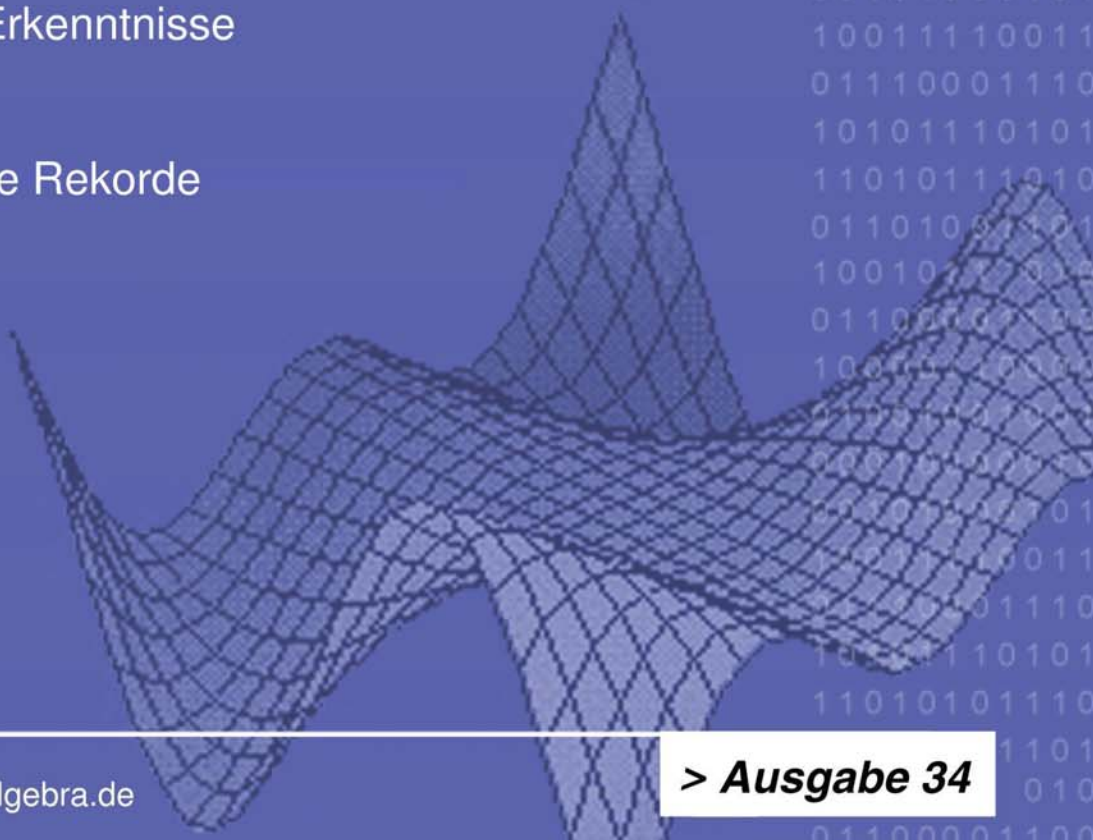
März 2004

Computeralgebra Rundbrief

GI_DMV_GAMM

- Fachgruppe – neue Tagungen
- Feli–X, GiNaC und gTybalt – neue Systeme
- Invariantentheorie – neue Entwicklungen
- Umfrage – neue Erkenntnisse
- Primzahlen – neue Rekorde



www.fachgruppe–computeralgebra.de

> Ausgabe 34

Computeralgebra-Rundbrief

Nr. 34 März 2004

Inhalt

Impressum

Der Computeralgebra-Rundbrief wird herausgegeben von der Fachgruppe Computeralgebra der GI, DMV und GAMM (verantwortlicher Redakteur: Dr. Markus Wessler, Universität Kassel, Fachbereich Mathematik/Informatik, Heinrich-Plett-Str. 40, 34132 Kassel, Telefon: 0561-8044192, Telefax: 0561-8044646, wessler@mathematik.uni-kassel.de).

Der Computeralgebra-Rundbrief erscheint halbjährlich, Redaktionsschluss 28.02 und 30.09. ISSN 0933-5994. Mitglieder der Fachgruppe Computeralgebra erhalten je ein Exemplar dieses Rundbriefs im Rahmen ihrer Mitgliedschaft. Fachgruppe Computeralgebra im Internet:
http://www.fachgruppe-computeralgebra.de.

Konferenzankündigungen, Mitteilungen, einzurichtende Links, Manuskripte und Anzeigenwünsche bitte an den verantwortlichen Redakteur.

Die Geschäftsstellen der drei Trägergesellschaften:

GI (Gesellschaft für Informatik e.V.)
Wissenschaftszentrum
Ahrstr. 45
53175 Bonn
Telefon 0228-302-145
Telefax 0228-302-167
gs@gi-ev.de
http://www.gi-ev.de

DMV (Deutsche Mathematiker-Vereinigung e.V.)
Mohrenstraße 39
10117 Berlin
Telefon 030-20377-306
Telefax 030-20377-307
dmv@wias-berlin.de
http://www.mathematik.uni-bielefeld.de/DMV/

GAMM (Gesellschaft für Angewandte Mathematik und Mechanik e.V.)
Technische Universität Dresden
Institut für Festkörpermechanik
01062 Dresden
Telefon 0351-463-33448
Telefax 0351-463-37061
GAMM@mailbox.tu-dresden.de
http://www.gamm-ev.de

Mitteilungen der Sprecher

Liebe Mitglieder der Fachgruppe Computeralgebra,

die Fachgruppenleitung traf sich am 20. Februar 2004 zu ihrer Frühjahrsitzung an der Universität Dortmund. Im Mittelpunkt standen diesmal (neben der Vorbereitung des vorliegenden Rundbriefs) die Tagungen.

Auf den jährlichen DMV-Tagungen ist die Fachgruppe fast regelmäßig mit einer eigenen Sektion vertreten. In diesem Jahr findet die DMV-Tagung in der Woche vom 13. bis 18. September 2004 in Heidelberg statt (`http://dmv2004.uni-hd.de`*). Der Hauptvortrag von M. van der Put zum Thema Ordinary Differential Equations and Groups, welcher am 16. September 2004 von 9–10 Uhr stattfinden wird, war von der Fachgruppe vorgeschlagen worden. Zum Vortragenden sei auf das Buch Galois Theory of Linear Differential Equations verwiesen, welches von Julia Hartmann im CA-Rundbrief 33 auf S. 22 referiert worden war. Ferner gibt es diesmal wieder eine eigene Sektion Computeralgebra, welche von Gregor Kemper (TU München) und Bettina Eick (TU Braunschweig) organisiert wird.*

Die wissenschaftliche Tagung, welche die Fachgruppe vom 15.–17. Mai 2003 in Kassel veranstaltete (`http://www.mathematik.uni-kassel.de/compmath/ca.htm`*), findet ihre Fortsetzung am 16.–18. Juni 2005 und wird wieder in Kassel durchgeführt werden. Diesen Termin sollten Sie sich für Ihre Langzeitplanung bereits jetzt notieren, weitere Details zu dieser Tagung werden dann im Oktoberheft mitgeteilt werden.*

Wir hatten im letzten Rundbrief die Publikation der Resultate unserer Umfrage zum Einsatz von Computeralgebrasystemen in der universitären Lehre, insbesondere auch in Lehramtsstudiengängen, angekündigt. Diese Ergebnisse finden Sie auf S. 23 des vorliegenden Rundbriefs.

Diese Umfrage war ja von den Teilnehmern der Tagung Computeralgebra in Lehre, Ausbildung und Weiterbildung III, 2002, welche in Kloster Schöntal stattfand, initiiert worden. Nun ist es Zeit für die Folgetagung, die diesmal unter dem Thema Computeralgebra in Lehre, Ausbildung und Weiterbildung IV: Konsequenzen aus PISA steht und die vom 13.-16.04.2004 in Haus Schönenberg in Ellwangen stattfindet. Details zu dieser Tagung, insbesondere das Tagungsprogramm, finden Sie im nächsten Abschnitt bzw. auf der Internetseite `http://www.fachgruppe-computeralgebra.de/CLAW/Schoenenberg2004`*.*

Wir konnten wieder einige interessante Berichte für den Rundbrief einwerben: Günter M. Ziegler gibt einen aktuellen Bericht über die Primzahlrekordjagd (S. 11). Dieser Artikel erschien kürzlich bereits in den DMV-Mitteilungen. Herzlichen Dank beim Herausgeber der DMV-Mitteilungen Folkmar Bornemann sowie beim Autor für die Erlaubnis, den Artikel im Rundbrief abzudrucken. Gregor Kemper berichtet über neue Entwicklungen in der algorithmischen Invariantentheorie (S. 7). Schließlich gibt Reinhard Oldenburg einen Bericht über ein neues System Feli-X, welches Computeralgebra und dynamische Geometrie verbindet (S. 17). Ein Novum ist ferner das Interview mit dem Mathematica-Entwickler Tom Wickham-Jones, welches Ulrich Kortenkamp geführt hat (S. 19).

Die Amtszeit der derzeitigen Fachgruppenleitung läuft im Frühjahr 2005 ab. Daher werden mit dem nächsten Rundbrief im Oktober 2004 wieder die Wahlunterlagen zur Neuwahl der Fachgruppenleitung verschickt. Kandidatenvorschläge sind herzlich willkommen, können von allen Mitgliedern eingereicht werden und werden von jedem Mitglied der Fachgruppenleitung per e-mail entgegengenommen. Auch weitere Anregungen aus unserem Leserkreis sind jederzeit willkommen.

Wir hoffen, Sie mit dem vorliegenden Heft wieder gut zu informieren.

Wolfram Koepf *H. Michael Möller*

Tagungen der Fachgruppe

Computeralgebra in Lehre, Ausbildung und Weiterbildung IV: Konsequenzen aus PISA
13.-16.04.2004, Haus Schönenberg bei Ellwangen
In Fortführung der Tagungstradition von Thurnau (1998, 2000) und Schöntal (2002) wird von der Fachgruppe Computeralgebra in der Woche nach Ostern eine Tagung zum Thema *Konsequenzen aus PISA* organisiert. Die Tagung findet im Haus Schönenberg bei Ellwangen statt (`http://www.haus-schoenenberg.de`). Sie beginnt am Osterdienstag, dem **13. April 2004 um 14:30 Uhr** und endet am Freitag, dem 16. April 2004 mit dem gemeinsamen Mittagessen. Es sind ausführliche Diskussionen zum Thema „Konsequenzen aus PISA“ geplant. Der Tagungsausflug besteht aus einer gemeinsamen Fahrt nach Rothenburg ob der Tauber. Alle weiteren Details, insbesondere Anmeldeformular sowie Tagungsprogramm, finden Sie auf der Internetseite der Tagung `http://www.fachgruppe-computeralgebra.de/CLAW/Schoenenberg2004`.

Untersuchungen wie PISA haben es gezeigt: Es ist was „faul“ mit den mathematischen Fähigkeiten der deutschen Schüler. Problemsolving ein Fremdwort in deutschem Mathematikunterricht? Wie reagieren die Hochschulen und die Studienseminare auf diese neue Herausforderung? Ist der Einsatz von Computeralgebrasystemen das Werkzeug, das die Gedanken der deutschen Schüler frei machen kann? Oder verhindert das in vielen Bundesländern nach PISA neu eingeführte Zentralabitur den Einsatz von Computeralgebra eher? Oder müssen wir uns nun doch wieder auf mathematische Fertigkeiten konzentrieren? Viele Fragen, auf die Lehrer und Hochschullehrer auf dieser Tagung gemeinsam nach Antworten suchen.

Folgende Vorträge sind geplant:

- Burkhard Alpers: Die mathematische Mikrowelt „Formel 1“ – Lernangebot und Nutzen in der Schüler-Ing.-Akademie (SIA)
- Manfred Bauch: Bildungsstandards und dynamische Mathematik
- Klaus Dürrschnabel: Mathematik an der Schnittstelle Schule-Hochschule – Aktivitäten in Baden-Württemberg
- Hans-Gert Gräbe: Variablenbegriff und Funktionsbegriff im symbolischen Rechnen
- Wolfgang Henn: CAS in der Lehrerausbildung: Wunsch und Wirklichkeit (Einführung in die Podiumsdiskussion)
- Heiko Knechtel: Mathematikunterricht mit CAS – Schnittstellenvereinbarung und Zentralabitur in Niedersachsen
- Robert Kragler: Animation mathematischer Sachverhalte mit Mathematica
- Hubert Langlotz, Wolfgang Moldenhauer, Wilfried Zappe: 5 Jahre CAS in Thüringen – Erfahrungen und Ausblick
- Eberhard Lehmann: Lineare Gleichungssysteme auf Bestellung – Bericht über eine Demonstrationsstunde auf einer Lehrerfortbildung
- Reinhard Oldenburg: CAS-Kompetenz – Was ist das?
- Heinz Schumann: „Einfache“ algebraische Kurven in dynamischer Behandlung
- Karel Tschacher: Wird mit dem Classpad 300 alles besser?
- Wilhelm Werner: Einfluss von CAS auf die Mathematikausbildung an Fachhochschulen
- Otto Wurnig: Neue Modelle zur Leistungsbeurteilung im CAS-integrierten Mathematikunterricht – Erfahrungen und erste Resultate aus den CA-Projekten in Österreich

Haus Schönenberg

Computeralgebra: 16.-18.06.2005, Kassel

Nach dem großen Erfolg der Computeralgebra-Tagung, welche im Mai 2003 in Kassel stattfand, plant die Fachgruppe, in der Zeit vom 16.-18. Juni 2005 wieder eine derartige Tagung in Kassel durchzuführen. Genaueres zu dieser Tagung wird dann im Oktoberheft mitgeteilt werden.

