Journal of Logic and Computation Advance Access published June 16, 2014

Computational logic and the social

URSULA MARTIN, *Department of Computer Science, University of Oxford, OX1 3QD UK.*
E-mail: Ursula.Martin@cs.ox.ac.uk

For Roy Dyckhoff, the most social of computer scientists

Abstract

For centuries, the highest level of mathematics has been seen as an isolated creative activity, to produce a proof for review and acceptance by research peers. Mathematics is now at a remarkable inflexion point, with new technology radically extending the power and limits of individuals. 'Crowdsourcing' pulls together diverse experts to solve problems; symbolic computation tackles huge routine calculations; and computers, using programs designed to verify hardware, check proofs that are just too long and complicated for any human to comprehend. 'Social machines' are new paradigm, identified by Berners-Lee, for viewing a combination of people and computers as a single problem-solving entity. This paper outlines a research agenda for a new vision of a mathematics social machine, a combination of people, computers, and archives to create and apply mathematics, and places it in the context of verification research, computational logic and Roy Dyckhoff's pioneering work on computer proof.

Keywords: Computational logic, proof systems, logic teaching.

1 MacLogic

In 1987 Roy Dyckhoff, with colleagues at the Universities of St Andrews and of Leeds, was awarded a two year grant by the United Kingdom's University Grants Committee through its Computers in Teaching Initiative, to support the Machine Assisted Logic Teaching (MALT) Project 'for the encouragement of computerised logic teaching'. Roy Dyckhoff, working with Bob Hale, Neil Leslie, Brenda Rapley, and Stephen Read created the MacLogic tutoring system.

Maclogic was first deployed for teaching logic to undergraduate students of philosophy at the University of St Andrews. It was taken up widely in the UK and overseas in the 1990s, and though no longer under active development, was still in use for teaching at Rutgers in 2012, thanks to emulation software developed by Branden Fitelson [25]. It also inspired a supporting text book, written by Graeme Forbes of Tulane [26]. MacLogic was highly regarded from the start, winning First Prize in the 1989 Philosophy Software Competition, organised by the Philosophy Documentation Center of Bowling Green State University, and 'recommended by its simplicity' in a 1993 comparative review of eight logic teaching systems as the best software for teaching proof theory [29]: its longevity was in marked contrast to other similar systems [12]. St Andrews also took a broader role through production of a short-lived journal, the '*Computerised Logic Teaching Bulletin*' [11].

The development in the 1980s of desktop PCs and workstations, replacing expensive centralized mainframes, allowed universities to provide dedicated teaching laboratories. In turn this drove initiatives such as the UK's Computers in Teaching Initiative, which ran from 1985 to 1996 [57], supporting national centres for a variety of disciplines, and giving individual academics the chance to develop and share software for discipline specific teaching, in particular exploiting for the first time multimedia and graphical user interfaces. As Dyckhoff noted 'We consider that typing skills, for example, are not the concern of the logic teacher ... . Mice, dialogue boxes and menus provide just the right kind of interface, and we have therefore selected the Apple Macintosh^{TM} as a target machine'

doi:10.1093/logcom/exu036

[22]. Perhaps in part due to an institutional commitment to the Apple Macintosh, St Andrews was something of a pioneer in such computer aided instruction. Dyckhoff's mathematical colleagues, Edmund Robertson and John O'Connor, produced a system called MacTutor to supplement student coursework with examples, graphics and so on: originally distributed as HyperCard stacks, it lives on as a widely used website of nearly 3,000 biographies of mathematicians [48]. Jim Aiton, Susie Whiten and Hania Allen were pioneers of computer-aided instruction for medical students [1]; Ron Morrison and Tony Davie had earlier developed S-algol, a programming language designed to reduce complexity for users, and used in teaching at St Andrews for over 20 years [14]; and Colin Allison and colleagues were pioneers of collaborative learning environments [2].

The student or educator opening the MacLogic manual [46] finds in the very first paragraph a succinct description of what it does.

