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 meeting *Analysis and Randomness*, `http://www.cs.auckland.ac.nz/~nies/ARAhome.html`, organised by A. Nies, was held in Auckland on 12–13 December 2011. Speakers: Laurent Bienvenu, Willem Fouche, Cameron Freer, Rupert Hölzl, A. Melnikov, Kenshi Miyabe, Jason Rute, Tom ter Elst, and Dan Turetsky.

1. The *12th Asian Logic Conference*, `http://msor.victoria.ac.nz/Events/ALC2011/WebHome`, was held 15–20 December 2011 in Wellington. Plenary Speakers: Hiroakira Ono, Mic Detlefsen, Akito Tsuboi, Noam Greenberg, Simon Thomas, Isaac Goldbring, Grigor Sargsyan and Wu Guohua. Several special sessions have been organised, including *Algorithmic Randomness* (by R. Downey and A. Nies), and *Computability and Algebraic Structures* (by R. Downey).

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

407. K. Svozil. Neutrino Dispersion Relation Changes Due to Radiative Corrections as the Origin of Faster-than-Light-in-Vacuum Propagation in a Medium. 09/2011

408. A.A. Abbott, C.S. Calude and K. Svozil. On Demons and Oracles. 11/2011
409. C.S Calude and E. Calude. The Complexity of Euler's Integer Partition Theorem. 11/2011
410. C.S Calude and E. Calude. The Complexity of Mathematical Problems: An Overview of Results and Open Problems. 11/2011
411. L. Staiger. On Oscillation-free Chaitin h-random Sequences. 11/2011
412. L. Staiger. Asymptotic Subword Complexity. 11/2011
413. D.H. Bailey, J.M. Borwein, C.S. Calude, M.J. Dinneen, M. Dumitrescu and A. Yee. An Empirical Approach to the Normality of π. 11/2011
414. S. Datt and M.J. Dinneen. Towards Practical P Systems: Discovery Algorithms. 12/2011
415. R. Nicolescu. Parallel and Distributed Algorithms in P Systems. 12/2011
416. M. Burgin, C.S. Calude and E. Calude. Inductive Complexity Measures for Mathematical Problems. 12/2011

2 A Dialogue with Reinhard Wilhelm about Compiler Construction and Dagstuhl

Reinhard Wilhelm is professor and leader of the chair for programming languages and compiler construction at Saarland University and the scientific director of the Leibniz Center for Informatics at Schloss Dagstuhl since its inception in 1990.

Professor Wilhelm has obtained numerous results in compiler construction, static program analysis, embedded real time systems, animation and visualization of algorithms and data structures. He is one of the co-developers of the MUG1, MUG2 and OPTRAN compiler generators, which are based on attribute grammars. He is a co-founder of the European Symposium on Programming, ESOP, and the European Joint Conferences on Theory and Practice of Software, ETAPS, a member of the ACM SIGBED Executive Committee and a member of the Scientific Advisory Board of CWI.

Professor Wilhelm is a fellow of the ACM (2000) and a member of Academia Europaea (2008); he was awarded the Alwin-Walther medal (2006), the Prix Gay-Lussac-Humboldt (2007), the Konrad-Zuse medal (2009), the Cross of the Order of Merit of the Federal Republic of Germany and the ACM Distinguished Service Award (2010); he has honorary doctorates from RWTH Aachen and Tartu University (2008).

Cristian Calude: You studied mathematics, physics and mathematical logic at University of Münster, computer science at Technical University Munich and Stanford University and obtained your PhD at TU Munich, quite a broad background. Please reminiscence about this period.

Reinhard Wilhelm: I studied at a time when the first curricula in computer science were being established. As a native of Westphalia, Westfälische Wilhelms Universität Münster with its strong tradition in Mathematics and Mathematical Logic was a natural starting point. Josef Stör, a numerical analyst from my home town, on the faculty of USC San Diego, recommended to switch to computer science, an advice I followed after passing the Vordiplom exam in Münster. At TH, later TU Munich, I was among the first students of the new curriculum in computer science. I finished this obtaining a Diploma degree, already oriented towards compiler construction. The German Academic Exchange Service (DAAD) offered one-year fellowships to study computer science in the US as they felt that the CS curricula did not yet have the same quality as the American curricula. I obtained such a fellowship and studied at Stanford University for one year. It was an exciting year with courses taught by Robert Floyd, Donald Knuth, Zohar Manna, John McCarthy, Robin Milner, and Niklaus Wirth. Looking back, the semantics people, Floyd, Manna, and McCarthy, seemed to have had the strongest influence on me. I gathered practical experience in compiler construction with my MS project, part of the port of the Zurich Pascal compiler to the IBM 360 machine.

