On refereeing computer science

Harold Thimbleby
Department of Computer Science, University of Swansea, Wales
h.thimbleby@swansea.ac.uk

August 19, 2008

Abstract

Computer science is a diverse, exciting, and rapidly expanding discipline. Inevitably the research community pulls itself in different directions, sometimes with the result that visionary research may be under-valued. This paper explores systemic issues of general applicability with the aim of understanding and hence valuing the diversity.
Keywords Refereeing; Innovator's dilemma; Sustainable CS; Snowflake development.

1 Introduction

Edsger Dijkstra, one of the towering figures in CS,[*] wrote that computer science solves problems that ten years earlier would have baffled the imagination [5]; problems rapidly move from being inexpressible to being solved. Computer science ideas, if they are to form part of the canon of successful research, have to catch that brief moment between being inexpressible and being routine.

Over thirty years ago the then fast pace of computer science was already recognised as creating divergence of opinion within the scientific community [30]. Or twenty years ago, when Robert Cailleau and Tim Berners-Lee demonstrated the World Wide Web at the ACM 1991 Hypertext conference [16], few if any people understood it or anticipated its rapid dominance.

... one year I submitted a paper (on distributed web site management to the World Wide Web conference in 1995), and referees reject it, asking who wants to do that? I revised and resubmitted the paper the following year, and referees then told me there were commercial products and the topic was not research. I believe the paper was good, but to get it published it would have to have landed on a referee's desk at just the right moment—between when the referee understood the topic and before it had moved on to mass production.

As evidenced by recent papers such as [3, 4, 10] (and many more) the debate on the nature of computer science is showing no sign of going away.

Even in the long-established and core area of programming languages there are still incompatible differences of opinion [12]. As Hayes says, calculus had its dispute over two notations (Newton's $\dot{x}$ and Leibnitz's $\frac{dx}{dt}$) [1], but it never

[*]E. W. Dijkstra, 1930–2002. 1972 A. M. Turing Award; 1975 Foreign Honorary Member of the American Academy of Arts and Sciences; 1982 IEEE Computer Pioneer Charter Recipient; 2002 ACM Principles of Distributed Computing Influential Paper Award; etc.

had 2,500 notations to argue over, nor the example of an outstanding expert such as Dijkstra denigrating people using notation he did not like as "mentally mutilated beyond hope of regeneration" [6]. In fact, psychology does have something to contribute to the debate: putting effort into our own work, then reviewing a new idea (that somebody else wishes to publish) creates "cognitive dissonance," a well-validated phenomenon [25], which is a negative feeling we act, often unconsciously, to resolve. One easy way to resolve cognitive dissonance in this case is to criticise the new idea—to the extent we can successfully criticise the idea, the less we need to question why we have invested effort in our own specialisms. Similar forces have been described as a brake on innovation generally: new innovations clash with existing value systems, creating dilemmas for the innovator [2].

Many of the results of computer science (henceforth, CS) are commercial products rather than research papers; often, one can appear to keep up with the cutting edge of CS more easily by being a consumer rather than by being a researcher. Indeed, buying the latest computer or software may accelerate research better than having a new idea. Some new research areas, like Web 2.0, arose out of the community rather than from a research programme that *intended* to create a Web 2.0. Moore's Law ensures that a (sufficiently long) calculation started today on a current computer can be overtaken by a new computer bought later. Indeed, the power of computers can prevent the development of whole areas of research [21]: the beautiful theory of elliptic functions may well not have been invented had Jacobi had access to a computer. Or perhaps elliptic functions might have been invented but not found a publication ready to recognise their value.

In short, CS has a problems, felt most especially at the cutting edge where new ideas are being developed. How should research be done? How should results be published? How should papers and proposals be reviewed? How can we make progress when we do not agree on the direction progress should take? What sensible answers can there be when the scale and speed of developments is beyond individuals to keep pace with? What advice can we give research students today who will have to face vivas a few years later? What values should guide research progress when the number of exciting research topics far exceeds the number of able researchers to pursue them adequately?

Some disciplines are deemed to be "essentially contestable," meaning that divergence of informed opinion is in principle unavoidable, and not merely a present lack of consensus on proper definitions [8]. A standard example is the unavoidable conflict in the definitions of democracy. Since computers can implement systems that implement, undermine, transform or contribute to democracy, we could conclude that CS itself is essentially contestable, certainly when it is applied in essentially contestable fields. But declaring a discipline essentially contestable is little more than counsel for despair; instead, we need to understand and accept diversity in CS.