> MacLogic is a proof assistant for first-order classical, intuitionistic and minimal logic, with or without equality, and either non-modal or modal (S4 or S5). It runs on any Apple Macintosh with at least 2 Mbyte RAM (an old version for 1Mby is available on request.) It works in two modes: as a proof checker and as a proof constructor. The proof checker verifies that a proof is indeed correct by checking it line by line in a 'bottom-up' fashion. Proofs are laid out in the style of E. J. Lemmon's *Beginning Logic*. However, the rules are not Lemmon's, but much closer to Gentzen's original natural deduction systems **NK, NJ** and **NM**. The proof constructor uses a 'top-down' approach, based essentially on Gentzen's sequent calculi such as **LJ**: once a proof has been constructed, it is transformed mechanically into one laid out in Lemmon's style.

As Dyckhoff explains in the manual, the interest in actually creating proofs, brought about by the interest in logics for reasoning about programs and types, was leading at the time to a renewed interest in the work of Gentzen. Natural deduction proofs in the style of books like Lemmon [42], while convenient for meta-logical reasoning about proof, are hard to understand and construct, even for very short proofs. Axiomatic proofs in the Hilbert style are even more forbidding. Gentzen developed Natural Deduction to capture the sort of reasoning typically performed by mathematicians, and the Sequent Calculus as an alternative more suited for constructing proofs: see [60] for the background. Gentzen wrote:

> My starting point was this: The formalization of logical deduction, especially as it has been developed by Frege, Russell, and Hilbert, is rather far removed from the forms of deduction used in practice in mathematical proofs. Considerable formal advantages are achieved in return. In contrast, I intended first to set up a formal system which comes as close as possible to actual reasoning. The result was a '*calculus of natural deduction.*' [27].

MacLogic uses Natural Deduction for checking proofs, and the Sequent Calculus for constructing them, and provides translations between the two. Dyckhoff, whose earlier mathematical work had been in categorical topology [20], [21], credits his work on MacLogic as introducing him to proof theory. Realizing that under certain circumstances one of the search algorithms in an early version of MacLogic did not terminate, he worked up a new algorithm following principles laid down by Vorob'ev [63]: the result is his most highly cited paper [23], which in late 2013 is reported by google scholar to be one of the top ten most highly cited papers in intuitionistic logic.

MacLogic was produced at an exciting time in computer science. Computer assisted proof had formed some of the earliest experiments in artificial intelligence: in 1955 Newell, Shaw and Simon's 'Logic Theorist' program applied their ideas of simulating human problem solving to produce proofs of results taken from Russell and Whitehead's 1911 Principia Mathematica. Herb Simon's

autobiography [56] explains how they simulated the program with a computer constructed of human components: Simon's family and graduate students who acted as subroutines and memories. Once the code was running on the RAND JOHNNIAC[1] computer, Simon wrote to Russell (who died in 1970, aged 97), who replied 'I wish Whitehead and I had known of this possibility before we both wasted ten years doing it by hand. I am quite willing to believe that everything in deductive logic can be done by a machine.' Remarkably, some of Logic Theorist's proofs were new, but Kleene rejected a paper co-authored by Logic Theorist and submitted to the 'Bulletin of Symbolic Logic' as 'not representing a new result' [56].

The notion of reasoning about programs was first presented by von Neumann [28] and Turing [62], and subsequent developments by Floyd, Hoare and others laid the foundations of computer science, and established the strong link with mathematical logic—to quote the artificial intelligence pioneer John McCarthy [45] 'It is reasonable to hope that the relationship between computation and mathematical logic will be as fruitful in the next century as that between analysis and physics in the last.' By the mid-1980s a variety of approaches and software tools, such as the computational logic systems ACL, HOL, NuPrl and Nqthm, had started to be developed for practical reasoning about programs. IBM in particular was an industrial pioneer of formal methods, and of developing software to support them, using logic to describe the expected behaviour of programs, and computational logic systems to show they behaved as expected. Large scale collaboration, while then infeasible, was envisaged, for example in a 1991 account of an early formal methods system, Mural [39]: an Appendix entitled 'The theorem prover's house' (credited to Alan Wills) identified many of the challenges in infrastructure, management and authentication needed to make collaborative computational logic a reality. Jones [40] is a thorough account of the early history of what is now a flourishing academic and industry community, with hardware and software verification to ensure error-free systems is a major endeavour in companies like Intel and Microsoft [34], as well as supporting a number of specialist small companies.

