

# The MICA 2008 Conference, Preface and Abstracts

Communicated by Mark Giesbrecht

The MICA 2008 conference was held May 1–3, 2008 in Stonehaven Bay, Trinidad. The conference honoured the scientific career of Keith Geddes. Colleagues, students and friends celebrated Professor Geddes' achievements in many areas: in fundamental research, in technology transfer, and in the training of the next generation of scientists, mathematicians and engineers.

Keith received his PhD in 1973 from the University of Toronto under the supervision of John C. Mason. Since that time, as a professor at the University of Waterloo, his research has spanned the areas of numerical approximation, algebraic algorithms for symbolic computation, hybrid symbolic-numeric computation and the design and implementation of computer algebra systems. Keith has actively supported our community through the ACM Special Interest Group on Symbolic and Algebraic Manipulation (SIGSAM), which he chaired from 1991 to 1993, and numerous conference program committees. He is perhaps best known as co-founder of the Maple computer algebra system. Through his teaching, research, service and software, the work of Keith Geddes has literally touched millions of individuals.

The timing of the conference marked many numerically significant milestones: Keith was born just over 60 years ago in Saskatchewan, he began his research career just under 40 years ago as a graduate student at the University of Toronto, he co-founded Maplesoft 20 years ago in Waterloo and now he has chosen to start his retirement at the end of 2008. This was clearly an occasion that called for celebration!

Almost four dozen scientific colleagues came together at MICA 2008 to pay tribute to Keith. This included eight distinguished invited speakers, some two dozen colleagues who have contributed scientific papers and posters, and many others who come to pay their respects. In addition, a great many colleagues sent their wishes, but could not attend in person.

Many people contributed time and effort to make the conference a success: The authors and speakers have prepared a great collection of high-quality contributions, the program committee spent time and effort reviewing the submissions, the members of the organizing committee all took on additional responsibilities, and many student volunteers helped with practical aspects. We thank Maplesoft for major funding for the meeting; without their support the meeting could not have taken place. We thank the University of Waterloo for additional financial contribution. We also thank ACM SIGSAM, the Ontario Research Centre for Computer Algebra (ORCCA), the University of Western Ontario and the University of the West Indies for their support.

Everyone who knows Keith and has had the privilege to work with him will attest to his qualities as a scholar and person. On behalf of all who participated in this conference, and all those who sent their best wishes, we thank Keith for his many contributions, and wish him a rich and active retirement!

Stephen Watt General Chair

Mark Giesbrecht Program Committee Chair

Marc Moreno Maza Proceedings Co-editor

## Organization

### Program Committee

Jacques Carette (McMaster U., Canada)  
Robert Corless (U. Western Ontario, Canada)  
James Davenport (U. Bath, UK)  
Jürgen Gerhard (Maplesoft, Canada)  
Mark Giesbrecht [Chair] (U. Waterloo, Canada)  
Laureano Gonzalez Vega (U. Cantabria, Spain)  
Hoon Hong (North Carolina State U., USA)  
Erich Kaltofen (North Carolina State U., USA)  
George Labahn (U. Waterloo, Canada)  
Ziming Li (AMSS Academia Sinica, China)

<b>General Chair</b>	Stephen Watt (U. Western Ontario, Canada)
<b>Sponsorship Chair</b>	George Labahn (U. Waterloo, Canada)
<b>Publicity Chair</b>	Ilias Kotsireas (Wilfrid Laurier U., Canada)
<b>Treasurer</b>	Jim Cooper (Maplesoft, Canada)
<b>Local Arrangements</b>	Bal Swaroop Bhatt (U. West Indies, Trinidad and Tobago) David Jeffrey (U. Western Ontario, Canada)
<b>Electronic Proceedings</b>	Marc Moreno Maza (U. Western Ontario, Canada) Stephen Watt (U. Western Ontario, Canada)
<b>Logistics</b>	Changbo Chen Cris Frusina Wei Pan Daniel Roche Eric Schost Yuzhen Xie

