[LeN98] Robert H. Lewis and George Nakos, "Solving the Six-Line Problem with the Dixon Resultant", in "High Performance Symbolic Computation and Challenges of Computer Algebra", 1998 IMACS ACA.

[MuP] MuPAD, anonymous FTP at athene.uni-paderborn.de:unix/MuPAD.

[NTL] NTL, http://www.cs.wisc.edu/~shoup/ntl/index.html.

[Par] Pari-Gp, anonymous FTP at ftp://megrez.math.u-bordeaux.fr/pub/pari.

[Sim92] Barry Simon, "Comparative CAS Reviews", Notices of the American Mathematical Society, Volume 39, Number 7, September 1992, 700-710.

[Sin] Singular, http://www.mathematik.uni-kl.de/~zca/Singular/Welcome.html.

[Wes94] Michael Wester, "A Review of CAS Mathematical Capabilities", Computer Algebra Nederland Nieuwsbrief, Number 13, December 1994, ISSN 1380-1260, 41-48.

2^W-ary algorithm for extended problem of integer GCD

Hirokazu MURAO
Information Technology Center, University of Tokyo
Yayoi 2-11-16, Bunkyo-ku, Tokyo 113-8658, JAPAN
murao@cc.u-tokyo.ac.jp

One of the well-known algorithms for integer GCDs is the binary algorithm, and it is a common recognition that the binary algorithm works very fast, and often much faster than the Euclidean algorithm with division, for long integers on the modern computer systems with RISC architecture. In these recent years, a new algorithm is developed, which genelizes the binary algorithm to the 2^W-ary one so that the bit-elimination and shift operations are performed on multiple (W) bits at one time.

In this poster abstract, we treat such an extended GCD problem for computing cofactors a and b as well as the GCD of given two integers X and Y such that $aX+bY = \gcd(X, Y)$. We apply the 2^W-ary method to this problem and develop a new algorithm. A binary algorithm itself for this problem is already known, however, its generalization to the 2^W-ary one is not straightforward, because the treatment of cofactors is required corredpondingly to the main sequence of X and Y. The main idea to circumvent the problem arisen by the problem extension is quite simple, and is the application of the 2^W-ary elimination technique to two additional stages. The resulting new algorithm is a natural combination of the existing methods.

More specifically, given two integers X and Y where $X > Y$, we use the 2^W-ary algorithm and compute the sequence $C_{i+1} = (C_{i-1} - q_iC_i)/2^W$ with some appropriate q_i, to finally obtain $\gcd(X, Y)$. Along with this main sequence, we must determine the cofactors a_i and b_i such that $C_i = a_iX + b_iY$. Notice that the calculation of a_{i+1} and b_{i+1} is not straightforward because $(a_{i-1} - q_ia_i)$ is not guaranteed to be divisible by 2^W. The essence of the 2^W-ary algorithm reminds us a natural solution to this problem. We apply the 2^W-ary elimination technique to this cofactor sequence.

One more problem is the treatment of the cases when X or Y is divisible by 2. In this case, we cannot perform bit-elimination by X or Y. In the case of a simple problem only for the GCD, we can eliminate 2's factor from X and Y initially, however we cannot in the case of the extended problem. Let $\bar{X}$ and $\bar{Y}$ be the respective factors of X and Y with 2's factor removed. We apply the 2^W-ary algorithm to $\bar{X}$ and $\bar{Y}$, and try to recover the cofactors for X and Y from those for $\bar{X}$ and $\bar{Y}$. The main idea to realize this is the application of the 2^W-ary bit-elimination technique, again.

Our new algorithm is useful, and required to efficiently compute the modular reciprocal $X^{-1} \bmod P$ for a given X and a prime P. This problem often appears in the arithmetic operations on an elliptic curve, and its efficient calculation is crucial for speeding-up the calculation of cipher codes.

Optimal Starting Approximation and Iterative Algorithm for Inverse Error Function

Bogdan A. Popov
Institute for Phisics and Mechanics
of Ukrainian Academy of Sciences
bogdan@popov.lviv.ua
Oksana Laushnyk
Lviv State University,
Applied Mathematics Department
oksana@yahoo.com

1 Introduction

The evaluation of inverse functions is often achieved via iterative methods. Here the ability to set the precision of intermediate calculations is of particular importance. Thus, the initial approximations can be computed to low precision, and then the precision can be increased in parallel with the approximation accuracy. The feature of such approach is that the magnitude of the error after some number of iterations is the known function of the initial error. Several iteration procedures and initial guesses for inverse error function were already proposed [3, 5]. But there were no investigations of selecting the optimal algorithm and initial guess for this function.

