To Herbert Gerhard (1921–1999)

Foreword

This work brings together two streams in computer algebra: symbolic integration and summation on the one hand, and fast algorithmics on the other hand.

In many algorithmically oriented areas of computer science, the *analysis of algorithms* – placed into the limelight by Don Knuth's talk at the 1970 ICM – provides a crystal-clear criterion for success. The researcher who designs an algorithm that is faster (asymptotically, in the worst case) than any previous method receives instant gratification: her result will be recognized as valuable. Alas, the downside is that such results come along quite infrequently, despite our best efforts.

An alternative evaluation method is to run a new algorithm on examples; this has its obvious problems, but is sometimes the best we can do. George Collins, one of the fathers of computer algebra and a great experimenter, wrote in 1969: "I think this demonstrates again that a simple analysis is often more revealing than a ream of empirical data (although both are important)."

Within computer algebra, some areas have traditionally followed the former methodology, notably some parts of polynomial algebra and linear algebra. Other areas, such as polynomial system solving, have not yet been amenable to this approach. The usual "input size" parameters of computer science seem inadequate, and although some natural "geometric" parameters have been identified (solution dimension, regularity), not all (potential) major progress can be expressed in this framework.

Symbolic integration and summation have been in a similar state. There are some algorithms with analyzed run time, but basically the mathematically oriented world of integration and summation and the computer science world of algorithm analysis did not have much to say to each other.

Gerhard's progress, presented in this work, is threefold:

- a clear framework for algorithm analysis with the appropriate parameters,
- the introduction of modular techniques into this area,
- almost optimal algorithms for the basic problems.

One might say that the first two steps are not new. Indeed, the basic algorithms and their parameters – in particular, the one called dispersion in Gerhard's work – have been around for a while, and modular algorithms are a staple of computer algebra. But their combination is novel and leads to new perspectives, the almost optimal methods among them.

A fundamental requirement in modular algorithms is that the (solution modulo p) of the problem equal the solution of the (problem modulo p). This is generally not valid for all p, and a first task is to find a nonzero integer "resultant" r so that the requirement is satisfied for all primes p not dividing r. Furthermore, r has to be "small", and one needs a bound on potential solutions, in order to limit the size and number of the primes p required. These tasks tend to be the major technical obstacles; the development of a modular algorithm is then usually straightforward. However, in order to achieve the truly efficient results of this work, one needs a thorough understanding of the relevant algorithmics, plus a lot of tricks and shortcuts.

The integration task is naturally defined via a limiting process, but the Old Masters like Leibniz, Bernoulli, Hermite, and Liouville already knew when to treat it as a symbolic problem. However, its formalization – mainly by Risch – in a purely algebraic setting successfully opened up perspectives for further progress. Now, modular differential calculus is useful in some contexts, and computer algebra researchers are aware of modular algorithms. But maybe the systematic approach as developed by Gerhard will also result in a paradigm shift in this field. If at all, this effect will not be visible at the "high end", where new problem areas are being tamed by algorithmic approaches, but rather at the "low end" of reasonably domesticated questions, where new efficient methods will bring larger and larger problems to their knees.

It was a pleasure to supervise Jürgen's Ph.D. thesis, presented here, and I am looking forward to the influence it may have on our science.

Paderborn, 9th June 2004 Joachim von zur Gathen

Preface

What fascinated me most about my research in symbolic integration and symbolic summation were not only the strong parallels between the two areas, but also the differences. The most notable non-analogy is the existence of a polynomial-time algorithm for rational integration, but not for rational summation, manifested by such simple examples as $1/(x^2 + mx)$, whose indefinite sum with respect to x has the denominator $x(x + 1)(x + 2) \cdots (x + m - 1)$ of exponential degree m, for all positive integers m. The fact that Moenck's (1977) straightforward adaption of Hermite's integration algorithm to rational summation is flawed, as discussed by Paule (1995), illustrates that the differences are intricate.

The idea for this research was born when Joachim von zur Gathen and I started the work on our textbook *Modern Computer Algebra* in 1997. Our goal was to give rigorous proofs and cost analyses for the fundamental algorithms in computer algebra. When we came to Chaps. 22 and 23, about symbolic integration and symbolic summation, we realized that although there is no shortage of algorithms, only few authors had given cost analyses for their methods or tried to tune them using standard techniques such as modular computation or asymptotically fast arithmetic. The pioneers in this respect are Horowitz (1971), who analyzed a modular Hermite integration algorithm in terms of word operations, and Yun (1977a), who gave an asymtotically fast algorithm in terms of arithmetic operations for the same problem. Chap. 6 in this book unites Horowitz's and Yun's approaches, resulting in two asymptotically fast and optimal modular Hermite integration algorithms. For modular hyperexponential integration and modular hypergeometric summation, this work gives the first complete cost analysis in terms of word operations.

Acknowledgements. I would like to thank:

My thesis advisor, Joachim von zur Gathen.

Katja Daubert, Michaela Huhn, Volker Strehl, and Luise Unger for their encouragement, without which this work probably would not have been finished.

My parents Johanna and Herbert, my sister Gisela, my brother Thomas, and their families for their love and their support.

My colleagues at Paderborn: the Research Group Algorithmic Mathematics, in particular Marianne Wehry, the MuPAD group,

in particular Benno Fuchssteiner, and SciFace Software, in particular Oliver Kluge.

My scientific colleagues all over the world for advice and inspiring discussions: Peter Bürgisser, Frédéric Chyzak, Winfried Fakler, Mark Giesbrecht, Karl-Heinz Kiyek, Dirk Kussin, Uwe Nagel, Christian Nelius, Michael Nüsken, Walter Oevel, Peter Paule, Arne Storjohann, and Eugene Zima.

Waterloo, 23rd July 2004 Jürgen Gerhard