News from New Zealand

by

C. S. Calude

Department of Computer Science, University of Auckland
Auckland, New Zealand
cristian@cs.auckland.ac.nz

1 Scientific and Community News

0. The Twelfth Asian Logic Conference will be held in Wellington, New Zealand from 15–20 December 2011. This meeting will be held jointly with a meeting of the Australasian Association for Logic (AAL), `http://www.victoria.ac.nz/victoria-conferences/conference.aspx?p=54&n=Asian%20Logic%202011`.

1. The latest CDMTCS research reports are (`http://www.cs.auckland.ac.nz/staff-cgi-bin/mjd/secondcgi.pl`):

392. A.A. Abbott and C.S. Calude. Von Neumann Normalisation of a Quantum Random Number Generator. 10/2010
393. C.S. Calude, M.J. Dinneen and A.M. Gardner. Opening the Book of Randomness (Extended Version). 10/2010
394. M.J. Dinneen and M. Khosravani. Hardness of Approximation and Integer Programming Frameworks for Searching for Caterpillar Trees. 11/2010

395. M.J. Dinneen, Y.-B. Kim and R. Nicolescu. Faster Synchronization in P Systems. 11/2010

396. A.A. Abbott, C.S. Calude and K. Svozil. A Quantum Random Number Generator Certified by Value Indefiniteness. 12/2010

2 A Dialogue with Professor Joseph F. Traub

Joseph F. Traub is the Edwin Howard Armstrong Professor of Computer Science at Columbia University and External Professor, Santa Fe Institute `http://cs.columbia.edu/~traub`*. He is the author or editor of ten monographs and some 120 papers in computer science, mathematics, physics, finance, and economics. In 1959 he began his work on optimal iteration theory culminating in the 1964 monograph which is still in print. Subsequently he pioneered work with Henryk Wozniakowski on optimal algorithms and computational complexity applied to continuous scientific problems (information-based complexity). He collaborated in creating significant new algorithms including the Jenkins-Traub algorithm for polynomial zeros, as well as the Kung-Traub, Shaw-Traub, and Brent-Traub algorithms. One of his current research areas is quantum computing. From 1971 to 1979 he headed the computer science department at Carnegie Mellon University and led it from a critical period to eminence (see Joseph Traub digital archive at CMU* `http://diva.library.cmu.edu/traub`*). From 1979 to 1989, he was the founding chair of the computer science department at Columbia University. From 1986 to 1992 he served as founding chair of the Computer Science and Telecommunications Board, National Academies, and served as chair again 2005 – 2009. Traub was founding editor-in-chief, Journal of Complexity, in 1985, and continues in that capacity.*

His numerous honors include election to the National Academy of Engineering in 1985, the 1991 Emanuel R. Piore Gold Medal from IEEE, and the 1992 Distinguished Service Award from the Computer Research Association (CRA). He is a Fellow of the Association for Computing Machinery (ACM), the American Association for the Advancement of Science (AAAS), the Society for Industrial and Applied Mathematics (SIAM), and the New York Academy of Sciences (NYAS). He has been Sherman Fairchild Distinguished Scholar at the California Institute of Technology, and received a Senior Scientist Award from the Alexander von Humboldt Foundation. He was selected by the Accademia Nazionale dei Lincei in Rome to present the 1993 Lezione Lincee, a cycle of six lectures. Traub received the 1999 Mayor's Award for Excellence in Science and Technology. The award was presented by Mayor Rudy Giuliani at a ceremony in New York City. In 2001 he received an honorary doctorate of science from the University of Central Florida.

Erol Gelenbe: To what do you ascribe your very successful career?

Joseph Traub: The short answer is mostly just plain dumb luck. Of course I also took advantage of some opportunities. I'll give you some examples. I entered Columbia in 1954, intending to take a Ph.D. in theoretical physics. In 1955, on the advice of a fellow student, I visited IBM's Watson Laboratories at Columbia. At the time this was one of the few places in the country where a student could get his hands on the computer. I was hooked. Due to that piece of luck, I've spent the last 55 years involved with computation. My Ph.D. thesis on computational quantum mechanics was done on the IBM 650, a 2000-word drum memory machine. I believe that the need to be very economical on this computer may have led to my early interest in optimal algorithms and computational complexity.