2 Iterative Algorithms for Inverse Error Function

The proposed iterative algorithms for functions evaluation are based on a *series expansion by residuals* [4]. Let suppose there is a function $y = f(x)$, which we wish to evaluate on some domain. The method of expansion by residuals is developed from functional identity such that the function $f(x)$ satisfies [3]

$$F(x, f(x)) = 0. \tag{1}$$

Let y_0 be an approximate value for $f(x)$ and let $z_0 = F(x, y_0)$; z_0 is called the residual of F at (x, y_0). Here we must be able to solve the residual equation for $x = \phi(y_0, z_0)$ and expand $y = f(x)$ in a series expansion in z_0.

By truncating the series expansion after m terms, we obtain

$$y_m = y_0 + \sum_{k=0}^{m} a_k z_0^k. \tag{2}$$

Let Δ_m be the signed error in y_m, so $y_m = y + \Delta_m$, for brevity $\Delta_0 = \Delta$. If Δ is sufficiently small it is possible to expand expression $z_0 = F(x, y_0) = F(x, y + \Delta)$ in power series in Δ. In general, if the weight error measure is used, say $y_m = y + \delta_m w$, we obtain an expression for δ_m as

$$\delta_m = \gamma_0 \delta^s + \gamma_1 \delta^{s+1} + \gamma_2 \delta^{s+2} + O(\delta^{s+3}) \tag{3}$$

It is quite reasonable to use y_m as the starting guess for next iteration. If we convert the expression (2) to rational polynomials $R_{m-l,l}(z0)$, $l = 1, 2, \ldots, m$, new iterative formulas for the evaluation of the functions are obtained.

As an example of the above mentioned method it is possible to consider the evaluation of inverse error function $\mathrm{inverf}(x)$ on the interval $(-1, 1)$. All the analytical transformations are done using computer algebra system Maple V Release 5 [1].

The error function is defined by

$$\mathrm{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt \tag{4}$$

for $x \in R$. As the integrand is an even function, erf is an odd function; and so inverf is an odd function. Thus, it is sufficient to develop an algorithm for its evaluation on the positive half of its domain $[0, 1)$.

There are no obvious variants for the identity (1) which satisfies inverf except the inverse of erf definition. The scale factor will simplify the form of the produced method. Thus, we can take

$$z_0 = \frac{\sqrt{\pi}}{2}(x - \mathrm{erf}(y_0))e^{y_0^2} \tag{5}$$

and $x = \mathrm{erf}(y_0) + \frac{2}{\sqrt{\pi}} z_0 e^{-y_0^2}$, and assuming z_0 is sufficiently small, we get

$$y = \mathrm{inverf}(x) = \sum_{k=0}^{\infty} \mathrm{inverf}^{(k)}(\mathrm{erf}(y_0)) \frac{(\frac{2}{\sqrt{\pi}} z_0 e^{-y_0^2})^k}{k!}. \tag{6}$$

The results for some iterative procedures of orders 2 and 3 and corresponding absolute errors which were prepared in Maple program are the following

$$m = 1,\ l = 0,\ y = y0 + z0$$

$$\Delta_1 = -y\,\Delta^2 + (-\frac{2}{3}y^2 - \frac{2}{3})\,\Delta^3 + O(\Delta^4)$$

$$m = 1,\ l = 1,\ y = y0 + \frac{y0\ z0}{y0 - z0}$$

$$\Delta_1 = -\frac{y^2 - 1}{y}\Delta^2 - \frac{2}{3}\frac{y^4 - 2y^2 + 3}{y^2}\Delta^3 + O(\Delta^4)$$

$$m = 2,\ l = 0,\ y = y0 + z0 + y0\ z0^2$$

$$\Delta_2 = (\frac{4}{3}y^2 + \frac{1}{3})\,\Delta^3 + \frac{1}{2}(4y^2 + 5)\,y\,\Delta^4 + O(\Delta^5)$$

