

Reliable Computation and Complexity on the Reals

Edited by

Norbert T. Müller¹, Siegfried M. Rump², Klaus Weihrauch³, and
Martin Ziegler⁴

1 Universität Trier, DE, mueller@uni-trier.de

2 TU Hamburg-Harburg, DE, rump@tuhh.de

3 FernUniversität in Hagen, DE, klaus.weihrauch@fernuni-hagen.de

4 KAIST – Daejeon, KR, ziegler@cs.kaist.ac.kr

Abstract

Naïve computations with real numbers on computers may cause serious errors. In traditional numerical computation these errors are often neglected or, more seriously, not identified. Two approaches attack this problem and investigate its background, Reliable Computing and Computable Analysis.

Methods in Reliable Computing are essentially mathematical theorems, the assumptions of which are verified on the computer. This verification is performed using the very efficient floating point arithmetic. If the verification succeeds, the assertions are true and correct error bounds have been computed; if not, a corresponding message is given. Thus the results are always mathematically correct. A specific advantage of Reliable Computing is that imprecise data are accepted; the challenge is to develop mathematical theorems the assumptions of which can be verified effectively in floating-point and to produce narrow bounds for the solution.

Computable Analysis extends the traditional theory of computability on countable sets to the real numbers and more general spaces by refining continuity to computability. Numerous even basic and simple problems are not computable since they cannot be solved continuously. In many cases computability can be refined to computational complexity which is the time or space a Turing machine needs to compute a result with given precision. By treating precision as a parameter, this goes far beyond the restrictions of double precision arithmetic used in Reliable computing. For practical purposes, however, the asymptotic results from complexity theory must be refined. Software libraries provide efficient implementations for exact real computations.

Both approaches are established theories with numerous important results. However, despite of their obvious close relations these two areas are developing almost independently. For exploring possibilities of closer contact we have invited experts from both areas to this seminar. For improving the mutual understanding some tutorial-like talks have been included in the program. As a result of the seminar it can be stated that interesting joint research is possible.

Seminar November 26–1, 2017 – <http://www.dagstuhl.de/17481>

1998 ACM Subject Classification F.2.1 Numerical Algorithms and Problems, F.1.3 Complexity Measures and Classes, G.4 Mathematical Software, F.4.1 Mathematical Logic

Keywords and phrases Computable Analysis, Verification Methods, Real Complexity Theory, Reliable Computing

Digital Object Identifier 10.4230/DagRep.7.11.142



Except where otherwise noted, content of this report is licensed under a Creative Commons BY 3.0 Unported license

Reliable Computation and Complexity on the Reals, *Dagstuhl Reports*, Vol. 7, Issue 11, pp. 142–167

Editors: Norbert T. Müller, Siegfried M. Rump, Klaus Weihrauch, and Martin Ziegler



Dagstuhl Reports

Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1 Executive Summary

Norbert T. Müller

Siegfried M. Rump

Klaus Weihrauch

Martin Ziegler

License © Creative Commons BY 3.0 Unported license

© Norbert T. Müller, Siegfried M. Rump, Klaus Weihrauch, and Martin Ziegler

The seminar was a meeting between two groups of researchers working in the related areas of reliable computing and of computational complexity on real numbers. While the first area originates in numerical analysis, the second area goes back to the roots of computer science and and computability.

Reliable computations aims to produce correct answers to numerical problems with mathematical rigor. This includes to prove that the problem is solvable and to compute mathematically correct error bounds for the solution. Reliable numerical computations solely use floating-point arithmetic to take advantage of the tremendous speed. Naturally that poses limits on the problems which can be solved, in particular the condition number. However, in contrast to purely numerical methods, no false answers are possible: Either a true error bound is computed or, a corresponding error message is given. There is a history of reliable numerical computations. In the early days, interval arithmetic was often used in a rather naive way. Still the computed results were correct, however, often wide or no bounds at all were computed. Meanwhile it is well understood how to derive effective methods for reliable numerical computations, avoiding wide bounds and pushing the set of solved problems to the limit of that of purely numerical algorithms. A number of interesting and hard mathematical problems have been solved using reliable numerical computations. This includes the famous Kepler conjecture, the existence of mutually distinct solutions to certain partial differential equations, and more. Needless to say that solving a mathematical problem requires rigorous solutions of all particular problems.

Computable analysis is a branch of computability theory studying those functions on the real numbers and related structures which can be computed by machines such as digital computers. The increasing demand for reliable software in scientific computation and engineering requires a sound and broad foundation not only of the analytical/ numerical but also of the computational aspects of real number computation. The branch of computable analysis based on the definition by Grzegorzczuk and Lacombe of computable real functions (TTE, “Type 2 Theory of Effectivity”) has turned out to be particularly useful for investigating computability on uncountable sets. As a central concept computability appears as a specialization of continuity. Meanwhile computability of numerous analytic problems has been investigated (from basic analysis, functional analysis, ordinary and partial differential equations, analytic functions, measure theory, dynamical systems etc.). All these examples demonstrate the usefulness of the concept.

Once a problem has been shown computable, a natural next question asks for the computational efficiency of such a solution. This is where real analysis meets (discrete) complexity theory with notions of runtime and memory/space: asymptotically with respect to $n \rightarrow \infty$ for approximating the output up to absolute error 2^{-n} . The famous Bailey-Borwein-Plouffe method for instance permits to compute billions of digits of transcendental within minutes; while Bloch’s constant, although proven computable, is still not known up to error 2^{-5} . In fact the distinction between polynomial and exponential time, in the discrete realm gauged for instance by complexity classes P, NP, #P, and PSPACE, re-emerges in

the real case: The bit-cost of computing the maximum of an arbitrary fixed smooth (i.e. infinitely often differentiable) polynomial-time computable $f : [0; 1] \rightarrow [0; 1]$ has been shown to correspond to P-vs-NP; that of Riemann integration to #P; and that of solving an ordinary differential equation to PSPACE. On analytic functions on the other hand these operations map polynomial-time computable instances back to polynomial-time computable results.

For practical purposes and in the spirit of “algorithm engineering”, the asymptotic results from complexity theory have to be refined by considering the efficiency of actual implementations. Corresponding software libraries are usually called “exact real arithmetic” (ERA) and implement real numbers in the sense of TTE. ERA implementations exist in many languages, like C, C++ JAVA, Haskell or OCaml. Internally, ERA has to perform operations on infinite data like $\{0, 1\}^\omega$. The user interface, however, hides the details and offers operations and functions on “exact” real numbers. In consequence, users do not need to care about aspects like rounding or truncation errors or the specification of precisions. Instead, they can concentrate on the mathematical part of the problem under consideration. As computable real functions have to be continuous, it is impossible to implement some widely used real functions (like testing on equality). In consequence, ERA cannot simply copy the double precision interface one-to-one, but needs to go its own ways. Additionally, for the reason of efficiency the representations used in TTE have to be carefully revised. The resulting speed is comparable to the use of multiple precision floating point numbers, but now without any need for manual precision control.