## Invited Talks

### 20 Years and 6 Definitions of Zero Later...

*Jim Cooper, Maplesoft*

### Maple as a prototyping language: a concrete and successful experience

*Gaston Gonnet, ETH Zurich*

Aruna PLC, a database company, embarked around 1999-2001 in the writing of an SQL (a relational database query language) query processor. A query processor normally consists of a parser, a simplifier of the query expression, a plan generation (how to execute the query) and a plan optimizer. Since we had many ideas about how to implement such a query processor and these were difficult to transmit to the programmer team, we decided to write a prototype of the QP in Maple to provide a "live" example of the algorithms. The prototype in Maple turned to be a full fledged prototype which is alive and active even today. We will describe several of the highlights of the prototype and how it contributed in many ways to the development of the query processor. Some software engineering aspects are also quite remarkable. The pervasive "symbolic" nature of the Maple implementation was also an extremely positive feature.

### How fast can we multiply and divide sparse polynomials?

*Michael B. Monagan, Simon Fraser University*

Most of today's computer algebra systems use either a sparse distributed data representation for multivariate polynomials or a sparse recursive representation. For example Axiom, Magma, Maple, Mathematica, and Singular use distributed representations as their primary representation. Macsyma, REDUCE, and TRIP use recursive representations. In 1984 David Stoutemyer suggested "recursive dense" as an alternative and showed that it was the best overall across the Derive test suite. Of the newer systems, only Pari uses recursive dense.

In 2003, Richard Fateman compared the speed of polynomial multiplication in many computer algebra systems. He found that Pari was clearly the fastest system on his benchmarks. His own implementation of recursive dense came in second. So is recursive dense best?

In this talk we take another look at the sparse distributed representation with terms sorted in a monomial ordering. Our algorithms for polynomial multiplication and division use an auxiliary data structure, a "chained heap of pointers". Using a heap for polynomial arithmetic was first suggested by Stephen Johnson in 1974 and used in the Altran system. But the idea seems to have been lost. Let  $h = fg$  where the number of terms in  $f$ ,  $g$ , and  $h$  are  $\#f$ ,  $\#g$ , and  $\#h$ . The heap gives us the following:

1. It reduces the number of monomial comparisons to  $O(\#f \#g \log \min(\#f, \#g))$ .
2. For dense polynomials, chaining reduces this to  $O(\#f\#g)$ .
3. By using  $O(1)$  registers for bignum arithmetic, multiplication can be done so that the terms of the product  $fg$  are output sequentially with no garbage created.
4. The size of the heap is  $O(\min(\#f, \#g))$  which fits in the cache.
5. For polynomials with integer coefficients, the heap enables multivariate pseudo-division - our division code is as fast as multiplication.

In the talk I would like to first show Maple's distributed data structure, Pari's recursive dense structure, Yan's "geobuckets" which are used in Singular for division, and our own distributed structure. Second I will show how heaps of pointers work. Third, our benchmarks suggest pointer heaps are really good. The timings are much faster than Pari, Magma, and Singular and much much faster than Maple. Fourth, I will show some of the other "necessary optimizations" needed to get high performance. The most important one is to encode monomials into a single word.

But there are others. Including the right way of extracting an element from a heap. This is joint work with Roman Pearce at Simon Fraser University.

### **Macsyma: A Personal History**

*Joel Moses, MIT*

The Macsyma system arose out of research on mathematical software in the AI group at MIT in the 1960's. Algorithm development in symbolic integration and simplification arose out of the interest of people, such as the author, who were also mathematics students. The later development of algorithms for the GCD of sparse polynomials, for example, arose out of the needs of our user community. During various times in the 1970's the computer on which Macsyma ran was one of the most popular nodes on the ARPANET. We discuss the attempts in the late 70's and the 80's to develop Macsyma systems that ran on popular computer architectures. Finally, we discuss the impact of the fundamental ideas in Macsyma on current research on large scale engineering systems.

### **Integrals, Sums and Computer Algebra**

*Peter Paule, Research Institute for Symbolic Computation (RISC)*

The types of mathematics being considered in in this talk are related to some of Keith Geddes' research interests, namely: computational aspects of algebra and analysis, including the solution of problems in integral and differential calculus, and closed-form summation. The thread of my presentation will be Victor Moll's article "The evaluation of integrals: A personal story" (Notices of the AMS, 2002) which begins with the remark, "... It was even more a surprise to discover that new things can still be said today about the mundane subject of integration of rational functions of a single variable and that this subject has connections with branches of contemporary mathematics as diverse as combinatorics, special functions, elliptic curves, and dynamical systems." In this talk I will add another ingredient to Moll's story, namely computer algebra. I will show how recently developed procedures can be used to retrieve observations which in Moll's original approach were derived with classical methods, like the positivity of the coefficients of a specialized family of Jacobi polynomials. In addition, as a result from a recent collaboration with Manuel Kauers (RISC), I will demonstrate that computer algebra can do even more, namely by proving Moll's longstanding log-concavity conjecture with a combination of various algorithms.