$$m = 2,\ l = 1,\ y = y0 - \frac{z0}{-1 + y0\ z0}$$

$$\Delta_2 = (\frac{1}{3}y^2 + \frac{1}{3})\,\Delta^3 + \frac{1}{2}\,y\,\Delta^4 + O(\Delta^5)$$

$$m = 2,\ l = 2,\ y = y0 - \frac{y0\ z0\,((-1 + y0^2)\,z0 + y0)}{(-1 + y0^2)\,z0^2 + y0\ z0 - y0^2}$$

$$\Delta_2 = \frac{1}{3}\frac{4y^4 - 5y^2 + 3}{y^2}\Delta^3 + \frac{1}{2}\frac{4y^6 - 5y^4 + 8y^2 - 6}{y^3}\Delta^4 + O(\Delta^5)$$

Among all the above mentioned iteration procedures it is needed to choose the most appropriate for evaluation of inverse error function. First we must reject all procedures which have a multiplier with function in the denominator of the first term of error expression. Using these expressions makes the corresponding iterative procedures disconvergent at the point $x = 0$. It is important to use such iterative procedure which will lead to the error 1e-8 by one iteration. The procedure $(m = 1, l = 0)$ must be rejected because the precision 1e-4 for the starting guess is hardly to achieve near the point $x = 1$. The most appropriate is procedure $(m = 2, l = 1)$; it is not complicated and its error is the smallest. One more advantage of this iterative algorithm is that the second term in its error expression is small.

3 Selecting the Starting Guess

For the procedure selected above we would like to construct such starting guess that the error 1e-8 is achieved by one iteration. This guess will be built for inverse error function as balanced approximation by Chebyshev spline [2].

For $\mathrm{inverf}(x)$ as for odd function the optimal approximation is the Chebyshev spline with links

$$V_{k,l}(x) = x \sum_{i=0}^{k} a_i x^{2i} / \sum_{i=0}^{l} b_i x^{2i}. \tag{7}$$

It is better to use the best Chebyshev weighted approximation with weight function $w(x) = 1/\sqrt{y^2 + 1}$ on each subinterval. Then the error expression for the selected method will be

$$\delta_{j+1} = \frac{1}{3}\delta_j^3 + \frac{1}{2}\frac{y}{(y^2 + 1)^{(\frac{3}{2})}}\delta_j^4 - \frac{1}{15}\frac{2y^4 + 4y^2 - 1}{(y^2 + 1)^2}\delta_j^5 + O(\delta_j^6) \tag{8}$$

and error curve after one iteration is to have equal oscillations. To achieve $\delta_1 < 10^{-8}$ the starting error δ_0 must be $\delta_0 < (3e-8)^{(1/3)} = 0.0031072325$.

Balanced weighted rational approximation by Chebyshev spline with links $V_{2,1}(x)$ of inverse error function for $x \in [0, 0.99979]$ with error $\delta_0 = 0.0031$ is

$$inverf_{01} = \begin{cases} 0, \\ \quad x < 0; \\ \dfrac{(-.95493118 + (.53160534 + .23343441\,x^2)\,x^2)\,x}{-1.0977154 + x^2}, \\ \quad x \le .97314979; \\ \dfrac{(1.6200516 + (-4.9295187 + 3.2890636\,x^2)\,x^2)\,x}{-1.0083317 + x^2}, \\ \quad x \le .99767065; \\ \dfrac{(29.849915 + (-61.896833 + 32.044810\,x^2)\,x^2)\,x}{-1.0007339 + x^2}, \\ \quad x \le .99978842; \\ 0, \\ \quad .99978842 < x. \end{cases} \tag{9}$$

For $x > 0.99979$ we can use the starting approximation from [3]

$$inverf_{02} = [-\ln(1-x^2) - \ln(\frac{\sqrt{\pi}}{2}\sqrt{-\ln(1-x^2)})]^{\frac{1}{2}}. \tag{10}$$

Thus, for evaluation of inverse error function we can use the selected iterative procedure of order 3 with starting approximation (9) - (10). The weight error on the interval $[0, 1)$ after one iteration is not larger then 1e-8.