2 Cutting-edge CS and the snowflake effect

Because much work in CS is very new, much of it does not seem "difficult" for anybody. In contrast, any advance in a well-established field such as mathematics builds on much previous work, and therefore requires expertise simply to know and understand the previous body of knowledge. Referees might think that good research has to be difficult (presumably their own research is pushing at the boundaries of their own abilities) so any new idea seems too easy to be worthwhile (especially if it has been well explained) and is therefore criticised. Worse, many CS developments end up being mass produced: anybody can use them without even understanding them! The consequence is that the differences between a genuine expert and an informed bystander are less than in established disciplines.

Such points are exacerbated by psychology. People, whether referees or not, tend to over-rate their own competence [14] relative to the population (in this case, of reviewers); secondly, people who are ignorant about a field do not have the meta-level abilities to assess their own incompetence accurately [7]. The consequence is that the differences between a genuine expert and an informed bystander who claims to be expert are less than in established disciplines.

Any good idea in CS raises new issues and will be stimulating—but it may also stimulate reviewers into thinking about new and unanticipated areas *particularly if they are not actual experts in the area of the idea*, and then the reviewers will think the new idea fails to address those issues that are of concern to *them*. Worse, if several people are reviewing, the chances that they are stimulated in the same way and agree is slim: thus reviewing processes that rely on consensus will behave conservatively, tending to filter out new research. Typically, three or more reviewers are chosen fairly, and hence more-or-less at random, from the population of potential reviewers. The probability that all reviewers are in the same intersection of subfields as the author is negligible.

The nature of CS ensures there are valid but alternative points of view. For example, CS can be evaluated for its quality of output, range, efficiency, speed, correctness, usability, maintainability, dependability, platform independence, novelty, performance on test cases—and we can add endlessly to this list: how about low energy consumption or advancing other fields of science by simulation? Some subfields of CS have specific views on experimental methods, mathematics, reproducibility and other dimensions. Reviewers are often inclined to view one of these dimensions as primary, and to dismiss work whose main claims are on other dimensions, not least because of cognitive dissonance. All the reviewer need do is show that the author has *clearly* failed to understand an entire subfield relevant to their work! Clearly this may be (or seem to be) a good reason for rejecting the work, but it is not a healthy way for CS to progress.

Some subfields (e.g., as in Knuth's classic books, where he argues for "definiteness" as an essential criterion [13]) clearly define their values and more-or-less ignore the diversity beyond their boundaries. But diversity is a particular problem when ideas cross-fertilise subfields; one could imagine, say, a new technical architecture for CSCSW not being publishable within CSCW (because it is too technical for that subfield) nor without (because it does not seem a sufficient advance on other architectures).

The natural sciences seek particular truths in the given natural world (figure 1), whereas artificial sciences [23] including CS create new ideas and new systems (figure 2). CS therefore does not know "the truth" until a context has been created; this is another way of expressing the issues of the previous section. If reviewers develop ideas (following arrows in figure 2) they are more likely to want to take CS in unpredictable directions than if it was a science.

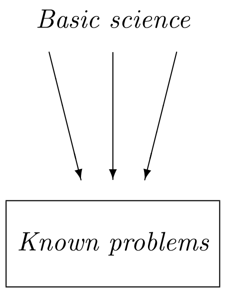

Figure 1. Consensus on the pursuit of truth in normal science; see §3. (Figure based on [27].)

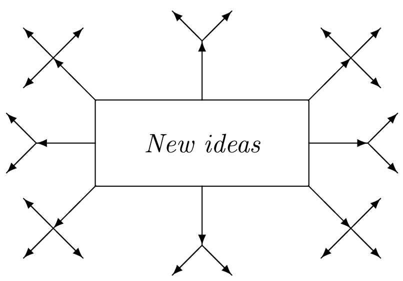

Figure 2. The "snowflake" of possibilities inspired by new ideas. (Figure based on [27].)

A CS researcher will be told by reviewers that there are different approaches that should have been considered—also illustrated in figure 4. Thus CS is more badly reviewed on average than natural sciences.