### **Linear Algebra**

*B. David Saunders, University of Delaware*

Computer algebra systems have become a major tool of science and engineering education and practice. Thoughts of CAS immediately bring to mind Maple, and thus Keith Geddes, but not so much linear algebra. I'm sure Keith has never long entertained a linear thought. That is good! To solve a linear system is perhaps the best understood of mathematical problems. However, this is largely because the concepts of matrix and linear system are overly general. In fact there is not one "solve linear system" problem, but many, depending on the structure of the matrix and application. It remains a challenge to compute solutions efficiently as hardware evolves, and the matter is rich in interesting computer science and mathematics and of growing importance to computer algebra. I will survey the history and the state of the art, and offer a view of the road ahead.

**Ten Commandments for Good Default Expression Simplification**

*David R. Stoutemyer, University of Hawaii*

This article motivates and identifies ten goals for good default expression simplification in computer algebra. Although oriented toward computer algebra, many of these goals are also applicable to manual simplification. The article then explains how the Altran partially-factored form for rational expressions was extended for Derive and the computer algebra in Texas Instruments products to help achieve these goals. In contrast to the distributed Altran representation, this recursive partially-factored semi-fraction form

- doesn't unnecessarily force common denominators,
- discovers and preserves significantly more factors,
- can represent general expressions, and
- can produce the entire spectrum from fully factored over a common denominator through complete partial fractions, including a dense subset of intermediate forms.

**Tropical Algebraic Geometry in Maple**

*Jan Verschelde, University of Illinois at Chicago*

Finding a common factor of two multivariate polynomials with approximate coefficients is a problem in symbolic-numeric computing. Taking a tropical view on this problem leads to efficient preprocessing techniques, alternatingly applying polyhedral methods on the exact exponents with numerical techniques on the approximate coefficients. With Maple we will illustrate our use of tropical algebraic geometry. This is work in progress jointly with Danko Adrovic.

## Contributed Papers

### Teaching first-year engineering students with “modern day” Maple

*Fred Chapman, Bruce Char and Jeremy Johnson*

We describe our experiences teaching technical computation to Drexel University engineering freshmen using Maple 11, in a one year sequence of three one credit courses. While progress has been made in providing sophistication and ease of use to programming novices, we note some problems that still remain.

### On the Representation of Constructible Sets

*Changbo Chen, Liyun Li, Marc Moreno Maza, Wei Pan and Yuzhen Xie*

The occurrence of redundant components is a natural phenomenon when computing with constructible sets. We present different algorithms for computing an irredundant representation of a constructible set or a family thereof. We provide a complexity analysis and report on an experimental comparison.

### The Maximality of Dixon Matrices on Corner-Cut Monomial Supports by Almost-Diagonality

*Eng-Wee Chionh*

The maximality of the Dixon matrix on corner cut monomial supports with at most three exposed points at the corners is established using almost-diagonal Dixon matrices. The search of these matrices is conducted using Maple.

### Computing Popov Form of General Ore Polynomial Matrices

*Patrick Davies, Howard Cheng and George Labahn*

The computation of the Popov form of Ore polynomial matrices is formulated as a problem of computing the left nullspace of such matrices. While this technique is already known for polynomial matrices, the extension to Ore polynomial matrices is not immediate because multiplication of the matrix entries is not commutative. A number of results for polynomial matrices are extended to Ore polynomial matrices in this paper. This in turn allows nullspace algorithms to be used in Popov form computations. Fraction-free and modular algorithms for nullspace computation can be used in exact arithmetic setting where coefficient growth is a concern. When specialized to ordinary polynomial matrices, our results simplify the proofs for the computation of Popov form while keeping the same worst case complexity.