The next stroke of luck was being hired in the research division of Bell Laboratories in 1959. This was a golden age at the Labs. You were free to work on whatever interested you; if your work had impact on the company, all the better. I had the freedom to spend some four years doing research which culminated in the creation of optimal iteration theory and the publication of a monograph in 1964. If I had been an assistant professor at a university, it would have been very dangerous to create a new area while publishing very little, but I could do it at Bell Labs.

The next stroke of luck occurred during a sabbatical at Stanford in 1966. I met a student, Michael Jenkins, who was looking for a Ph.D. advisor. We developed the Jenkins-Traub algorithm as well as high-quality portable software for polynomial zeros.

In 1970 I was at the University of Washington. I advertised for a GRA, and interviewed about a dozen students. I selected H. T. Kung, and the following year, brought him with me to CMU. He eventually joined the CMU faculty and today is a chaired professor at Harvard.

In the spring of 1971, I was selected to be head of the computer science department at CMU. I was 38 years old and had this opportunity because I'd gotten into the field so early. If I'd been in a mature field, I would never have had such an opportunity. Alan Perlis, the department head, was leaving to become founding chair of the department at Yale. Al didn't publish much but was a towering figure at CMU. Allen Newell, Alan Perlis, and Herbert Simon had founded the department in 1965. Perlis and I overlapped by just a few days but he gave me invaluable advice. One of the things I recall is that he advised me to rapidly tenure Bill Wulf, who was then an assistant professor. Soon Bill was a full professor, and he later succeeded me as chair of the Computer Science and Telecommunications Board of the National Research Council, and then became President of the National Academy of Engineering.

The department was quite small, including Gordon Bell, Nico Habermann, Allen Newell, Raj Reddy, Herbert Simon, and William Wulf. Just prior to 1971, many faculty had left the department to take positions elsewhere. Those professors who remained formed a core of world-class scientists recognized as leaders of the discipline. I worked with the faculty to recruit new members and diversify research funding. I was deeply concerned whether we would remain a leading department. Perhaps it's just as well that I didn't know of a commitment made by the senior faculty to stay at CMU for at least one year to see if the department could be turned around. I often had the feeling that the department and I had been created for each other. By the time I left Carnegie in 1979, we had some fifty teaching and research faculty.

Another opportunity occurred in 1972. Lawrence Livermore Laboratory planned to acquire a STAR computer and hired me as a consultant. I became fascinated with parallel computing, which I saw as a very interesting new direction for computing. Perhaps I was too early. I remember giving a talk at a major research university on why parallel computing was going to be very important. The first question after the lecture was from a very well known professor: Joe, you don't really think we're ever going to use these computers to solve our problems?

In 1972 occurred a stroke of luck that was to change my scientific work. I received a registered package containing a paper and letter from someone named Henryk Wozniakowski in Warsaw. A Polish professor had pointed him to my 1964 monograph. The paper proved conjectures I'd framed in the 1964 book but in much greater generality. Henryk visited me at CMU in 1973 and that was to be the beginning of a collaboration that has extended for almost forty years. We were to start and build the field of information-based complexity.

By 1979 the department was thriving by every measure. I had been head at CMU for seven years, and I could think of moving on to new challenges. Then came the next stroke of good timing. With some exceptions, the Ivy League universities had lagged in the building of computer science departments. Now Columbia decided to start a department, and invited me to return and to build it. I accepted, and went to Columbia in 1979.

By the mid-80s many papers were being written on information-based complexity and there was no obvious place to publish them. I was not particularly interested in starting a journal, but there was a need. I noted with surprise that there was, as far as I could tell, no journal with the word complexity in the title. In 1985, I started the Journal of Complexity. It's now in its 26th year and is much broader than only publishing papers in information-based complexity.

In 1986, a different opportunity came knocking. I was asked to start a computer science board at the National Academy of Sciences. More precisely, it was a board of the National Research Council, which is the working arm of the National Academies. I was told two previous boards had failed–I was determined there would not be a third failure. I called the new board The Computer Science and Technology Board (CSTB) and appointed leaders from academia and industry to serve as members. About a year after the establishment of CSTB, I was fortunate to hire a superb staff director, Marjory Blumenthal. Around 1988, Frank Press, the president of the National Academy of Sciences, told me that the board on telecommunications (BOTCAP) was failing, and that the decision had been made to terminate it. He asked us to add telecommunications to our responsibilities. Marjory and I wanted to preserve the abbreviation CSTB, so we renamed the board the Computer Science and Telecommunications Board, which has remained its name to the present.