Themen und Anwendungen der Computeralgebra

Neue Entwicklungen in der algorithmischen Invariantentheorie

Gregor Kemper (München)

kemper@ma.tum.de

Dieser Artikel soll über einige neuere Entwicklungen in der algorithmischen Invariantentheorie berichten, die sich nach Erscheinen des Buchs [4] ergeben haben. In erster Linie soll ein Algorithmus zum Berechnen von Invariantenringen reduktiver Gruppen in positiver Charakteristik vorgestellt werden.

Der bisherige Stand. In der Invariantentheorie betrachtet man die folgende Situation: G ist eine lineare algebraische Gruppe über einem Körper K, den wir der Einfachheit halber als algebraisch abgeschlossen voraussetzen wollen. G operiert auf einer affinen K-Varietät X durch einen Morphismus $G \times X \to X$. Wenn wir den Ring der regulären Funktionen auf X mit $K[X]$ bezeichnen, so ist der Invariantenring gegeben durch

$$K[X]^G := \{f \in K[X] \mid f(g(x)) = f(x) \text{ für alle } x \in X, g \in G\}.$$

Ein wichtiger Spezialfall (man könnte fast sagen: der Standardfall) ist der, dass X ein endlich-dimensionaler K-Vektorraum V und die G-Operation linear ist. Klassische Fragestellungen sind:

1. Wann ist $K[X]^G$ endlich erzeugt als K-Algebra?
2. Wie findet man Erzeuger von $K[X]^G$?
3. Welche Punkte von X können durch Invarianten getrennt werden?

Zur (auch nur ansatzweisen) Beantwortung dieser Fragen sollte man verschiedene Klassen von Gruppen betrachten. Als die wichtigste Klasse hat sich hierbei die Klasse der *reduktiven* Gruppen herausgestellt (siehe [4, Abschnitt 2.2]). Es gilt nämlich nach Hilbert und Nagata, dass $K[X]^G$ immer endlich erzeugt ist, falls G reduktiv ist (siehe [4, Abschnitt 2.2]). Umgekehrt konnte Popov [9] zeigen, dass eine Gruppe G, bei der $K[X]^G$ für alle G-Varietäten X endlich erzeugt ist, reduktiv sein muss. Die reduktiven Gruppen sind also die „richtige“ Klasse für das Betreiben von Invariantentheorie (was Invariantentheoretiker allerdings nicht davon abhält, sich auch intensiv mit Invarianten nicht reduktiver Gruppen zu befassen). Wie sieht es aus mit der zweiten Fragestellung nach dem Finden von erzeugenden Invarianten? Hier lohnt es sich zwei Unterklassen der reduktiven Gruppen zu betrachten. Zum einen sind das die *linear reduktiven* Gruppen. Für diese wurde 1999 von Derksen ein Algorithmus zur Konstruktion von erzeugenden Invarianten gefunden (siehe [4, Abschnitt 4.1]). In Charakteristik 0 ist jede reduktive Gruppe linear reduktiv, womit das Problem also in diesem Fall in befriedigender Allgemeinheit gelöst wäre. Als zweite wichtige Unterklasse betrachten wir die endlichen Gruppen, bei denen die Invariantentheorie vor allem im *modularen Fall* (d.h. $|G|$ ist ein Vielfaches der Charakteristik von K) schwierig und damit interessant ist. Hier wurden nach Vorarbeiten von Sturmfels [10] Algorithmen durch den Autor gefunden (siehe [4, Abschnitte 3.3 und 3.5]), die den Fall linearer Operationen (X ein Vektorraum) abdecken.

Soweit in groben Zügen der Stand der Dinge, wie er sich im Buch [4] darstellt. Wir haben also eine schmerzliche Lücke, nämlich das Fehlen eines Algorithmus zum Berechnen von Invarianten reduktiver Gruppen, die nicht linear reduktiv sind (was nur in positiver Charakteristik auftreten kann und auch häufig auftritt). Diese Lücke ist inzwischen für den Fall linearer Operationen

geschlossen worden, und der sich ergebende Algorithmus [6] ist Hauptthema dieses Berichts.

Separierende Invarianten. Die algorithmische Invariantentheorie hat in der letzten Zeit neuen Impetus durch ein stärkeres Heranziehen der dritten Fragestellung nach Trennungseigenschaften von Invarianten erhalten. Genauer gesagt hat es sich als gewinnbringend erwiesen, diese Frage zunächst einmal rein definitorisch zu bearbeiten, statt sie zu beantworten. Wir geben folgende sehr allgemeine Definition.

Definition 1. *X und K seien Mengen, und F sei eine Menge von Funktionen $X \to K$. Eine Teilmenge $S \subseteq F$ heißt* F-separierend, *falls für alle $x, y \in X$ gilt:*

$$\text{Falls } f(x) = f(y) \quad \text{für alle } f \in S, \qquad \text{so auch für alle } f \in F.$$

In dem uns besonders interessierenden Fall ist K wie oben ein algebraisch abgeschlossener Körper und F ein Invariantenring. Ist K ein kommutativer Ring, so bildet die Menge K^X der Funktionen $X \to K$ eine K-Algebra. Dann gilt für jede Teilmenge $S \subseteq K^X$: S ist $K[S]$-separierend, wobei $K[S]$ die von S erzeugte Unteralgebra von K^X bezeichnet. Mit anderen Worten: Jedes Erzeugendensystem einer Teilalgebra A von K^X ist A-separierend, d.h. „separierend“ ist schwächer als „erzeugend“. Die Hauptidee des oben erwähnten Algorithmus zur Berechnung von Erzeugern von Invariantenringen reduktiver Gruppen ist, zunächst ein separierendes System von Invarianten zu konstruieren. Bevor wir uns dem Algorithmus zuwenden, sollen noch einige Fakten genannt werden, die illustrieren, wie vereinfachend die Abschwächung der Betrachtungsweise von „erzeugend“ auf „separierend“ wirkt. Beispielsweise gilt der folgende Satz.

Satz 2. *Es seien X eine Menge, K ein noetherscher kommutativer Ring und $A \subseteq K^X$ eine endlich erzeugte K-Algebra von Funktionen $X \to K$. Dann existiert zu jeder Teilmenge $F \subseteq A$ eine endliche F-separierende Teilmenge $S \subseteq F$.*

In der uns interessierenden Situation ist $A = K[X]$ der Ring der regulären Funktionen auf der Varietät X und $F = K[X]^G$. Der Satz besagt also insbesondere, dass es in jedem Invariantenring ein endliches System separierender Invarianten gibt, und das, obwohl wir wissen, dass nicht jeder Invariantenring endlich erzeugt ist! Der Beweis des Satzes 2 läuft fast exakt wie der des etwas spezielleren Satzes 2.3.15 in [4]. Er ist sehr einfach, aber leider völlig inkonstruktiv und erinnert insofern etwas an den ursprünglichen von Hilbert gegebenen Endlichkeitsbeweis (siehe [4, Theorem 2.2.10]). Ein illustratives Beispiel für den Gegensatz von separierenden und erzeugenden Funktionen mag das folgende sein:

Beispiel 3. Es sei $A = K[x, y]$ die Algebra der Polynomfunktionen auf $X = K^2$ (K ein unendlicher Körper). Dann ist

$$F := K + x \cdot A = K[x, xy, xy^2, xy^3, \ldots]$$

eine nicht endlich erzeugbare Unteralgebra. Man überlegt sich jedoch leicht, dass die Funktionen x und xy eine F-separierende Teilmenge bilden.

Als weitere Kostprobe gehen wir auf die Situation von endlichen Gruppen ein. Hier erhalten wir nun ein völlig konstruktives Resultat. Es sei K ein Integritätsbereich und $A = K[f_1, \ldots, f_n]$ eine endlich erzeugte Unteralgebra von K^X (X wieder irgendeine Menge). Weiter operiere eine endliche Gruppe G durch Algebren-Automorphismen auf A. Wir nehmen zwei Unbestimmte T und U her und bilden das Polynom

$$F(T, U) := \prod_{g \in G} \left(T - \sum_{i=1}^{n} g(f_i) \cdot U^{i-1} \right), \qquad (1)$$

dessen Koeffizienten offenbar im Invariantenring A^G liegen. Nun gilt:

Satz 4. *Die Koeffizienten von $F(T, U)$ bilden eine A^G-separierende Teilmenge.*

Der (einfache) Beweis sei dem Leser überlassen. Dieser Satz ist insbesondere interessant im Fall einer linearen Gruppenoperation. Dann ist $A = K[V] = K[x_1, \ldots, x_n]$ ein Polynomring über einem Körper K, und offenbar sind alle Koeffizienten von $F(T, U)$ homogen vom Grad $\leq |G|$. Wir erhalten also, dass für separierende Invarianten die noethersche Gradschranke (siehe [4, Abschnitt 3.8]) gilt, und zwar unabhängig von der Charakteristik von K. Dies gewinnt an Brisanz, wenn man in Betracht zieht, dass die noethersche Gradschranke für erzeugende Invarianten im modularen Fall in eklatanter Weise scheitert, wie Richman als erster gezeigt hat (siehe [4, Abschnitt 3.9.1]).

Ein weiteres Beispiel für das gute Verhalten von separierenden Invarianten ist die Tatsache, dass der Satz von Weyl über Polarisierung für separierende Invarianten über beliebigen Grundkörpern gilt (und zwar in dem Sinne, dass ein separierendes System durch Polarisierung wieder in ein separierendes übertragen wird), obwohl eben dieser Satz für erzeugende Invarianten in positiver Charakteristik falsch ist. Der Beweis wurde kürzlich von Campbell, dem Autor und Wehlau erbracht. Die Publikation hierzu ist noch in Vorbereitung. Abgesehen von den guten Eigenschaften separierender Invarianten ist es auch unter anwendungsbezogenen Gesichtspunkten sinnvoll, sich mit ihnen zu befassen, denn in vielen beispielsweise geometrischen Anwendungen der Invariantentheorie sind separierende Invarianten in jeder Hinsicht hinreichend.

Bevor wir uns endgültig dem Algorithmus zum Berechnen von Invarianten reduktiver Gruppen zuwenden, sei noch eine Abschweifung über das Berechnen von Invarianten*körpern* endlicher Gruppen gestattet. Es sei

jetzt K irgendein Körper und $L = K(f_1, \ldots, f_n)$ eine endlich erzeugte Körpererweiterung. Weiter sei G eine endliche Gruppe von K-Automorphismen von L. Mit Unbestimmten T und U bilden wir das Polynom $F(T, U)$ wie in (1) und außerdem für $i = 1, \ldots, n$

$$H_i(T) := \prod_{y \in G(f_i)} (T - y)\,,$$

wobei $G(f_i)$ die G-Bahn von f_i bezeichnet.

Satz 5. *Ist M die Menge aller Koeffizienten von $F(T, U)$ und der $H_i(T)$, so gilt $L^G = K(M)$. Hat K die Charakteristik 0, so kann sogar auf die Hinzunahme der Koeffizienten der $H_i(T)$ verzichtet werden.*

Auch der Beweis hierfür (welcher mit elementarer Galoistheorie geführt werden kann) sei dem Leser überlassen. Wendet man Satz 5 auf den linearen Fall an, so erhält man als Folgerung die noethersche Gradschranke für den Invariantenkörper, wieder unabhängig von der Charakteristik. Satz 5 scheint in der Literatur bis jetzt noch nicht aufgetaucht zu sein. Es ist vorstellbar, dass er auch Anwendungen in der endlichen Galoistheorie hat, da ja keinerlei Voraussetzungen über die algebraische Unabhängigkeit der f_i gemacht werden.

Der Algorithmus. Bis jetzt wissen wir nur für endliche Gruppen, wie man separierende Invarianten berechnet. Um dies auch für allgemeinere Gruppen zu können, müssen wir uns dem dritten Problem aus der Einleitung, welche Punkte durch Invarianten trennbar sind, zuwenden. Und auf diese Frage ist die Antwort im Falle reduktiver Gruppen seit langem bekannt. Wir setzen wieder die Standardsituation voraus, dass G eine lineare algebraische Gruppe mit einer Operation auf einer affinen Varietät X durch einen Morphismus ist. Ist G nun reduktiv, so gilt für zwei Punkte $x, y \in X$ (siehe [8, Corollary 3.5.2]):

$$\begin{aligned} f(x) = f(y) \quad &\text{für alle} \quad f \in K[X]^G \\ &\iff \quad \overline{G(x)} \cap \overline{G(y)} \neq \emptyset, \end{aligned} \tag{2}$$

wobei $\overline{G(x)}$ und $\overline{G(y)}$ die Abschlüsse der Bahnen (bezüglich der Zariski-Topologie) bedeuten. Mit anderen Worten: Das offensichtliche topologische Hindernis zur Trennung zweier Punkte (das von der Stetigkeit der Invarianten herrührt) ist in Wirklichkeit das einzige. Wir betrachten nun die Situation einer linearen Operation, also $X = V$ ein endlich-dimensionaler K-Vektorraum. Um den Algorithmus zur Konstruktion von separierenden Invarianten zu verstehen, muss man wissen, dass es relativ leicht ist, zu einem gegebenen Grad d alle homogenen Invarianten in $K[V]^G$ vom Grad d zu berechnen. Man kann dies tun, indem man ein allgemeines Polynom vom Grad d mit unbestimmten Koeffizienten aufstellt und dann, grob gesagt, Invarianzbedingungen fordert. Man erhält dann nach einer Normalformberechnung ein homogenes lineares Gleichungssystem mit den unbestimmten Koeffizienten als Unbekannten (siehe [6, Algorithm 2.7]). Ein etwas schnelleres Verfahren für dieselbe Aufgabe wurde von Bayer [1] angegeben. Damit ist für die Konstruktion separierender Invarianten nur noch erforderlich, einen Verifikationsschritt zu haben, der entscheidet, ob die Konstruktion fertig ist. Unter Benutzung des Kriteriums (2) erhalten wir den folgenden Algorithmus. Dabei nehmen wir an, dass die reduktive Gruppe G (als affine Varietät) durch ein Ideal $I_G \subseteq K[t_1, \ldots, t_m]$ in einem Polynomring gegeben ist, und dass die G-Operation auf $V = K^n$ durch eine Matrix $(a_{i,j})_{i,j=1,\ldots,n} \in K[t_1, \ldots, t_m]^{n \times n}$ definiert ist.