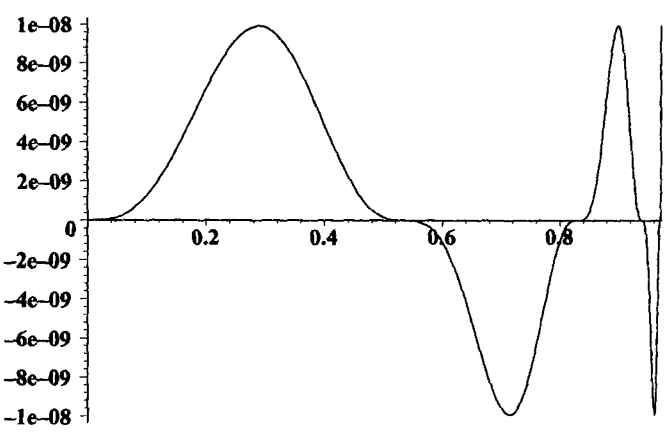

Figure 1. The error curve after one iteration, $x \in [0, 0.97]$

So the use of the presented method makes it possible to evaluate the inverse error function on the interval $x \in (-1, 1)$, that is not available in many computer systems. The proposed evaluation procedure for inverse error function is in several times faster than known ones [3, 5].

References

[1] Monagan M. B., Geddes K. O., Heal K.M. et al., *Maple V. Programming Guide*, Springer-Verlag, New York, 1998.

[2] Popov B.A., *Balanced Approximation by Splines*, Naukova Dumka, Kyiv, 1989. [Russian]

[3] Popov B.A., Hare D. E. G., *Using Computer Algebra to Construct Function Evaluation Methods*, Maple Tech., vol. **3**, No. 3, 1996, 18 - 23.

[4] Popov B.A., Tesler G. S., *Function Approximations for Technical Applications*, Naukova Dumka, Kyiv, 1980. [Russian]

[5] Strecok A.J., *On the Calculation of the Inverse of the Error Function*, Math. Comp., **22**, 1968, 144-158.

Parallel computation of Boolean Gröbner bases

Yosuke Sato
Dept.of Computer Science Ritsumeikan University
1-1-1 Nojihigashi,Kusatsu,Shiga,525-8577, Japan
ysato@theory.cs.ritsumei.ac.jp

In polynomial rings over commutative Von Neumann regular rings, we can construct Gröbner bases using special kinds of monomial reductions. This method is implemented as a Gröbner bases computation algorithm in "SET CONSTRAINT SOLVER Version 1.0" ([Sa 97, Sb 97]).
Meanwhile any commutative Von Neumann regular ring is known to be isomorphic to a subring of a direct product of fields ([SW 75]), and Gröbner bases of its polynomial rings can be characterized as follows.

Theorem
Let F be a finite set of polynomials over a commutative Von Neumann regular ring R which is a subring of a direct product $\prod_{i \in S} K_i$ of fields K_i $i \in S$. For a set G of boolean closed polynomials, the following two conditions are equivalent.

- G is a reduced Gröbner basis of an ideal (F).
- G_i is a reduced Gröbner basis of an ideal (F_i) for each element i of S.

For a polynomial h over R, h_i denotes a polynomial over K_i given from h by replacing every coefficient r of h with its i'th coordinate. For a set H of polynomials over R, H_i denotes $\{h_i | h \in H\}$.
(See Theorem 2.3 of [W 89] or Theorem 3.5 of [S 98] for more details.)

By this theorem, we can also construct a Gröbner basis G by computing Gröbner basis G_i for each coordinate i independently. When the structures of direct products are simple, the latter method seems more efficient, especially under the environment we can use parallel computation.

We implemented this parallel computation method as a boolean Gröbner bases computation algorithm of "SET CONSTRAINT SOLVER Version 2.0" ([S 99]). Boolean Gröbner bases called there are Gröbner bases of ideals of a polynomial ring over a boolean ring that essentially consists of a power set of a certain finite set. When the cardinality of this finite set is n, the boolean ring is isomorphic to the direct products $GL(2)^{n+1}$ of the Galois field $GL(2)$. So there need $n+1$ independent computations of Gröbner bases of polynomial rings over $GL(2)$.