Another opportunity came in 1990, when I was asked to teach in the summer school of the Santa Fe Institute (SFI). I was greatly stimulated by the variety of topics studied at SFI and have been associated with the Institute ever since. Currently I'm serving as an external professor.

In the early 90s, we were lucky to be given a collateralized mortgage obligation (CMO) by Goldman Sachs. This involved computing integrals in 360 dimensions. A Ph.D. student, Spassimir Paskov, computed the integrals by Monte Carlo (MC) and quasi-Monte Carlo (QMC). To our surprise, and later to the surprise of Wall Street, QMC always beat MC by one to three orders of magnitude.

Of course the greatest stroke of luck of all was meeting and marrying Pamela McCorduck. I'm also blessed with two loving children, Claudia and Hillary, and four wonderful grandchildren.

EG: You've done a great deal of research. Could you tell us about some of your early work?

JT: Shortly after I joined Bell Labs in 1959, a colleague asked me how to compute the numerical solution of a certain problem, which involved the solution of a complicated nonlinear equation. I could think of a number of ways to solve the problem. What was the optimal algorithm, that is a method that would minimize the required computational resources? To my surprise, there was no theory of optimal algorithms. (The phrase computational complexity which is the study of the minimal resources required to solve computational problems was not introduced until 1965.) I set out to construct a theory of optimal algorithms for the solution of the nonlinear equation $f(x) = 0$. I had the key insight that the maximal order of an iteration depended on the available information about f and not on the structure of the iteration. (Maximal order is closely related to computational complexity.) This was such a powerful idea that I was sure someone else would announce it. I scanned the world literature, fearing such a publication. To my immense relief, no one published this idea. Wouldn't it be useful for researchers to be automatically notified of papers in which they would be interested? This led to work on such a publication notification system [1, 3].

I first presented the work on what was to become optimal iteration theory at the 1961 National ACM Conference [17]. In those days the national ACM conference was a big deal, and many of the researchers in computing attended. I kept building the theory and soon had a manuscript of some 120 pages. Being a very young and naive researcher, I sent this manuscript to Mario Juncosa, the editor-in-chief of the Journal of the ACM, who wrote back that it would be a long time before he read the manuscript. Just then, Prentice-Hall asked me if I wanted to write a book. I set to work and wrote the monograph, Iterative Methods for the Solution of Equations [18]. The editor of the Prentice-Hall Series in Automatic Computation was George Forsythe, who was to become founding chair of the Computer Science Department at Stanford. I'm pleased that this book is still in print, going on fifty years since its publication. The book would have been better titled Optimal Iteration Theory. It marked the beginning of lifetime work on optimal algorithms and computational complexity for continuous problems. The Introduction begins with: The general area into which this book falls may be labeled algorithmics. By algorithmics we mean the study of algorithms... Don Knuth credits me with coining algorithmics [6].

So far I'd worked on general nonlinear equations. For such problems, convergence could not be guaranteed. Was there a class of nonlinear equations for which one could guarantee convergence? The answer is yes, for polynomial equations [19]. In 1966 I was a visiting professor at Stanford, where I met a Ph.D. student, Michael Jenkins. We continued the work on global convergence which led to what is usually called the Jenkins-Traub algorithm [5]. The algorithm consists of three stages, of which the third is the most important. It can be shown that this stage is equivalent to applying Newton iteration to a sequence of rational functions, which is converging to a first-degree polynomial whose zero is one of the desired answers. Although Newton iteration requires the evaluation of a derivative at each step, the Jenkins-Traub algorithm does not require the evaluation of any derivatives. It can be shown that under mild conditions, the algorithm always converges, and that the rate of convergence is faster than the quadratic rate of Newton. See the Wikipedia article [34] for more. Jenkins wrote a high-quality portable program implementing the algorithm. This algorithm is still one of the most widely used methods for this problem and is included in many textbooks.

In 1972 I became fascinated with parallel computing and organized a symposium at CMU in 1973, which may have been the first on this subject [20]. I spoke on this topic on the 1974 IFIP Congress [21].