It is useful to give concepts names so that they can be referred to concisely. The cutting edge ideas artificial sciences create in numerous directions is the "snowflake effect" (figure 2). Two points can be quickly made: just because a referee can point out or follow a snowflake path that is different from a proposer's plan does not criticise the proposer (figure 4)—in fact, it *confirms* the potential of developments from the position the proposer has identified; secondly, the growing frontier of the snowflake needs exponential resources—some ideas have to be dropped because eventually there will not be enough people to follow them up. As competition increases, reviewers become stricter. Over time, as established research is successful and funded, and research in newer areas not funded, then the success around old ideas gets relatively higher and the success of innovation drops: this is the "Matthew Effect" [18, 19], an effect that reduces diversity and the appreciation of diversity: named after Jesus's saying in Matthew 13:12, "Whoever has will be given more, and he will have an abundance. Whoever does not have, even what he has will be taken from him." The frustration of risk-taking researchers proposing new breakthroughs increases, and they naturally retreat from promising research programmes.

In the natural science's focus on truth, people become experts relatively easily by hill climbing, and fields become well-defined. Outsiders cease to understand the nuances of the speciality, and referees are picked more accurately—badly chosen referees know they are inappropriate and say so. In contrast, in the snowflake situation, new areas are recent and less deeply explored, so the difference between the author's insights and a non-specialist referee's insight is less; the non-specialist is less likely to realise or admit they are the wrong referee—after all, the author is only a few months ahead of them, rather than a whole career ahead of them.

3 A philosophy of science perspective

Science is not just the pursuit of truth [15]: we recognise wider issues about how we as a community of researchers work together, and, in particular, work within frameworks of assumptions—so called "paradigms" that influence and form our world view. Kuhn's classic perspective [15], of course, does not capture all the distinctions we might wish to make. A paradigm change emerges, as Kuhn puts it, from crisis in normal science. Indeed, the transformation may actually be felt like a crisis, as the scientific community resists the change. If interdisciplinary research like CS is being done, it is possible that the proposer and reviewer are working within different paradigms. While the proposer sees the multiparadigm approach or paradigm shift as positive, a reviewer may see the pull towards a different paradigm as crisis, and cognitive dissonance predicts that reviewers will work to reduce crises.

In normal science, most scientists agree what the important questions are, and they agree on what the interesting unanswered questions are. Typically, such questions have resisted resolution, but the questions are clear. An example from CS is the P=NP problem: it is well defined and there is a large community making "normal" progress on it; however, if it is proven that P=NP, this may cause a crisis and a change in direction for some researchers. To give two *advanced* questions from other sciences, consider that we currently do not know much about the atmospheric composition of the satellites of Saturn; and although chimpanzee communities have unique "cultural habits," we do not know whether their "culture" is determined by genes or by habitat.

In both cases it would obviously be very interesting to know the answers (indeed these are projects recently funded). There are many such well defined hard questions in the normal sciences. In contrast, much of what happens in CS happens because we create new concepts, and these new concepts *then* pose new questions. The advanced questions, inevitably, concern new topics that we hardly understand. For, if we understood the questions (e.g., well enough to write a program), then the question has effectively become a commodity that can be downloaded and run by anyone. AI is the classic example: it has been described as a field where as soon as any algorithm is known, it is not AI.

Advanced CS is always at the edge of Kuhnian crisis, at the edge of the snowflake. While this is part of the excitement of the field, it ensures that authors have a very hard time describing their ideas clearly. Often referees will see alternative ways of solving the same problem, partly because the proposer has not been able to focus the partly-undefined question finely enough.

4 The habit of irreproducibility and its dangers

In CS we do not have a strong tradition of building on or reproducing earlier work, a habit that encourages reviewers to pursue different agendas than authors. In most disciplines knowledge that is shared is widely known in an agreed form. In artistic subjects, what is appreciated may not be knowledge but æsthetics, which is also shared and consensual: art that is only appreciated by one person is quirky. Ziman defines science as founded on "consensual knowledge" [31]; that is, as the full apparatus and procedures for arriving at community consensus on reliable knowledge that that community wants to use.In Ziman's conception, science that is only appreciated by one person is either ready for community testing or is non-science. In contrast, much in CS is essentially private. For example, one of my sons did a CS degree, and took a functional programming course. However, the lecturer teaching the course had implemented his own language, and I couldn't answer the previous year's exam questions!