Algorithmus 6 (Separierende Invarianten).

Input: Das Ideal I_G und die Matrix $(a_{i,j})$ wie oben.

Output: Homogene Invarianten $f_1, \ldots, f_k \in K[V]^G$, die eine $K[V]^G$-separierende Teilmenge bilden.

1. Bilde den Polynomring

   $$K[t_1, \ldots, t_m, x_1, \ldots, x_n, y_1, \ldots, y_n] =: K[\underline{t}, \underline{x}, \underline{y}]$$

   in $2n + m$ Variablen, setze $x'_i := \sum_{j=1}^n a_{i,j} x_j$ und bilde das Ideal

   $$I_0 := (I_G) + (y_1 - x'_1, \ldots, y_n - x'_n) \subseteq K[\underline{t}, \underline{x}, \underline{y}].$$

2. Berechne Erzeuger $g_1, \ldots, g_r$ des Eliminationsideals $I := I_0 \cap K[x_1, \ldots, x_n, y_1, \ldots, y_n]$. (Wegen seiner Verwendung im Derksen-Algorithmus nennen wir I das *Derksen-Ideal*.)

3. Bilde den Polynomring

   $$K[x_1, \ldots, x_n, y_1, \ldots, y_n, z_1, \ldots, z_n] = K[\underline{x}, \underline{y}, \underline{z}]$$

   in $3n$ Unbestimmten und das Ideal

   $$\begin{aligned} J_0 := &(g_1(\underline{x}, \underline{z}), \ldots, g_r(\underline{x}, \underline{z})) \\ &+ \big(g_1(\underline{y}, \underline{z}), \ldots, g_r(\underline{y}, \underline{z})\big) \subseteq K[\underline{x}, \underline{y}, \underline{z}]. \end{aligned}$$

4. Berechne das Eliminationsideal $J := J_0 \cap K[\underline{x}, \underline{y}]$. ($J$ beschreibt die Varietät aller Paare $(v, w) \in V \times V$, die nicht durch Invarianten getrennt werden können.)

5. Produziere nacheinander homogene Invarianten $f_1, \ldots, f_k$ mit steigenden Graden, bis gilt:

   $$J \subseteq \sqrt{(f_1(\underline{x}) - f_1(\underline{y}), \ldots, f_k(\underline{x}) - f_k(\underline{y}))}.$$

   Dann bilden $f_1, \ldots, f_k$ ein System von $K[V]^G$-separierenden Invarianten.

Für die Berechnung der Eliminationsideale und für den Inklusionstest in Schritt 5 werden Gröbnerbasen-Techniken verwendet. Der Inklusionstest erfordert *nicht* die Berechnung eines Radikalideals. Für Informationen über einschlägige Methoden sei auf die Standardliteratur (etwa [2, 11, 7] oder [4, Kapitel 1]) verwiesen. Die

Verwendung der drei Variablensätze $\underline{x}$, $\underline{y}$, $\underline{z}$ und das zweifache Berechnen eines Eliminationsideals mögen überraschen oder gar abschrecken. Dies spiegelt jedoch die Bedingung in (2) wider, die man umformulieren kann zu: „$\exists\, z \in \overline{G(x)} \cap \overline{G(y)}$“ – wodurch sowohl die drei Variablensätze als auch das Eliminationsideal J (als Ausdruck des Existenzquantors) eine Interpretation finden. Algorithmus 6 hat die ersten zwei Schritte mit dem Derksen-Algorithmus [4, Algorithm 4.1.9] gemeinsam – überhaupt ist die Verwandtschaft unverkennbar. Unser Algorithmus lässt sich relativ leicht so abwandeln, dass er auch bei Operationen auf affinen Varietäten X separierende Invarianten liefert. Man kann Algorithmus 6 auch laufen lassen, wenn G nicht reduktiv ist. In vielen Fällen wird dann jedoch das Abbruchkriterium in Schritt 5 nie erreicht, so dass der Algorithmus nicht terminiert.

Wie oben bereits angedeutet, wollen wir nun aus einem System separierender Invarianten Erzeuger des Invariantenrings gewinnen. Wie lässt sich das bewerkstelligen? Tatsächlich zeigen geometrische Überlegungen, dass separierende und erzeugende Invarianten gar nicht so weit auseinander liegen. Ist nämlich $A \subseteq K[X]^G$ eine Unteralgebra, so induzieren die Inklusionen $A \subseteq K[X]^G \subseteq K[X]$ dominante Morphismen

$$X \to X/\!\!/G \to \mathrm{Spec}(A),$$

wobei wir $X/\!\!/G := \mathrm{Spec}\left(K[X]^G\right)$ für den „kategoriellen Quotienten“ schreiben. Falls G reduktiv ist, so ist der Morphismus $X \to X/\!\!/G$ sogar surjektiv (siehe [8, Theorem 3.5(ii)]). Ist A separierend (also erzeugt von $K[X]^G$-separierenden Invarianten) so muss also der Morphismus $X/\!\!/G \to \mathrm{Spec}(A)$ (genauer: dessen Einschränkung auf die Maximalspektren) injektiv sein. Falls X irreduzibel ist, so folgt, dass die Körpererweiterung $\mathrm{Quot}(A) \subseteq \mathrm{Quot}(K[X]^G)$ endlich und rein inseparabel ist. Falls wir es mit einer linearen Operation zu tun haben (also $X = V$ ein Vektorraum) und A von homogenen Invarianten erzeugt wird, so folgt aus der Injektivität von $X/\!\!/G \to \mathrm{Spec}(A)$ bei den Vertizes, dass $K[V]^G$ außerdem ganz über A ist (siehe [6, Lemma 1.3]). Insgesamt ergibt sich, dass man von einer homogen erzeugten separierenden Unteralgebra $A \subseteq K[V]^G$ zu dem vollen Invariantering gelangt, indem man zuerst den ganzen Abschluss (= Normalisierung) $B := \widetilde{A}$ von A (in $\mathrm{Quot}(A)$) bildet, und davon dann im Falle positiver Charakteristik p den „inseparablen Abschluss“

$$\widehat{B} := \{f \in K[V] \mid f^q \in B \text{ für eine } p\text{-Potenz } q\}\,.$$

(Im Fall von Charakteristik 0 setzen wir $\widehat{B} := B$.) Glücklicherweise gibt es für beide Schritte Algorithmen: Für die Normalisierung können wir den Algorithmus von de Jong [5] (siehe auch [4, Abschnitt 1.6]) und für den inseparablen Abschluss den in [6, Algorithm 4.6] gegebenen Algorithmus verwenden. Insgesamt erhalten wir folgendes Verfahren.

Algorithmus 7 (Erzeugende Invarianten für reduktive Gruppen).

Input: Wie in Algorithmus 6.

Output: Erzeuger des Invariantenrings $K[V]^G$.

1. Berechne homogene separierende Invarianten $f_1, \ldots, f_k \in K[V]^G$ mit Algorithmus 6.
2. Setze $A := K[f_1, \ldots, f_k]$ und berechne die Normalisierung $B := \widetilde{A}$, etwa mit de Jongs Algorithmus [5].
3. Berechne den inseparablen Abschluss $\widehat{B}$ von B mit Algorithmus 4.6 von [6]. Dann ist $K[V]^G = \widehat{B}$.

Auf den Algorithmus zur Berechnung des inseparablen Abschlusses [6, Algorithm 4.6] soll hier nicht eingegangen werden, da dieser ein wenig technisch ist. Abgesehen von einer unvollständigen und nicht optimierten Test-Implementierung in Magma [3] ist Algorithmus 7 noch nicht implementiert. Daher lässt sich bis jetzt auch wenig über die Effizienz sagen. Erste Eindrücke weisen darauf hin, dass die Laufzeiten in derselben Größenordnung wie die des Derksen-Algorithmus liegen. Es besteht die Hoffnung, dass Algorithmus 7 einen Einstieg in die Invariantentheorie klassischer Gruppen in positiver Charakteristik ermöglicht.

Zum Abschluss dieses Berichts seien noch einige größere offene Probleme genannt.

- Man finde einen Algorithmus zur Berechnung erzeugender Invarianten von reduktiven Gruppen, aber ohne die Beschränkung auf lineare Operationen. Auch für endliche Gruppen ist dies noch offen.
- Man finde einen Algorithmus zur Konstruktion separierender Invarianten, aber ohne die Beschränkung auf reduktive Gruppen. Mit anderen Worten, Satz 2 soll für den Fall $F = K[X]^G$ konstruktiv gemacht werden.
- Man implementiere und optimiere Algorithmus 7.

Literatur

[1] Thomas Bayer, *An Algorithm for Computing Invariants of Linear Actions of Algebraic Groups up to a Given Degree*, J. Symb. Comput. **35** (2003), 441–449.

[2] Thomas Becker, Volker Weispfenning, *Gröbner Bases*, Springer-Verlag, Berlin, Heidelberg, New York 1993.

[3] Wieb Bosma, John J. Cannon, Catherine Playoust, *The Magma Algebra System I: The User Language*, J. Symb. Comput. **24** (1997), 235–265.

[4] Harm Derksen, Gregor Kemper, *Computational Invariant Theory*, Encyclopaedia of Mathematical Sciences **130**, Springer-Verlag, Berlin, Heidelberg, New York 2002.

[5] Theo de Jong, *An Algorithm for Computing the Integral Closure*, J. Symb. Comput. **26** (1998), 273–277.

[6] Gregor Kemper, *Computing Invariants of Reductive Groups in Positive Characteristic*, Transformation Groups **8** (2003), 159–176.

[7] Martin Kreuzer, Lorenzo Robbiano, *Computational Commutative Algebra 1*, Springer-Verlag, Berlin 2000.

[8] P. E. Newstead, *Introduction to Moduli Problems and Orbit Spaces*, Springer-Verlag, Berlin, Heidelberg, New York 1978.

[9] Vladimir L. Popov, *On Hilbert's Theorem on Invariants*, Dokl. Akad. Nauk SSSR **249** (1979), English translation Soviet Math. Dokl. **20** (1979), 1318–1322.

[10] Bernd Sturmfels, *Algorithms in Invariant Theory*, Springer-Verlag, Wien, New York 1993.

[11] Wolmer V. Vasconcelos, *Computational Methods in Commutative Algebra and Algebraic Geometry*, Algorithms and Computation in Mathematics **2**, Springer-Verlag, Berlin, Heidelberg, New York 1998.

Der folgende Artikel ist unter gleichem Titel in den Mitteilungen der Deutschen Mathematiker-Vereinigung, Heft 4-2003, erschienen. Wir bedanken uns für die Erlaubnis, ihn hier abdrucken zu dürfen.

Primzahl-Rekordjagd

Günter M. Ziegler (Berlin)

ziegler@math.tu-berlin.de

Der Dezember 2003 beschert uns mehrere Primzahl-Rekorde. So wurde unter der Regie von Jens Franke (Bonn) das RSA-576 *Entschlüsselungsproblem gelöst: die Faktorisierung einer 174-stelligen Dezimalzahl. Die größte bekannte Primzahl ist ebenfalls neu, eine Mersennesche Primzahl mit insgesamt 6.320430 Stellen:*

$$M = 2^{20.996.011} - 1.$$

Die Medien (unter anderem Spiegel-Online vom 3. Dezember) schreiben die Entdeckung einem Studenten der Verfahrenstechnik an der Michigan State University namens Michael Shafer zu – aber das ist nur ein Teil der Wahrheit.

Mersennesche Zahlen

Seit Januar 1996 läuft im Internet eine Suche nach immer größeren Mersenneschen Primzahlen. In dem verteilten Rechenprojekt unter dem Titel GIMPS („Great Internet Mersenne Prime Search", `www.mersenne.org`), können Freiwillige übers Internet die GIMPS-Computerprogramme abrufen und „ihre" Zahlen zum Testen zugeteilt bekommen, ihre PCs damit Sklavenarbeit leisten lassen, und die Rückmeldung übers Internet abliefern.

Zur Erinnerung: zu Ehren des französischen Mönches Marin Mersenne (1588-1648) heißen die Zahlen der Form $M_n = 2^n - 1$ *Mersennesche Primzahlen* – wenn sie prim sind. Dafür ist notwendig (schöne Übungsaufgabe aus der elementaren Zahlentheorie), dass n selbst prim ist. Aber hinreichend ist das nicht: $n = 11$ liefert das erste Gegenbeispiel. Im Jahr 1644 behauptete Mersenne, dass M_n für $n = 2$, 3, 5, 7, 13, 17, 19, 31, 67, 127 und 257 prim sei, aber keine andere Primzahl unter 257 (womit er exakt fünfmal danebengelegen hat).

Mersennesche Primzahlen sind ziemlich selten:

Marin Mersenne, 1588 – 1648 (Quelle: `http://www-groups.dcs.st-and.ac.uk/~history/PictDisplay/Mersenne.html`*)*

Man weiß nicht, ob es unendlich viele gibt, und man kennt inzwischen die ersten 38 davon, und nur zwei weitere, darunter die neu gefundene $M_{20.996.011}$, die auch die größte bekannte Primzahl überhaupt ist.

Dass man Zahlen mit mehr als 6 Millionen Stellen effektiv auf Primalität testen kann, ist die eigentliche wissenschaftliche (und programmiererische) Höchstleistung hinter dem neuen Rekord – dass $n = 20.996.011$ prim sein muss, ist ja nur eine klitzekleine Aufwärmübung für den neuen Rekord.