CC: You discovered connections between code selection and regular tree automata, which are relevant for code generation.

RW: My group at Saarland University developed a formally-based approach to compiler optimizations expressed as transformations of attributed trees. The necessary tree pattern-matching algorithm—identifying places where transformations could be applied—used a subset construction on non-deterministic tree automata as I learned later from Helmut Seidl. I found some informal proposals in the literature proposing to express code selection by tree parsing. This led to a beautiful and efficient approach using deterministic bottom-up tree automata, which could be nicely combined with dynamic programming to identify least-cost code sequences. However, reality, i.e. processor-architecture design, made this beautiful approach obsolete as real processor architectures did not offer the required regularity.

CC: Although your research is quite practical, the theoretical component is strong. How do you manage this?

RW: Well, the colleagues in the CS department at Saarland University have a strong conviction, that nothing is as practical as a good theory. This conviction has been a recipe for success. Our curriculum has always had a strong theoretical foundation on which one could build solid practical work.

CC: Reinhard Wilhelm and Dieter Maurer's book *Compiler Design*—written in German and translated into English and French—is a good illustration of the interplay between theory and applications: it offers a solid theoretical foundation for compilers for imperative, object oriented, functional and logic-based languages.

RW: I was not content with the Dragon Book, the dominant compiler textbook, which was and is by and large void of the theoretical foundations for compiler design. The underlying theory, however, is quite beautiful. So, I decided to write a book that I would like to teach from. I was fortunate to have Dieter Maurer in my group, who coauthored the first two editions. Currently, I cooperate with Helmut Seidl and Sebastian Hack on a rather complete rewrite for the third edition. The virtual machines in this third edition are made more uniform. The code-optimization part introducing static program analysis and program transformations has been largely extended. The code-generation chapters will be completely restructured and rewritten due to new insights into the code-generation process obtained in Sebastian Hack's dissertation.

CC: Please explain the shape analysis based on three-valued logic you designed.

RW: Static program analysis, which received most of its theoretical foundations by Patrick and Radhia Cousot in the 70s, computes invariant properties of all behaviors of a program. Abstract interpretation, as the Cousots formulated it, uses an abstraction of the semantics of the programming language to determine these invariants at all program points. Due to the impossibility to be sound and complete at the same time, sound static analysis approximate these properties; they give up completeness, but maintain soundness.

A largely unexplored area was the static analysis of heap-manipulating programs. These offer particular challenges, namely dynamically created anonymous objects and linked data structures of unbounded size. During a sabbatical I spent in Israel I was fortunate to meet Mooly Sagiv, then a student at the Technion. He asked me for a good thesis topic, and I proposed to develop a specification language for static program analyses. Mooly and I cooperated on this topic for something like 16 years, joined by Tom Reps, whom I knew from our attribute-grammar times.

The breakthrough in our research came with the discovery that predicate logic was a good basis to express program semantics, and that a reinterpretation of the same semantics over a 3-valued logical domain—the third value expressing *don't know*—could be used as an abstract interpretation. Our approach was parametric in the abstraction properties, i.e., different sets of predicates could be used to obtain different abstractions, which would (approximately) different properties of the program.

The *shapes* occurring in the name *Shape Analysis*were somethin like generalized *types* of data structures in the heap. Example are *singly-linked list without shared nodes*, *balanced binary tree* etc.

CC: Your ACM fellowship citation refers to your research on compiler construction and program analysis. Can you discuss one or two important results in this area?

RW: A result of my group that had quite some impact is the development of an approach to derive run-time guarantees for real-time embedded systems, that is, to show that such systems satisfy their timing constraints. These are often quite tight; in the automotive domain, they range down to microseconds. At the same time, the execution platforms used to realize these systems have a huge variability of execution times: the execution time of an instruction depends on the state of the platform and may vary by a factor of 100 or more.

The engineers at Airbus in Toulouse called us to help them because they knew that their traditional methods, based on measurement, were not sound for the new architectures they were deploying in their planes. We were able to solve this problem and provide tools through a spinoff company, AbsInt, that Airbus could use. The meanwhile long cooperation between Airbus, AbsInt, and my group at the University was so successful that several time-critical subsystems of the Airbus A380, the *big Airbus*, were certified with the AbsInt tool, which thereby became the only tool worldwide to be *validated* for the certification of these avionics applications. This work is considered as one of the major success stories of formal methods.

CC: How do you see your book *Informatics: 10 Years Back. 10 Years Ahead* (Springer 2001), 10 years after its publication?

RW: That is hard to answer! I would have to reread the prognoses contained in it. In the domain of verification, I have recently coauthored a manifesto, *Formal Methods—Just a Euroscience?* attempting to describe the state of the art. This could be compared with the articles in the monograph you refer to.