The lecturer was teaching a public concept, functional programming, but doing so with a private notation.

Much of CS is based on computer programs that are written records of the work that researchers have done. Unlike any other science (though like mathematics) we can in principle publish our work directly. It would seem that much of CS ought to be easy to build on, because reproduction is merely a matter of the researcher putting programs on a server, and the future researcher downloading and running it. However many factors undermine this simple view:

1. Computer programs are very complex, and the gist of a publication or research advance may not be the *entire* program but only a relatively tiny part, the rest being a support structure of no research value.

2. Computer scientists tend to be sloppy. Relatively few computer scientists use laboratory note books to record their work, and very few record their work in a rigorous way that would withstand legal scrutiny in an IPR dispute. The text of a test run that generates data for a publication may itself never be recorded.

3. Hardware and software environments advance. A program working on a computer when a paper is submitted may not work when the paper is published; operating system versions, compilers and so on will all have advanced.

4. Some programs have intrinsic value. Research translates into intellectual property rights in CS faster than in many other disciplines. A hopeful researcher may be reluctant to give readers access to programs whether object or source code that gives away value.

5. Publication often involves typographical niceties that original program results cannot achieve; this encourages manipulation of results to conform to publishing styles.

6. The purpose of a publication is argument, not a literal presentation of a program for others to replicate directly. The size of most programs precludes describing them clearly.

7. The field has a tradition of publishing ideas without confirming whether they work. Given the pressure to publish, why work to higher standards of reproducibility?

Some of these issues can be partly or fully addressed by using tools [26].

The last point above sounds unduly harsh and begs evidence in its support. Tichy sampled 400 CS papers published to 1995 [29] and found that 50% of them proposing models did not test them, compared to a rate of about 10% from other fields of science. Tichy's paper goes on to provide a rebuttal of many of the excuses for not doing experimental work scientifically. My own survey of authors contributing to the *Journal of Machine Learning Research* [28] showed that the authors of about one third of papers papers published 2000–2004 describing systems were not in a position to share the system described in the paper. Some authors said their systems simply didn't work; some said operating systems had moved on; some said their employer forbade distribution of the system, and so on. A study covering the same period of time for the journals *Computer Journal* and *Software—Practice and Experience* obtained similar results. The differences in the journals, the fields and the different natures of the surveys suggests that the result is typical of CS.

Sandewall [22] gives a possible explanation, based on his concept of "design iteration." Initially, a researcher has a hypothesis H_0 about how a system to undertake a task should be organised. The researcher builds a prototype system S_0 and then finds that they can see some aspects of their initial hypothesis was wrong and needs correcting. In principle, one might want to start again, but for a substantial programme of research it might not be reasonable to discard the work and start over. Instead at time t it may be better to continue with a new hypothesis H_t and a revised implementation S_t. Hopefully S_t will relate closely to H_t, but inevitably there is a time lag as work on S catches up with the revised hypothesis driving the work. Sandewall agues that eventually the relevance of S_t to H_t may become so weak that it is better to start implementation again, and thus start another cycle of design iteration. Hopefully, design iteration converges. This is fine if the cycle time is measured in weeks or months and when few cycles are required, but often there are pressures to publish before the design iteration has converged. Indeed, the end of a single cycle of iteration is an ideal place to communicate with other researchers working on the same problem. The consequence, then, is a research literature with systems work that to greater and lesser extents does not work—and which the research community has little interest in making work, since it has moved on to subsequent design iterations.

Without a habit of reproducing and building on others' work we are unlikely to detect misconduct or even to deter people from being "inaccurate in their own favour." Would we be able to detect our own Jan Hendrik Schön, who published papers with data generated automatically rather than from experiments (which is what he claimed)? As Goodstein puts it [9], the temptation is to publish what you think is the truth without going to the trouble of doing the rigorous work to prove it. Feynman [20] is more direct: non-reproducible work wastes the time of other researchers. (The temptation to gloss reproducibility is severe with research proposals: how can you get funds to do research when rigorous preliminary work might be taken by the reviewers as tantamount to achieving what you propose to do?)

While it is not routine to reproduce work in CS the habit becomes to do new work. To the extent that referees share this habit they are conditioned to do things in a different way, and if they express such views they will tend to undermine new ideas.