Primalitätstests

Nun weiß man seit Kurzem, dass es exakte Primzahltests gibt, die in Polynomzeit laufen – siehe *DMV Mitteilungen* 4/2002, S. 14–21. Diese stellen einen theoretischen Durchbruch dar, sind aber für den Einsatz in der Praxis (noch) nicht geeignet. Im GIMPS-Projekt wird für jedes prime n eine Kaskade von klassischeren Tests durchlaufen, die unter `www.mersenne.org/math.htm` sehr schön und kapierbar beschrieben werden.[1] In *Phase I* sucht man nach kleinen Primteilern q von $2^n - 1$. Diese müssen (wieder eine hübsche Übungsaufgabe) $q \equiv 1 \bmod 2n$ und $q \equiv \pm 1 \bmod 8$ erfüllen. Mithilfe eines auf solche Faktoren zugeschnittenen „Sieb des Eratosthenes“ werden dann Primteiler von M_n bis ca. 40.000 erkannt. Dabei kann ausgenutzt werden, dass Teilbarkeitstests für Zahlen vom Typ $2^n - 1$ in Binärarithmetik sehr effektiv durchgeführt werden können.

In *Phase II* wird dann ein Spezialfall der sogenannten $(p-1)$–Methode von Pollard (1974) verwendet, mit der man Faktoren $q = 2kn + 1$ finden kann, für die $q - 1 = 2kn$ aus vielen kleinen Primfaktoren besteht, oder aber (in einer verbesserten Version) bis auf einen etwas größeren Primfaktor stark zusammengesetzt ist: Wenn man q sucht, so dass alle Primfaktoren kleiner als B sind, so bildet man dafür das Produkt $E := \prod_{p<B} p$ aller Primzahlen, die kleiner als B sind, und berechnet dann $x := 3^{E2n}$. Im ggT von $x - 1$ und $2^n - 1$ fängt man dann den gesuchten Teiler von $2^n - 1$.

`http://www.mersenne.org`

Erst in *Phase III* verwendet man dann ein Verfahren, mit dem man sicher entscheiden kann, ob $2^n - 1$ prim ist, den sogenannten Lucas–Lehmer Test (1878, 1930/1935) für Mersenne-Zahlen: M_n ist genau dann prim, wenn $\ell_{n-1} \equiv 0 \bmod M_n$ gilt, wobei die ℓ_k durch $\ell_1 = 4$ und $\ell_n = \ell_{n-1}^2 - 2$ rekursiv definiert werden. Um das effektiv zu berechnen, muss man riesige Zahlen schnell modulo $2^n - 1$ quadrieren. Dazu werden die Zahlen in große Blöcke unterteilt, und dann arbeitet man mit Spezialversionen einer schnellen Fourier Transformation („Fast Fourier Transform“, FFT), in diesem Fall mit einer FFT bezüglich einer irrationalen Basis, die von Richard Crandell und Barry Fagin (*Mathematics of Computation* 1994) eingeführt wurde. Auf den WWW-Seiten des *Mathematica*-Projekts `mathworld.wolfram.com`, die die aktuelle Rekordmeldung verbreiten, wird suggeriert, GIMPS würde mit einer *Mathematica*-Implementierung arbeiten, aber das ist eine arge Dehnung der Tatsachen. (Es hat nur Crandell die Methode auch für die Primzahltests von *Mathematica* implementiert.) In der Tat arbeitet GIMPS mit hochoptimiertem Assembler-Code, aus Prozessorarchitekturgründen in Gleitkommaarithmetik, deren Fehler getrennt erkannt und aufgefangen werden müssen.

[1] Zur algorithmischen Primzahltheorie empfehlen die Experten RICHARD CRANDELL & CARL POMERANCE: “Prime Numbers. A Computational Perspective”, Springer-Verlag, New York 2001. Aus Computeralgebra-Perspektive finden sich Primzahltests (und sehr viel mehr Spannendes) in JOACHIM VON ZUR GATHEN & JÜRGEN GERHARD: “Modern Computer Algebra”, Cambridge University Press, 2. Auflage 2003.

Primalität und Faktorisierung

Phasen I und II des GIMPS-Verfahrens spucken also im Fall von zusammengesetztem M_n wirklich Teiler aus – *wenn* sie welche finden –, die dritte und entscheidende Phase aber nicht mehr. Die Antwort heißt da dann nur noch „zusammengesetzt!“, ohne einen expliziten (Prim-)Teiler als Beweis. Es wird also ein Primalitätstest durchgeführt, aber kein vollständiges Faktorisierungsverfahren.

Und das ist auch gut so: Nicht einmal für den Spezialfall von Mersenne-Zahlen kennt man effektive Verfahren zum Faktorisieren. Ein Verfahren, mit dem man *beliebige* Zahlen mit ein paar Hundert Stellen faktorisieren könnte, wäre interessant und bedrohlich, weil die kryptographischen Verfahren, die die Sicherheit von Online-Banking und Internet garantieren sollen, darauf beruhen, dass das Faktorisieren und verwandte Probleme (wie die Berechnung von „diskreten Logarithmen“) offenbar schwer sind.

RSA

Ein Beispiel dafür ist das von Ron Rivest, Adi Shamir und Leonard Adleman 1978 publizierte Verschlüsselungsverfahren „mit öffentlichen Schlüsseln“, das sich inzwischen in fast jedem elementaren Zahlentheorie-Lehrbuch findet, gleichzeitig aber auch in der Praxis vielfältig zum Einsatz kommt – siehe die Homepage `http://www.rsasecurity.co` der Firma von Rivest, Shamir und Adleman. Die Sicherheit des Verfahrens gegen unerlaubtes Entschlüsseln hängt davon ab, dass es mit heutiger Technologie sehr schwer ist, Produkte von Zahlen mit 150-200 Stellen in ihre Primfaktoren zu zerlegen. Die Firma „RSA Securities“ hat sogar Preise auf Beispielprobleme[2] ausgesetzt. Der erste davon ist/war ein Preis von 10.000 Dollar für das Faktorisieren der Zahl „RSA-576“

188198812920607963838697239461650439807163563
379417382700763356422988859715234665485319060
606504743045317388011303396716199692321205734
031879550656996221305168759307650257059

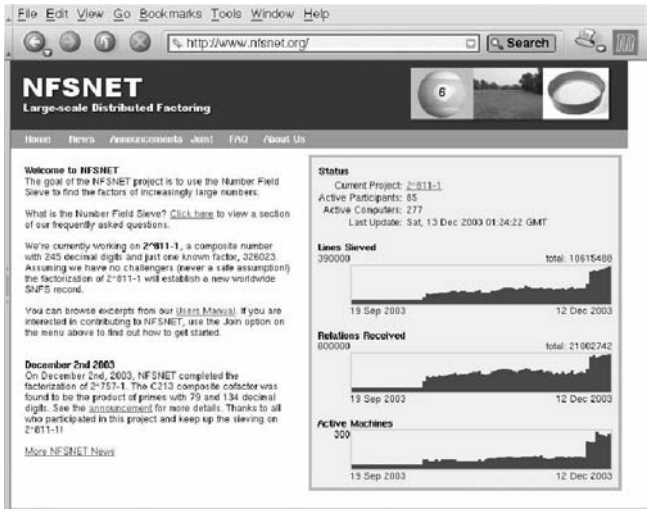
mit 174 Dezimalziffern, bzw. 576 Binärziffern (bits). Und dieses Problem hat Jens Franke von der Universität Bonn jetzt geknackt, wie *Heise Online* am 8. Dezember gemeldet hat: die Zahl hat Faktoren

398075086424064937397125500550386491199064362
342526708406385189575946388957261768583317

und

472772146107435302536223071973048224632914695
302097116459852171130520711256363590397527

(mit je 87 Ziffern), und die sind prim – was wiederum mit den aktuellen Methoden ganz leicht zu zeigen ist. Franke verwendete dabei das „General Number Field Sieve (GNFS)“. Dieses wurde von Lenstra, Lenstra, Manasse & Pollard 1990 eingeführt, und hat eine Laufzeit von $\exp(O(\sqrt[3]{n \log n}))$ für n-stellige Zahlen; es ist also nicht ganz polynomial, aber *fast*. Unter Verwendung des GNFS wurden auch schon die kleineren Testprobleme von RSA-100 bis RSA-512 geknackt (letzteres im August 1999).



`http://www.nfsnet.org`

Und es gibt noch mehr aktuelle Rekorde, die sich ebenfalls aufs Faktorisieren beziehen: Unter Anderem versucht man eben Mersenne-Zahlen nicht nur auf Primalität zu untersuchen, sondern auch vollständig in Primfaktoren zu zerlegen. NFSNET (`http://www.nfsnet.org` ist auch ein Internet-Projekt, dem es nun (Erfolgsmeldung vom 2. Dezember) in Internet-Gemeinschaftsarbeit gelang die Mersennesche Zahl $2^{757} - 1$ vollständig zu faktorisieren: Die Primfaktoren 9815263 und 561595591 dieser Zahl kannte man schon länger, aber der 212-stellige Rest war ein hartes Stück Arbeit: er wurde jetzt in die Primfaktoren

572213702200206782424822797509585774915131282
780938840696234625318212891696459 3

und

240338216409835080887362734030059654466890023
563443321305650666431938139011197710904242694
120545430727149147426656777742473252923275 59

zerlegt. Dieser Erfolg basiert auf dem „Special Number Field Sieve (SNFS) “ – einer schnelleren Spezialversion des GNFS, die nur für spezielle Zahlen, etwa vom Typ $b^n \pm 1$, anwendbar ist.

Rekordjagd

Die Rekordjagd geht weiter. Die „Electronic Frontier Foundation“ (`http://www.eff.org/`) hat schon im Jahr 2000 einmal 50.000 Dollar für die erste Primzahl mit einer Million Stellen ausgezahlt. Für die Identifikation einer Primzahl mit mehr als 10 Millionen Dezimalstellen hat sie 100.000 Dollar ausgesetzt. Dies heizt die Stimmung an, und das GIMPS-Projekt sucht Mitstreiter, die ihre Computer für die Rekordjagd einsetzen wollen.

Genauso ist man natürlich hinter den größeren RSA-Problemen hinterher; als nächstes wartet da RSA-640,

[2] `http://www.rsasecurity.com/rsalabs/challenges/factoring/numbers.html`

eine Zahl mit 193 Dezimalstellen, auf ihre Zerlegung. Darauf sind 20.000 Dollar ausgesetzt.

Und die nächste Mersenne-Zahl auf der Abschuss- bzw. Zerlegungsliste von NFSNET ist $2^{811} - 1$. Auch für dieses Projekt werden noch Mitstreiter gesucht.

Viele arme kleine PCs werden also mit Zahlen gefüttert und mit Primzahltests und mit Zerlegungsverfahren gequält werden, nur damit Herrchen vielleicht einen Teil des Ruhms (und des Preisgeldes) einkassieren kann.

Neues über Systeme

GiNaC – eine C++-Bibliothek für symbolisches Rechnen

Christian Bauer (Mainz)

Über GiNaC Handelsübliche Computeralgebrasysteme bieten typischerweise weitreichende algebraische Fähigkeiten und eine umfassende Sammlung an Funktionen aus allen Bereichen der reinen und angewandten Mathematik, eingebettet in eine interaktive Benutzerumgebung mit einfachen Programmiermöglichkeiten. GiNaC (ein rekursives Akronym für „*GiNaC is not a computer algebra system*“) verfolgt hier einen anderen Ansatz: Es erweitert die existierende Programmiersprache C++ um Klassen und Funktionen zum symbolischen Rechnen.

Ein Beispiel Hier ist ein vollständiges C++-Programm, das mit Hilfe von GiNaC Laguerre-Polynome nach der Formel von Rodrigues berechnet und diese im LaTeX-Format ausgibt:

```
#include <iostream>
using std::cout; using std::endl;

#include <ginac/ginac.h>
using namespace GiNaC;

ex laguerre(const symbol & x, unsigned n)
\{
    ex L_n = diff(pow(x, n)*exp(-x), x, n)
             / (exp(-x) * factorial(n));
    return L_n.normal();
\}

int main()
\{
    symbol x("x");
    cout << latex << laguerre(x, 4) << endl;
\}
```

Die Ausgabe des Programms ($1+\frac{3}{2}x^2-\frac{1}{6}x^3-3x$) kann direkt in LaTeX-Dokumente übernommen werden.

Die wichtigsten von GiNaC zur Verfügung gestellten Datentypen sind die Klasse `ex` (kurz für „*expression*“), die einen beliebigen algebraischen Ausdruck speichert und die Klasse `symbol`, die eine symbolische Variable repräsentiert. Da die Namen von C++-Variablen nach der Kompilierung nicht mehr zugänglich sind, muss beim Erzeugen von Symbolen ein Name angegeben werden, der zur Ausgabe des Symbols verwendet wird, in diesem Fall „*x*“.

Symbolische Ausdrücke lassen sich mit GiNaC in genau der gleichen Weise hinschreiben wie bei rein numerischen Ausdrücken in C++ üblich. Die Funktion `diff(e, x, n)` berechnet die n-fache symbolische Ableitung des Ausdrucks e nach der Variablen x und die Methode `.normal()` bringt rationale Ausdrücke auf die Form eines vollständig gekürzten Bruches, wobei in diesem Fall der gemeinsame Faktor e^{-x} herausdividiert wird.