Computational logic systems are now also being used by a small but influential community of mathematicians. A research team coordinated by Tom Hales has almost completed a ten-year formalization of his proof of the Kepler conjecture, using several computational logic systems to confirm his major 1998 paper [33]. In September 2012 Georges Gonthier announced that after six years effort the team that he leads has completed a formalization in the Coq system of one of the most important and longest proofs of 20th century algebra, the 255 page odd-order theorem [30], summarized as

> Number of lines ~ 170 000
> Number of definitions ~ 15 000
> Number of theorems ~ 4 200
> Fun ~ enormous!

The growth of interest in formal methods led in turn led to discussions about the right way to educate computer scientists and software engineers in the new techniques required. By the mid-1980s these ideas had made their way to undergraduate textbooks [40]. Gries's 1991 paper in the house magazine of the leading professional body for computing [32], and Dijkstra's famous paper on 'the cruelty of really teaching computer science' [18], were among many reports and papers of the period calling for computer science education to address not just the ideas of logic and proof, but also the practicalities of doing proofs. Dyckhoff recognises that 'The main problem faced by students is ... that of constructing a proof of a given proposition. ... The key skill to be learnt ... is that of proof development' [22],

[1] Named in honour of von Neumann, and now on show in the Computer History Museum in Palo Alto

and he describes MacLogic as an 'introduction to the kind of proof construction that all computer scientists should do when learning logic, because essentially the same kind of work is involved in programming' [46].

This approach was not uncontested. Some, like Maurice Wilkes, while supportive in principle, were concerned that the costs of verification would outweigh the advantages [66]. There was considerable debate in the mathematical community over the use of a computer (though not for a formal proof) by Appel and Haken [3] to settle the long standing four colour conjecture, with mathematicians arguing that, unlike a hand proof, the computer proof could not realistically be checked, and did not provide insight or understanding. Similar concerns were raised about proofs of programmes: for example in an influential paper in 1979 De Millo Lipton and Perlis [15], argued that 'Mathematical proofs increase our confidence in the truth of mathematical statements only after they have been subjected to the social mechanisms of the mathematical community', expressing concern over 'symbol chauvinism', and suggesting that computer science was too new for it to be wise to prioritize verification to the exclusion of investigating other approaches to reliable software.

The importance of the social context of proof was already identified by Hume, in his 1739 Treatise on Human Nature [37]

> There is no Algebraist nor Mathematician so expert in his science, as to place entire confidence in any truth immediately upon his discovery of it, or regard it as any thing, but a mere probability. Every time he runs over his proofs, his confidence encreases; but still more by the approbation of his friends; and is rais'd to its utmost perfection by the universal assent and applauses of the learned world.

A comprehensive account of these debates in both mathematics and computer science is given by sociologist Donald MacKenzie in his 2001 book 'Mechanizing Proof' [47], where he concludes that used as a prosthesis the computer is benign, but that'trust in the computer cannot entirely replace trust in the human collectivity'.

In the 1980s when MacLogic was developed, the notion of using computers to support the human collectivity was the subject of science fiction—it is a single user program to run on a single machine, and was originally distributed pre-internet by post on a single floppy disk. The rest of this article concerns computer support for mathematical collaboration, in particular the potential of social machines, shedding new light on what Dyckhoff has called the 'interesting art of constructing proofs' [46].

2 Mathematics and social machines

By contrast with the 1980s, when the potential of mathematics and logic in computing was starting to be recognized, mathematics and theoretical computer science research today is recognized as underpinning modern programming languages, secure systems and the Web. Advances depend on hard foundational mathematics, and draw on newer areas such as statistics and dynamical systems, alongside traditional combinatorics and logic, supplemented by simulation and experiment. This potential reach of mathematics is increasing, thus increasing the challenge to researchers of deploying the right combinations of techniques within and beyond their own specialism to solve increasingly hard and broad problems, which are not stated in isolation but require an integrated approach. The rapid pace of technology creates key opportunities, if mathematicians are able to collaborate with each other, and with other computing disciplines, both academic and industry, to produce timely results. Increasing pace and reach has the potential to increase

impact, if potential users of research can find the researchers and the research they need: google scholar suggests there have been nearly 200,000 papers in theoretical computer science in the past ten years.