In the 70s I became interested in algebraic complexity. Mary Shaw was a student in my class when I spoke about Horner's method for evaluating polynomials, which was known to be optimal. I conjectured that if one wanted to evaluate all the derivatives of a polynomial, the optimal method would take a quadratic number of multiplications. Mary showed me that she could beat that, and we worked together to get the number of multiplications to linear [16]. Mary is now the Alan

J. Perlis Professor at CMU.

Next, my student H. T. Kung and I showed that computing the first n terms of any algebraic function was no harder than multiplying nth degree polynomials [7]. This problem has a long history; Isaac Newton missed a key point.

Richard Brent and I were able to show that computing the q-th composite of a power series was no harder than computing a single composition where q is any number [4]. One day, I got a call from Don Knuth who had heard about our work. I told him I could mail him a preprint in a few days. Don replied that he didn't want to wait, since he was working on that part of his book just then. He had redone parts of our analysis and just wanted to check that it agreed with what we had done.

This ended my work on algebraic complexity because I was about to move into an entirely new direction.

EG: Can you tell me what happened next?

JT: I mentioned earlier that Henryk Wozniakowski visited me at CMU in 1973. That was to be the beginning of a collaboration that has spanned almost forty years. Initially, we continued the work on optimal iteration theory. Then in 1976 there came an event that changed the course of our research. A Ph.D. student named Arthur Werschulz, now a professor at Fordham University and part of our research group at Columbia University, gave a seminar, where he used some of the techniques from nonlinear equations to attack the complexity of integration. Our reaction was that integration is inherently different from solving nonlinear equations; one doesn't solve integration iteratively. Because these problems are so different, there must be a general structure that underlies this and many other problems. Our search for the general structure led to our monograph [29]. We called this new field analytic complexity. This was to differentiate it from algebraic complexity, which was a very active research area in the late 60s and 70s.

Algebraic complexity deals with algebraic problems such as the complexity of matrix multiplication, where information about the input is complete, while analytic complexity deals with problems from analysis, such as the complexity of high-dimensional integration, where information about the continuous input is partial. Let me elaborate this last point. In calculus, students are taught to compute univariate integrals exactly. But most integrals cannot be expressed in terms of elementary functions; they have to be approximated numerically. This is especially true of real world high-dimensional integrals, such as the integrals common in mathematical finance. We sample the integrand; that is why the information about the mathematical input is partial. Other problems studied in analytic complexity include optimal algorithms and computational complexity of systems of ordinary differential equations, high-dimensional approximation, partial differential equations, continuous optimization, and nonlinear equations.

Greg Wasilkowski joined Henryk and me to write the monograph [30]. We renamed the field epsilon-complexity. One day, Pamela asked me why epsilon-complexity? I replied that epsilon denotes a small quantity and that it measures the error in the answer. She did not seem impressed. Since Pamela is the author of numerous books, I took her lack of enthusiasm seriously and started thinking about a new name. One day I was chatting with my friend, Richard Karp, who, as you know, was a pioneer in the study of NP-completeness. He suggested information-based complexity, which we adopted as the name of the field. It was the name of our monograph [31]. For brevity we often refer to the field as IBC. Typically, IBC theory is developed over abstract linear spaces such as Hilbert or Banach spaces. The applications are often for problems with a very large number of variables.

Because the information is partial, IBC is able to use powerful adversary arguments at the information level. The general idea behind an adversary argument is the following: the adversary creates a situation where the inputs are indistinguishable but the outputs are quite different. It is therefore impossible to compute a good approximation because if we claim an approximation to one output as the answer, the adversary will say the second output is the correct one. Adversary arguments are often used to find a good lower bound on the information complexity and hence tight lower bound on the computational complexity (see, for example, [15], section 2). This may be contrasted with the rest of theoretical computer science where researchers work on discrete problems with complete information and have to settle for conjectures on the complexity hierarchy. We find these adversary arguments very natural but we've learned that this way of thinking is so different that many of our colleagues in theoretical computer science find them difficult. See [28] for an expository account of IBC.

IBC has grown vastly over the past twenty years. See [26] for a brief history and [9] for a survey.

In the early 1990s, I had a Ph.D. student named Spassimir Paskov. Spassimir was very strong in theory, but I wanted to broaden him. We had gotten a collateralized mortgage obligation (CMO) from Goldman Sachs. (A CMO is a bond that represents claims to specific cash flows from large pools of home mortgages.) This involved computing integrals in 360 dimensions. I asked Spassimir to compute the integrals using quasi-Monte Carlo (QMC) and Monte Carlo (MC). It was believed by experts that QMC, which uses deterministic sampling, was not good for dimension greater than 12. To the amazement of our research group, Paskov reported that QMC beat MC by one to three orders of magnitude. The results were presented to a number of Wall Street firms, who were initially skeptical. Other researchers then got similar results. QMC is not a panacea for all high dimensional integration. It is still an open question to explain why QMC is superior to MC for financial instruments. See the Wikipedia article Quasi-Monte Carlo Methods in

Finance for a survey [35] .