### Compressed Modular Matrix Multiplication

*Jean-Guillaume Dumas, Laurent Fousse and Bruno Salvy*

Matrices of integers modulo a small prime can be compressed by storing several entries into a single machine word. Modular addition is performed by addition and possibly subtraction of a word containing several times the modulus. We show how modular multiplication can also be performed. In terms of arithmetic operations, the gain over classical matrix multiplication is equal to the number of integers that are stored inside a machine word. The gain in actual speed is also close to that number.

First, modular dot product can be performed via an integer multiplication by the reverse integer. Modular multiplication by a word containing a single residue is also possible. We give bounds on the sizes of primes and matrices for which such a compression is possible. We also make explicit the details of the required compressed arithmetic routines and show some practical performance.

**Black Box Matrix Contributions: Two Improvements***Wayne Eberly*

Two enhancements for black box matrix computations are described.

**Barycentric Birkhoff Interpolation***Laureano Gonzalez-Vega, R Corless, John C. Butcher, Dhavide A. Aruliah and Azar Shakoori*

We extend to the Birkhoff Interpolation Scheme the construction of barycentric interpolants when the considered scheme is poised.

**Solving the separation problem for two ellipsoids involving only the evaluation of six polynomials***Laureano Gonzalez-Vega and Esmeralda Mainar*

By using several tools coming from Real Algebraic Geometry and Computer Algebra (Sturm–Habicht sequences), a new condition for the separation of two ellipsoids in three-dimensional Euclidean space is introduced. This condition is characterized by a set of equalities and inequalities depending only on the matrices defining the two considered ellipsoids and does not require in advance the computation (or knowledge) of the intersection points between them.

Moreover, this characterization is specially well adapted for computationally treating the case where the considered ellipsoids depend on one or several parameters. But specific techniques for dealing with the big involved expressions when rational motions are involved are required.

**Systematic Tensor Simplification: A Diagrammatic Approach***Anthony Kennedy and Thomas Reiter*

Simplification of tensor expressions is important in many applications of computer algebra, and many systems have been written to undertake this task. It is crucial to make use of the all the symmetries of an expression both to reduce the computational complexity of the simplification algorithm and the size of the simplified expression. Most if not all existing systems do this using various heuristic approaches, including Keith Geddes' contributions to the subject: we propose instead a systematic non-heuristic approach using completeness and recoupling relations for the irreducible representations of the appropriate symmetry group. This reduces any tensor expression into a sum of basis tensors, corresponding to tree graphs in our diagrammatic notation, with coefficients that are rational functions of 0-j(dimensions), 3-j, and 6-j coefficients. These n-j coefficients are readily computed and reused for all symmetry groups of interest, and for the case of  $Sl(N)$  we give a new efficient algorithm for computing them.

**The Modpn Library: Bringing Fast Polynomial Arithmetic into Maple***Xin Li, Marc Moreno Maza, Raqeeb Rasheed and Eric Schost*

One of the main successes of the computer algebra community in the last 30 years is the discovery of algorithms, called modular methods, that allow to keep the swell of the intermediate expressions under control. Without these methods, many applications of Computer Algebra would not be possible and the impact of computer algebra in scientific computing would be severely limited. Amongst the computer algebra systems which have emerged in the 70's and 80's Maple and its developers have played an essential role in this area.

Another major advance in symbolic computation is the development of implementation techniques for asymptotically fast polynomial arithmetic. Computer algebra systems and libraries initiated in the 90's, such as Magma and NTL, have been key actors in this effort.

In this extended abstract, we present Modpn, a Maple library dedicated to fast arithmetic for multivariate polynomials over finite fields. The main objective of Modpn is to provide highly efficient routines for supporting the implementation of modular methods in Maple.

We illustrate the impact of fast polynomial arithmetic on a simple modular method by comparing its implementations in Maple with classical arithmetic and with the Modpn library. Then, we discuss the design of Modpn, its

strengths and necessary future improvements.

### **Computer algebra and experimental mathematics**

*Petr Lisonek*

We discuss a new paradigm for experimental mathematics research supported by computer algebra. Instead of using the computer algebra system for studying one specific mathematical phenomenon interactively, we aim at acquiring a broader understanding of it by searching (in a “data mining” style) a large data set, such as the On-line encyclopedia of integer sequences. We give a case study documenting the viability and usefulness of this approach.