2 Table of Contents

Executive Summary

Norbert T. Müller, Siegfried M. Rump, Klaus Weihrauch, and Martin Ziegler . . . 143

Overview of Talks

Reliable Visual Analytics within a Verification and Validation Management <i>Ekaterina Auer and Wolfram Luther</i>	148
Orbital stability investigations for travelling waves in a nonlinearly supported beam <i>M. Plum, B. Breuer, J. Horak, K. Nagatou, and P. J. McKenna</i>	148
Command-like Expressions for Real Infinite-precision Calculations <i>Andrej Bauer, Sewon Park, and Simpson, Alex (University of Ljubljana, SI)</i>	149
Bounds for eigenvalues of an eigenvalue problem with non-smooth coefficients <i>Henning Behnke</i>	150
Computing positive invariant sets with intervals <i>Benoît Zerr, Luc Jaulin, and Thomas Le Mézo</i>	150
On the Taylor model approach for solving ODEs <i>Florian Bünger</i>	150
Uncertainty Quantification: Probabilistic Forecasts of Energy Demand <i>George F. Corliss</i>	151
Nonlinear Symbolic Transformations for Simplifying Functions – Applied for Interval Based Global Optimization <i>Tibor Csendes and Elvira Dobjánné Antal</i>	151
Daisy – a framework for sound accuracy analysis and optimization of finite-precision programs <i>Eva Darulova</i>	152
Computability of geometric Lorenz attractors <i>Daniel Graça</i>	152
Interval computations with compensated algorithms <i>Stef Graillat</i>	153
On the Computational Complexity of the Range Computation Problem <i>Peter Hertling</i>	153
Formal Verification of a Rigorous ODE Solver <i>Fabian Immler</i>	153
Towards certified exact real computation <i>Sunyoung Kim</i>	154
Functional exact real computation, towards verification of total correctness <i>Michal Konecny</i>	154
Computable Numbers, Computable Sets, and Computable Functions and How It Is All Related to Interval Computations <i>Vladik Kreinovich</i>	154

Need to Combine Interval and Probabilistic Uncertainty: What Needs to Be Computed, What Can Be Computed, What Can Be Feasibly Computed, and How Physics Can Help <i>Vladik Kreinovich</i>	155
An Approach to Programming Configurable Computers for Numeric Applications <i>Fritz Mayer-Lindenberg</i>	157
On implementing TTE <i>Norbert T. Müller</i>	157
On the numerical norm estimation of the inverse operator in Hilbert space <i>Mitsuhiro T. Nakao</i>	158
Parametrised complexity for the naive Cauchy representation <i>Eike Neumann and Florian Steinberg</i>	158
Test Matrices for Numerical Linear Algebra <i>Katsuhisa Ozaki</i>	159
A tutorial on reliable numerical computation <i>Paul Zimmermann</i>	159
Computational complexity of solving polynomial differential equations over unbounded domains <i>Amaury Pouly and Daniel Graça</i>	160
Numerical (arte-)facts and reliable computing <i>Siegfried M. Rump</i>	160
Co-Polish spaces in Complexity Theory <i>Matthias Schröder</i>	160
Bit complexity of Computing Solutions for Symmetric Hyperbolic Systems of PDEs with Guaranteed Precision <i>Svetlana Selivanova</i>	161
Parametrised second-order complexity theory with applications to the study of interval computation <i>Florian Steinberg and Eike Neumann</i>	162
Verified computations for solutions to 1-dimensional advection equations with variable coefficients <i>Akitoshi Takayasu</i>	163
Average case complexity for Hamiltonian dynamical systems <i>Holger Thies, Akitoshi Kawamura, and Martin Ziegler</i>	163
Small divisors and normal forms <i>Warwick Tucker</i>	164
Soft Foundations for Geometric Computation <i>Chee K. Yap</i>	164
On Formal Verification in Imperative Multivalued Programming over Continuous Data Types <i>Martin Ziegler, Gyesik Lee, Norbert T. Müller, Eike Neumann, Sewon Park, and Norbert Preining</i>	165

Norbert T. Müller, Siegfried M. Rump, Klaus Weihrauch, and Martin Ziegler 147


Tutorial on Computational Complexity in Analysis
Martin Ziegler and Akitoshi Kawamura 166

Participants 167

3 Overview of Talks

3.1 Reliable Visual Analytics within a Verification and Validation Management

Ekaterina Auer and Wolfram Luther

License  Creative Commons BY 3.0 Unported license
© Ekaterina Auer and Wolfram Luther


A recently described four tier verification and validation management (VVM) defines requirements for categorization and classification of processes as a result of precise assessment procedures and addresses recommending techniques, user interaction and collaboration via adequate human machine interfaces. Huge data and program code require new visual analysis methods. Reliable visual analytics is paired with an assessment of (meta)data and code quality, adequate data types and methods to propagate and bound uncertainty. In our talk, we present an ontology-based architecture with a query engine and modern human machine interaction and requirements from the VVM for various use cases: GPS sensing and localization, spatial decision making, risk communication and perception, analysis of steel samples using SILENOS[©], virtual museums and labs toolbox ViMEDEAS, biomechanics: e.g., femur prosthesis (PROREOP) as well as modeling and implementation of a microscopic traffic simulation (OLSIMv4).

References

- 1 E. Auer, W. Luther, Numerical Verification Assessment in Computational Biomechanics in: Proc. of Dagstuhl Seminar 08021: Numerical Validation in Current Hardware Architectures, Lecture Notes in Computer Science 5492. Springer (2009) 145–160
- 2 B. Weyers, W. Luther, Risk Communication and Perception in Low- and High-Immersion Virtual Environments, APSSRA6, 28-30 May 2016, Shanghai, China, H.W. Huang, J. Li, J. Zhang & J.B. Chen (editors) (2016)
- 3 G. Rebner, D. Sacher, B. Weyers, W. Luther, Verified stochastic methods in geographic information system applications with uncertainty, Structural Safety 52 (2015) 244–259
- 4 D. Sacher: A generative approach to virtual museums using a new metadata format. A curators', visitors' and software engineers' perspective. Logos, Berlin (2017) ISBN 978-3-8325-4627-4, 288p.
- 5 M. Thureau, Chr. Buck, and W. Luther, IPFViewer: Incremental, approximate analysis of steel samples, in Proceedings of SIGRAD 2014, Linköping University Electronic Press (2014) 1–8
- 6 Publications and projects SCG: <http://www.scg.inf.uni-due.de/>

3.2 Orbital stability investigations for travelling waves in a nonlinearly supported beam

M. Plum, B. Breuer, J. Horak, K. Nagatou, and P. J. McKenna

License  Creative Commons BY 3.0 Unported license
© M. Plum, B. Breuer, J. Horak, K. Nagatou, and P. J. McKenna

For a nonlinear beam equation on the whole real line with exponential nonlinearity, we prove existence of at least 36 travelling wave solutions for the specific wave speed $c=1.3$. Our proof makes heavy use of computer assistance: starting from numerical approximations, we

use a fixed point argument to prove existence of solutions “close to” the approximate ones. Moreover we investigate the orbital stability of these solutions via computation of their Morse indices, using classical theoretical results by Grillakis, Shatah, and Strauss. Also for these stability investigations we make use of both analytical and computer-assisted techniques.

3.3 Command-like Expressions for Real Infinite-precision Calculations

Andrej Bauer (University of Ljubljana, SI), Sewon Park (KAIST – Daejeon, KR), and Simpson, Alex (University of Ljubljana, SI)

License © Creative Commons BY 3.0 Unported license

© Andrej Bauer, Sewon Park, and Simpson, Alex (University of Ljubljana, SI)

We present an idealized programming language, *Clerical*, for exact real-number computation. Clerical is an imperative language with conditional statements, **while** loops and local mutable variables. Its primitive datatypes are the Booleans, the integers and the reals. Basic arithmetical operations and comparison tests are provided. As is usual and necessary, comparison operators on reals may be non-terminating.

Clerical supports Dijkstra’s guarded non-deterministic choice

case $b_1 \Rightarrow c_1 \mid b_2 \Rightarrow c_2$ **end**

The intended meaning is that c_1 may execute if b_1 is true, and c_2 may execute if b_2 is true, and one of the branches will execute as long as b_1 or b_2 is true. When b_1 and b_2 hold, either branch may execute, which leads to *multivalued* computations that may return one of several possible results. The **case** construct is used to circumvent non-decidability of $<$ on the reals. In a typical application we perform an *approximate* test $x < y$ with precision ϵ by running

case $x < y + \epsilon \Rightarrow c_1 \mid y < x + \epsilon \Rightarrow c_2$ **end**.

One of the tests will always succeed and the corresponding branch will be executed. When both tests succeed, Clerical may choose either branch.

In Clerical real numbers are constructed with a limit constructor

lim($n : \text{int}, e(n)$)

where $e(n)$ is real-valued.