Moore's Law, which has explained the exponential increase in computer power over some five decades, is coming to an end. Starting in 2001 our research group has been applying IBC ideas to solve continuous problems such as the Schrödinger equation and path integrals on quantum computers. Among other objectives we want to answer the question posed by Nielsen and Chuang [8]. Of particular interest is a decisive answer whether quantum computers are more powerful than classical computers. To answer this question, one must know the classical and quantum complexities, which can sometimes be obtained using IBC techniques. A survey on solving continuous problems on a quantum computer may be found in [14].

EG: Can you tell me about some of the organizations you've built?

JT: As I mentioned earlier, by getting involved in computing so early I had opportunities I would not have had in a mature discipline. The first was being selected as head of the computer science department at CMU when I was 38 years old. As I told you earlier, the department was very small but the faculty formed a core of world-class scientists. Crucial was adding outstanding faculty and diversifying research funding. We decided to revamp the Ph.D. program. One of the innovations was the creation of the Black Friday Meeting, which was held at the end of each semester. The entire faculty reviewed every Ph.D. student. Every student received a letter regarding his or her progress. I thought this was a very effective management tool. Allen Newell, Herb Simon and I talked about the greening of CMU and Pittsburgh, using computers. That has come to pass, big time.

In 1979, Columbia invited me to start a new computer science department. At the time there were two efforts in computer science: an Electrical Engineering and Computer Science Department and a group in the Statistics Department. The two groups were at loggerheads, and unable to recruit good junior faculty. The plan was that both these efforts were to be terminated.

I was able to negotiate some very good things for the new department. We would get our own building, and I would select the architect. I pointed out that the teaching load in computer science departments at leading private research universities was one course a semester, and got the same at Columbia. I accepted the position and started on July 1, 1979.

It was to be the toughest challenge I ever had. At CMU, the department had enough DEC computers to heat the building. At Columbia, the entire engineering school had a single DEC machine; an 11/45 model, as I recall. There were three tenured faculty inherited from the terminated efforts, as well as a number of junior faculty. None of the junior faculty belonged in a department with national ambitions; they were all gone within two years. I set out to hire outstanding new Ph.D.s. Since many universities and corporate research laboratories were hiring at this time, the competition was very tough, but we succeeded in hiring some outstanding young faculty. With almost no faculty, we were trying to teach several thousand students who were taking our courses. I didn't advertise to our newly-hired hotshots that they'd be teaching some 200 students per course.

But we had some great successes. We received a substantial grant from IBM. I took the new faculty to meet Bob Kahn, the DARPA IPTO director. He was so impressed with the new faculty that he decided to give us major funding. Furthermore, for the first time Columbia had a connection to the ARPAnet. We started bachelors, masters, and Ph.D. programs, and taught computer science to all of Columbia University.

As I mentioned earlier, I started the Journal of Complexity in 1985. In the early years, all papers funneled through me. I realized I was a bottleneck, so I created an Editorial Board, who could independently accept or reject papers or require revisions. The journal is going strong in its 26th year.

In 1986 I was asked to create a computer science board for the National Research Council (NRC). It's now called the Computer Science and Telecommunications Board (CSTB). Our first report, for which the late Michael Dertouzos did much of the writing, was called The National Challenge in Computer Science and Technology. We had a fairly difficult time getting it through the very thorough NRC review process. Much of this report was devoted to policy, whereas my impression was that NRC was more comfortable with technology. CSTB continued to work on policy as well as technology, and in time, that became highly appreciated at the NRC. I rotated off CSTB in 1992 and then served as chair again in 2005 – 2009. To see what reports CSTB has completed and what projects are currently underway, visit `www.cstb.org`.

I had a hand in building four organizations: the CS department at CMU, the CS department at Columbia, the Journal of Complexity, and the Computer Science and Telecommunications Board. The common ingredient for success was excellent people.

EG: You mentioned your role in building organizations. Did you also play a role in the creation of other entities?