The whole program is written in KLIC which is a parallel logic programming language developed at ICOT(Institute for new generation computer technology) and released as a free software ([KLIC]). Using this program, we had several parallel computation experiments. Some of their data are given in the tables below. The number of *variables* is the number of indeterminates of a polynomial ring, which generally affects the size of the Gröbner basis computation. The number of *elements* is the cardinality of the above mentioned finite set, which determines how many independent computations we need. The data are of three kinds of computations, *sequential* for sequential computations of the new algorithm, *parallel* for parallel computations of the new algorithm and *old algorithm* for computations of the old algorithm. The computation time is measured by minutes:seconds. The computation memory is measured by kilobytes. The process number is the number of processes running through PVM([PVM]). The CPU number is the number how many CPU's are actually used. We also give the number of S-polynomials and boolean closures created through the computations. For parallel computations, the number of S-polynomials is the total sum of the numbers of S-polynomials created through each independent computation. The computation of *parallel*† is a parallel computation with only one CPU, although many processes are running through PVM. For this computation, the computation time given in the table is the maximum of the computation time of all processes. At the computation of the old algorithm in Example 3, the memory was exceeded.
We used 7 computers(PentiumII 400MHZ x 4, PentiumII 333MHZ x 3). Each of them has 512 Megabytes memory. The operating system is FreeBSD 3.1.
Through the experiments, we found;

(1) New method is much faster than the old one even when the computation is sequential.

(2) New method needs less memory than the old one.

(3) Under the actual parallel computation, we can get reasonable speed up.

In the poster session, we would like to introduce our new parallel computation method for the construction of boolean Gröbner bases. Our plan is as follows.

1. Brief introduction of theoretical background.
We give a short sketch of our Gröbner bases including the above theorem and related results.

2. Brief introduction of KLIC
We give a short introduction of KLIC.

3. Demonstration
We give a demonstration of parallel computations of our program using a note personal computer where PVM is running.

Example 1(14 variables 10 elements)						
	s-polynomials	boolean closures	time	memory	process	CPU
seq.	4769	0	0:24	7116	1	1
par.†	4769	0	0:05	7452	11	1
old	3848	495	2:08	25576	1	1

Example 2(20 variables 15 elements)						
	s-polynomials	boolean closures	time	memory	process	CPU
seq.	54280	0	10:45	7128	1	1
par.†	54280	0	1:16	6984	16	1
par.	54280	0	5:10	6988	3	3
par.	54280	0	3:03	6984	7	7
old.	31313	2126	35:16	50216	1	1

Example 3(25 variables 20 elements)						
	s-polynomials	boolean closures	time	memory	process	CPU
seq.	302973	0	698:59	155000	1	1
par.	302973	0	217:03	99800	7	7
old	-	-	∞	∞	1	1

Here seq. stands for sequential, par. – for parallel and old – for old algorithm.

References

[KLIC] KLIC Version 3.002.
http://www.icot.or.jp/AITEC/COLUMN/KLIC/klic.html

[PVM] Parallel Virtual Machine
http://www.epm.ornl.gov/pvm/

[S 96] Sato, Y. (1996). Application of Groebner basis in constraint of non-numerical domains. presented in The 2nd IMACS Conference on Applications of Computer Algebra.

[Sa 97] Sato, Y. (1997). SET CONSTRAINT SOLVER Version 1.0.
http://www.icot.or.jp/AITEC/FGCS/funding/itaku-H8-index-E.html

[Sb 97] Sato, Y. (1997). Set Constraint Solver - Groebner bases for non-numerical domains -. International Symposium on Symbolic and Algebraic Computation(ISSAC 97), Poster Abstracts pp 13-14.

[S 98] Sato, Y. (1998). A new type of canonical Gröbner bases in polynomial rings over Von Neumann regular rings. International Symposium on Symbolic and Algebraic Computation(ISSAC 98), Proceedings pp 317-321.

[S 99] Sato, Y. (1999). SET CONSTRAINT SOLVER Version 2.0.
http://www.icot.or.jp/AITEC/FGCS/funding/itaku-H10-index-E.html

[SW 75] Saracino, D. and Weispfenning, V. (1975). On algebraic curves over commutative regular rings, Model

Theory and Algebra, a memorial tribute to A.Robinson, Springer LNM Vol 498, 307–387.

[W 89] Weispfenning, V. (1989). Gröbner bases in polynomial ideals over commutative regular rings, EUROCAL '87, J.H. Davenport Ed., Springer LNCS Vol 378, 336–347.