The result is defined to be the limit of the sequence $e(0), e(1), e(2), \dots$, assuming the sequence converges with a required rate of convergence for all possible values of the terms $e(n)$. For the limit operator to be useful, we must allow $e(n)$ to be arbitrarily complex code. However, in order to make sense of the convergence of the sequence, $e(n)$ should be free of side-effects. We are led to the idea of a *command-like expression*, a value-returning command that may modify only its own local state.

The value of a Clerical program is a non-empty set of possible values, including non-termination \perp . Each datatype is interpreted naively: the booleans take values from the set $\{\text{false}, \text{true}\}$, the integers from \mathbb{Z} , and the reals from \mathbb{R} . In particular, we need no domain theory or space representations, a familiar set-theoretic model does the job. In a related talk we present Hoare-style proof rules for deriving correctness of Clerical programs. The rules are sound with respect to the set-theoretic semantics.

We will showcase an implementation of Clerical that executes programs in the style of Norbert Müller’s iRRAM. It approximates real numbers with intervals, using interval

arithmetic. When the intervals get too inaccurate for the comparison tests to succeed, the entire computation is aborted and restarted anew with better initial precision. Such an execution strategy is correct with respect to the semantics of programs in the sense that it always computes one of the possible values of the program.

3.4 Bounds for eigenvalues of an eigenvalue problem with non-smooth coefficients


Henning Behnke (TU Clausthal, DE)

License  Creative Commons BY 3.0 Unported license
© Henning Behnke

In the design of integrated optical chips an eigenvalue problem with piecewise continuous functions arises. Typical features of the technological problem are the possible occurrence of eigenvalue clusters and the necessity of quite stringent relative error tolerances for the eigenvalues. For the computation of bounds we use finite elements based on the Rayleigh-Ritz and Temple-Lehmann-Goerisch methods for upper and lower eigenvalue bounds, respectively. Rounding errors are controlled with interval arithmetic.

3.5 Computing positive invariant sets with intervals

Benoît Zerr, Luc Jaulin, and Thomas Le Mézo

License  Creative Commons BY 3.0 Unported license
© Benoît Zerr, Luc Jaulin, and Thomas Le Mézo
Main reference Thomas Le Mézo, Luc Jaulin, Benoît Zerr: “An Interval Approach to Compute Invariant Sets”, IEEE Trans. Automat. Contr., Vol. 62(8), pp. 4236–4242, 2017.
URL <http://dx.doi.org/10.1109/TAC.2017.2685241>

Given a set X , and a dynamical deterministic system S , the largest invariant set inside X is the set of all x_0 such that all trajectories going through x_0 stay inside S for all time instant t . In this presentation, I will show how interval methods can be used to compute an inner and an outer characterization of an invariant set. The procedure that will be presented is fast, guaranteed and does not require any interval integration of the dynamics. The main idea is to use the new notion of maze, which is a composition of graphs, pavings of boxes and polygons. The set of mazes forms a lattice with respect to the inclusion so that contractor-based methods could be used.

3.6 On the Taylor model approach for solving ODEs

Florian Bünger (TU Hamburg-Harburg, DE)

License  Creative Commons BY 3.0 Unported license
© Florian Bünger

Taylor models have been used successfully to calculate verified inclusions of the solutions of initial value problems for ordinary differential equations (ODEs).

Especially Berz, Makino, and their group focused on that and invented several accompanying methods like “shrink wrapping”, “blunting”, and “preconditioning”.

We give a short description of Taylor models, their arithmetic, the algorithm for solving ODEs and the aforementioned accompanying methods.

3.7 Uncertainty Quantification: Probabilistic Forecasts of Energy Demand

George F. Corliss (Marquette University – Milwaukee, US)

License © Creative Commons BY 3.0 Unported license
 © George F. Corliss
Joint work of Mohammad Saber and Richard Povinelli
Main reference Mohammad Saber: “Quantifying Forecast Uncertainty in the Energy Domain”, Ph.D. Dissertation, Department of Electrical and Computer Engineering, Marquette University, Milwaukee, Wisc., 2017.
URL http://epublications.marquette.edu/dissertations_mu/746/

We forecast daily natural gas demand for utilities at time horizons up to a week. Intervals could capture 100% confidence intervals, but our customers prefer a cumulative density distribution (CDF) to capture a richer expression of uncertainty. The CDF $f(x)$ expresses the probability that the actual demand d will be less than or equal to amount x . Our preferred probabilistic forecasting engine uses historical point-valued forecasts from a linear regression or artificial neural network model and bins empirical residuals, e.g., by temperature. In each bin, a Johnson transformation maps the empirical residuals to a (nearly) normal distribution. To generate a probabilistic forecast, we generate a point forecast, which we use to mean-shift the normal distribution appropriate to the forecast temperature. We assess the quality of our forecasts using Saber’s Graphical Calibration Measure: If we make probabilistic forecasts for a year, we have 365 forecasts $P(d \leq x_p) = p$, for example, $p = 0.95$. If our forecasting engine is good, we should be right 95% of the time and wrong 5% of the time.

3.8 Nonlinear Symbolic Transformations for Simplifying Functions – Applied for Interval Based Global Optimization

Tibor Csendes (University of Szeged, HU) and Elvira Dobjänné Antal

License © Creative Commons BY 3.0 Unported license
 © Tibor Csendes and Elvira Dobjänné Antal
Main reference Elvira Antal, Tibor Csendes: “Nonlinear Symbolic Transformations for Simplifying Optimization Problems”, Acta Cybern., Vol. 22(4), pp. 715–733, 2016.
URL <http://dx.doi.org/10.14232/actacyb.22.4.2016.1>

For interval arithmetic based reliable computation the expression of the function to be optimized can be critical. We have an automatic algorithm based on symbolic calculation to simplify nonlinear functions. The talk will give detailed results on the effect of this presolving technique on the efficiency of an interval arithmetic based branch-and-bound algorithm. Although we still have a long way to go, the first numbers are encouraging [1].

References

- 1 Tibor Csendes and Elvira Dobjänné Antal: Nemlineáris szimbolikus transzformációk optimalizálási feladatokra (In Hungarian: Nonlinear symbolic transformations for optimization problems). Szigma 48(2017) 33-46

3.9 Daisy – a framework for sound accuracy analysis and optimization of finite-precision programs

Eva Darulova (MPI-SWS – Saarbrücken, DE)

License © Creative Commons BY 3.0 Unported license

© Eva Darulova

Main reference Daisy – Framework for Analysis and Optimization of Numerical Programs (Tool Paper), TACAS 2018 (to appear)

Floating-point or fixed-point computations are an integral part of many embedded and scientific computing applications, as are the roundoff errors they introduce. They expose an interesting tradeoff between efficiency and accuracy: the more precision we choose, the closer the results will be to the ideal real arithmetic, but the more costly the computation becomes. Unfortunately, the unintuitive and complex nature of finite-precision arithmetic makes manual optimization infeasible such that automated tool support is indispensable. This talk presents an overview of Daisy, a framework for sound accuracy analysis and optimization of finite-precision programs. We will provide a high-level view of its main features: roundoff error analysis as well as rewriting and mixed-precision optimization.

3.10 Computability of geometric Lorenz attractors

Daniel Graça (University of Algarve, PT)

License © Creative Commons BY 3.0 Unported license

© Daniel Graça

Joint work of Graça, Daniel; Rojas, Cristobal; Zhong, Ning

Main reference D. S. Graça, C. Rojas, and N. Zhong: “Computing geometric Lorenz attractors with arbitrary precision”, *Trans. Amer. Math. Soc.* 370, pp. 2955–2970, 2018.

