Foreword

Erich L. Kaltofen

Department of Mathematics, North Carolina State University
Raleigh, North Carolina 27695-8205, USA
Department of Computer Science, Duke University
Durham, North Carolina 27708-0129

On the initiative of Wen-shin Lee, Austin Lobo and Lihong Zhi and generously hosted by Mark Giesbrecht and Stephen Watt at the University of Waterloo, the Milestones in Computer Algebra (MICA 2016) conference was held July 16–18, 2016 to celebrate my research. This special issue contains papers submitted in connection to MICA 2016. My deep appreciation goes to the organizers, who also include Shaoshi Chen and Arne Storjohann, and to the authors who have submitted papers.

The talks at MICA discussed the past of the discipline of symbolic computation up to 2016, and the papers in this volume form a snapshot of several aspects of current research in symbolic computation: arithmetic such as greatest common divisors, root finding, computational number theory, semantics of expressions in complex variables, applications to chemistry, including my favorite topic of my youth, polynomial factoring, and one of my current topics, certifying computations delegated to a cloud-based computing server. I feel honored to have those papers in a volume associated with me.

My PhD adviser, the late Bobby F. Caviness, in his foreword to our Computer Algebra Handbook (Grabmeier et al., 2003) quotes David Berlinski (2001):

> "Two ideas lie gleaming on the jeweler's velvet. The first is the calculus, the second, the algorithm. The calculus and the rich body of mathematical analysis to which it gave rise made modern science possible; but it has been the algorithm that has made possible the modern world."

It is indeed the algorithm, performed on computers, that made possible the discipline of symbolic computation. Computational complexity distinguishes problems that have a fast algorithmic solution from those that do not. By a twist of fate, the most significant algorithm which our discipline has produced, Bruno Buchberger's (1965) algorithm for computing a Gröbner Basis and all its modern variants, solves NP-complete problems. The mystery remains, today, to know if a given input is hard before running the algorithm to the limit of time and space resources.

Ten years ago, I asked a rising young star researcher what he thought would be the "next big thing:" The symbolic computation discipline would embrace differential equations, was his answer then, which is partly right: today large hybrid difference-differential models are composed by symbolic computation software. The models are solved numerically by iteration. Gradient-descent, a form of Newton's method, can solve equations faster than Buchberger's algorithm, but that may be an illusion. For hard inputs, isolated solutions such as digits of the factors of an integer, cannot be computed fast.

Email address: kaltofen@ncsu.edu, kaltofen@cs.duke.edu (Erich L. Kaltofen)
URL: https://www.cs.duke.edu/~elk27 (Erich L. Kaltofen)

The computerization of mathematics in all areas, from proof-checking and automated proof generation to computer algebra and analysis, stands at the place where it stood when I was young. Our highly-dedicated research community is relatively small, when compared with communities such as in theoretical computer science or numerical optimization or signal processing. The software has high impact, Mathematica and its wolfram|alpha interface, Maple, Magma and Sage and its many underlying opensource components, but again the development community is relatively small, when compared with artificial intelligence and data analysis.

The discipline of symbolic computation lives at the interface of mathematics and computer science. In order to master development and achieve progress, one needs to understand both well. I see some obstructions that have slowed growth:

1. Mathematicians remain focused on theorems and proofs, using symbolic computation for generating examples. Programming new algorithms which add to the infrastructure remains a secondary activity. The Macaulay 2 and Singular teams are an exception.

2. Numerical analysts do not hybridize their algorithms, which would mean to include symbolic computation components. They search for new modeling applications of their trademark numerical algorithms. The sparse Fourier transform algorithms may be an exception.

3. Symbolic computation researchers like to start from scratch, rebuilding the bottom layer of the infrastructure: linear and polynomial arithmetic. The Maplesoft and Wolfram Research Companies are a major exception. So are applications to cryptographic analysis.

I do not know if those attitudes will evolve in the future; I have wished for a long time that they would.

Over my career as a researcher, our community has pioneered trademark algorithms on a diversity of problems: randomization that is based on polynomial identity testing is ubiquitous in our algorithms, from black box linear algebra (Wiedemann, 1986) to factorization of straight-line and black box multivariate polynomials (Kaltofen and Trager, 1990) to sparse multivariate interpolation (Zippel, 1990). Polynomial factorization was put into polynomial-time via the Berlekamp (1967) algorithm and the lattice basis reduction algorithm (Lenstra et al., 1982; van Hoeij, 2002). Prony's sparse interpolation algorithm, the decoder of the 1959 BCH error correcting codes, is randomized for sparsity determination (Kaltofen and Lee, 2003; Hao et al., 2016), and adapted as a sparse polynomial error correction code (Kaltofen and Pernet, 2014). Determinants of n-dimensional integer matrices, which can have $\geq n$ digits, can be computed asymptotically essentially as fast as with fixed precision floating point arithmetic (Storjohann, 2005). One can probabilistically check a proof for matrix-positive-definiteness, which is the core problem in global optimization via sum-of-squares and semi-definite programming, in essentially-linear time in the binary input size (Dumas and Kaltofen, 2014). The tri-variate generating function for Gessel walks on the 2-dimensional grid (number of n moves among $\leftarrow, \swarrow, \nearrow, \rightarrow$ that end at grid point i, j) is an algebraic function (Bostan and Kauers, 2010). In real algebraic geometry, there is a new roadmap algorithm (Safey El Din and Schost, 2017). There are many more, and our symbolic computation community can be rightly proud of the accomplishments. Observing the next generation of researchers, I have no doubt new important algorithms and software will keep coming. Maybe some are already in this issue; history will tell.