Was kann GiNaC (und was kann es nicht)? Das Ziel des GiNaC-Projekts ist die Entwicklung eines offenen und erweiterbaren Werkzeugs, mit dem symbolische Rechnungen direkt in C++ implementiert werden können. Die wichtigsten Eigenschaften von GiNaC umfassen: eine sehr schnelle komplexe Arithmetik basierend auf der CLN-Bibliothek, effiziente Handhabung selbst großer (mehrere tausend Terme) multivariater Polynome und rationaler Funktionen, symbolisches Differenzieren, Entwicklung von Funktionen in Taylor- und Laurentreihen, Matrizen, Vektoren und Lösen linearer Gleichungssysteme, indextragende Objekte wie z. B. Tensoren, viele vordefinierte mathematische Funktionen (trigonometrische Funktionen, Logarithmen, Fakultät etc.), variable Ausgabemöglichkeiten (z. B. als LaTeX-Formel oder als optimierter C++-Code für anschließende numerische Rechnungen), Speichereffizienz durch (für den Benutzer unsichtbare) Referenzzählung von Objekten und leichte Erweiterbarkeit durch eigene symbolische Funktionen und Klassen. Schließlich ist GiNaC kostenlos und im Quelltext verfügbar.

Da GiNaC ursprünglich für Anwendungen in der Hochenergiephysik entwickelt wurde, bietet es außerdem zahlreiche für dieses Aufgabenfeld spezialisierte Objekte und Funktionen an, wie z. B. Clifford- und Farbalgebren und Polylogarithmen.

GiNaC ist zur Entwicklung von umfangreichen Applikationen vorgesehen, bei denen symbolische Rech-

nungen erforderlich sind. Daher wurde kein Wert auf eine interaktive Benutzerumgebung ähnlich zu der „traditioneller“ Computeralgebrasysteme gelegt. Eine solche existiert zwar, ist jedoch in erster Linie für Testzwecke gedacht. Selbstverständlich kann GiNaC aber als Grundlage für ein interaktives System dienen, wie das Paket gTybalt beweist.

Darüberhinaus fehlen in GiNaC zur Zeit noch einige Fähigkeiten wie die Faktorisierung von Polynomen, die Berechnung von Grenzwerten oder symbolisches Integrieren, wie sie von den meisten anderen Computeralgebrasystemen angeboten werden. Auf die Implementierung von Funktionen wie einem allgemeinen `simplify()` zum Vereinfachen von Ausdrücken, deren Ergebnis nicht klar vorhersagbar ist (nach welchem Maß ist ein Ausdruck „einfach“?) wurde jedoch bewusst verzichtet.

Warum Computeralgebra in C++? Die Motivation zur Entwicklung von GiNaC waren die Erfahrungen mit „*xloops*“, einem ebenfalls an der Universität Mainz erstellten Programm zur automatisierten Berechnung von Schleifenintegralen in der Quantenfeldtheorie. Die dabei notwendigen, sehr umfangreichen algebraischen Rechnungen wurden ursprünglich in der Sprache des kommerziellen Computeralgebrasystems Maple implementiert, was sich schnell als sehr problematisch herausstellte.

Zum einen sind die in Maple und ähnlichen Systemen eingebauten Programmiersprachen zwar einfach zu lernen und für kleine Projekte recht praktisch, aber zur Entwicklung von Anwendungen wie *xloops*, die mehrere tausend Programmzeilen umfassen, eher ungeeignet. Möglichkeiten zur Modularisierung von Programmen und zur Informationskapselung fehlen fast vollständig, als einzige Datenstruktur stehen meist nur anonyme Listen zur Verfügung, und auch die angebotenen Hilfen zur Fehlersuche in Programmen sind üblicherweise sehr bescheiden.

Zum anderen führt die Verwendung eines proprietären Systems zu einer Abhängigkeit von einem bestimmten Programm (teilweise sogar einer bestimmten Programmversion) und seinem Hersteller, die für langlebige Projekte nicht wünschenswert ist. Syntax und Semantik der Sprache von Maple haben sich im Laufe letzten Jahre mehrfach geändert. Für *xloops* waren bereits zwei getrennte Versionen für Maple V Release 1 und Release 3 erforderlich. Mit neueren Maple-Versionen ist der gleiche Code nicht mehr lauffähig.

Die Entscheidung für C++ als Implementationssprache fiel aus mehreren Gründen:

- C++ ist standardisiert, was das Projekt gegenüber Willkürlichkeiten seitens des Herstellers der Programmierumgebung schützt.
- Entwicklungssysteme für C++ existieren nicht nur in großer Zahl, sondern auch für alle relevanten Plattformen.
- C++ hat sich nicht nur in der Physik zur Entwicklung von umfangreichen Applikationen bewährt.
- Die Möglichkeit zum Überladen von Operatoren erlaubt es in C++ symbolische Ausdrücke in ihrer gewohnten Art und Weise hinzuschreiben, also z. B. `x+y` statt `add(x, y)` oder ähnlicher Konstrukte.
- Als universelle Programmiersprache ist C++ auch für Anwendungen geeignet, die nur teilweise auf symbolischen Rechnungen beruhen. Insbesondere müssen algebraische Resultate oft numerisch weiterverarbeitet werden, was in C++ ebenfalls effizient möglich ist. Das für viele Programm„pakete“ der Hochenergiephysik typische Chaos aus einer Vielzahl von Sprachen und Systemen (C++, Fortran, Mathematica, Maple, FORM ...) wird somit vermieden.

Verfügbarkeit und Ausblick GiNaC ist unter der GNU General Public License (GPL) auf `http://www.ginac.de` verfügbar. Aktuell ist Version 1.1.6.

Für die nähere Zukunft sind insbesondere eine noch stärkere Integration mit C++ und eine leichtere Erweiterbarkeit geplant. So wird Version 1.2 unter Anderem STL-kompatible Iteratoren zum Durchlaufen von Ausdrücken und Templates zur einfachen Erzeugung neuer algebraischer Klassen bereitstellen.

gTybalt – ein frei verfügbares Computeralgebrasystem

Stefan Weinzierl (München)

gTybalt ist ein frei verfügbares Computeralgebrasystem, das unter der GNU General Public Licence steht [1]. Entwickelt wurde gTybalt im Bereich der Elementarteilchenphysik. In diesem Fachgebiet sind symbolische Rechenmethoden mit Hilfe eines Computers mittlerweile ein unumgänglicher Bestandteil täglicher Arbeit. Eine Analyse der Anforderungen an ein Computeralgebrasystem für diese Rechnungen zeigt, dass in erster Linie das Computeralgebrasystem in der Lage sein muss, große Datenmengen zu verarbeiten. Zweitens ist

es meistens der Fall, dass die Algorithmen zur Lösung eines Problems von den Anwendern selbst entwickelt werden. Dies erfordert eine Programmiersprache, die die Entwicklung komplexer Algorithmen unterstützt. Andererseits ist es dagegen nicht unbedingt notwendig, dass das Computeralgebrasystem vorgefertigte Routinen für alle Bereiche der Mathematik besitzt. Drittens ist es im Allgemeinen der Fall, dass die Zeit, die für die Entwicklung und Implementierung der Algorithmen benötigt wird, die tatsächliche Laufzeit des Programms bei weitem übersteigt. Deshalb ist auch eine komfortable Entwicklungsumgebung hilfreich.

gTybalt wurde im Hinblick auf diese Anforderungen entwickelt. Die wichtigsten Eigenschaften von gTybalt sind:

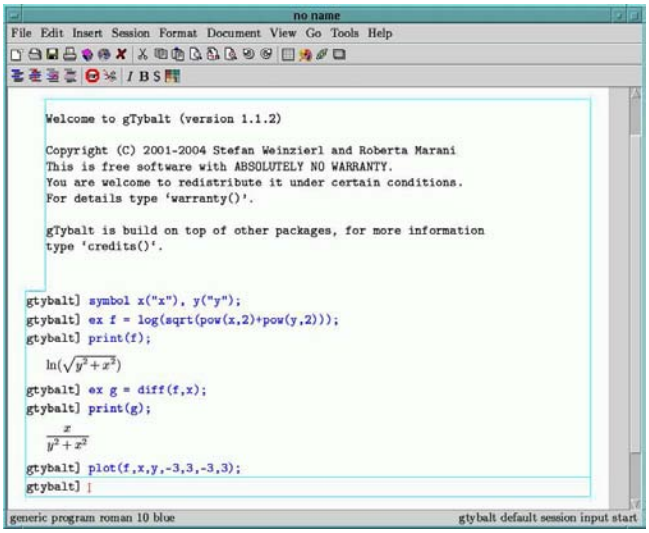
- Objektorientierte Programmiersprache: gTybalt erlaubt, symbolische Rechnungen in C++ zu programmieren.
- Effizienz für große Datenmengen: Algorithmen, die mit Hilfe von gTybalt entwickelt worden sind, können mit Hilfe eines C++ Compilers kompiliert werden und unabhängig von gTybalt ausgeführt werden.
- Kurze Entwicklungszyklen: gTybalt kann C++ Anweisungen interpretieren und C++ Skripte ausführen. Lösungen können interaktiv oder durch Skripte mit Hilfe von gTybalt erarbeitet werden. Nachdem die Algorithmen getestet und eventuelle Programmierfehler beseitigt worden sind, können sie kompiliert und mit größeren Datenmengen eingesetzt werden.
- Ausgabe der Resultate in TEX-fonts: gTybalt stellt mathematische Formeln mit Hilfe von TEX-fonts auf dem Bildschirm dar. Die dargestellten Formeln lassen sich einfach zu einer LATEX-Datei konvertieren.

Die Funktionalität eines Computeralgebrasystems basiert auf mehreren Modulen. Beispiele für verschiedene Module sind ein Baustein für die Ausgabe von mathematischen Formeln auf dem Bildschirm, ein Modul für die Analyse und die Interpretation der Eingabe sowie natürlich ein Baustein für die eigentlichen symbolischen Rechnungen. Ein komplettes Computeralgebrasystem von Grund auf zu entwickeln, erfordert ein nicht unbedeutendes Maß an Arbeit. Im Rahmen eines Opensource-Projektes lässt sich die Entwicklungsarbeit jedoch signifikant reduzieren, da für viele Unterprojekte geeignete Programmpakete bereits existieren. gTybalt ist ein Programm im „Stile eines Basars“ [2] und basiert auf den folgenden Programmpaketen:

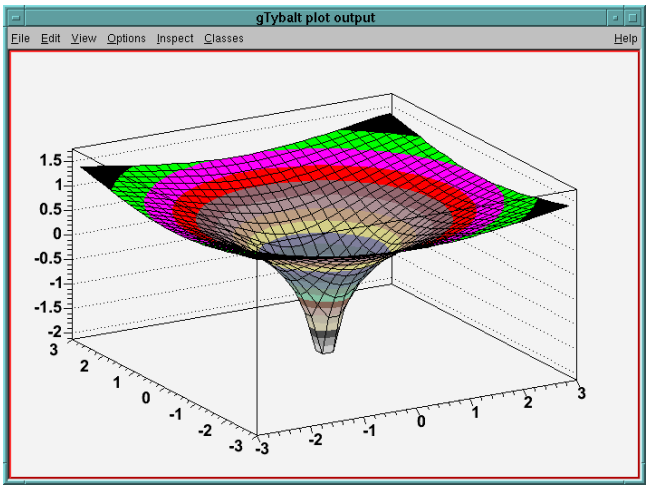
- Der TeXmacs-Editor [3] wird verwendet, um mathematische Formeln mit TEX-Qualität auf dem Bildschirm darzustellen.
- Alternativ kann gTybalt von einem Textfenster aus gestartet werden. In diesem Fall wird die Programmbibliothek eqascii [4] verwendet, um mathematische Formeln lesbar in einem Textfenster darzustellen.
- Jedes interaktive Programm benötigt für die Eingabebefehle einen Interpreter. gTybalt verwendet den CINT C/C++ Interpreter [5], welcher die Ausführung von Skripten und Eingabebefehlen in C++ erlaubt.
- Der Kern eines Computeralgebrasystems ist das Modul für symbolische und algebraische Manipulationen. Diese Funktionalität wird von der Programmbibliothek GiNaC [6] bereitgestellt.
- Ein Aspekt aller Computeralgebrasysteme ist die Fähigkeit, Zahlen beliebiger Größe exakt zu verarbeiten. Hierfür greift GiNaC (und damit auch gTybalt) auf die Class Library for Numbers (CLN library) [7] zurück.
- Die graphische Darstellung von Funktionen ist sehr hilfreich, um Ergebnisse schnell verstehen zu können. Die graphischen Fähigkeiten von gTybalt werden durch die Einbeziehung des Root-Paketes [8] erreicht.
- Die GNU scientific library [9] wird für Monte Carlo Integration verwendet.
- Fakultativ kann gTybalt mit einem Modul für die Entwicklung transzendenter Funktionen kompiliert werden. Dies erfordert die Installation der Programmbibliothek nestedsums [10].
- Fakultativ kann gTybalt mit einem Modul für die Faktorisierung von Polynomen kompiliert werden. Dies erfordert die Installation der Programmbibliothek NTL [11].

Aufgrund der modularen Struktur lässt sich gTybalt problemlos um zusätzliche Komponenten erweitern.

Die umseitige Abbildung zeigt eine Beispielsitzung für gTybalt. In dieser Sitzung werden zunächst zwei Symbole x und y definiert. Danach wird f als $\ln\sqrt{x^2+y^2}$ definiert. Ein allgemeiner Ausdruck ist immer vom Typ `ex`. Auch ist zu beachten, dass gerade definierte Ausdrücke nicht noch einmal auf dem Bildschirm ausgedruckt werden. Dies geschieht nur, wenn der Anwender dies mit `print` explizit verlangt. Der Ausdruck g wird als Ableitung von f nach x definiert und anschließend mit Hilfe von `print` auf dem Bildschirm ausgegeben. Am Schluß wird die Funktion f mittels eines `plot`-Befehles noch graphisch dargestellt.



Eine Beispielsitzung mit gTybalt.



Die graphische Darstellung einer Funktion zweier Variablen mittels gTybalt.