Newell, Shaw and Simon wanted to build a machine to simulate human problem solving. 'Social machines' are a new paradigm, identified by Berners-Lee in his 1999 book 'Weaving the Web' [5], for viewing a combination of people and computers as a single problem-solving entity. Berners-Lee describes a dream of collaborating through shared knowledge:

> Real life is and must be full of all kinds of social constraint—the very processes from which society arises. Computers can help if we use them to create abstract social machines on the Web: processes in which the people do the creative work and the machine does the administration… The stage is set for an evolutionary growth of new social engines. The ability to create new forms of social process would be given to the world at large, and development would be rapid.

For science, social machines provide platforms for sharing knowledge, leading to innovation, discovery, and commercial opportunity, a perspective set out in Michael Nielsen's 2011 book 'A new kind of science' [53]. For example, Galaxy Zoo allows members of the public to look for features of interest in images of galaxies, and Foldit is an online game where users solve protein folding problems. Future more ambitious social machines will combine social involvement and sophisticated automation, and are now the subject of major research, for example in Southampton's SOCIAM project [58] following an agenda laid out by Hendler and Berners-Lee [35]. This approach builds on e-Science work such as Goble's myExperiment [16], a collaborative research space for scientific workflow management and experiment, widely used in bioinformatics to document and share data, annotations, experiments and results. Bicarregui and colleagues provided a convincing update [6] to 'The theorem prover's house', explaining how the semantic web and the infrastructure of e-science provided the elements needed, should we wish to make collaborative verification a reality.

For centuries, the highest level of mathematics has been seen as an isolated creative activity, to produce a proof for review and acceptance by research peers. The work of cognitive scientists, sociologists, philosophers and the narrative accounts of mathematicians themselves, highlight the paradoxical nature of the production of mathematics—while the goal of mathematics is to discover mathematical truths justified by rigorous argument, mathematical discovery involves 'soft' aspects such as creativity, informal argument, error and analogy. For example, in an interview in 2000 [65] Andrew Wiles describes his 1989 proof of Fermat's theorem in almost mystical terms '... and sometimes I realized that nothing that had ever been done before was any use at all. Then I just had to find something completely new; it's a mystery where that comes from'. Michael Atiyah remarked at a workshop in Edinburgh in 2012 [24]'I make mistakes all the time' and 'I published a theorem in topology. I didn't know why the proof worked, I didn't understand why the theorem was true. This worried me. Years later we generalised it—we looked at not just finite groups, but Lie groups. By the time we'd built up a framework, the theorem was obvious. The original theorem was a special case of this. We got a beautiful theorem and proof'.

While computation is recognized as having transformed other sciences, the use of computation at the highest levels of mathematics has been subject to vigorous discussion, with its role often viewed as minimal—see [49] for a survey of the debate. In particular to take this view ignores the significant role of computing in providing an evidence base that does not necessarily form part of a final rigorous proof. Thus, for example, the computer verification of many cases of the Birch Swinnerton-Dyer conjectures was important for Wiles's later work.

However, today, mathematics seems now at a remarkable inflexion point, with new technology radically extending the power and limits of individuals. Viewing mathematics as a social machine may

be the way to extend this capacity even further, so that in the future, instead of using these techniques in stand-alone fashion, they might be integrated with each other, with access to papers and other archives of mathematical material, and perhaps the means to take account of human creativity and fallibility.

The mathematical community were 'early adopters' of the internet for disseminating papers and sharing data, and recent experiments use the internet to support 'crowdsourcing' (albeit among a highly specialised and educated crowd), for collaboration and the production of mathematics. For example,