Algebraic invariants of graphs: a computer aided study

Nicolas M. Thiéry
Laboratoire de Mathématiques Discrètes (EA619),
Univ Lyon I, FRANCE
`Nicolas.Thiery@jonas.univ-lyon1.fr`

Let $\mathbb{K}$ be a field of characteristic 0 and $\{x_{\{1,2\}}, \dots, x_{\{n-1,n\}}\}$ be a set of $\binom{n}{2}$ variables indexed by the pairs $\{i,j\}$ of $\{1,\dots,n\}$. The symmetric group $\mathfrak{S}_n$ acts naturally on these variables by $\sigma.x_{\{i,j\}} := x_{\{\sigma(i),\sigma(j)\}}$. Finally, let $\mathbb{K}[x_{\{i,j\}}]$ be the ring of polynomials in the $x_{\{i,j\}}$. We study the subring $\mathfrak{I}_n := \mathbb{K}[x_{\{i,j\}}]^{\mathfrak{S}_n}$ of invariant polynomials. Our motivation comes from applications to the problems of isomorphism and reconstruction of weighted graphs. Our primary goal is to construct complete systems of invariants (systems which separates valuated graphs up to isomorphy), and in particular minimal generating systems.

For $n = 4$, such a minimal generating system has been constructed by hand by Aslaksen, Chan et Gulliksen [ACG96], and can now be computed in a few seconds by the usual invariant theory softwares (e.g. Kemper's packages in `Maple` and `Magma` [Kem98b]). However, for $n \geq 5$, those softwares are unable to compute minimal generating sets, even partial. We wrote `PerMuVAR`, a library of invariant theory routines for `MuPAD`, using the usual algorithms [Stu93, Kem98b], but specialized for permutation groups. This allowed us to go a little step further: for $n = 5$, we computed a generating system containing a thousand of polynomials of degree ≤ 22, and a minimal system, containing 57 polynomials of degree ≤ 9, which is likely to be generating.

Another step further would be to refine the majoration $\beta(n) \leq \binom{\binom{n}{2}}{2}$ of the bound $\beta(n)$ on the degree of the polynomials in a minimal generating set. One way is to construct a system of parameters with low degrees. The study of the Hilbert series of the invariant ring up to $n = 19$, combined with the conjecture of Mallows and Sloane, suggest that there exists such a system consisting of homogeneous polynomials of degrees $1, 2, \dots, n, 2, 3, \dots, \binom{n-1}{2}$. This would give $\beta(n) \leq \binom{\binom{n-1}{2}}{2}$. We propose a natural construction for a system of parameters with such degrees, but we could only check it for $n \leq 5$. Indeed, the usual test relies on a Gröbner basis computation, which is intractable for $n \geq 6$. Different computations in small cases suggest a much better majoration: $\beta(n) \leq \binom{n}{2} - 1$.

The Hilbert Series can also be used to show that various systems do not generate the invariant ring. This led us, in a related invariant ring, to construct a counter-example to a lemma of Grigoriev [Gri79, Lemma I]. Finally, we show that the field of invariant fractions is generated by the elementary symmetric polynomials together with a very simple polynomial of degree 2. We note that they do not form a complete system of invariant.

References

[ACG96] Helmer Aslaksen, Shih-Ping Chan, and Tor Gulliksen. Invariants of S_4 and the shape of sets of vectors. *Appl. Algebra Engrg. Comm. Comput.*, 7(1):53–57, 1996.

[GS84] A. M. Garsia and D. Stanton. Group actions of Stanley - Reisner rings and invariants of permutation groups. *Adv. in Math.*, 51(2):107–201, 1984.

[Gri79] D. Ju. Grigoriev. Two reductions of the graph isomorphism to problems for polynomials. *Zap. Nauchn. Sem. Leningrad. Otdel. Mat. Inst. Steklov. (LOMI)*, 88:56–61, 237–238, 1979. Studies in constructive mathematics and mathematical logic, VIII.

[Kem98b] Gregor Kemper. Computational invariant theory. *Queen's Papers in Pure and Applied Math.*, February 1998.

[Stu93] Bernd Sturmfels. *Algorithms in invariant theory.* Springer-Verlag, Vienna, 1993.