5 Unreliable evaluation

There is a difference between any objective value of CS and its subjective review. The objective value is what it is or can actually achieve; its subjective valuation is what reviewers think an idea can achieve. The subjective value is crucial; researchers require resources to pursue their work. All measures of support a researcher need depend on peer reviewing: for proposals (which ask for explicit funding), for papers (which ask to be published and presented to the broader community), and for career progression—from doctoral vivas, pro-

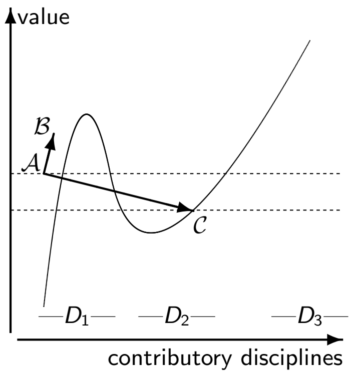

Figure 3. Researchers aim to increase the value of their work from $\mathcal{A}$, and climb the surface. Hill climbers aim at $\mathcal{B}$, non-local optimisers may aim at $\mathcal{C}$, which has worse immediate performance. (Figure based on [27].)

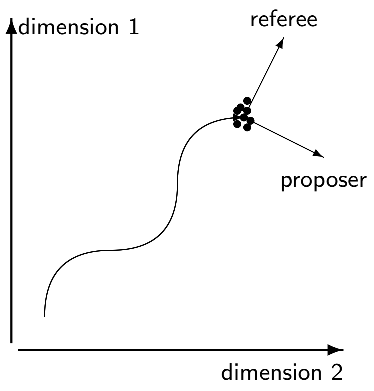

Figure 4. Research progresses along various dimensions (only two are shown); currently it is being pursued in closely related ways represented by the blobs. The proposer sees potential in one direction; a referee legitimately sees potential in another.

motion committees and job applications.

If a researcher submits an idea in an "intersection area" (in general there may be more fields than two), finding expert reviewers will be hard.

1. If the intersection area is novel, then few people will be established in this speciality. Selecting reviewers cannot be reliably based on their track record.

2. A proposal may be sent to some reviewers in one area and some in another; typically, each reviewer will spot problems because there are ways to advance their own areas that have not been addressed.

3. There are few reviewers in any specific intersection area, so reviewers are more likely to be chosen who intersect one area with *another* area—and such reviewers might think of themselves as suitably interdisciplinary even when they do not know both relevant disiciplines.

The "scientific search space" is a multidimensional space where we try to maximise scientific value. Value might be value for money, theoretical power or interest. It doesn't matter at this level of abstraction: we want to do better CS, and that means going upwards in the space as represented in figure 3. The space is multidimensional, but the graph projects it down to two dimensions; the horizontal axis is labelled "contributory disciplines," measuring dispersion from a notional "origin" discipline. In particular, three regions of discipline are picked out: D_1, D_2, D_3, where D_1 is an established region of normal science; regions D_2 and D_3 are new areas.

The graph allows us to represent familiar issues. Work in D_1 has value $\mathcal{A}$, and a researcher spots a new opportunity around D_2. Because this is a new idea, it will have less current value ($|\mathcal{C}| < |\mathcal{A}|$), although the future that can be reached from $\mathcal{C}$ (in area D_3) may well surpass what can possibly be achieved directly from the current state of the art in D_1. Seen like this, the researcher has good grounds to work in region D_2.

Unfortunately, reviewers compare the worse value at $\mathcal{C}$ with the well-known value at $\mathcal{A}$. Moreover, since many people work in area D_1, the local slope of the graph (i.e., $\mathcal{AB}$) is known to be much more promising than the slope $\mathcal{AC}$. There is more research activity in D_1 and status is clearer—both the status of the principles underlying work in D_1, as well as the political status (who are the top scientists, how one becomes a top scientist, who are the prominent reviewers to choose, etc).

Overall, the research idea in D_2 is likely to have reviewers picked from D_1 who do not rate it well. There are fewer people in the region D_2, so there is less activity there, and the probability of picking a good reviewer in this region is correspondingly less. There may be established journals in D_1, but perhaps none in D_2. Good reviewers are not visible: they do not stand out—people who understand an intersection area may have been publishing in other areas because they have more established and conducive outlets. New, adventurous work will necessarily be spread thinner, and this spreading also dilutes a researcher's track record as a reviewer would see their apparent low productivity in many "relevant" areas. In particular, new areas and expanding areas are under-represented in biliometric information.