- A number of senior mathematicians produce influential and widely read blogs: for example Fields Medallists Sir Tim Gowers has been a leader in an international debate about mathematics publishing. In the summer of 2010 a paper was released plausibly claiming to prove one of the major challenges of theoretical computer science, that $P \neq NP$. It was withdrawn after rapid analysis coordinated by senior scientist-bloggers coordinated from Lipton's blog [44].
- *polymath* collaborative proofs, a new idea led by Gowers, use a wiki for collaboration among mathematicians from different backgrounds and have led to major advances [31]
- discussion fora, including new ideas such as user ratings for finding the right expert, allow rapid informal interaction and problem solving; in three years *mathoverflow.net* has 16,000 users and has hosted 27,000 conversations
- the widely used 'Online Encyclopaedia of Integer Sequences' (OEIS) invokes subtle pattern matching against over 200,000 user-provided sequences on a few digits of input: so for example (3 1 4 1) returns π [54]. Other reference material includes the Dynamic Dictionary of Mathematical Functions [17], which provides interactive tables of elementary and special functions.
- the *arXiv* holds around 750K preprints in computer science, mathematics etc.. By providing open access ahead of journal submission, it has markedly increased the speed of refereeing, widely identified as a bottleneck to the pace of research [13]

All can be viewed within Berners-Lee's framework of social machines, combining human and machine with the machine doing the administration. For example, OEIS involves users with queries or proposed new entries; volunteers curating the system; governance and funding mechanisms through a trust; alongside traditional computer support for a database, matching engine and web interface. In turn OEIS is linked to other mathematical information, such as research papers.

Current challenges raised by the mathematical community, for example see [24], include the importance of collaborative systems that 'think like a mathematician', can handle unstructured approaches such as the use of 'sloppy' natural language, support the exchange of informal knowledge and intuition not recorded in papers, and engage diverse researchers in creative problem solving.

The social is crucial in the production of mathematics, and in recent years a research field of mathematical practice has emerged, drawing together sociologists, philosophers, educationalists, cognitive scientists and mathematicians themselves to understand the production of mathematics, knowledge that in turn can feed into designing better social machines for mathematics. Collaborative systems such as *polymath* contribute to mathematics research, and also provide a rich evidence base for further understanding of mathematical practice. The emergence of forums and blogs will, if used as a serious research tool by mathematicians, provide many more clues as to the way in which they think, since they display and record thought processes in informal mathematics in a completely unprecedented way. This can be used to characterize patterns of reasoning which emerge via social interaction and collaborative mathematics.

For example, our analysis of a *polymath* proof [50] found only 47% of the conversational 'turns' were proof steps, with the rest being made up of conjectures, discussions of examples, and social moves, which, although they did not contribute technical content, were essential to keeping the discussion going. Confirmation was also found of Lakatos's notions of how mathematics advances, for example his analysis of the role of counterexamples and error [41].

3 The next steps

One can imagine many kinds of mathematics social machines, and the kinds of parameters to be considered in thinking about them in a uniform way include, for example: precise versus loose queries and knowledge; human versus machine creativity; specialist or niche users versus general users; logical precision versus cognitive appeal for output; formal versus natural language for interaction; checking versus generating conjectures or proofs; formal versus informal proof; 'evolution' versus 'revolution' for developing new systems; and governance, funding and longevity.

Current social and not-so-social machines occupy many different points in this design space, with Logic Theorist, MacLogic and polymath quite far apart from each other. Each dimension raises broad and enduring challenges, in both traditional logic and semantics, and broader areas such as cognitive science. And yet even such early experiments did not shy away from the difficulties: MacLogic took on board the difference between proof checking and proof generation, and addressed the cognitive appeal of the output through careful decisions about the logic, the layout and even the fonts that the system used.

Exploring such issues in the framework of social machines will include matters such as:

– *Designing social computations* Social machine models [35] view users as entities within the model and allow consideration of social interaction, enactment across the network, engagement and incentivization, and methods of software composition that take into account evolving social aggregation. For mathematics this has far reaching implications—handling known patterns of practice, and enabling others as yet unimagined, as well as handling issues such as error and uncertainty, and variations in user beliefs. Social mathematics shows a variety of communities, interactions and purposes—looking for information, solving problems, clarifying information and so on [50]—displaying much broader interactions than those supported by typical mathematical software. The notions of workflow, core to e-science work such as myExperiment [16], have hardly been explored for mathematics or verification. Solutions might incorporate a variety of artificial intelligence and and cognitive science inspired approaches to 'soft' aspects such as creativity, analogy and concept formation [8]. For example, mathematicians often mention the importance of error for creativity [24]: this has also stimulated Bundy's recent artificial intelligence work on ontology repair [7]
– *Accessing data and information* Semantic web technology enables databases supporting provenance, annotation, citation and sophisticated search. Mathematics data includes papers, records of mathematical objects from systems such as Maple, and scripts from computational logic systems. The International Mathematical Union is leading discussions on the digitization of the mathematical literature [38]. Commercial systems such as Matlab and Maple, or research packages such as GAP, provide significant computational support and archives of data and examples. Successful formal proof efforts, such as the work of Hales and Gonthier, provide rich archives of material if we knew how to use them effectively. There has been considerable research in mathematical knowledge management [43], to support, for example, digitised mathematical archives, search, re-use and executable papers, but current experiments in social