### **Numerical Analysis with Maple**

*Ales Nemecek and Mirko Navara*

We summarize more than 10 years of our experience with a course of Numerical Analysis with the use of Maple. Besides software packages, we discuss also the principles of education.

### **Summation of Linear Recurrence Sequences**

*Robert Ravenscroft and Ed Lamagna*

We describe a simple and efficient algorithm for evaluating summations of sequences defined by linear recurrences with constant coefficients. The result is expressed in finite terms using the sequence name. The algorithm can evaluate all homogeneous recurrences and a large variety of inhomogeneous recurrences.

### **Max-Plus Linear Algebra in Maple and Generalized Solutions for First-Order Ordinary BVPs via Max-Plus Interpolation**

*Georg Regensburger*

If we consider the real numbers extended by minus infinity with the operations maximum and addition, we obtain the max-algebra or the max-plus semiring. The analog of linear algebra for these operations extended to matrices and vectors has been widely studied.

We outline some facts on semirings and max-plus linear algebra, in particular, the solution of max-plus linear systems. As an application, we discuss how to compute symbolically generalized solutions for nonlinear first-order ordinary boundary value problems (BVPs) by solving a corresponding max-plus interpolation problem. Finally, we present the Maple package `MaxLinearAlgebra` and illustrate the implementation and our application with some examples.

### **Automatic Regression Test Generation for the SACLIB Computer Algebra Library**

*David Richardson and Werner Krandick*

We use aspects to weave tracing code into the SACLIB library of computer algebra programs. For each SACLIB function that is called during a high-level application the tracing code records the signature of the function together with its inputs and outputs. The recorded data are used to generate a test bed for each SACLIB function that can be used for regression testing.

### **Adaptive Polynomial Multiplication**

*Daniel S. Roche*

Finding the product of two polynomials is an essential and basic problem in computer algebra. While most previous results have focused on the worst-case complexity, we instead employ the technique of adaptive analysis to give an improvement in many “easy” cases where other algorithms are doing too much work. Three ideas for adaptive polynomial multiplication are given. One method, which we call “chunky” multiplication, is given a more careful

analysis, as well as an implementation in NTL. We show that significant improvements can be had over the fastest general-purpose algorithms in many cases.

### **A Survey of Recent Advancements of Multivariate Hensel Construction and Applications**

*Tateaki Sasaki*

Multivariate Hensel construction (the generalized Hensel construction) is a very important tool in computer algebra. It plays essential roles in multivariate GCD computation, multivariate factorization, and so on. In early 1970's, important advancements in computational technique were made: formulation by interpolation polynomials by Moses and Yun, and parallel Hensel construction by Yun and Wang. Twenty years later, two important advancements have been made: in 1993, Sasaki and Kako formulated the Hensel construction at singular points, which they called "extended Hensel construction (EHC)". In 2008, Sasaki and Inaba expressed the multivariate Hensel factors in the roots of initial factors. These advancements lead us to wide applications, theoretically as well as practically. In this article, we describe these advancements and survey representative applications attained so far or being attained now.

### **Geometric properties of locally minimal energy configurations of points on spheres and special orthogonal groups**

*Elin Smith and Chris Peterson*

In this paper, we construct locally minimal energy configurations of  $t$  points on the unit sphere  $S^{n-1} \subseteq \mathbb{R}^n$ . We utilize basic linear algebra and the computer program *Groups, Algorithms, Programming* (GAP) to generate the subgroups of  $SO(n), O(n)$  which permute these points. We also consider the colored complete graph  $K_t$  induced by the configuration and the subgroup of the symmetric group,  $S_t$ , which preserves the edge colored  $K_t$ . Next we consider locally minimal energy configurations of points on  $SO(n)$  (as a manifold). After a shift of the configuration, we consider the group generated by the corresponding elements of  $SO(n)$  (as a group). In some cases we are able to utilize the LLL algorithm (via Maple) to recover exact representations from the numerical data produced by the algorithms. Finally, we consider basic examples of locally minimal energy configurations of points on other manifolds.