URL <https://doi.org/10.1090/tran/7228>

In many applications, ranging from biology to physics, one is interested in knowing about the long term behavior of a given system. However, many of those systems have complex dynamics, making their analysis quite difficult through the exclusive use of analytical methods. With the introduction of high speed and affordable computers, a common approach is to use numerical simulations to obtain new information about a system under study. This numerical approach led to new important insights. For example, evidence that “strange attractors” like the Lorenz attractor can occur came from numerical experiments. However, it is also commonplace to use floating point arithmetic in those simulations, where real numbers are substituted by approximations having fixed finite precision. This is a source of rounding and truncation errors which, especially in the case of chaotic systems like the Lorenz attractor, can be greatly amplified along time. Therefore the accuracy of the results obtained in this manner can be put into question. In response to one of the 18 unsolved problems that the Fields medalist S. Smale suggested for the 21st century, W. Tucker proved that the Lorenz attractor exists, using a combination of normal form theory and rigorous numerics. To reach that conclusion, it is shown that the Lorenz system behaves like a geometric Lorenz model. In this talk we will digress over these results and also show that geometric Lorenz attractors are computable.

3.11 Interval computations with compensated algorithms

Stef Graillat (UPMC – Paris, FR)

License © Creative Commons BY 3.0 Unported license
© Stef Graillat

In this talk, we will present some results on interval computations with compensated algorithms. Compensated algorithms consist in computing the rounding errors of individual operations and then adding them later on to the computed result. This makes it possible to increase the accuracy of the computed result efficiently. Computing the rounding error of an individual operation is possible through the use of a so-called error-free transformation (EFT). The EFT needs a rounding to the nearest to be exact. In this talk, we will show that EFTs are still robust with directed rounding and that they can be used to perform interval computations. This is a joint work with Fabienne Jézéquel.

3.12 On the Computational Complexity of the Range Computation Problem

Peter Hertling (Universität der Bundeswehr – München, DE)

License © Creative Commons BY 3.0 Unported license
© Peter Hertling

The following problem is one of the basic problems of interval computations: given a function $f(x_1, \dots, x_n)$ of n real variables, given n intervals, and given some desired output precision, compute the range of f over the box of these intervals at least with the desired precision. Gaganov (1981, 1985) considered the case where the input function f is a polynomial given by its coefficients and showed that this problem is at least as hard as any NP-problem. Kreinovich, Lakeyev, Rohn, and Kahl (1998) analysed the computational complexity of many further variants of this problem. First we show that the general problem is not harder than NP-problems. Then we consider some variants where a sequence of polynomials and a sequence of interval boxes are fixed and show that their complexity is closely connected to some other well-known open questions from structural complexity theory.

3.13 Formal Verification of a Rigorous ODE Solver

Fabian Immler (TU München, DE)

License © Creative Commons BY 3.0 Unported license
© Fabian Immler


Main reference Fabian Immler: “A verified ODE solver and the Lorenz attractor”, Journal of Automated Reasoning, Springer, 2018.

URL <https://doi.org/10.1007/s10817-017-9448-y>

This presents a formalization of ordinary differential equations (ODEs) and the verification of rigorous (with guaranteed error bounds) numerical algorithms in the interactive theorem prover Isabelle/HOL. The formalization comprises flow and Poincaré map of dynamical systems. The verified algorithms are based on Runge-Kutta methods and affine arithmetic. They certify numerical bounds for the Lorenz attractor and thereby lift the numerical part of Tucker’s proof of Smale’s 14th problem onto a formal foundation.

3.14 Towards certified exact real computation


Sunyoung Kim (Ewha Womans University, KR)

License  Creative Commons BY 3.0 Unported license
© Sunyoung Kim

When we use computers to do computing with real numbers, it is likely that we cannot guarantee the correctness of the results. In this talk, we give an overview of our recent project about certified exact real arithmetic. The main goal of our project is to develop and extend verified libraries for exact real arithmetic.

3.15 Functional exact real computation, towards verification of total correctness

Michal Konecny (Aston University – Birmingham, GB)

License  Creative Commons BY 3.0 Unported license
© Michal Konecny

I have reported on the status of my attempts to develop formally verified numerical programs, taking advantage of the (relative) simplicity of the semantics of exact real computation and functional programming. In my implementation, I use a cut-down version of a functional version of the CIDR language which is currently being developed jointly with Brausse, Collins, Mueller, Neumann, Park and Ziegler in the CID project which is also similar to the Clerical language being developed by Bauer, Park and Simpson and the ERA language used in (Lee et al 2017). The programming language is deeply embedded in Isabelle and is given formal operational and non-deterministic denotations semantics. As the semantics works with a given resource limit, it is possible to formally specify program termination. The approach has been validated by formally verifying in Isabelle the functional correctness and termination of a program that computes the square root by Newton iteration.

References

- 1 Gyesik Lee, Norbert Müller, Eike Neumann, Sewon Park, Norbert Preining, Martin Ziegler *On Formal Verification in Imperative Multivalued Programming over Continuous Data Types*. arXiv:1608.05787 [cs.NA], 2017

3.16 Computable Numbers, Computable Sets, and Computable Functions and How It Is All Related to Interval Computations

Vladik Kreinovich (University of Texas – El Paso, US)

License  Creative Commons BY 3.0 Unported license
© Vladik Kreinovich

From the physical viewpoint, real numbers x describe values of different quantities. We get values of real numbers by measurements. Measurements are never 100% accurate, so after a measurement, we get an approximate value r_k of x . In principle, we can measure x with higher and higher accuracy. So, from the computational viewpoint, a real number is a sequence of rational numbers r_k for which, e.g., $|x - r_k| \leq 2^{-k}$. By an algorithm processing

real numbers, we mean an algorithm using r_k as an “oracle” (subroutine). This is how computations with real numbers are defined in *computable analysis*.

Once we know the measurement result \tilde{x} and the upper bound Δ on the measurement error $\Delta x \stackrel{\text{def}}{=} \tilde{x} - x$, we can conclude that the actual value x belongs to the interval $[\tilde{x} - \Delta, \tilde{x} + \Delta]$. In interval analysis, this is all we know: we performed measurements (or estimates), we get intervals, and we want to extract as much information as possible from these results. In particular, we want to know what can we conclude about $y = f(x_1, \dots, x_n)$, where f is a known algorithm.

In computable (constructive) analysis, we take into account that eventually, we will be able to measure each x_i with higher and higher accuracy. In other words, for each quantity, instead of a *single* interval, we have a *sequence* of narrower and narrower intervals, a sequence that eventually converging to the actual value. From this viewpoint, *Interval analysis is applied constructive analysis* (Yuri Matiyasevich, of 10th Hilbert problem fame).

In this talk, we describe, from this viewpoint, what is a computable set, what is a computable function, and give examples of interval-related positive and negative results of computable analysis.

3.17 Need to Combine Interval and Probabilistic Uncertainty: What Needs to Be Computed, What Can Be Computed, What Can Be Feasibly Computed, and How Physics Can Help

Vladik Kreinovich (University of Texas – El Paso, US)

License © Creative Commons BY 3.0 Unported license
© Vladik Kreinovich

Need to Combine Interval and Probabilistic Uncertainty: What Needs to Be Computed, What Can Be Computed, What Can Be Feasibly Computed, and How Physics Can Help

In many practical situations, the quantity of interest y is difficult to measure directly. In such situations, to estimate y , we measure easier-to-measure quantities x_1, \dots, x_n which are related to y by a known relation $y = f(x_1, \dots, x_n)$, and we use the results X_1, \dots, X_n of these measurement to estimate y as $Y = f(X_1, \dots, X_n)$. How accurate is this estimate?

Traditional engineering approach assumes that we know the probability distributions of measurement errors $X_i - x_i$, however, in practice, we often only have partial information about these distributions. In some cases, we only know the upper bounds D_i ; in such cases, the only thing we know about the actual value x_i is that it is somewhere in the interval $[X_i - D_i, X_i + D_i]$. Interval computation estimates the range of possible values of y under such interval uncertainty.