JT: Because I got into computing so early, I had such opportunities. For example, I was one of the founders of the Computer Research Association (CRA) in 1972. We decided to create what became the Federated Computing Research Conference (FCRC) at a meeting which I believe was held in Washington, D.C. I was a founding member of the scientific advisory committee (ISAT) of DARPA in 1986.

EG: Do you have any regrets about something you did not pursue?

JT: There is something important I should have done. Starting in 1985, I noticed various ways in which our information infrastructure was vulnerable to electronic or physical attack. I imagined myself to be a terrorist, or an enemy country, and targeted aspects of what we would today call the national information infrastructure. I felt it was just because we were the most advanced country in our use of information technologies, we were and are the most vulnerable. I also felt we were vulnerable to physical attack. Let me give you an example. I was given the opportunity to visit the floor of the New York Stock Exchange one morning just before it opened. The only visible security was one guard, equipped with a revolver. There may, of course, have been security that was not visible to me, but I doubt it. I thought what a tempting target the symbolic heart of our capitalist society this would make, and the damage a couple of hand grenades would inflict. The actual processing of trades was executed across the river in Brooklyn but I doubted that it was sufficiently secure against physical or electronic attack.

I did not go public with my concerns because I was worried about giving individuals or countries ideas. That was foolish; our enemies are very smart. I now feel I should've spent a considerable amount of my energy and time alerting the country.

I finally went public when I gave the keynote address at a symposium at the National Academy of Sciences celebrating the tenth anniversary of CSTB in 1996 [23]. I pointed out the vulnerability of what I called the virtual estate, which consists of bank accounts, equities, CDs, pension accounts, etc. I called it the virtual estate because it's recorded in electrons. If you were a terrorist, and wanted to do a great deal of damage to American institutions and individuals, a natural target would be the virtual estate. Our virtual estate is just one example of a potential target. Others include the power grid, and our communications systems.

Who should be in charge of protecting our infrastructure? I argued for strong Federal government leadership, centered in the executive branch.

EG: Can you say something about the future of computing, especially as it relates to your interests?

JT: That is such a deliciously open-ended question–I'll confine myself to just four issues.

The first has to do with scaling laws. Perhaps the most famous scaling law is Moore's Law, an empirical law which has driven computing for almost half a century. Moore's Law is running up against a number of fundamental physical limits. For a while we will benefit from multicores, many cores, and massive parallelism. But there are considerable impediments to parallel computing. Parallel machines are difficult to program, and some problems are difficult to decompose. That is why there's much interest in radically different kinds of computing, such as quantum, photonic, molecular, and biological computing. Of course Moore's Law is not the only important scaling law. Another example is the doubling rate of bandwidth, which is much shorter than that of chip density. How should we plan our computing and networking in light of this effect?

A second area is information-based complexity (IBC). Quite a few years ago, I ended an MIT lecture by stating some open questions. Afterwards, Marvin Minsky told me he saves his good questions for his students. I replied that there were lots more where those came from. That's always been the case in IBC. We end many papers and talks with a list of open problems. For example, Erich Novak and Henryk Wozniakowski are writing a three-volume monograph, Tractability of Multivariate Problems, Volume I [10] listed thirty open problems, while Volume II, which has just been published, has sixty-one more. Volume III, which is expected in 2012, will list many additional open problems. Why are there so many open problems? I believe it's because we're asking new questions about many continuous scientific problems, a vast domain. Furthermore, when the technology changes, or might change, that alters what algorithms are permitted. A good example is quantum computing, where, for example, we're investigating the power of quantum computing for solving the problems of quantum mechanics.

A third area is cybersecurity, far more than just the protection of the virtual estate. Two very different issues have recently been studied by the Computer Science and Telecommunications Board. One is deterrence strategies for the U.S. government towards preventing cyberattacks. A recent letter report on this topic is now available; see www.cstb.org. Another is in regard to U.S. acquisition and use of cyberattack capabilities; a recent CSTB report is also available. Cybersecurity can only become still more important.

I'll end with concern about computer science majors. There is an odd dissonance between the feelings of prospective students and their parents, on the one hand, and university computer science faculty, on the other. Students seem reluctant to study computer science for two reasons: concern jobs will be outsourced; and a feeling that the big advances are behind us. But my colleagues and I feel that computer science is more exciting than ever. Bill Gates has expressed concern about not being able to hire American computer science majors. We worry that women are not attracted to computer science, which cuts us off from half the brains of the country. This is at the very time that China is making huge investments in computer science education. The key to the country's future is innovation, and it's vital that computing attract some of our country's best and brightest.