CC: Please summarise your manifesto.

RW: The manifesto gives an overview of how far different formal methods, in particular the verification methods, have been taken up by industry. There are notable differences between hardware and software industries and also some between Europe and America. The acceptance of verification methods is related to the costs of potential failures. The chip manufacturers have broadly adopted verification methods after the Pentium bug cost Intel a lot of money. The Ariane 5 disaster due to a software bug was very helpful to raise problem awareness in some parts of the embedded-systems industry. There is a somewhat surprising distribution of strongholds for the different verification methods; model checking is stronger in the US, abstract interpretation stronger in Europe, deductive verification initially stronger in the US, but now strong in Europe.

One particular insight I gained in my work with industry and which is described in the manifesto is that the different verification techniques have a different distribution of roles, researcher, tool developer, user. In academia, typically the researcher also develops the tools based on his findings, and, of course, he is an enthusiastic user of his own tools. Some of the biggest disappointments resulted from the expectations raised by enthusiastic researchers/tool developers when the tools were deployed in industry and engineers could not use them.

CC: Since 1990 you have been the scientific director of the Leibniz Center for Informatics at Schloss Dagstuhl. What was the initial motivation of starting this center? How did it evolve in the last twenty years?

RW: The Leibniz Center for Informatics was formed after the famous Mathematics Research Institute in Oberwolfach, in the Black Forest. Theoretical computer scientists had been guests there for a number of years and felt the desire to have an Oberwolfach for Informatics. The German Informatics Society (GI) set up a search committee to identify an appropriate place for it. Several offers were made by the states Baden-Württemberg, Rheinland-Pfalz, and Saarland. The search committee selected Schloss Dagstuhl, a late-baroque mansion, at that time a retirement home run by a nuns order. The Saarland government agreed to buy the ensemble for the center and the German National Science Council supported the decision to set the center up in Dagstuhl.

Apparently, the Informatics community had waited for this center. Against my expectations it filled up rather quickly. An extension building was opened in 1995 together with a new kitchen and a restaurant. The greater capacity also filled up quickly so that lead times of far more than a year became common. You must know that meetings in Dagstuhl, the so-called Dagstuhl Seminars, result from successful applications to a Scientific Directorate, which meets twice a year to decide about the submitted proposals.

CC: Yes, I indeed know as I was privileged to be invited to a few seminars. As a participant to both Oberwolfach and Dagstuhl, I noted similarities but also differences...

RW: Definitely, Oberwolfach was our role model when we set up Dagstuhl. When I had been convinced to run Dagstuhl, I went to Oberwolfach together with my colleague on the administrative side, Wolfgang Lorenz, to get advice from Martin Barner, the long-time director of the Mathematical Research Institute, on what to do and, even more importantly, what not to do. Among the latter was his recommendation not to establish entailed estates, that is, long running series of meetings, which ran too long to be ever stopped. We, therefore, established an iron rule that the organizing team of a series had to, at least incrementally, change from instance to instance. This was not always well received by organizing teams, but proved fruitful in the long run.

Another notable difference to Oberwolfach was that we charged participation fees right from the beginning. Computer scientists usually are better funded than mathematicians, and our fees were more symbolic than covering real costs.

Let me report an anecdote about where Dagstuhl profited from Oberwolfach. I was amazed by the fantastic music room on Oberwolfach. Great instruments and an extensive musical library! Actually, I had met Don Knuth and told him about our plans for Dagstuhl, and he had sent me a letter saying that he had always enjoyed playing the grand piano in Oberwolfach. The White Hall in Dagstuhl, a beautiful baroque hall, offered itself for our music room. I set out to buy instruments, a grand piano—not as grand as the one in Oberwolfach—, a decent violin, a cello. The executive in the ministry in charge of supervising our efforts complained about us acquiring a grand piano. I sent him a copy of Knuth's letter to prove that luminaries like him would find their way to Dagstuhl because of the grand piano. This stopped the complaints.

Another anecdote on setting up the music library in Dagstuhl. Musical scores are very expensive. So I thought about how to save on buying a basic library. I knew that the German publishers had licensed editions to the Eastern countries not meant to be reimported, at least not large scale. At that time I was playing with a Hungarian pianist. I told him my problem and asked him to see how he could import Eastern editions of scores for Dagstuhl. Next time, a friend of his came to visit him, he had the trunk of his car full with scores, somewhat biased towards Southeast Europe, all that for just 1000 DEM. I was nervous about what would happen to the fellow and the smuggled scores at the Austro-Hungarian border, and, in fact, Austrian customs asked the fellow to open the trunk of his car. On top of all the scores, there was a twelve-pack of cigarettes. They made him pay a fine for smuggling cigarettes.