[Gvb95] Manfred Gvbel. Computing Bases for Rings of Permutation-Invariant Polynomials. *JSC*, 19(4): 285-291, 1995.

Weyl closure of a D-ideal

Harrison Tsai
Department of Mathematics
University of California at Berkeley
Berkeley, CA 94720
`htsai@math.berkeley.edu`

Let D_n denote the n-th Weyl algebra $k\langle x_1, \dots, x_n, \partial_1, \dots, \partial_n\rangle$ over an algebraically closed field k of characteristic 0, and let R denote the ring of differential operators $k(x_1, \dots, x_n)\langle\partial_1, \dots, \partial_n\rangle$. We study the following operation.

Definition 0.0.1. *Let $I \subset D_n$ be a left ideal. The Weyl closure of I is the ideal*

$$Cl(I) = R \cdot I \cap D_n$$

From the analytic perspective with $k = \mathbf{C}$, the left ideal I corresponds to a system of linear partial differential equations. If I has finite rank, i.e. $dim_{k(x)} R/R \cdot I < \infty$, and if U is a simply connected domain of $\mathbf{C}^n$ away from the singular locus of I, then the Cauchy-Kovalevskii-Kashiwara theorem implies that I has a finite dimensional vector space $Sol(I; U)$ of holomorphic solutions on U. The Weyl closure of I is then the set of all operators in D_n which annihilate the common solution space $Sol(I;U)$, much in the same way that the radical of a commutative ideal J is the set of all functions which vanish on the common zeroes of J.

In our forthcoming article "Weyl closure of a linear differential operator", we give an algorithm to compute the Weyl closure when $n = 1$. This algorithm has been implemented in MAPLE. We also provide some applications, namely the algorithmic computation of Jordan-Hölder series for holonomic D_1-modules, a description of the possible initial ideals of left D_1-ideals with respect to the order filtration, and a description of the isomorphism classes of left D_1-ideals.

Our poster will summarize the above work and will also report on more recent progress. Currently, we have an algorithm to compute the Weyl closure for general n. The algorithm is largely based on the extensive work of Oaku and Takayama concerning computation in D_n-modules. Our plan for the future is to make an implementation in Macaulay2 and to explore applications such as those mentioned in the $n = 1$ case.

Deciding Linear-Exponential Problems

Volker Weispfenning
Fakultät für Mathematik und Informatik
Universität Passau
D-94030 Passau, Germany
weispfen@alice.fmi.uni-passau.de

1 Background

Tarski's theorem [Tar48] about the decidability of the first-order theory of the ordered fields $\mathbb{R}$ of real numbers is one of the most fundamental results of algorithmic real algebra and the starting point of the computer algebra of polynomial questions about real numbers. It raises the question whether this result can be extended to a language including in addition the real exponential function e^x. Despite intensive research and remarkable partial results (compare [vdD82, vdD86]), this problem remained open until 1996, when Macintyre and Wilkie published a conditionally positive solution [MW96]. It relies on the yet unproven Schanuel's conjecture, that asserts roughly speaking that there are no unexpected algebraic relations between finitely many reals and there exponentials. The ingenious result of Macintyre and Wilkie combines deep methods and results of model theory, the analysis of Pfaffian functions and algebra. The resulting decision method is very far from an explicit formulation that may be turned into an implementable algorithm. Partial subproblems – in particular concerning the solvability of real exponential equation systems – have been studied more in the spirit of computer algebra by Richardson (compare [Ric95, Ric98]); but here again Schanuel's conjecture comes up as a hypothesis.
Here we consider a small but highly non-trivial fragment of the decision problem of the reals as ordered exponential field that is concerned with exponential-linear problems. We give an unconditionally positive solution of the decision problems for this fragment. Moreover we extend this decision procedure to the case, where we allow the integer-part function – and hence integer variables – in our language. This extended result is not covered by the Macintyre-Wilkie theorem.
Our decision procedure is quite explicit and uses only elementary analysis, real and mixed real-integer quantifier elimination [LW93, Wei99], and Lindemann's theorem on the transcendence of e^c for real algebraic c. An implementation of the decision procedure is planned in an extension of the REDLOG- package of REDUCE (see `http://www.fmi.uni-passau.de/~redlog/`)

2 Main Result

We consider a formal language for the domain of real numbers.

