economic Research document in *Ibid.* Mirowski, p. 793.