Cristian Calude: In your 1998 paper Non-Computability and Intractability: Does It Matter to Physics? you write I'm not convinced that non-computability need be of concern. What is your current position regarding this statement.

JT: I will briefly summarize the issue for the benefit of the reader. See also [28], Chapter 9. Consider, for example, partial differential equations with computable initial conditions but non-computable solutions. The equations can be very simple. Examples are the wave equation with initial conditions which are not twice differentiable and the backwards heat equation. The renowned physicist and mathematician Roger Penrose is concerned by the result that the wave equation with computable initial conditions can have non-computable solutions; he called this a rather startling result, [11]. Is this really a startling result? Let's have a deeper look.

The differential equations mentioned about are special cases of ill-posed equations. Werschulz [33] proved that if a problem is ill-posed it is impossible to compute an ε-approximation to the solution at finite cost even for arbitrarily large error ε. But this is a worst-case result. There is a surprising result that every ill-posed problem is well-posed on the average for every Gaussian measure; see [27] for a survey of the work leading to this result. Thus the non-solvability of ill-posed problems is a worst-case phenomenon. It melts away in the average case for reasonable measures. Nothing has happened to make me change my mind that non-computability may not be a cause for concern for physicists. As in the above example it may simply be a worst case phenomenon.

CC: It seems to me your concern was that non-computability could be bad for physics. What about the possibility of being an asset?

JT: I agree that non-computability can cut in two ways. Its analogous to the situation in cryptography. We want secure encryption to protect out private information. On the other hand, we want to read encrypted messages between terrorists to foil the planning of attacks.

CC: Starting with Ralph Gomary's tripartite division of science into the known, the unknown which may someday became known, and the (most interesting) unknowable, the part which will never be known you've written My goal is to move the distinction between the unknown and the unknowable from philosophy to science. Did you make further steps towards achieving this goal, and if yes, can you summarize them?

JT: I'd like to begin with some background regarding my interest in this issue. That quote comes from [25]. The first time I wrote about this issue was in 1991 [22]. I gave a talk on What is Scientifically Knowable at a symposium celebrating the 25th anniversary of the Computer Science Department at CMU and this reference is in the anniversary commemorative. Gödel's work has had a profound impact on mathematics. It established fundamental limits on mathematical proofs. It was an enrichment of mathematics. I hoped that establishing limits to science by proving that the answers to certain scientific questions were unknowable would be an enrichment of science. How might one prove that the answer to a question is unknowable? I've proposed several possible attacks. The first attack is the following. A scientific question does not come equipped with a mathematical model.

Researchers develop models for scientific questions. Consider then all formal models that capture the essence of a scientific question. Prove that every formal model is undecidable, or computationally intractable. It seems to me that would be one way of proving the answer is unknowable. However, although this might be a possible attack in principle it is far from evident that it could actually be carried out for any nontrivial question. A different attack is proposed in [24]. Rather than a direct attack, which considers all mathematical models, perhaps an indirect attack would have more chance of success. Computational complexity might serve as a guidepost. The intractability theorems of information-based complexity are not proven by varying algorithms. Instead general theorems are proven from which we can infer intractability of specific mathematical problems. Can this procedure be adapted to derive negative properties (undecidability, intractability) which any mathematical model, for a certain scientific question, must possess? I still think the question of how to distinguish the unknown from the unknowable is an interesting question. However, since the papers I wrote in the 90's I've moved into other research areas.

References

[1] W. S. Brown, J. R. Pierce, J. F. Traub, The future of scientific journals, Science, 158: 1153-1159 (1966).

[2] L. Blum, M. Shub, and S. Smale, On a theory of computation and complexity over the real numbers: NP-completeness, recursive functions, and universal machines, Bull. Amer. Math. Soc. 21: 1-46 (1989).

[3] W. S. Brown, J. F. Traub, MERCURY – A system for the computer aided distribution of technical papers, Journal ACM, 16: 13-25 (1969).

[4] R. Brent and J. F. Traub, On the complexity of composition and generalized composition of power series, SIAM Journal Computation, 9: 54-66 (1980).