machines for mathematics have little such support. Yet effective search, mining and data re-use would transform both mathematics research and related areas of software verification. Research questions are both technical, for example tracking provenance or ensuring annotation remains timely and correct [10], and social, for example many *mathoverflow* responses cite published work, raising the issue of why users prefer asking *mathoverflow* to using a search engine.

– *Accountability, provenance and trust* Participants in social machines need to be able to trust the processes and data they engage with and share. Key concepts are *provenance*, knowing how data and results have been obtained, which contributes to *accountability*, ensuring that the source of any breakdown in trust can be identified and mitigated [64]. There is a long tradition of openness in mathematical research which has made endeavours like *polymath* or the *arXiv* possible and effective—for example posting drafts on the *arXiv* ahead of journal submission is reported as speeding up refereeing and reducing priority disputes [24]. Trusting mathematical results requires considering provenance of the proof, a major issue in assessing the balance between formal and informal proofs, and the basis for research into proof certificates [51]. Privacy and trust are significant for commercial or government work, where revealing even broad interests may already be a security concern.

There is much to be done, and a substantial body of research lies ahead of us, but the outcomes could transform the nature and production of mathematics. Researchers are already bringing the social to computational logic: for example Tom Hales crowdsourced, with a team of mathematicians in Vietnam, parts of the formal proof of the Kepler conjecture; a team led by Voevedsky collaborated intensively during a semester in Princeton to instigate the new field of homotopy type theory, underpinned by a body of formal proof in Agda [36]; and Jacques Fleuriot and his colleagues in Edinburgh have recently launched the ProofPeer system for collaborative proving [55].

The University of St Andrews, Scotland's oldest University and the third oldest in the English-speaking world, celebrated its 600th birthday in 2013 by, among other things, awarding honorary degrees to Sir Timothy Berners-Lee and Sir Timothy Gowers. One of its 15th-century founders was philosopher Laurence of Lindores, Principal Regent of Pedagogy and Rector and Governor of the University, where according to Baxter [4] 'The system which he professed was notable for its scepticism and empiricism, [and] its preference for the patient interrogation of nature'; as Papal Inquisitor of Heretical Pravity in Scotland he had responsibility for *De heretico comburendo* [19].

In the 20th-century St Andrews was recognized, thanks to the work led by Roy Dyckhoff, as a pioneer in the pedagogy of logic and computer instruction. Roy is renowned among his many friends and collaborators, and the community at large, as the most convivial of computer scientists: an embodiment of computational logic and the social. Many many people have enjoyed constructive, generous and patient conversations about computer science, logic, mathematics and philosophy, and their history, practice and precise presentation, and look forward to many more productive and creative discussions.

References

[1] J. Aiton, S. Whiten, and H. Allen. Student-centred multimedia projects for teaching and learning in pre-clinical medicine. *Medical Teacher*, **18**, 338–340, 1996.

[2] C. Allison and M. Livesey. Coping with concurrency in real time groupware. *Symposium on Experience with Distributed and Multiprocessor Systems* San Diego: USENIX Association, 1993.