References

Berlekamp, E. R., 1967. Factoring polynomials over finite fields. Bell Systems Tech. J. 46, 1853–1859, republished in revised form in: E. R. Berlekamp, *Algebraic Coding Theory*, Chapter 6, McGraw-Hill Publ., New York, 1968.

Berlinski, D., 2001. The Advent of the Algorithm: The 300-Year Journey from an Idea to the Computer. Harcourt, San Diego, CA, originally published 2000 with the subtitle 'The Idea That Rules the World'.

Bostan, A., Kauers, M., Sep. 2010. The complete generating function for Gessel walks is algebraic. Proceedings of the AMS 138, 3063–3078, with an Appendix by Mark van Hoeij.

Buchberger, B., 1965. Ein Algorithmus zum Auffinden der Basiselemente des Restklassenringes nach einem nulldimensionalen Polynomideal. Dissertation, Univ. Innsbruck, Austria.

Dumas, J.-G., Kaltofen, E., 2014. Essentially optimal interactive certificates in linear algebra. In: Nabeshima (2014), pp. 146–153, URL: http://users.cs.duke.edu/~elk27/bibliography/14/DuKa14.pdf.

Grabmeier, J., Kaltofen, E., Weispfenning, V. (Eds.), 2003. Computer Algebra Handbook. Springer Verlag, Heidelberg, Germany, 637 + xx pages + CD-ROM. Includes E. Kaltofen and V. Weispfenning §1.4 Computer algebra – impact on research, pages 4–6; E. Kaltofen §2.2.3 Absolute factorization of polynomials, page 26; E. Kaltofen and B. D. Saunders §2.3.1 Linear systems, pages 36–38; R. M. Corless, E. Kaltofen and S. M. Watt §2.12.3 Hybrid methods, pages 112–125; E. Kaltofen §4.2.17 FoxBox and other blackbox systems, pages 383–385. URL: http://users.cs.duke.edu/~elk27/bibliography/01/symnum.pdf.

Hao, Z., Kaltofen, E., Zhi, L., 2016. Numerical sparsity determination and early termination. In: Rosenkranz, M. (Ed.), ISSAC'16 Proc. 2016 ACM Internat. Symp. Symbolic Algebraic Comput. Association for Computing Machinery, New York, N. Y., pp. 247–254, URL: http://users.cs.duke.edu/~elk27/bibliography/16/HKZ16.pdf.

van Hoeij, M., 2002. Factoring polynomials and the knapsack problem. J. of Number Theory 95, 167–189.

Kaltofen, E., Lee, W., 2003. Early termination in sparse interpolation algorithms. J. Symbolic Comput. 36 (3–4), 365–400, special issue Internat. Symp. Symbolic Algebraic Comput. (ISSAC 2002). Guest editors: M. Giusti & L. M. Pardo. URL: http://users.cs.duke.edu/~elk27/bibliography/03/KL03.pdf.

Kaltofen, E., Pernet, C., 2014. Sparse polynomial interpolation codes and their decoding beyond half the minimal distance. In: Nabeshima (2014), pp. 272–279, URL: http://users.cs.duke.edu/~elk27/bibliography/14/KaPe14.pdf.

Kaltofen, E., Trager, B., 1990. Computing with polynomials given by black boxes for their evaluations: Greatest common divisors, factorization, separation of numerators and denominators. J. Symbolic Comput. 9 (3), 301–320, URL: http://users.cs.duke.edu/~elk27/bibliography/90/KaTr90.pdf.

Lenstra, A. K., Lenstra, Jr., H. W., Lovász, L., 1982. Factoring polynomials with rational coefficients. Math. Ann. 261, 515–534.

Nabeshima, K. (Ed.), 2014. ISSAC 2014 Proc. 39th Internat. Symp. Symbolic Algebraic Comput. Association for Computing Machinery, New York, N. Y.

Safey El Din, M., Schost, É., 2017. A nearly optimal algorithm for deciding connectivity queries in smooth and bounded real algebraic sets. J. ACM 63 (6), 48:1–48:37, URL: https://doi.org/10.1145/2996450.

Storjohann, A., 2005. The shifted number system for fast linear algebra on integer matrices. J. Complexity 21 (5), 609–650.

Wiedemann, D., 1986. Solving sparse linear equations over finite fields. IEEE Trans. Inf. Theory IT-32, 54–62.

Zippel, R., 1990. Interpolating polynomials from their values. J. Symbolic Comput. 9 (3), 375–403.