Weiterführende Literatur: Das Benutzermanual für gTybalt enthält detaillierte Informationen zur Bedienung und Installation von gTybalt. Darüber hinaus profitiert jeder Anwender sicherlich von den Benutzerhandbüchern der Programmpakete GiNaC, TeXmacs und Root, auf denen gTybalt aufbaut. Eine allgemeine Einführung zu Anwendungen von Computeralgebra im Bereich der Elementarteilchenphysik findet sich in [12].

Literatur

[1] S. Weinzierl, gTybalt, Comput. Phys. Commun. **156**, 180 (2004), cs.sc/0304043; `http://fis.unipr.it/~stefanw/gtybalt`.

[2] E. Raymond, The Cathedral and the Bazaar, `http://catb.org/~esr/writings/cathedral-bazaar`.

[3] J. van der Hoeven, Cahiers GUTenberg **39-40**, 39 (2001); TeXmacs (1999), `http://www.texmacs.org`.

[4] P. Borys, eqascii (2001), `http://dione.ids.pl/~pborys/software/linux`.

[5] M. Goto, C++ Interpreter – CINT, CQ publishing, ISBN 4-789-3085-3 (in Japanisch); M. Goto, CINT, `http://root.cern.ch/root/Cint.html`.

[6] C. Bauer, A. Frink, and R. Kreckel, J. Symbolic Computation **33**, 1 (2002), cs.sc/0004015; GiNaC library, `http://www.ginac.de`.

[7] B. Haible, CLN library (1999), `http://www.ginac.de/CLN`.

[8] R. Brun and F. Rademakers, Nucl. Inst. & Meth. in Phys. Res. **A389**, 81 (1997); Root, `http://root.cern.ch`.

[9] M. Galassi et al., GNU scientific library, `http://sources.redhat.com/gsl`.

[10] S. Weinzierl, Comput. Phys. Commun. **145**, 357 (2002), math-ph/0201011; nestedsums library, `http://fis.unipr.it/~stefanw/nestedsums`.

[11] V. Shoup, NTL library (1990), `http://www.shoup.net`.

[12] S. Weinzierl, Computer algebra in particle physics (2002), hep-ph/0209234.

Feli-X – ein Computeralgebra-gestütztes dynamisches Geometrieprogramm

Reinhard Oldenburg (Göttingen)

Algebra und Geometrie wechselwirken in der Mathematik auf vielfache Weise. Auf beiden Gebieten kann der Computer sowohl beim Forschen wie beim Lernen Unterstützung bieten. Den Computeralgebrasystemen (CAS) auf der einen Seite stehen dabei – für die ebene Geometrie – dynamische Geometrieprogramme (DGS) wie Cabri, Cinderella und Euklid gegenüber. DGS erlauben das interaktive Konstruieren und die Veränderung

der Konstruktionen durch Verziehen von Basiselementen mit der Maus, wobei alle abhängigen Elemente angepasst werden. Aus didaktischer Sicht ist bedauernswert, dass diese Werkzeuge die Kluft zwischen Algebra und Geometrie verstärken statt sie zu überbrücken.

Als Antwort auf diese Situation habe ich nach Vorüberlegungen aus dem Jahr 2000 im Sommer 2002 mit der Implementation eines Geometrieprogramms Feli-X begonnen, das auf dem CAS Mathematica aufsetzt. Ziel ist die möglichst enge Integration von geometrischen und algebraischen Arbeitsweisen mit Schülern der Sekundarstufe II und Studenten der Anfangssemester als Zielgruppe.

Der Ansatz von Feli-X Die Arbeit mit Feli-X geschieht in zwei Fenstern, zum einen einem Mathematica-Notebook für algebraische Rechnungen (kurz Algebrafenster) und einem Geometriefenster, das ähnliche Möglichkeiten bietet wie andere DGS auch. Beide Fenster werden bidirektional synchron gehalten. Die Änderung von Koordinaten im Zugmodus oder die Erstellung neuer Objekte im Geometriefenster wirkt sich unmittelbar im Algebrafenster aus. Dort stehen u. A. folgende Variablen zur Verfügung: Objects (die aktuell vorhandenen geometrischen Objekte), Vars (die Variablen der Objekte), Co (die aktuellen Koordinaten der Objekte), DGAncestors (der gerichtete Graph der Konstruktion) und Equations (die Gleichungen, die zwischen den Variablen gelten).

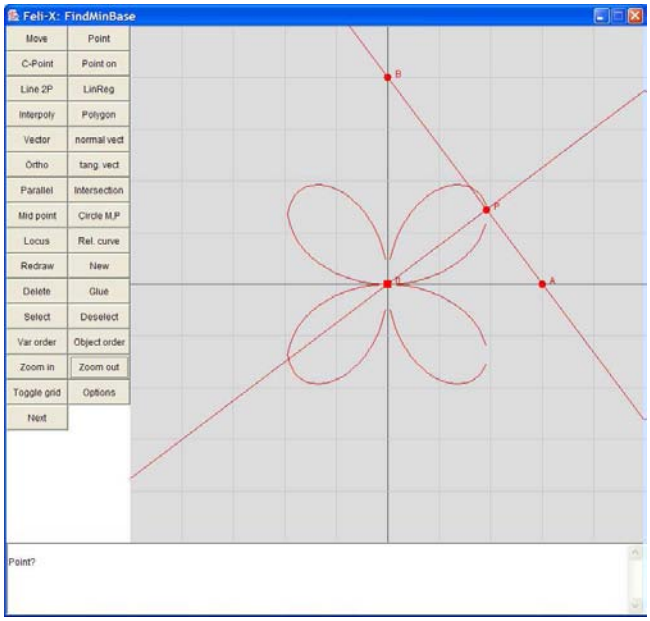
An diesen Variablen darf der Benutzer rumfummeln. Wenn er den Graphen in DGAncestors ändert, werden einige der wählbaren Zug-Strategien ihr Verhalten ändern. Wenn die Koordinaten geändert werden, bewegt sich das Bild im Geometriefenster. Wenn eine weitere Gleichung hinzu gefügt wird, ist die Bewegungsfreiheit der Konstruktion eingeschränkt. Gerade dieses Feature ist auch für jüngere Schüler interessant, da es erlaubt, die Bedeutung einer Gleichung mit der Maus erfahren zu können. Beispielsweise kann man mit addEquation[dist[P, F]==dist[P, g]] festlegen, dass der Punkt P nur an solche Orte mit der Maus schiebbar ist, die von einem gegebenen Punkt F und einer gegebenen Gerade g den gleichen Abstand haben, die also auf der Parabel zu diesem Brennpunkt und dieser Leitgeraden liegen. Im Gegensatz zum (auch sehr wichtigen) impliziten Plotten ermöglicht das einen operativen Zugang zur Exploration von Gleichungen und zur Modellierung mit Gleichungen. Diese brauchen auch nicht zwingend algebraisch zu sein. Beispielsweise kann man die Ausbreitung von Licht in Glaskörpern beliebiger Form modellieren, indem man Lote auf die Oberfläche setzt und die Brechungsgesetze in Kraft setzt.

Ein Beispiel Das folgende Skript konstruiert (im Sinne von Feli-X) eine Lotfußpunktkurve:

```
c=5; Nu=addObject["point",0,0];
addEquation[xc[Nu]==0];
addEquation[yc[Nu]==0];
A=addObject["point",3,0];
addEquation[yc[A]==0];
B=addObject["point",0,4];
addEquation[xc[B]==0];
l=addObject["line2P",A,B];
addEquation[dist[A,B]==c];
lot=addObject["orthogonalPG",Nu,l];
P=addObject["intersection",lot,l];
```

In dieser Konstruktion kann an A, B und P mit der Maus gezogen werden. Bei herkömmlichen DGS müsste man durch die Konstruktion dagegen A oder B als Basispunkt auszeichnen und könnte nur an diesem ziehen. Die größere Zugfreiheit macht sich besonders angenehm bemerkbar bei der Modellierung von Gelenkmechanismen, wo man in der Realität ja auch an verschiedenen Stellen ziehen kann.

Durch die Konstruktion ist P nicht mehr frei, sondern auf eine bestimme Bahn eingeschränkt, in Feli-X heißt sie eine Relationenkurve, weil es die (algebraische) Kurve ist, die die P auferlegte Relation darstellt. Feli-X kann sie zeichnen (siehe Bild) und ihre Gleichung berechnen.



Darstellung einer Kurve in Feli-X

Dazu verwendet Feli-X im Wesentlichen zwei Algorithmen: Zum einen direkt den Eliminate-Befehl von Mathematica, zum anderen einen schrittweisen, heuristischen Algorithmus, der danach trachtet immer nur wenige Variablen pro Schritt zu eliminieren und nur so viele Gleichungen mitzunehmen wie nötig. Allerdings gibt es natürlich Grenzen dieser symbolisch arbeitenden Verfahren.

Neben den Relationenkurven kann Feli-X auch Ortskurven im herkömmlichen Sinne der DGS berechnen. Dabei wird die Bahn eines Punktes berechnet, wenn ein anderer längs eines Trägerobjekts (Gerade, Kreis, ...) bewegt wird. Feli-X berechnet in diesem Fall

eine Parametergleichung der Kurve, was dann z. B. die graphische Darstellung erleichtert.

Weitere Features Die Werkzeugkiste von Feli-X umfasst u. A. die folgenden Hilfsmittel, die z. T. von herkömmlichen geometrischen Konstruktionsmitteln deutlich abweichen: Normalen- und Tangentenvektoren an Geraden, Kreise, Funktionsgraphen, parametrische Kurven, implizite Kurven, Binden von Punkten an die eben genannten Objekttypen, Interpolationspolynome und Regressionsgeraden.

Feli-X verfügt über eine Vielzahl verschiedener Strategien zur Aktualisierung einer Zeichnung nach dem Ziehen mit der Maus. Die meisten unterstützen das eben schon angesprochene Ziehen an Punkten, die in klassischen DGS abhängig wären. In der Regel gibt es eine Vielzahl von Möglichkeiten die konstruierten Bedingungen zu erfüllen. Dann kommen Zusatzinformationen ins Spiel wie etwa die Reihenfolge der Variablen (die aber natürlich interaktiv verändert werden kann).

Anwendungen Hier ist eine Liste von geometrisch-algebraischen Anwendungen von Feli-X, die mit herkömmlichen DGS schwierig zu bewältigen sind:

- Exploration der Kreisinversion, wobei man an Bild und Urbild mit der Maus ziehen kann. Eine konstruierte Lemniskate wird korrekt in eine Hyperbel abgebildet und die Hyperbelgleichung bestimmt.
- Kissoiden, Hypokissoiden und Pascalsche Schnecke zeichnen und deren Gleichung bestimmen.
- In Zusatzfenstern nicht nur die Entwicklung einer Variablen (z. B. einer bestimmten Entfernung) verfolgen, sondern beliebige Mathematica-Ausdrücke evaluieren, also z. B. auch Grafiken erzeugen.

Schlussbetrachtung Feli-X ist freie Software und kann von `www.oldenburg-goettingen.gmxhome.de` herunter geladen werden (Voraussetzung ist eine Installation von Mathematica 5.0 (mit Einschränkungen 4.x) mit J/Link). Allerdings befindet sie sich noch in einem frühen Entwicklungsstand, so dass ein Einsatz mit Schülern nicht praktikabel ist. Da Mathematica für Schulen zu teuer ist, soll das System zum einen nach MuPAD portiert werden, zum anderen ist geplant, einige der Ideen in das DGS Cinderella zu exportieren. Die nächste Cinderella-Version wird nämlich externe Algorithmen einbinden können. Darüber sollte es z. B. möglich sein, einen Punkt an eine Gleichung zu binden, in dem Sinne, dass seine Koordinaten die Gleichung erfüllen müssen. Feli-X ist ein innovativer Zugang zum explorativen Arbeiten mit Mathematik. Die Stärke des Systems, seine Flexibilität und Offenheit, kann, so wird immer wieder gefürchtet, auch seine Schwäche sein: Bleibt das System vernünftig bedienbar, wenn ein unerfahrener Nutzer damit arbeitet? Dies wird sich zeigen, wenn die Weiterentwicklung soweit gediehen ist, dass erstmals Schüler mit dem System arbeiten können. Neben einer Reifung der Software müssen dazu auch tragfähige didaktische Konzepte erarbeitet werden. Dazu gehört die Einsicht, dass ein solches System nicht geeignet ist, den traditionellen geometrischen Konstruktionsbegriff zu entwickeln. Es ist statt dessen ein System zum algebraischen Modellieren geometrischer Situationen und zum geometrischen Explorieren algebraischer Bedingungen.

Mathematica 5.0 – ein Interview mit Tom Wickham-Jones, Wolfram Inc.

Ulrich Kortenkamp (Berlin)

*Mathematica 5.0 ist nun schon länger auf dem Markt. Wir unterhielten uns mit Tom Wickham-Jones (***TWJ***) von Wolfram, Inc. über die Software. Das Interview für den Rundbrief (***CAR***) führte Ulrich Kortenkamp.*

CAR: *To me it seems that a main goal of the new release of Mathematica was to make it better suited for industrial application, as the numeric computation part of Mathematica was enhanced. This seems to be a little bit unusual for a formerly mostly symbolic general purpose computer algebra system.*

TWJ: This is partly a question of perception, how people perceive Mathematica, and partly a question of definition, what Mathematica is. You perceive Mathematica as a computer algebra system, but other people may perceive it differently. In fact, many users would see it as a tool that can solve a wide variety of problems, and for that it requires a wide variety of functionality. The functionality certainly includes computer algebra, but it also includes much more. Over the past ten years we have been steadily developing the numerical computation parts of Mathematica. In part, this is a response to requests from users of Mathematica, but also it is because we felt that a system which integrates symbolic and numerical computation would be genuinely power-

ful (and interesting to work on). We also believed that building numerical and symbolic functionality as part of an integrated library would be more constructive than joining two separate libraries. This is a major direction in which our computation development has been moving for a long time.