[3] K. Appel and W. Haken. The solution of the four-color-map problem. *Scientific American*, **237**, 108–121, 1977.
[4] J. H. Baxter. The philosopher Laurence of Lindores. *The Philosophical Quarterly*, **5** 348–354, 1955.
[5] T. Berners-Lee and M. Fischetti. *Weaving the Web: The Original Design and Ultimate Destiny of the World Wide Web by its Inventor*. Harper, 1999.
[6] J. Bicarregui et al. Return to the theorem prover's house: application of the learning grid to formal methods. In *Proceedings of the Workshop on Teaching Formal Methods: Practice and Experience*, 2003.
[7] A. Bundy et al. Dynamic first-order ontology repair. *J. Semantic Web and Inf Systems*, **3**, 1–35, 2007.
[8] A. Bundy. Automated theorem provers: a practical tool?. *Annals of Mathematics and Artificial Intelligence*, **61** 3–14, 2011.
[9] A. Bundy et al. Reasoning with context in the semantic web. *J. Web Semantics*, **12**, 2012.
[10] J. Cheney et al. Provenance as dependency analysis. *Mathematical Structures in Computer Science*, **21**, 1301—37, 2011.
[11] *Computerised Logic Teaching Bulletin*, 1988-1991.
[12] M. Cohen. Why Bertie and Twootie failed at university, *Times Higher Education*, July 1996 http://www.timeshighereducation.co.uk/features/why-bertie-and-twootie-failed-at-university/99980.article.
[13] J. Crowley et al. Mathematics journals: what may change. *Notices of the AMS*, **58**, 1127–1130, 2011.
[14] A. Davie, R. Morrison, and B. Meek. *Recursive Descent Compiling*, Ellis Horwood, 1981.
[15] R. A. De Millo et al. Social processes and proofs of theorems and programs. *Communications of the ACM*, **22**, 271–28, 1979.
[16] D. De Roure and C. Goble. Software design empowering scientists. *IEEE Software*, **26** 88–95, 2009.
[17] A. Benoit et al. The dynamic dictionary of mathematical functions (DDMF) *ICMS 2010, the Third International Congress on Mathematical Software*, Springer Lecture Notes in Computer Science 6327, 54–69, 2010.
[18] E. W. Dijkstra. On the cruelty of really teaching computing science http://www.cs.utexas.edu/users/EWD/ewd10xx/EWD1036.PDF.
[19] A. I. Dunlop. *Acta Facultatis Artium Universitatis Sanctiandree*. Oliver and Boyd, 1964.
[20] R. Dyckhoff. Total reflections, partial products and hereditary factorisations. *Topology and Its Applications*, **17**, 101–113, 1984.
[21] R. Dyckhoff and W. Tholen. Exponentiable morphisms, partial products and pullback complements. *Journal of Pure and Applied Algebra*, **49**, 103–116, 1987.
[22] R. Dyckhoff. Implementing a simple proof assistant. *Proceedings of the Workshop on Programming for Logic Teaching, Leeds, 6-8 July 1987*, University of Leeds, 1987.
[23] R. Dyckhoff. Contraction-free sequent calculi for iIntuitionistic logic. *Journal of Symbolic Logic*, **57**, 795–807, 1992.
[24] http://events.inf.ed.ac.uk/sicsa-mcp/.
[25] http://fitelson.org.
[26] G. Forbes. *Modern Logic*, Oxford University Press, 1994.
[27] G. Gentzen. Untersuchungen uber das logische Schließen, 1934/5 (English translation in [59]).