### **Differentiation of Kaltofen's Division-Free Determinant Algorithm**

*Gilles Villard*

Kaltofen has proposed a new approach in [Kaltofen 1992] for computing matrix determinants. The algorithm is based on a baby steps/giant steps construction of Krylov subspaces, and computes the determinant as the constant term of a characteristic polynomial. For matrices over an abstract field and by the results of Baur and Strassen, the determinant algorithm, actually a straight-line program,

### **Symbolic Polynomials with Sparse Exponents**

*Stephen Watt*

Earlier work has presented algorithms to factor and compute GCDs of symbolic Laurent polynomials, that is multivariate polynomials whose exponents are integer-valued polynomials. These earlier algorithms had the problem of high computational complexity in the number of exponent variables and their degree. The present paper solves this problem, presenting a method that preserves the structure of sparse exponent polynomials.

## Posters

### Determining when projections are the only homomorphisms

*David Casperson*

A projection algebra  $M$  is a finite algebra with the property that every homomorphism from a subalgebra of a finite power of  $M$  to  $M$  is the restriction of a co-ordinate projection. We show that determining whether  $M$  is a projection algebra is computable by giving a term condition that is equivalent to being a projection algebra.

### Triangular Decompositions for Solving Parametric Polynomial Systems

*Changbo Chen, Marc Moreno Maza, Bican Xia and Lu Yang*

We extend to parametric constructible sets and parametric semi-algebraic sets the notion of comprehensive triangular decomposition, which was originally limited to algebraic varieties. We propose algorithms for computing all these decompositions. This establishes a unified framework for solving parametric polynomial systems by means of triangular decompositions. In addition, we investigate the relations between the notions of border polynomial, discriminant set and minimal discriminant variety.

### Automatic Variable Order Selection for Polynomial System Solving

*Mark Giesbrecht, John May, Marc Moreno Maza, Daniel Roche and Yuzhen Xie*

The goal of a general purpose solver is to allow a user to compute the solutions of a system of equations with minimal interactions. Modern tools for polynomial system solving, namely triangular decomposition and Groebner basis computation, can be highly sensitive to the ordering of the variables. Our goal is to examine the structure of a given system and use it to compute a variable ordering that will cause the solving algorithm to complete quickly (or alternately, to give compact output). We explore methods based on the dependency graph of coincident variables and terms between the equations. Desirable orderings are gleaned from connected components and other topological properties of these graphs, under different weighting schemes.

### A Note on the Functional Decomposition of Symbolic Polynomials

*Stephen Watt*

It often arises that the general form of a polynomial is known, but the particular values for the exponents are unknown. For example, we may know a polynomial is of the form  $3x^{(n^2+n)/2} - y^{2m} + 2$ , where  $n$  and  $m$  are integer-valued parameters. We consider the case where the exponents are multivariate integer-valued polynomials with coefficients in  $\mathbb{Q}$  and call these “symbolic polynomials.” Earlier work has presented algorithms to factor symbolic polynomials and compute GCDs. Here, we extend the notion of univariate polynomial decomposition to symbolic polynomials and presents an algorithm to compute these decompositions. For example, the symbolic polynomial  $f(x) = 2x^{n^2+n} - 4x^{n^2} + 2x^{n^2-n} + 1$  can be decomposed as  $f = g \circ h$  where  $g(x) = 2x^2 + 1$  and  $h(x) = x^{n^2/2+n/2} - x^{n^2/2-n/2}$ .

**A Preliminary Report on the Set of Symbols Occurring in Engineering Mathematics Texts***Stephen Watt*

Certain forms of mathematical expression are used more often than others in practice. We propose that a quantitative understanding of actual usage can provide information to improve the accuracy of software for the input of mathematical expressions from scanned documents or handwriting and allow more natural forms of presentation of mathematical expressions by computer algebra systems. Earlier work has examined this question for the diverse set of articles from the mathematics preprint archive arXiv.org. That analysis showed the variance between mathematical areas. The present work analyzes a particular mathematical domain more deeply. We have chosen to examine second year university engineering mathematics as taught in North America as the domain. We have analyzed the set of expressions occurring in the most popular textbooks, weighted by popularity. Assuming that early training influences later usage, we take this as a model of the set of mathematical expressions used by the population of North American engineers. We present an preliminary empirical analysis of the symbols and  $n$ -grams occurring in these expressions.