Terms are obtained from the constants 0, 1, variables $x, x_1, x_2, \ldots$ and the exponential expression e^x for the specified variable x by means of addition, scalar multiplication by rational constants, and the formation of integer parts [ ]. For example

$$[[3x_1 - 1] + e^x] + \frac{2}{7}x_2 - 5e^x$$

is a term.

Atomic Formulas are of the form $s = t$ or $s < t$, where s, t are terms.

Formulas are obtained from atomic formulas by composition with the boolean operators $\wedge, \vee, \neg$ and quantification over the variables $x, x_1, x_2, \ldots$.

Normal Formulas are formulas without free variables, where the variable x is quantified outermost. Moreover, if the integer-part operation occurs in a normal formula then the quantifier with respect to x has to be bounded by rational constants. For example
$\exists x \forall x_1$
$(-20 < x \leq 100 \wedge [2x_1 + e^x] - (x_1 + [3x_1 - 5]) > 0 \wedge x_1 \leq 5e^x)$
is a normal formula.

Theorem

There is an algorithm that decides upon input of a normal formula whether or not this formula holds in the domain of real numbers.

Proof Sketch. The decision method first eliminated all quantifiers refering to the linear variables x_i by the method in [Wei99]. This leaves us with a univariate linear-exponential problem. Next any (nested) occurrences of the integer part function are removed by a case distinction over finitely many integer constants; here we use the boundedness of the quantifier with respect to the variable x.
Finally the remaining quantifier referring to the linear-exponential variables x is eliminated by a kind of virtual substitution of finitely many test points à la [Wei97], taking into account the convexity of functions described by univariate

terms without integer-parts, and elementary analysis. Lindemann's theorem is used to decide the equations and inequalities between constant expressions that arise in this procedure.

3 A Simple Example

Consider the normal formula

$$\exists x(cx + \frac{1}{2} > e^x \ \wedge \ -x < e^x)$$

for a rational constant c. Then our decision procedure will say that this formula holds in the reals iff

$$c \leq -1 \ \vee \ (c > -1 \ \wedge \frac{1}{2(c+1)} > e^{\frac{-1}{2(c+1)}})$$

By Lindemann's theorem the equation

$$\frac{1}{2(c+1)} = e^{\frac{-1}{2(c+1)}}$$

is impossible for rational c. So the inequalities above can be decided by computing enough of the expansion of

$$e^{\frac{-1}{2(c+1)}}.$$

References

[LW93] Rüdiger Loos and Volker Weispfenning. Applying linear quantifier elimination. *The Computer Journal*, 36(5):450–462, 1993. Special issue on computational quantifier elimination.

[MW96] A. Macintyre and A.J. Wilkie. On the decidability of the real exponential field. In *Kreiseliana: About and around Georg Kreisel*, pages 441–467. A.K. Peters, 1996.

[Ric95] Daniel Richardson. An ISSAC'95 tutorial: Algorithmic methods for finding real solution of systems involving exponential and other elementary functions. Preprint, July 1995.

[Ric98] Daniel Richardson. Local theories and cylindrical decompositon. In B.F. Caviness and J.R. Johnson, editors, *Quantifier Elimination and Cylindrical Algebraic Decomposition*, Texts and Monographs in Symbolic Computation, pages 351–364. Springer, Wien, New York, 1998.

[Tar48] Alfred Tarski. A decision method for elementary algebra and geometry. Technical report, University of California, 1948. Second edn., rev. 1951.

[vdD82] L. van den Dries. Remarks on Tarski's problem concerning (r,+,.,exp). In G. Longi, G. Longo, and A. Marcja, editors, *Logic Colloquium*, pages 97–121, Amsterdam, u.a., 1982. North-Holland.

[vdD86] L. van den Dries. A generalization of the Tarski-Seidenberg theorem and some nondefinability results. *Bulletin of AMS*, 15:189–193, 1986.

[Wei97] Volker Weispfenning. Quantifier elimination for real algebra—the quadratic case and beyond. *Applicable Algebra in Engineering Communication and Computing*, 8(2):85–101, February 1997.

[Wei99] Volker Weispfenning. Mixed real-integer linear quantifier elimination. In S. Dooley, editor, *ISSAC'99*. ACM-Press, 1999. to appear.