I have been very pleased with and proud of Mathematica 5, because it brings many of these ideas to fruition. For a long time I have felt we could build such an application, and to actually succeed is very satisfying.

I could make other points, for example, to point out that traditional numerical computation has always made use of symbolic techniques, though typically when it does they are not considered as symbolic techniques and are often described in language that hides the relation. Examples of this are graph theoretic approaches (the basis of direct sparse solvers and automatic differentiation. In Mathematica, we try to make this relation explicit.

For the question of definition, I do not think that Wolfram Research has ever claimed that Mathematica is a computer algebra system. We have given more general definitions. 'A system for doing mathematics by computer', was one early statement that I always liked. We certainly felt that it contained computer algebra, but from the very beginning worked to add other areas of functionality. In addition to numerical computation, we added graphics, a document interface, etc.

CAR: *How do the new ODE/PDE solvers compare to other, external, solvers? Do you take advantage of having the solver tightly integrated into a system capable of doing symbolic manipulations?*

TWJ: I think the new ODE/PDE solver is really quite unique in a number of important directions.

1) The collection of built-in ODE methods is very large. I do not believe that any other single system provides such a wide collection.

2) The system supports a plug-in mechanism. This is very important for people who want to develop new methods, allowing them to concentrate on the mathematics and not have to write code to process input, produce output, etc. This makes the development of new methods much more efficient because the framework is much more supportive–similar to the way that programming in Java is typically more efficient than programming in assembler.

3) The system provides strong support for nested methods, i.e., a method that relies on some underlying method. For example, a projection method uses a base method to take a step which is then corrected to maintain some structural property of the system. Now, projection methods certainly existed before Mathematica 5, but the ease with which you can work with them, for example trying different base methods, is quite significant.

4) Support for systems of equations is specified with vector or matrix input. That is, instead of writing all the equations explicitly, you can specify the equations as vectors or matrices. This leads to efficiencies in processing the equations (which can lead to computational efficiencies). It also gives great notational simplification to the user of the system. In addition, it is obviously useful if you actually want to use differential equation techniques for solving matrix problems.

In addition, there are the long-standing benefits of solving differential equations in Mathematica. In a way, these are so obvious that they get overlooked. These include the fact that you just enter differential equations to the solver as you would see them in a paper or in a book. In many systems you have to rewrite the equations manually, to a traditional form for a numerical solver. It also includes the fact that the output of the solver is not just a collection of numbers, it is a first-class Mathematica object, a representation of the function that satisfies the equation (implemented as a piecewise polynomial). This can be used in all other mathematical computations, e.g., it can be differentiated, used in an optimization equation or used to solve another ODE.

I'm not an expert on differential equations in Mathematica. More information can be found at `http://documents.wolfram.com/v5/Built-inFunctions/AdvancedDocumentation/DifferentialEquations/NDSolve/index.html`.

CAR: *For our readers it is of high interest that it is now possible to solve equations and inequalities symbolically over fields other than the reals. How do the methods compare to the methods used in earlier versions of Mathematica?*

TWJ: Mathematica 4 only solved systems of equations over complex numbers, and the only methods used were linear algebra for linear systems, Groebner bases for polynomial systems, and a few heuristics for transcendental systems. Mathematica 5 can solve systems of algebraic equations and inequalities over the reals using cylindrical algebraic decomposition (Mathematica 4 has an earlier version of CAD, which was hidden in the `Experimental'` context), can solve systems involving quantifiers, has algorithms and heuristics giving complete solutions of (some) transcendental systems over the complex and the real numbers, has 25 or so algorithms and heuristics for solving various types of Diophantine equations and inequalities, and can solve quantified linear and polynomial systems over integers modulo m using modular linear algebra and Groebner basis methods. (Mathematica 4 had some of the functionality using the `Mode` → `Modular` option, but it was a rather awkward syntax, and not always reliable.)

CAR: *Sophisticated pattern matching is crucial for almost everything in Mathematica. How does "A New Kind of Science", the book of Stephen Wolfram, influence the further development of the Mathematica kernel?*

TWJ: The relationship between NKS and Mathematica is quite interesting. In one sense NKS is a large application of Mathematica. It might not have been possible without Mathematica, but its work gives us many ideas for extending and developing Mathematica to ma-

ke it suitable for such a project. In addition, we have been taking technology developed as part of NKS and adding this to Mathematica. For example, we have added an optimized cellular automaton function, and this might be useful for running computations found in NKS.

CAR: *The user interface of Mathematica has not changed for a long time. Of course, Mathematica Notebooks are quite powerful, but they lack some new, modern interactive features. We have seen customizable user interfaces for Maple with the introduction of Maplets. Do you plan to incorporate user interface changes in Mathematica 6?*

TWJ: This is something on which we are working very hard. For many years Mathematica users have been building extended user interfaces with Java using our J/Link interface (which uses Java reflection to give a very tight integration with Mathematica). Recently, we released a beta version of GUIKit, a toolkit which sits on top of J/Link. This enormously simplifies the process of building user interfaces in Mathematica. See `http://www.wolfram.com/solutions/mathlink/guikit/`.

This supports many features, such as an extensible component widget system, automatic layout, support for modal and non-modal interactions, and a scripting framework. In addition, since this combines with J/Link for full interaction and access to Java classes and objects, the whole kit is very flexible and powerful and can be used for building serious applications.

I could talk about it for a long time, but just the GUIKit layout manager is really very sophisticated (it uses a custom layout manager that requires very little intervention from the user).

I do not think that Maplets support many features such as these.

CAR: *What about OpenMath support? Is there a chance for easy data and formula exchange between Mathematica and other Computer Algebra systems in the future?*

TWJ: I get questions about this from time to time. My general answer is that I am not opposed to seeing development like this, and given the XML support in Mathematica it should not be that difficult. However, my personal view is that I am not certain how useful it can really be. Consequently, while we at Wolfram Research will not work on this ourselves, I would be very happy to see some other person work on it and would be willing to give some support to a venture like this. Perhaps one of your readers might be interested.

CAR: *Speaking of interactivity and Web integration. webMathematica is a server-side solution for providing computer algebra services. Do you think that the future lies in service-based computation? Wasn't it a big step forward to have processing power on the desktop instead of having jobs processed at the computer center?*

TWJ: It clearly was a big step to have large processing power on the desktop. Of course, if your client is a PDA or mobile phone, it may still lack processing power. However, lack of processing power on the client is not the only motivation towards server computation. Running on the server takes care of distribution, configuration and security problems, and it allows updates to software to be applied immediately. It also allows applications to study how users really use them, and provides a convenient way to collect feedback from users.

Thus I think the future really has a role for server computation. Perhaps this is part of a distributed computation system. This might be driven by classical web-based HTML interfaces, but also part of a general Web Service framework. This is also something on which we have been working. See `http://www.wolfram.com/solutions/mathlink/webservices/`.

CAR: *Grid computing and supercomputing has become affordable even for smaller institutions, as clusters of PCs running Linux, or the new Virginia Tech Supercomputer, built of 1.100 Apple G5 machines, show. The new gridMathematica supports parallel distributed computing with Mathematica 5. How does the Parallel Computing Toolkit compare to existing technologies, like PVM? Does the kernel parallelize tasks automatically, say, with the new ODE/PDE solvers?*

TWJ: At present the kernel does not parallelize tasks automatically. This is something that we are working to try and improve in addition to using some of our new technology to improve the scheduling, etc., of parallel tasks. gridMathematica is somewhat related to existing parallel systems such as PVM or MPI. It differs in that it is strongly connected to Mathematica. Thus for developing parallel solutions, many of the features that make Mathematica interesting, such as its large collection of mathematical functionality and support for rapid prototyping of a solution, would make gridMathematica attractive.

CAR: *When will the G5-optimized version of Mathematica be ready?*

TWJ: We are working very hard on this. We want to make sure that it really benefits from all the specialized features of the G5. Making sure that all the dedicated libraries that Mathematica uses are fully optimized is quite a lot of work. I don't want to give a definite date, because I usually do not give definite dates.

CAR: *What about students and schools? Other companies seem to push their products more aggressively into these markets. Mathematica is, at least in Germany, still considered as too expensive to be used in schools. MuPAD or Maple seem to be more interested in schools, and also more successful. Is Mathematica just being considered a PProproduct for industry and research?*

TWJ: This is a detailed question, one that I am not very qualified to answer, but I will try.

Wolfram Research is very interested in the high school market for which we have a number of products and purchase schemes specifically targeted. As well as Mathematica for the Classroom, which is a full version of Mathematica, we also offer The Mathematica Tea-

cher's Edition which is specifically customized for high school use, and CalculationCenter which is a version of Mathematica optimized for easier use on less demanding applications. As far as I am aware, MuPad and Maple do not offer schools anything other than their standard product.

Perhaps this is a question of perception. Maybe our marketing has not been as successful in Germany as it has in other countries.

CAR: *Finally, what is the most important new feature in your eyes?*

TWJ: I am very pleased with our support for sparse matrices. This is something that we had debated and worked on for many many years. We knew what the functionality should do–solving systems, carry out multiplication, etc.–but to design the interface to the functionality to be correct is not so trivial. Sometimes people think that we spend too much time worrying about these design issues, but this is often a very narrow view which is mainly relevant if you just want to solve one class of problem. In the long term, designing a good interface is very worthwhile, especially for a very general system.

Sparse matrices are of course a key technology that supports many areas of computation, especially if you want to solve very large computation problems.

If I could mention another new feature I would also mention a meta-feature, our advanced documentation system that we are starting to develop. This is an adjunct to TThe Mathematica Booktthat provides a much more in-depth description of the functionality and how to use it. Users of the Mathematica system have been requesting this for many years. We will be adding to this as we develop the system. For an example, see `http://documents.wolfram.com/v5/Built-inFunctions/AdvancedDocumentation/LinearAlgebra/`.

CAR: *Tom, thanks a lot for your answers!*

Kurzmitteilungen

Umstieg mit Problemen (Thomas Hahn)

Mathematica 5 wurde angepriesen als „an advanced algorithm release with a large number of major new technologies." Leider hat das nicht nur positive Seiten, so geht neuerdings vieles anders oder fehlerhaft. Der Umstieg von Mathematica 3 auf 4 war jedenfalls deutlich schmerzloser. Im Folgenden sollen an wenigen Beispielen Schwierigkeiten beim Wechsel illustriert werden.

Im einfachsten Fall haben Optionen andere Defaults. So hat `Eigenvalues` neuerdings die Optionen `Cubics -> False` und `Quartics -> False`, und der Benutzer wundert sich, dass in den Eigenwerten plötzlich Objekte der Form `Root[f, k]` auftreten, die die Nullstellen von Polynomen darstellen.

Schwerwiegender sind da schon Fehler in der symbolischen Umformung, so hat z. B. `Apart` einen Vorzeichenfehler, demzufolge `Simplify[expr - Apart[expr]]` nicht immer Null ergibt. Ein nicht einmal besonders langer Ausdruck (LeafCount 597), für den dieses Problem auftritt, ist hier nicht abgedruckt, aber auf Anfrage vom Autor zu erfahren. Kommentar vom Wolfram-Support: „This error has been fixed for the internal development version of Mathematica. We have not yet determined when this correction might become available in a released version of Mathematica."

Aber auch die Numerik hat noch ihre Schwierigkeiten: `PolyLog[2, 0.540032735091924645152 1 - 0.77755820666184486045961 I]` liefert so z. B. zunächst „Recursion limit exceeded" und, wenn man selbiges hochsetzt, `0.10^2 + 0.10^2 I`, während die Variante mit machine precision ganz anstandslos `0.3584 - 0.953611 I` ausspuckt.

Selbst einfache Funktionen verwundern. Legt man für ein Symbol mehr als 17 Definitionen an, im einfachsten Fall etwa mit `Array[(f[#] := g)&, 18]`, so liefert `Definition[f]` bzw. `??f` eine Liste, in der `Set` (`=`) statt `SetDelayed` (`:=`) steht.

Wer seinen Upgrade noch bis zum nächsten Release hinauszögern kann, wird sicherlich viele Bugs ausgemerzt finden. Insgesamt hält sich die Begeisterung in Grenzen, 1250 Bucks pro Upgrade einer Netzwerk-Lizenz für viele Bugs gezahlt zu haben.

Und wenn jemand nun zu seiner alten Version 4 zurückkehren will (lizenzrechtliche Fragen seien hier einmal ausgeklammert): man muss unter Linux auch wieder den alten License Manager installieren, denn Linux-Lizenzen sind nunmehr in Klasse A (vorher Klasse X).