In other situations, in addition to the intervals, we also have partial information about the probabilities. In this talk, we describe how to solve this problem in the linearized case, what is computable and what is feasibly computable in the general case, and, somewhat surprisingly, how physics ideas – that initial conditions are not abnormal, that every theory is only approximate – can help with the corresponding computations.


References

- 1 Finkelstein, A.M., Kreinovich, V.: Impossibility of hardly possible events: physical consequences. Abstracts of the 8th International Congress on Logic, Methodology, and Philosophy of Science, Moscow, 1987, 5(2), 23–25 (1987)

- 2 Kosheleva, O., Zakharevich, M., Kreinovich, V.: If many physicists are right and no physical theory is perfect, then by using physical observations, we can feasibly solve almost all instances of each NP-complete problem, *Mathematical Structures and Modeling* 31, 4–17 (2014)
- 3 Kreinovich, V.: Toward formalizing non-monotonic reasoning in physics: the use of Kolmogorov complexity. *Revista Iberoamericana de Inteligencia Artificial* 41, 4–20 (2009)
- 4 Kreinovich, V.: Negative results of computable analysis disappear if we restrict ourselves to random (or, more generally, typical) inputs, *Mathematical Structures and Modeling* 25, 100–103 (2012)
- 5 Kreinovich, V., Finkelstein, A. M.: Towards applying computational complexity to foundations of physics. *Notes of Mathematical Seminars of St. Petersburg Department of Steklov Institute of Mathematics* 316, 63–110 (2004); reprinted in *Journal of Mathematical Sciences* 134(5), 2358–2382 (2006)
- 6 Kreinovich, V., Kunin, I. A.: Kolmogorov complexity and chaotic phenomena. *International Journal of Engineering Science* 41(3), 483–493 (2003)
- 7 Kreinovich, V., Kunin, I. A.: Kolmogorov complexity: how a paradigm motivated by foundations of physics can be applied in robust control. In: Fradkov, A. L., Churilov, A. N., eds. *Proceedings of the International Conference “Physics and Control” PhysCon’2003*, Saint-Petersburg, Russia, August 20–22, 2003, 88–93 (2003)
- 8 Kreinovich, V., Kunin, I. A.: Application of Kolmogorov complexity to advanced problems in mechanics. *Proceedings of the Advanced Problems in Mechanics Conference APM’04*, St. Petersburg, Russia, June 24–July 1, 2004, 241–245 (2004)
- 9 Kreinovich, V., Longpré, L., Koshelev, M.: Kolmogorov complexity, statistical regularization of inverse problems, and Birkhoff’s formalization of beauty. In: Mohamad-Djafari, A., ed., *Bayesian Inference for Inverse Problems*, *Proceedings of the SPIE/International Society for Optical Engineering*, San Diego, California, 1998, 3459, 159–170 (1998)
- 10 Kreinovich, V., Pownuk, A., Kosheleva, O.: Combining interval and probabilistic uncertainty: what is computable?, In: Pardalos, P., Zhigljavsky, A., Zilinskas, J. (eds.): *Advances in Stochastic and Deterministic Global Optimization*, Springer Verlag, Cham, Switzerland, 2016, p. 13–32. <http://www.cs.utep.edu/vladik/2015/tr15-66a.pdf>
- 11 Li, M., Vitanyi, P.: *An Introduction to Kolmogorov Complexity and Its Applications*, Springer (2008)
- 12 Pour-El, M. B., Richards, J. I.: *Computability in Analysis and Physics*, Springer, Berlin (1989)
- 13 Sun, L., Dbouk, H., Neumann, I., Schoen, S., Kreinovich, V.: Taking into account interval (and fuzzy) uncertainty can lead to more adequate statistical estimates, *Proceedings of the 2017 Annual Conference of the North American Fuzzy Information Processing Society NAFIPS’2017*, Cancun, Mexico, October 16–18, 2017. <http://www.cs.utep.edu/vladik/2017/tr17-57a.pdf>
- 14 Weihrauch, K.: *Computable Analysis*, Springer-Verlag, Berlin (2000)

3.18 An Approach to Programming Configurable Computers for Numeric Applications

Fritz Mayer-Lindenberg (TU Hamburg-Harburg, DE)

License  Creative Commons BY 3.0 Unported license
© Fritz Mayer-Lindenberg

While conventional programming languages deal with a few types of numbers only, typically the ones implemented on their standard target processors, configurable computers, in particular FPGA based ones, can implement a large variety of number codes to make resource-aware application specific selections of the types to be used, and add an additional layer of programming this way. This motivated the design of a new programming language, besides the general need for simple programming tools for FPGA based systems. First, the required circuit design for the processors to be configured and for an infrastructure linking them to each other and to the memory resources is taken out of the task of programming numeric algorithms by building on a library of precompiled configurations. For the algorithmic programming, dealing with the various number codes as separate predefined types would result in a reduced level of abstraction. Instead, a single, abstract type of number is used as proposed in [1], at the same time rising abstraction to the level of mathematical algorithms on real numbers and clearly distinguishing numbers from their codes. The various number codes are represented by individual operations only, namely the corresponding roundings. By predefining tuples of numbers and operations on them, non-standard tuple codes such as block floating point codes can be supported as well to further reduce circuit complexity and memory requirements [2]. The language then proceeds to structures for the required control of parallel processing on heterogeneous sets of rather simple compute nodes for the different number codes [3]. In particular, it includes statements to identify the processor networks configured in an FPGA with sufficient detail to support native code generation for each individual processor, to select library components, and to simulate the operation of an entire parallel target.

References

- 1 F. Mayer-Lindenberg, A Management Scheme for the Basic Types in Programming Languages, SOFSEM 2005
- 2 F. Mayer-Lindenberg, A Modular Processor Architecture for HPC on FPGA, CDES'12
- 3 F. Mayer-Lindenberg, A programming language for embedded processor networks, ICSICT 2016

3.19 On implementing TTE

Norbert T. Müller (Universität Trier, DE)

License  Creative Commons BY 3.0 Unported license
© Norbert T. Müller

Type-2-Theory of Effectivity (TTE) is the accepted model for computability and computational complexity on real numbers. One of its many advantages is that there do exist implementations being consistent with this theory.

The talk presents main common aspects of these implementations with several examples (like dynamical systems and ODE solving) and an outlook on future directions.

3.20 On the numerical norm estimation of the inverse operator in Hilbert space

Mitsuhiro T. Nakao (*Waseda University – Tokyo, JP*)

License © Creative Commons BY 3.0 Unported license

© Mitsuhiro T. Nakao

Joint work of Mitsuhiro T. Nakao, Yoshitaka Watanabe, Takehiko Kinoshita

We present some relations between the exact norm for the inverse operator in Hilbert space and its numerical estimation by some approximate methods. In order to verify numerically an exact solution of the nonlinear operator equations, particularly for partial differential equations, around some neighborhood of the approximate solution, it is important to calculate some approximate norm for an inverse of linearized operators. In this talk we clarify the conditions so that such approximate estimates by the finite element methods converge to the exact norm. This result enables us the appropriateness of the estimation by concerned numerical methods.

References

- 1 Y. Watanabe, T. Kinoshita, M. T. Nakao, A posteriori estimates of inverse operators for boundary value problems in linear elliptic partial differential equations, *Mathematics of Computation*, 82 (2013), 1543-1557. DOI:10.1090/S0025-5718-2013-02676-2
- 2 T. Kinoshita, Y. Watanabe, M. T. Nakao, Some remarks on the rigorous estimation of inverse linear elliptic operators, SCAN 2014, LNCS 9553, the proceedings of the conference '16th GAMM-IMACS International Symposium on Scientific Computing, Computer Arithmetic and Validated Numerics' (2016), 225-235. DOI: 10.1007/978-3-319-31769-4-18

3.21 Parametrised complexity for the naive Cauchy representation

Eike Neumann (*Aston University – Birmingham, GB*) and Florian Steinberg (*TU Darmstadt, DE*)

License © Creative Commons BY 3.0 Unported license

© Eike Neumann and Florian Steinberg

Joint work of Eike Neumann, Florian Steinberg

Main reference Eike Neumann, Florian Steinberg: "Parametrised second-order complexity theory with applications to the study of interval computation", CoRR, Vol. abs/1711.10530, 2017.