Reviewers are selected because of their prominence; they are likely to be good, and to be good they are likely to be specialists with a clear view, formed over their career, of the normal science that they can do. Their criticisms will be emphatic, authoritative, and devastating. There is always a much better way of doing part of the research that the reviewer can spot; there is always something to point out that the proposer has not read or does not know.

Figure 3 represents a well known problem in AI. A reviewer is (typically) a "hill climber" who can see how to make progress from their local perspective. The researcher, however, has a new idea, but of course it hasn't been done yet, so the first step seems like a step backwards *for the reviewer*. The reviewer has grounds for criticising the proposed project, since they do not have the perspective of the proposer to see the future potential: the initial drop gives the reviewer ammunition to be negative. A key assumption figure 3 makes clear is that the surface treated as continuous; a reviewer in D_1 thinks D_2 is an extension of their work or in competition with it, rather than a new field. In contrast in many established disciplines, such as mathematics, further specialisation gives status.

6 Improving refereeing processes

Publishers and funding bodies have resource limitations. Not all good proposals can be funded, not all good papers can be published—and publishers and funders also wish to avoid false positives: bad proposals and bad papers should not be supported. Given that decisions have to be made, can refereeing processes be improved? Below are some suggestions; the main insight from this illustrative list of ideas is that improvements to processes are possible—others should be sought, as appropriate to specific needs.

- Publications and proposals could be "star rated" by the author. A 5 star paper would be one where the author has lodged a complete program with the journal's file server, and that this program has been refereed; at the other end of the scale, a 0 star paper needs no program in principle. A 1 star paper might be a program from which screen shots or other examples have been generated but the author has not released the program. The point is that star ratings clearly communicate the reproducibility status; moreover, it is a simple scheme for doing so. If status accrues to publishing 5 star papers, then this would improve standards. (Of course one could choose a different scale.)

- Peer review systems should have appeals processes that don't simply repeat the risks of diverse referees, but make the past refereeing disagreements an explicit issue to address. Referees should be specific about their *precise* specialisms. Conferences, and in other fast-paced situations where appeals are inappropriate, could have planned outlets for debate where refereeing disagreements are debated, so that subsequent calls can make progress.

- Reviewing is an optimisation problem, and CS has algorithms available for this. For example, use review techniques based on global strategies: simulated annealing suggests that when a idea gets mixed reviews (which would conventionally guarantee rejection, because the average score is too low), make a random decision to decide whether to discard the positive or negative reviews. Some of the reviewers are right, but they can't *all* be right, so a process independent of the reviewers should reject some of the reviews randomly.

- Where possible, the authors and reviewers should interact; at least the authors should be able to respond to reviewers. Systems such as appropriately modified wikis could be used. Conferences might have five minute presentations, followed by survivors giving full presentations.

- The issues this paper has identified within CS are rampant in the cutting edges of other disciplines [17, 24]; this suggests that cross-disciplinary work exploring the issues and how they can best be handled will be both fruitful and worthwhile.

7 Conclusions

CS is diverse and interdisciplinary, and this creates opportunities and problems. This paper has reviewed many ways that conventional reviewing leads to conservative decisions, resulting in unnecessary hurdles for cutting edge work. Unsuccessful researchers are likely to think that if their own high quality work is not being accepted then they should referee others' work to higher standards, thus creating a vicious circle. If everybody does what appears rational to them, without considering the bigger picture, then CS suffers—"the tragedy of the commons" [11]. CS then gets less support, recognition and funding (and, in turn, worsening bibliometrics). Successful researchers are likely to disagree with this view, for a successful researcher can argue that peer review works—they have already been selected by that very process, so what else would they say?

The bigger picture can be summarised by the "snowflake effect" (§2) but solutions, not just descriptions, are also required; we gave a few in section §6 to show that solutions are possible—more must be sought, specialised as appropriate to particular subfields, traditions and so forth.

The issues are complex and fast-paced. Disagreement between referees must be expected and handled constructively. If the CS community does not urgently address the issues, resources will increasingly favour subjects that rate themselves more highly. However justifiable reviewers feel their negative individual assessments are, collectively they are undermining the intellectual and financial resources of the CS community.