CC: In addition to the music, the dedication to the fine arts is visible in Dagstuhl. What is the origin for this?

RW: Although I have a sister who is an artist my connection to the fine arts was not very strong. That changed when the extension building in Dagstuhl was finished. It is, I think, a beautiful modern architecture based on a traditional concept, a monasterial building. The architects saw Dagstuhl as a scientific monastery. Our monastery has a cloister, a very nice opportunity for arts exhibitions. But what got me really involved with fine art and not so fine artists was the procedure for equipping the new building with artistic objects. Germany has a law requiring that public buildings should be furnished with pieces of art. A certain percentage of the construction money should go to the arts. A jury was set up, some groups of artists were asked to submit proposals. The architect and I were made members of the jury. When the submissions were discussed, I felt that something fishy was going on. I didn't know what. The jury, against my vote, selected some proposal that would deal with computer science in a pubertal way. I was quite upset and told the jury that this work would never see the center. I was declared a philistine, ignorant of modern trends in the fine arts. A four month battle behind the scenes led to the rejection of the jury's proposal by the minister in charge. As a revenge, the jury decided to let the money in the arts budget fall back to the construction budget. We were left with empty walls! I then invented an arts donation scheme, see `http://www.dagstuhl.de/en/about-dagstuhl/kunst/`, which, with a little help by our friends, has helped us to acquire quite a few nice pieces mostly from exhibitions we have had in the cloisters.

CC: What is the "job description" of the scientific director of the Leibniz Center for Informatics?

RW: The Scientific Director is responsible for the scientific program in Schloss Dagstuhl. That is the primary duty. Unlike a conference hotel, the Scientific Directorate, and the whole scientific staff at Schloss Dagstuhl feel responsible to guarantee high-quality meetings. The participants, who spend considerable effort to travel to this remote place, expect a high return for this travel investment. A disappointed participant will most likely not accept another invitation.

The Scientific Director chairs the Scientific Directorate at its meetings, moderates the discussion, and executes the decisions taken.

He also develops or takes up new directions and functions of the center. The Leibniz Center has extended its activities beyond the original function in several directions. It has become an open-access publisher. The high-quality conference series, LIPIcs, provides a low-cost, open-access alternative to established publishers, who, under financial pressure of their owners, were forced to change their publication policy to increase their revenue.

Another new direction is the cooperation with DBLP, the renowned bibliographic database established by Michael Ley at the University of Trier. The Leibniz Center has agreed to secure the long-term existence of this important source of information for computer science. With support from the Leibniz Association and the Klaus Tschira Foundation, DBLP has strongly increased the coverage of computer science publications.

CC: Over the years you have witnessed many interesting events in Schloss Dagstuhl. Are there any such memories which you would like to share with us?

RW: Let me report about two events, one rather sad, one positive. We scheduled a meeting on Computer Science and Astronomy at the time of last total solar eclipse covering central Europe. This meeting included computer scientists, astronomers, and historians. As it brought together different communities that would hardly meet anywhere else it was a quite typical event for Dagstuhl.

One particular talk attempted to refute the then popular claim of some pseudo-historians that three centuries, around 700–1000 ad, had been invented. A historian had collected recordings about solar and lunar eclipses from that time. These were checked against an exciting software reproducing the planetary constellations at any time and any location. And indeed, all recorded eclipses were properly reproduced by this software. Another exciting experience at this event was that we selected exactly the right place to watch the eclipse. More or less all others in Britain, in France, and in Germany did not see anything due to rain and clouds while we had a 20 minutes hole in the clouds through which we could perfectly watch the eclipse.

Now to the sad side. As we know from history, total solar eclipses were always seen as bringing with them mischief, catastrophes, and plagues. To support this old superstition, one participant had an accident coming to the meeting, one fell ill during the meeting, and one had to leave early because his father died.

As mentioned above, it is very common that Dagstuhl meetings bring together different communities that don't have any conference where they would meet. Dagstuhl thus often establishes absolutely necessary communication. Let me report about a meeting about Scheduling. Scheduling is an important topic, which occurs in manufacturing and in logistics—this is typically dealt with in the Operations Research community—, but also in computer science, and in computer science again in different subdomains, e.g. real-time scheduling, compilation, and algorithms. A meeting in 2010 brought together the algorithms community, the real-time scheduling community, and the operations-research community. Some real-time scheduling participants were asked to list their most interesting open problems, which were unknown to the algorithms community. They wrote up a report about their most urgent open problems, and in the proposal to the successor meeting the proposers proudly presented 10 publications that had resulted from this meeting solving at least 5 of the listed open problems of the real-time scheduling community.

CC: Many thanks.