[5] M. A. Jenkins and J. F. Traub, A three-stage variable-shift iteration for polynomial zeros and its relation to generalized Rayleigh iteration, Numerische Mathematik, 14: 252-263 (1970).

[6] D. E. Knuth, Algorithms in mathematics and computer science, in Lecture Notes in Computer Science, G. Goos and J. Hartmanis, editors, 122: 82-99 (1979).

[7] H. T. Kung and J. F. Traub, All algebraic functions can be computed fast, Journal ACM, 25: 245-260 (1978).

[8] M. A. Nielsen, E. L. Chuang, Quantum Computation and Quantum Information, Cambridge University Press (2000).

[9] E. Novak, I. Sloan, J. F. Traub, H. Wozniakowski, Essays on the Complexity of Continuous Problems, European Mathematical Society (2009).

[10] E. Novak and H. Wozniakowski, Tractability of Multivariate Problems, vol. 1, Linear Information, European Mathematical Society (2008).

[11] R.L. Penrose, The Emperor's New Mind, Oxford University Press (1989).

[12] S. Paskov, J. F. Traub Faster evaluation of financial derivatives, Journal of Portfolio Management, 22: 113-120 (1995).

[13] A. Papageorgiou, J. F. Traub, Beating Monte Carlo, RISK, 9: 63-65 (1996).

[14] A. Papageorgiou, J. F. Traub, Quantum algorithms and complexity for continuous problems, in Springer Encylopedia of Complexity and Systems Science, 8: 7118-7135 (2009).

[15] A. Papageorgiou, J. F. Traub, Qubit complexity of continuous problems, J. Fixed Point Theory and Applications, 6, no. 2: 295-304 (2009).

[16] M. Shaw and J. F. Traub, On the number of multiplications for the evaluation of a polynomial and some of its derivatives, Journal ACM, 21: 161-167 (1974).

[17] J. F. Traub, Functional iteration and the calculation of roots, Proceedings, National ACM Conference: 5A-1 – 5A-4 (1961).

[18] J.F. Traub, Iterative Methods for the Solution of Equations, Prentice-Hall (1964). Reissued, American Mathematical Society (1998).

[19] J. F. Traub, A class of globally convergent iteration functions for the solution of polynomial equations, Math. Comp., 20: 113-138 (1966).

[20] Complexity of Sequential and Parallel Numerical Algorithms, J. F. Traub, editor, Academic Press (1973).

[21] J. F. Traub, Parallel algorithms and parallel computational complexity, Proceedings IFIP Congress: 685-687 (1974).

[22] J.F. Traub, What is scientifically knowable? in CMU Computer Science: A 25th Anniversary Commemorative, Addison-Wesley: 489-503 (1991).

[23] J. F. Traub, From infoware to infowar, in Defining a Decade: Envisioning CSTB's Second Ten Years, National Academy Press: 1 – 7 (1997).

[24] J.F Traub, Do negative results from formal systems limit scientific knowledge?, Complexity, 3, No. 1: 29-31 (1997).

[25] J.F. Traub, The unknown and the unknowable, The Sciences, 39, No.1: 39-44 (1999).

[26] J. F. Traub, A brief history of information based complexity, in Essays on the Complexity of Continuous Problems, European Mathematical Society: 61-71 (2009).

[27] J.F. Traub and A.G. Werschulz, Linear ill-posed problems are solvable on the average for all Gaussian measures, Math. Intelligencer, 16, No. 2: 42-48 (1994).

[28] J. F. Traub and A. G. Werschulz, Complexity and Information, Cambridge University Press (1998).

[29] J. F. Traub and H. Wozniakowski, A General Theory of Optimal Algorithms, Academic Press (1980).

[30] J. F. Traub, G. Wasilkowski, and H. Wozniakowski, Information, Uncertainty, Complexity, Addison-Wesley (1983).

[31] J. F. Traub, G. Wasilkowski, and H. Wozniakowski, Information-based Complexity, Academic Press (1988).

[32] P. van Emde Boas, Machine models and simulations. Pp.1-66 of: van Leeweun, J. (ed.) Handbook of Theoretical Computer Science: vol. A, Algorithms and Complexity, MIT Press (1990).

[33] A.G. Werschulz, What is the complexity of ill-posed problems?, Numer. Funct. Anal. Opt., 9:945-967 (1987).

[34] Jenkins-Traub algorithm, Wikipedia.

[35] Quasi-Monte Carlo methods in finance, Wikipedia.