URL <http://arxiv.org/abs/1711.10530>

The aim of this work is to provide a rigorous complexity framework for “non-rigorous” numerical algorithms, which come without explicit error bounds or convergence rates.

The standard representation of real numbers used in computable analysis, where a real number is encoded by a fast converging Cauchy sequence of dyadic rational numbers, leads to a very robust and realistic notion of computability and complexity on the reals which is closely related to rigorous numerical analysis.

In order to model non-rigorous computation one could attempt to simply drop the requirement of fast convergence. The resulting representation - the so-called naive Cauchy representation - is however well-known to be very ill-behaved computationally.

We show that the space of naive Cauchy reals can be enriched with a natural parameter, which essentially encodes a rate of convergence, to obtain a reasonable computability and complexity structure.

Although the resulting parametrised space of naive Cauchy reals is not even topologically equivalent to the space of real numbers with the standard representation, it has the same

(polytime) computable points. Moreover, a real function is (polytime) computable with respect to the parametrised naive Cauchy representation if and only if it is (polytime) computable in the usual sense.

We further show that the space of continuous real functions on the compact unit interval admits a minimal parametrised representation such that the evaluation functional

$$\text{eval}: C([0, 1]) \times [0, 1] \rightarrow \mathbb{R}$$

becomes polytime computable when $[0, 1]$ and \mathbb{R} are given the parametrised naive Cauchy representation.

3.22 Test Matrices for Numerical Linear Algebra

Katsuhisa Ozaki (Shibaura Institute of Technology – Saitama, JP)

License © Creative Commons BY 3.0 Unported license
© Katsuhisa Ozaki

Main reference Katsuhisa Ozaki, Takeshi Ogita: “Generation of Linear Systems with Specified Solutions for Numerical Experiments”, *Reliable Computing*, Vol. 25, pp. 148–167, 2017.

This talk concerns test matrices for numerical linear algebra, especially, linear systems and eigenvalue problems. If exact solution is known in advance, it is very useful for checking the accuracy and stability of numerical algorithms. Residual is often used for the check of the accuracy of numerical results. However, numerical solutions with small residual and big error can be obtained. We propose methods that produce problems with the exact solution based on error-free transformation of floating-point arithmetic. For linear systems, our methods generate a matrix, a solution and a right-hand side vector whose all elements are representable in floating-point numbers. We showed the exact stepwise errors for the BiCGSTAB method for several sparse matrices. For eigenvalue problems, our methods produce a matrix and its specified eigenvalues using the Hadamard matrix.

3.23 A tutorial on reliable numerical computation

Paul Zimmermann

License © Creative Commons BY 3.0 Unported license
© Paul Zimmermann

This tutorial presents different software tools to perform reliable numerical computations: machine double precision using the IEEE 754 standard, fixed-precision interval arithmetic, arbitrary precision floating-point or interval arithmetic using MPFR, MPFI, or the Arb library. All these kinds of arithmetic can be used within the SageMath computer algebra system.

3.24 Computational complexity of solving polynomial differential equations over unbounded domains

Amaury Pouly (MPI-SWS – Saarbrücken, DE) and Daniel Graça (University of Algarve, PT)

License © Creative Commons BY 3.0 Unported license

© Amaury Pouly and Daniel Graça

Main reference Amaury Pouly, Daniel S. Graça: “Computational complexity of solving polynomial differential equations over unbounded domains”, *Theor. Comput. Sci.*, Vol. 626, pp. 67–82, 2016.

URL <http://dx.doi.org/10.1016/j.tcs.2016.02.002>

In this abstract we present a rigorous numerical algorithm which solves initial-value problems defined with polynomial differential equations (i.e. initial-value problems of the type $y' = p(t, y)$, $y(t_0) = y_0$, where p is a vector of polynomials) for any value of t . The inputs of the algorithm are the data defining the initial-value problem, the time T at which we want to compute the solution of the IVP, and the maximum allowable error $\epsilon > 0$. Using these inputs, the algorithm will output a value \tilde{y}_T such that $|\tilde{y}_T - y(T)| < \epsilon$.

3.25 Numerical (arte-)facts and reliable computing

Siegfried M. Rump (TU Hamburg-Harburg, DE)

License © Creative Commons BY 3.0 Unported license

© Siegfried M. Rump

URL <http://www.ti3.tuhh.de>

We show examples of linear systems where the computed approximation has no correct digit, but nevertheless the residual (computed in floating-point) is exactly equal to zero. Similarly, the true inverse A^{-1} rounded to the nearest floating-point matrix $R := fl(A^{-1})$ may produce a residual $I - RA$ of norm larger than 1, but an approximate inverse $\tilde{R} = inv(A)$ computed by Matlab satisfies $\|I - \tilde{R}A\| < 1$. Despite, the entries of \tilde{R} are wrong by more than a factor 2.

A remedy to incorrect approximations are verification methods. If the precision is not sufficient to compute correct error bounds for a solution, a corresponding message is given. Wrong results are not possible.

The power of verification methods is demonstrated by problems from quantum chemistry. Here floating-point algorithms fail completely, where verification methods provide tight inclusions of the solution. This is for problem sizes up to 30 million unknowns with tens of thousands of constraints.

3.26 Co-Polish spaces in Complexity Theory

Matthias Schröder (Universität der Bundeswehr – München, DE)

License © Creative Commons BY 3.0 Unported license


© Matthias Schröder

Co-Polish spaces play an important role in Type Two Complexity Theory. A Co-Polish space is defined to be a sequential topological space that is regular and for which the compact-open topology on the function space $C(X, IR)$ has a countable base. Co-Polish spaces turn out to be exactly those Hausdorff qcb-spaces X that admit a Simple Complexity Theory. Simple

Complexity Theory means that time complexity for functions on X can be measured in the desired output precision plus a *discrete* parameter on the input. For general spaces X , for example for non-locally-compact metric spaces X , an indiscrete parameter on the input is necessary.

3.27 Bit complexity of Computing Solutions for Symmetric Hyperbolic Systems of PDEs with Guaranteed Precision

Svetlana Selivanova

License  Creative Commons BY 3.0 Unported license

© Svetlana Selivanova

Joint work of Svetlana Selivanova, Victor Selivanov

The algorithms used in mathematics-oriented software can be divided into two big classes: symbolic algorithms which aim to find precise solutions, and approximate algorithms which aim to find “good enough” approximations to precise solutions. The symbolic algorithms are implemented e.g. in computer algebra systems while the approximate algorithms — in numerical mathematics packages. The both classes of algorithms are widely used in applications and in mathematical research. The symbolic algorithms correspond well to computations on discrete structures (with mathematical foundations in the classical computability and complexity theory) while the approximate algorithms — to computations on continuous structures (with mathematical foundations in the field of computability and complexity in analysis evolving under the slogan “Exact real computation”).

An important idea relating the both classes of algorithms is to look for approximate solutions to a numerical problem with “guaranteed precision”. The bit complexity of an algorithm is fundamental because it estimates the amount of computational resources needed to implement the algorithm on a computing device. Here we investigate the bit complexity of finding guaranteed precision solutions for Cauchy and boundary-value problems for symmetric hyperbolic systems of PDEs (see e.g. [1]) Such systems can be used to describe a wide variety of physical processes like those considered in the theories of elasticity, acoustics, electromagnetism etc. Accordingly, many people from theoretical and numerical mathematics worked on the existence and uniqueness theorems as well as on numerical methods of computing solution operators for problems related to such systems (the explicit solution formulas exist only in some simplest particular cases).