[28] H. H. Goldstine and J. von Neumann. Planning and coding of problems for an electronic computing instrument, Report for U.S. Army Ord. Dept. 1947 (republished in [61].)
[29] D. Goldson, S. Reeves, and R. Bornat, A review of several programs for the teaching of logic. *The Computer Journal*, **36**, 373–386, 1993.
[30] G. Gonthier et al. A machine-checked proof of the odd order theorem. *ITP 2013, 4th Conference on Interactive Theorem Proving*, Springer Lecture Notes in Computer Science 7998, 2013.
[31] T. Gowers and M. Nielsen. Massively collaborative mathematics. *Nature*, **461**, 879–881, 2009.
[32] D. Gries. Teaching calculation and discrimination: a more effective curriculum. *Communicatons of the ACM*, **34**, 44–55, 1991.
[33] T. Hales et al. A revision of the proof of the Kepler conjecture. *Discrete and Computational Geometry*, **44**, 2010.
[34] J. Harrison. *Handbook of Practical Logic and Automated Reasoning*. Cambridge University Press, 2009.
[35] J. Hendler and T. Berners-Lee. From the semantic web to social machines: a research challenge for AI. *Artificial Intelligence*, **174**, 156–161, 2010.
[36] Univalent Foundations Project *Homotopy Type Theory* Univalent Foundations Project, 2013.
[37] D. Hume. *Treatise on human nature*. Penguin, 1969.
[38] International Mathematical Union. *Final report of the Future World Heritage Digital Mathematics Library Symposium* 2012, www.mathunion.org.
[39] C. B. Jones et al. *mural: A Formal Development Support System*. Springer 1991. (Appendix E on the Theorem Prover's House is attributed to A C Wills).
[40] C. B. Jones. The early search for tractable ways of reasoning about programs. *IEEE Annals of the History of Computing*, **25**, 26–49, 2003.
[41] I. Lakatos. *Proofs and Refutations*. Cambridge University Press, 1976.
[42] E. J. Lemmon. *Beginning Logic*, Chapman and Hall, 1971.
[43] M. Kohlhase and F. Rabe. Semantics of OpenMath and MathML3. *Mathematics in Computer Science*, **6**, 235–260, 2012.
[44] http://rjlipton.wordpress.com.
[45] J. McCarthy. A basis for a mathematical theory of computation. In *Proceedings of the Western Joint Computer Conference*, 225–238, 1961.
[46] MacLogic manual, available at http://rd.host.cs.st-andrews.ac.uk/logic/mac/docs/.
[47] D. MacKenzie, *Mechanizing Proof: Computing, Risk, and Trust*. MIT Press, 2001.
[48] http://www-history.mcs.st-and.ac.uk/.
[49] U. Martin. Computers, reasoning and mathematical practice. In *Computational Logic* Springer 1999, 301–346.
[50] U. Martin and A. Pease. Mathematical practice, crowdsourcing and social machines. In *Proceedings of Intelligent Computer Mathematics* Springer Lecture Notes in Computer Science, 7961, 98–119, 2013.
[51] D. Miller. Broad spectrum proof certificates. *Certified Programmes and Proofs*, Springer Lecture Notes in Computer Science, 7086, 54–69, 2011.
[52] C. Neylon. Re-use as impact, *altmetrics11, workshop at ACM Web Science Conference*, 2011.
[53] M. Nielsen. *Reinventing Discovery: The New Era of Networked Science*. Princeton University Press, 2011.
[54] OEIS Foundation Inc, The On-Line Encyclopedia of Integer Sequences, http://oeis.org.
[55] S. Obua. ProofPeer - A Cloud-based Interactive Theorem Proving System, arXiv:1201.0540.
[56] H. A. Simon. *Models of My Life*. Basic Books, 1991.

[57] J. Smith. From flowers to palms: 40 years of policy for online learning. *Research in Learning Technology*, **13**, 93–108, 2005.
[58] SOCIAM: the theory and practice of social machines http://tinyurl.com/cmmwbcf.
[59] M. Szabo, ed. *The Collected Works of Gerhard Gentzen*. North-Holland, 1969.
[60] G. Sundholm. Systems of deduction. In *Handbook of Philosophical Logic, vol I*, D. Gabbay and F. Guenthner, eds, pp. 133–188. Reidel, Dordrecht, 1983.
[61] A. H. Taub, ed. *John von Neumann: Collected Works, volume V: Design of Computers, Theory of Automata and Numerical Analysis*. Pergamon Press, 1963.
[62] A. M. Turing. Checking a large routine. *Report of a Conference on High Speed Automatic Calculating Machines* University Mathematical Laboratory, Cambridge, 67–69, 1949.
[63] N. N. Vorob'ev. A new algorithm for derivability in the constructive propositional calculus. *American Mathematical Society Translations*, **94**, 37–71, 1970.
[64] D. Weitzner et al. Information accountability. *Communications of the ACM*, **51**, 82–87, 2008.
[65] A. Wiles. Interview on PBS, 2000 http://www.pbs.org/wgbh/nova/physics/andrew-wiles-fermat.html.
[66] M. V. Wilkes. Software engineering and structured programming. *IEEE Transactions on Software Engineering*, **2**, 274–276, 1976.

Received 21 April 2014