Acknowledgements

Jason Brotherton, Richard Bornat, Alan Bundy, Paul Cairns, Jon Crowcroft, Richard Harper, Tony Hoare, Matt Jones and Steve Woolgar provided many useful comments. Anonymous referees have also made significant contributions: several earlier versions of this paper have been rejected—not many papers benefit from so many referees! This paper was very substantially developed from a short conference paper [27], which explored the subfield of HCI. Harold Thimbleby is a Royal Society-Leverhulme Trust Senior Research Fellow, and gratefully acknowledges the support of this funding for the work reported here.

References

[1] J. Bardi. *The Calculus Wars.* High Stakes, London, 2006.

[2] C. M. Christensen. *The Innovator's Dilemma.* Harvard Business School Press, 1997.

[3] J. Crowcroft. On the nature of computing. *Communications of the ACM*, 48(2):19–20, 2005.

[4] P. J. Denning. Is computer science science? *Communications of the ACM*, 48(4):27–31, 2005.

[5] E. W. Dijkstra. Computing science: Achievements and challenges EWD1284. *ACM Symposium on Applied Computing*, pages 2–9, 1975.

[6] E. W. Dijkstra. Selected writings in computer science. 1982.

[7] D. Dunning and J. Kreuger. Unskilled and unaware of it: How difficulties in recognizing one's own incompetence lead to inflated self-assessments. *Journal of Personality and Social Psychology*, 77(6):1121–1134, 1999.

[8] W. B. Gallie. Essentially contested concepts. *Proceedings of the Aristotelian Society*, 56:167–198, 1955-6.

[9] D. Goodstein. In the matter of J Hendrik Schön. *Physics World*, page 16, November 2002.

[10] J. Grudin. Crossing the divide. *ACM Transactions on Computer-Human Interaction*, 11(1):1–25, 2004.

[11] G. Hardin. The tragedy of the commons. *Science*, 162:1243–1247, 1968.

[12] B. Hayes. The semicolon wars. *American Scientist*, 94:299–304, 2006.

[13] D. E. Knuth. *The Art of Computer Programming*, volume 1. Addison-Wesley, 3rd edition, 1997.

[14] J. Kruger. Enhancement bias in the description of self and others. *Personality and Social Psychology Bulletin*, 24:505–516, 1998.

[15] T. S. Kuhn. *The Structure of Scientific Revolutions.* University of Chicago Press, 2nd edition, 1970.

[16] J. J. Leggett. In *Proceedings of the third annual ACM conference on Hypertext*, 1991.

[17] A. Lightman and O. Gingerich. When do anomalies begin? *Science*, 255(5045):690–695, 1991.

[18] R. K. Merton. The Matthew Effect in science: The reward and communication systems of science are considered. *Science*, 159(3810):56–63, 1968.

[19] R. K. Merton. The Matthew Effect in science, II: Cumulative advantage and the symbolism of intellectual property. *ISIS*, 79:606–623, 1988.

[20] L. Mlodinow. *Some time with Feynman.* Penguin, 2003.

[21] M. Petkowvšek, H. S. Wilf, and D. Zeilberger. *A=B.* A K Peters: Wellesley, MA, 1996.

[22] E. Sandewall. *The Methodology of Design Iteration for Systems-oriented Research in Computer Science.* 2006.

[23] H. A. Simon. *Sciences of the Artificial.* MIT Press, 3rd edition, 1996.

[24] L. Smolin. *The trouble with physics.* Allan Lane, 2006.

[25] C. Tavris and E. Aronson. *Mistakes were made (but not by me).* Harcourt, 2007.

[26] H. Thimbleby. Explaining code for publication. *Software—Practice & Experience*, 33(10):975–1001, 2003.

[27] H. Thimbleby. Supporting diverse HCI research. *Proceedings BCS HCI Conference*, 2:253–254, 2004.

[28] H. Thimbleby. Give your computer's IQ a boost: Review of *Journal of Machine Learning Research. Times Higher Education Supplement*, 9 May, 2004.

[29] W. F. Tichy. Should computer scientists experiment more? *IEEE Computer*, 31(5):32–40, 1998.

[30] P. Wegner. Research paradigms in computer science. *Proceedings of the 2nd international conference on Software engineering*, pages 322–330, 1976.

[31] J. Ziman. *Reliable Knowledge.* Canto ed., Cambridge University Press, 1991.