In [2] we developed an approach to the study of computability of the Cauchy and dissipative boundary-value problems for such systems based on finite-dimensional approximations (the so called difference schemes widely used in numerical analysis) and established the computability of solution operators in the rigorous sense of the TTE approach to computable analysis [3]. The main obstacle in proving the computable dependence of solutions on the input matrices is the fact that all known stable difference schemes for finding the approximate solutions use eigenvectors of some matrices and matrix pencils but these eigenvectors are known to be non-computable [4]. To overcome the obstacle, we considered in [2] restrictions of the solution operators to computably presentable real closed number fields and have shown that such restricted solution operators are computable. This fact together with close relationships of such fields to the field of computable reals (also established in [2]) imply that the solution operators are computable for any fixed computable input matrices.

We develop the approach from [2] to establish some reasonable upper bounds for some guaranteed-precision problems related to symmetric hyperbolic systems. A version of such a

problem asks, given a fixed number of space variables, given algebraic real input matrices, rational polynomials as initial-value functions, and a precision p , to find an algebraic grid function such that the difference between the poly-linear interpolation of grid function and the precise solution is at most p . We establish the EXPTIME upper complexity bound for such problems and show that the estimate becomes polynomial under some additional restrictions often used in the practice of numerical methods. To our knowledge, these are the first such bounds in the literature.

Our approach makes a heavy use of some known and our own algorithms of computer algebra (exact computations with integers, rationals, algebraic reals and polynomials, polynomial-time computability of spectral decomposition of symmetric matrices and matrix pencils in the field of algebraic reals), together with some algorithms from numerical mathematics and computable analysis used in [2]. Altogether, our proofs demonstrate a fruitful mix of methods from symbolic and numerical computation. Although our methods do not seem to yield practically feasible algorithms for guaranteed precision, we hope that investigations in this direction are fruitful for both theoretical research and applications. In particular, on the implementation level it seems useful and rewarding to enhance the existing systems of “exact real computations” (like iRRAM) by packages based of highly developed algorithms of computer algebra. We are not aware of the existence of such “hybrid” systems built under the slogan of “guaranteed precision numerical computations”.

References

- 1 Godunov S. K., ed.: Numerical Solution of Higher-dimensional Problems of Gas Dynamics (in Russian). Nauka, Moscow, 1976.
- 2 Selivanova S., Selivanov V.: Computing Solution Operators of Boundary-value Problems for Some Linear Hyperbolic Systems of PDEs. Logical Methods in Computer Science, 13 (4:13) 2017, pp. 1–31. Earlier version on arXiv:1305.2494 (2013).
- 3 Weihrauch K.: Computable Analysis. Berlin, Springer, 2000.
- 4 Ziegler M., Brattka V.: A computable spectral theorem. Proc. CCA-2001, Lecture Notes in Computer Science, v. 2064 (2001), 378–388.

3.28 Parametrised second-order complexity theory with applications to the study of interval computation

Florian Steinberg (TU Darmstadt, DE) and Eike Neumann (Aston University – Birmingham, GB)

License © Creative Commons BY 3.0 Unported license
© Florian Steinberg and Eike Neumann

Main reference Eike Neumann, Florian Steinberg: “Parametrised second-order complexity theory with applications to the study of interval computation”, CoRR, Vol. abs/1711.10530, 2017.

URL <http://arxiv.org/abs/1711.10530>

We extend the framework for complexity of operators in analysis devised by Kawamura and Cook (2012) to allow for the treatment of a wider class of representations. The main novelty is to endow represented spaces of interest with an additional function on names, called a parameter, which measures the complexity of a given name. This parameter generalises the size function which is usually used in second-order complexity theory and therefore also central to the framework of Kawamura and Cook. The complexity of an algorithm is measured in terms of its running time as a second-order function in the parameter, as well as in terms of how much it increases the complexity of a given name, as measured

by the parameters on the input and output side. As an application we develop a rigorous computational complexity theory for interval computation. In the framework of Kawamura and Cook the representation of real numbers based on nested interval enclosures does not yield a reasonable complexity theory. In our new framework this representation is polytime equivalent to the usual Cauchy representation based on dyadic rational approximation. By contrast, the representation of continuous real functions based on interval enclosures is strictly smaller in the polytime reducibility lattice than the usual representation, which encodes a modulus of continuity. Furthermore, the function space representation based on interval enclosures is optimal in the sense that it contains the minimal amount of information amongst those representations which render evaluation polytime computable.

This talk was based on the arXiv paper <https://arxiv.org/abs/1711.10530> and a followup talk was given by Eike Neumann.

3.29 Verified computations for solutions to 1-dimensional advection equations with variable coefficients

Akitoshi Takayasu (University of Tsukuba, JP)

License © Creative Commons BY 3.0 Unported license
© Akitoshi Takayasu

In this talk, we provide a methodology of verified numerical computations for solutions to 1-dimensional advection equations with variable coefficients. The advection equation is typical partial differential equations (PDE) of hyperbolic type. There are few results for verified computations to initial-boundary value problem of hyperbolic PDEs. Our methodology is based on the spectral method and semigroup theory. Numerical examples show that the rigorous error estimate showing the well-posedness of the exact solution is given with high accuracy and high speed.

3.30 Average case complexity for Hamiltonian dynamical systems

Holger Thies (University of Tokyo, JP), Akitoshi Kawamura (Kyushu University, JP), and Martin Ziegler (KAIST – Daejeon, KR)

License © Creative Commons BY 3.0 Unported license
© Holger Thies, Akitoshi Kawamura, and Martin Ziegler

Average case complexity in analysis was recently introduced by Schröder, Steinberg and Ziegler. We study the complexity of Hamiltonian systems like the famous n-body problem in this context. We use a simple parameterized worst-case complexity result for initial value problems with analytic right-hand side and the fact that Hamiltonian systems are volume preserving to relate the complexity of a system to the volume of singularities in phase space. As an application we show that the planar circular restricted three-body problem is computable in polynomial-time on average.

3.31 Small divisors and normal forms

Warwick Tucker (*Uppsala University, SE*)

License © Creative Commons BY 3.0 Unported license
© Warwick Tucker

Joint work of Warwick Tucker, Zbigniew Galias

In this talk, we will discuss the computational challenges of computing trajectories of a non-linear ODE in a region close to a fixed-point. By introducing a carefully selected close to identity change of variables, we can bring the non-linear ODE into an “almost” linear system. Determining the domain of existence for such a change of variables poses some interesting computational challenges.

3.32 Soft Foundations for Geometric Computation

Chee K. Yap (*New York University, US*)

License © Creative Commons BY 3.0 Unported license
© Chee K. Yap

Joint work of 1. Roots (V. Sharma, A. Eigenwillig, M. Sagraloff, R. Becker, J. Xu) 2. Surfaces (M. Burr, S. Choi, L. Lin, V. Sharma, G. Vegter) 3. Motion Planning (Y.-J. Chiang, C. Wang, J.-M. Lien, Z. Luo, C.-H. Hsu) 4. Voronoi Diagrams (E. Papadopoulou, H. Bennett, V. Sharma, J.-M)

URL <http://cs.nyu.edu/exact/>

For over two decades, Exact Geometric Computation (EGC) has provided a paradigm in Computational Geometry for the correct implementation of geometric algorithms. It is the most successful approach to numerical nonrobustness issues, leading to software libraries and practical algorithms. We review some reasons to extend this paradigm:

- EGC algorithms may not be Turing computable (e.g., transcendental functions)
- EGC may be too inefficient (e.g., shortest path problems)
- EGC entails numerous/difficult algebraic analysis (e.g., Vor diagram of polyhedra)
- Exact computation is inappropriate for the physical world (e.g., robot motion planning)

This talk describes a program to develop “soft” approaches for addressing these issues. “Soft” refers to numerical, certified approaches that nevertheless provide some modified notions of “hard” (topological/combinatorial) guarantees in the output. We illustrate these ideas by work in four areas:

- root isolation and clustering (ISSAC’09,’11,’12,’16, SNC’11, CiE’13, JSC’17)
- isotopic approximation of curves and surfaces (ISSAC’08, SoCG’09, SPM’12, ICMS’14)
- Voronoi diagrams (ISVD’13, SGP’16)
- robot motion planning (SoCG’13, WAFR’14, FAW’15, WAFR’16)

Common themes in this list include: we replace the Real RAM model by one based on numerical iteration on interval approximations. Algorithms are framed in the algorithmic paradigm of subdivision. We introduce an input resolution parameter (epsilon) but use it in novel “soft” ways. We design soft versions of classical hard geometric predicates in order to construct effective and practical algorithms. Some consequences of such a computational paradigm are:

- scope of computational geometry is vastly broadened to non-linear non-algebraic problems.
- unsolvable/hard problems in the Real RAM model becomes feasible
- soft algorithms are implementable and practical

One challenge is to revisit other classical problems of computational geometry with this view point. Another is to produce complexity analysis of such algorithms. Successes in complexity analysis for roots suggest that similar sharp “amortized” results can be obtained in higher dimensions.

References

- 1 Chee K. Yap, Vikram Sharma: Robust Geometric Computation. *Encyclopedia of Algorithms* 2016: 1860–1863
- 2 Ruben Becker, Michael Sagraloff, Vikram Sharma, Juan Xu, Chee Yap: Complexity Analysis of Root Clustering for a Complex Polynomial. *ISSAC* 2016: 71–78
- 3 Long Lin, Chee Yap, Jihun Yu: Non-local isotopic approximation of nonsingular surfaces. *Computer-Aided Design* 45(2): 451–462 (2013)
- 4 Huck Bennett, Evanthia Papadopoulou, Chee Yap: Planar Minimization Diagrams via Subdivision with Applications to Anisotropic Voronoi Diagrams. *Comput. Graph. Forum* 35(5): 229–247 (2016)
- 5 Cong Wang, Yi-Jen Chiang, Chee-Keng Yap: On soft predicates in subdivision motion planning. *Comput. Geom.* 48(8): 589–605 (2015)

3.33 On Formal Verification in Imperative Multivalued Programming over Continuous Data Types

Martin Ziegler (KAIST – Daejeon, KR), Gyesik Lee, Norbert T. Müller (Universität Trier, DE), Eike Neumann (Aston University – Birmingham, GB), Sewon Park (KAIST – Daejeon, KR), and Norbert Preining

License © Creative Commons BY 3.0 Unported license

© Martin Ziegler, Gyesik Lee, Norbert T. Müller, Eike Neumann, Sewon Park, and Norbert Preining


Main reference Norbert Th. Müller, Sewon Park, Norbert Preining, Martin Ziegler: “On Formal Verification in Imperative Multivalued Programming over Continuous Data Types”, *CoRR*, Vol. abs/1608.05787, 2016.

URL <http://arxiv.org/abs/1608.05787v2>

Inspired and guided by the iRRAM C++ library (Müller 2001), we formally specify a programming language for the paradigm of Exact Real Computation (ERC): reliably operating on encapsulated continuous data types such as (not necessarily algebraic) real numbers - imperatively and exactly (no rounding errors) with primitives computable in the sense of Recursive Analysis including a necessarily modified multivalued (=non-functional) semantics of tests. Three simple numerical problems demonstrate the elegance and convenience of writing programs handling real (and not just, say rational or algebraic) numbers: integer rounding, solving systems of linear equations, and continuous root finding. We establish Turing-completeness over the reals: a partial function is computable (in the sense of Recursive Analysis) iff it can be expressed in ERC. For rigorously specifying and arguing about such computations in Mathematical Logic, we then propose a decidable first-order theory over two sorts, integers and real numbers. We extend the rules of Hoare Logic to support the formal derivation of correctness proofs in ERC; and we have them, including their real quantification, verified in the Coq Proof Assistant.

3.34 Tutorial on Computational Complexity in Analysis

Martin Ziegler (KAIST – Daejeon, KR) and Akitoshi Kawamura (Kyushu University, JP)

License  Creative Commons BY 3.0 Unported license

© Martin Ziegler and Akitoshi Kawamura

URL <http://theoryofcomputation.asia/survey3.pdf>

Computability Theory in Analysis explores the ultimate capabilities and limitations of computing with real numbers, functions, and subsets. Complexity Theory in Analysis refines these investigations with respect to efficiency. It complements the classical Complexity Theory over discrete structures and adapts its generic goal to continuous data, based on two pillars: (i) Design and rigorous analysis of algorithms for approximating the solution up to guaranteed absolute error $1/2^{-n}$. The computational resources (runtime, memory) thus incurred constitute upper bounds on the computational complexity inherent to the problem; and (ii) establishing – preferably tight – lower bounds, that is, prove any algorithmic solution to require that many resources: This exhibits the algorithm from (i) as optimal, possibly subject to standard hypotheses such as $P \neq NP \neq PSPACE \neq EXP$ or by adapting adversary arguments from Information-Based Complexity to the bit-cost model.

Participants

- Andrej Bauer
University of Ljubljana, SI
- Henning Behnke
TU Clausthal, DE
- Jens Blanck
Swansea University, GB
- Franz Brauße
Universität Trier, DE
- Florian Bünger
TU Hamburg-Harburg, DE
- Pieter Collins
Maastricht University, NL
- George F. Corliss
Marquette University –
Milwaukee, US
- Tibor Csendes
University of Szeged, HU
- Eva Darulova
MPI-SWS – Saarbrücken, DE
- Hubert Glesener
Universität Trier, DE
- Daniel Graça
University of Algarve, PT
- Stef Graillat
UPMC – Paris, FR
- Peter Hertling
Universität der Bundeswehr –
München, DE
- Fabian Immler
TU München, DE
- Luc Jaulin
ENSTA Bretagne – Brest, FR
- Akitoshi Kawamura
Kyushu University, JP
- Sunyoung Kim
Ewha Womans University, KR
- Michal Konecny
Aston University –
Birmingham, GB
- Margarita Korovina
Russian Academy of Sc. –
Novosibirsk, RU
- Vladik Kreinovich
University of Texas – El Paso, US
- Ivo List
University of Ljubljana, SI
- Wolfram Luther
Universität Duisburg-Essen, DE
- Fritz Mayer-Lindenberg
TU Hamburg-Harburg, DE
- Atsushi Minamihata
AIST – Tokyo, JP
- Norbert T. Müller
Universität Trier, DE
- Mitsuhiro T. Nakao
Waseda University – Tokyo, JP
- Eike Neumann
Aston University –
Birmingham, GB
- Katsuhisa Ozaki
Shibaura Institute of Technology
– Saitama, JP
- Sewon Park
KAIST – Daejeon, KR
- Michael Plum
KIT – Karlsruher Institut für
Technologie, DE
- Amaury Pouly
MPI-SWS – Saarbrücken, DE
- Robert Rettinger
Fachhochschule Dortmund, DE
- Fabrice Rouiller
INRIA – Paris, FR
- Siegfried M. Rump
TU Hamburg-Harburg, DE
- Michael Sagraloff
MPI für Informatik –
Saarbrücken, DE
- Matthias Schröder
Universität der Bundeswehr –
München, DE
- Svetlana Selivanova
Sobolev Institute of Mathematics
– Novosibirsk, RU
- Florian Steinberg
TU Darmstadt, DE
- Akitoshi Takayasu
University of Tsukuba, JP
- Holger Thies
University of Tokyo, JP
- Warwick Tucker
Uppsala University, SE
- Joris van der Hoeven
Ecole Polytechnique –
Palaiseau, FR
- Klaus Weihrauch
FernUniversität in Hagen, DE
- Nobito Yamamoto
The University of
Electro-Communications –
Tokyo, JP
- Yuuka Yanagisawa
Waseda University – Tokyo, JP
- Chee K. Yap
New York University, US
- Martin Ziegler
KAIST – Daejeon, KR
- Paul Zimmermann
INRIA Nancy – Grand Est, FR