Computeralgebra in der Schule

Niedersachsen – Zentralabitur mit CAS

Heiko Knechtel (Bückeburg)

In Niedersachsen werden ab dem Jahr 2006 die Abiturprüfungen mit zentralen Aufgabenstellungen durchgeführt. Niedersachsen schließt sich damit den Forderungen der KMK nach zentralen Überprüfungen von Bildungsstandards an. Niedersachsen hat im Fach Mathematik bereits vor fast zehn Jahren begonnen, im Unterricht und in Abiturprüfungen Computeralgebrasysteme zu integrieren, und hat damit in vielen Bereichen auch die Unterrichtskultur geprägt. Z.Zt. werden dort in sehr vielen Leistungskursen schon Computeralgebrasysteme überwiegend auf kleinen leistungsfähigen Taschencomputern wie TI 92 Plus, TI Voyage 200 oder Casio Classpad für den Mathematikunterricht genutzt. Damit wurde der Weg „weg von klassischen Kurvendiskussionen hin zu kontextbezogenen Problemlösestrategien“ geebnet. Diesem Trend stehen zentrale Prüfungsaufgaben naturgemäß zum Teil entgegen, da der Ruf nach mehr Outputsteuerung in der Bildung naturgemäß zu mehr Breite statt Tiefe in den Bildungsinhalten führt. Damit kam schnell die Befürchtung auf, dass Computeralgebrasysteme wieder aus dem Unterricht verbannt werden könnten. In Niedersachsen hat man sich aber für einen anderen Weg entschieden. Der zielgerichtete Einsatz von Technologie soll unterstützt werden und auch im Zentralabitur berücksichtigt werden. Schon in den Klassenstufen 7 - 10 sollen die Schülerinnen und Schüler neben grafischen Taschenrechnern als Standardwerkzeug Computeralgebrasysteme als didaktisches und methodisches Werkzeug in längeren Unterrichtssequenzen kennen lernen. Dieses wird explizit in den neuen *Rahmenrichtlinien Mathematik für die Klassen 7-10* am Gymnasium gefordert. Im Abitur ist der Einsatz von entsprechenden Werkzeugen wie Computeralgebrasystemen oder Grafikrechnern auch zukünftig erlaubt, wenn sie vorher im Mathematikunterricht integriert wurden. Die Schule muss lediglich vorher mitteilen, dass sie diese Technologie im Unterricht und in den Prüfungen einsetzt. Dann erhält sie am Prüfungstag die auf ihre Technologie abgestimmten Aufgaben. Erreicht wird dieses dadurch, dass zu den Prüfungsthemen technologieabhängige Vertiefungen gefordert werden, die sich von Jahr zu Jahr ändern. Im ersten Prüfungsjahrgang werden z.B. im Bereich Analysis folgende Vertiefungen erwartet:
Grundkurs: Beschreibung und Analyse von Wachstum (auch unter dem Aspekt der Änderungsrate)
Leistungskurs: Modellierung von Trassierungen; Krümmungsfunktion; Volumina.
Bei der Nutzung von einfachen technisch-wissenschaftlichen Taschenrechnern treten an diese Stelle andere Vertiefungen. Darüber hinaus sind die Bewertungen der einzelnen Leistungen natürlich von der gewählten Technologieklasse abhängig. Dieses Verfahren ist zwar für die entsprechende Auswahlkommission sehr aufwendig, da es eine Vielzahl unterschiedlicher Aufgaben und Bewertungsschemata erfordert, aber es sichert andererseits die Möglichkeiten den vor vielen Jahren eingeschlagenen Weg fortzusetzen. Wenn in wenigen Jahren alle Schüler der Sekundarstufe II mindestens einen Grafikrechner zur Verfügung haben werden, kann die Vielzahl der Aufgaben deutlich reduziert werden. Dann werden sich die Aufgaben nur noch in Abhängigkeit von der Technologie im Bearbeitungsumfang und in der „Untersuchungstiefe“ unterscheiden.

Computeralgebra in der Lehre

Umfrage zum Einsatz von Computeralgebra in der Lehre an den mathematischen Fachbereichen der deutschen Hochschulen

Hans-Wolfgang Henn (Dortmund)

Dieser Bericht ist das Resümee einer Fragebogenaktion, mit der wir uns im Frühjahr 2003 an die mathematischen Fachbereiche aller deutschen Universitäten gewandt hatten. Die Anfrage ging mit einem erläuternden Begleitschreiben von Herrn Koepf möglichst an uns bekannte Kontaktpersonen, sonst an die Dekane

der Fachbereiche. Im auszufüllenden Fragebogen, der auf unserer Homepage noch verfügbar ist, wurde nach den vorhandenen Computeralgebrasystemen, nach deren Einsatz in mathematischen Vorlesungen, nach speziellen Vorlesungen zum Thema CAS und nach speziellen fachdidaktischen Veranstaltungen zum Thema CAS gefragt. Es gingen von 36 Hochschulen Antwortbögen ein, dabei schickten vier Fachbereiche je zwei Antwortbögen, von der Universität Bonn bekam ich sogar vier Antwortbögen zugesandt. In einem zweiten Durchgang wurde im Oktober 2003 an den Universitäten, von denen noch keine Antwort vorlag, eine Person gezielt angeschrieben. Diese zweite Runde ergab jedoch nur acht neue Antwortbögen; von 24 Universitäten lieg nach wie vor keine Information vor. Daher sind die folgenden Ergebnisse notwendigerweise unvollständig. Oft zeigen die Antwortbogen eher die subjektive Sicht des Antwortenden bezüglich seines persönlichen Umfelds als die Situation am Fachbereich; viele Antwortbögen sind unvollständig ausgefüllt, so dass die Aussagekraft nicht besonders stark ist.

Zunächst sollen einige Daten aus den vorliegenden Fragebogen zusammengefasst werden.

In allen Fachbereichen, die geantwortet haben, werden Computeralgebrasysteme verwendet. Dabei liegen für die hauptsächlich verwendeten Systeme meistens Campuslizenzen, in jedem Fall aber ausreichend viele Lizenzen vor. Bis auf drei Universitäten wird stets Maple genannt, mehrfach als einziges System. Mathematica ist das am zweithäufigsten genannt System, dreimal wird dieses CAS als einziges erwähnt. Den dritten Platz belegt Derive, das in etwa einem Drittel der Antworten erwähnt wird, einmal sogar ausschließlich. Naturgemäß wird Derive (einige Male auch in Form des Derive-Taschencomputers TI92) hauptsächlich in der Lehrerausbildung eingesetzt. MuPad wird nur fünfmal genannt. Es folgt das weite Spektrum der Spezial-Computeralgebrasysteme, die eher selten genannt werden und sich an einigen Hochschulen konzentrieren; beispielsweise werden in den Fragebögen von Bayreuth, Darmstadt, Erlangen und Heidelberg sechs bis acht Systeme aufgeführt.

Die Intensität des Einsatzes von Computeralgebrasystemen in fachmathematischen Veranstaltungen ist sehr unterschiedlich. Er dient meistens in Vorlesungen und bungen an der einen oder anderen Stelle zur Visualisierung und für komplexere Beispiele. Der Einsatz ist eher sporadisch; von einem durchgängigen Einsatz wird kaum berichtet. Oft werden die Veranstaltungen Analysis und Lineare Algebra aus dem Grundstudium genannt. Der Einsatz von CAS in weiteren Veranstaltungen streut stark, da die Antworten von der Einzelperson abhängen. Genannt werden z. B. Numerik, Stochastik, Zahlentheorie, Algebra, Differentialgleichungen und Kryptographie. Eine Nennung heißt aber nur, dass der antwortende Dozent in seiner entsprechenden Vorlesung ein CAS verwendet.

Spezialvorlesungen für Computeralgebra werden an etwa 40% der antwortenden Fachbereiche angeboten; hierbei handelt es zum Teil um Veranstaltungen zum mathematischen Nutzung eines CAS, zum Teil um Vorlesungen über die Algorithmen und mathematischen Grundlagen von CAS. Generell besteht bei den Studierenden mehr Nachfrage nach Einführungskursen in ein konkretes CAS als an der Vermittlung theoretischer Hintergründe der Computeralgebra.

In etwa 70% der Fragebögen wird über den Einsatz von Computeralgebrasystemen in fachdidaktischen Veranstaltungen berichtet. Ein „nein“ im Fragebogen liegt zum Teil auch daran, dass an der entsprechenden Institution keine Lehrer ausgebildet werden. Ansonsten ist die Intensität der Beschäftigung mit CAS äußerst unterschiedlich. Manchmal werden CAS an der einen oder anderen Stelle in fachdidaktischen Veranstaltungen thematisiert, manchmal werden spezielle Einführungskurse für ein CAS für Lehramtsstudierende genannt. Es gibt aber auch Veranstaltungen, die ganz fachdidaktischen Fragen des CAS-Einsatzes gewidmet sind. Leider wird oft berichtet, dass Lehramtsstudierende eher wenig an Erfahrungen mit CAS interessiert sind. Teilweise liegt dies an den Studienordnungen. Zukünftige Gymnasiallehrer haben zwei Fächer und in vielen Bundesländern einen relativ großen Anteil im Fach Erziehungswissenschaften zu studieren. Der Stundenanteil für die Mathematik ist daher eher bescheiden und weitgehend festgelegt. Wenn Veranstaltungen aus dem Bereich Computeralgebra nicht dem verbindlichen Kanon angehören, so werden sie auch nicht gewählt. Abhilfe könnte folgendes Verfahren sein, das an einigen Hochschulen praktiziert wird: Zu Beginn des Studiums müssen die Studierenden ein Softwarepraktikum belegen, bei dem sie u. A. in das an der Hochschule übliche CAS eingeführt werden. In den Anfängervorlesungen werden an geeigneten Stellen CAS zur Visualisierung und zum Ausführen von Algorithmen eingesetzt; in den Übungen sind gewisse Aufgaben mit CAS zu bearbeiten. In den fachdidaktischen Veranstaltungen im Hauptstudium ist dann die fachliche Kompetenz vorhanden, um fachdidaktische Kompetenz über den CAS-Einsatz in der Schule erwerben zu können. Diese Kompetenz ist für künftige Lehrerinnen und Lehrer unbedingt notwendig, da Computeralgebrasysteme zumindest in Form der kleinen CAS-Rechner immer mehr und teilweise schon flächendeckend in die Schulen drängen. Die Fachgruppe plant, demnächst Empfehlungen zur CAS-Ausbildung von zukünftigen Lehrerinnen und Lehrer zu verabschieden.

Zum Tod von Richard Dimick Jenks – dem Entwickler von Scratchpad II/AXIOM und einem Vorkämpfer der Computeralgebra

Am 30. Dezember 2003 verstarb im Alter von 66 Jahren Richard D. Jenks nach langer, schwerer Krankheit. Jenks wurde 1966 an der University of Illinois at Urbana-Champaign in Mathematik mit einer Arbeit zum Thema „Quadratic Differential Systems for Mathematical Models“ promoviert. Nach einer zweijährigen Tätigkeit am Brookhaven National Laboratory in Long Island schloss er sich der IBM an. In seinen ersten Jahren war er dort mit der Entwicklung eines ersten Computeralgebrasystems Scratchpad beschäftigt [3]. Nach einem Aufenthalt in den frühen siebziger Jahren an der Utah University bei Anthony C. Hearn, dem Reduce-Entwickler, wurden die späteren objektorientierten Konzepte von Scratchpad II/AXIOM erstmalig in einem experimentellen System Mode-Reduce erprobt. Bis zu seinem Ruhestand im Jahr 2002 blieb Jenks bei IBM Research in Yorktown Heights. Viele Jahre leitete er dort das Computeralgebra-Team. Er ermöglichte unzählige Gastaufenthalte von Computeralgebraikern aus der ganzen Welt. Bei der Konzeption und der Entwicklung von Scratchpad II/AXIOM war er maßgeblicher Ideengeber und Chefentwickler.

Richard Dimick Jenks

Die Ideen der `Categories` und `Domains` als Datentypkonstruktoren ermöglichte eine sehr nahe Realisierung mathematischer Konzepte in mathematischen Rechenstrukturen in AXIOM. Ein Beispiel ist der Datentyp `R := Fraction Polynomial Integer`, durch den zunächst der Polynomring in beliebig vielen Unbestimmten über den ganzen Zahlen und dann der Quotientenkörper darüber konstruiert wird. Der entscheidende Punkt hierbei ist, dass beispielsweise ein Datentypkonstruktor `Fraction` einen Integritätsring als Parameter erwartet, der als `Category` in AXIOM realisiert ist. Damit kann man sich bei der Implementierung der Arithmethik auschließlich auf die generisch lediglich als Signaturen definierten Operationen dieser `Category` stützen und verwendet dann bei der Ausführung ausschließlich die vom konkret übergebenen `Domain` geerbten konkreten Implementierungen dieser Signaturen. Damit ist der Code für die Arithmetik der rationalen Zahlen der gleiche wie für rationale Funktionen; man hat nur den jeweils geeigneten Integritätsring als Parameter des Datentypkonstruktors `Fraction` zu wählen. Der so konstruierte Ring `R` kann dann weiter benutzt werden, um etwa den Datentyp `S:=SquareMatrix(3, R)` – 3×3-Matrizen mit rationalen Funktionen als Komponenten – zu realisieren. Da dieser selbst wieder ein Monoid bzgl. der Multiplikation darstellt und in AXIOM auch ein Mitglied der gleichnamigen `Category` in AXIOM ist, kann dieser Datentyp benutzt werden, um formale Linearkombinationen von solchen Matrizen als Elemente des Monoidrings `MonoidRing(R,S)` zu realisieren. Der gleiche Datentypkonstruktor `MonoidRing`, nun aber angewendet auf den Ring `Integer` und das Monoid `FreeMonoid Symbol` liefert en passant den Polynomring in nichtkommutierenden Unbestimmten über den ganzen Zahlen.

Eine solche weitgehende, konsequente und dennoch performante Umsetzung mathematischer Strukturen auf dem Computer ist bis heute in keinem der Systeme, die sich kommerziell durchgesetzt haben, erreicht. Am nächsten kommt noch das System MAGMA. AXIOM ist inzwischen als freie Software auf `http://savannah.nongnu.org/projects/axiom` erhältlich. (Eine Auswahl von Referenzen zu Scratchpad und AXIOM: [16], [9], [6], [10], [12], [13].)

Richard D. Jenks war Vorsitzender (chair person) von ACM SIGSAM, Mitglied im ISSAC Steering Committee und leitete zusammen mit J. A. van Hulzen die EUROSAM’84 [15], eine Vorläuferkonferenz der heutigen ISSAC-Konferenzreihe. Zusammen mit David Chudnovsky organisierte er die großen und erfolgreichen Konferenzen „Computers and Mathematics“ in New York [11], 1986 Stanford und 1989 im MIT in

Boston [7]. Ein weiterer Höhepunkt war das unter seiner Leitung von IBM Europe duchgeführte Institute on Symbolic Mathematical Computation in Oberlech in Österreich.

In Erinnerung an Richard D. Jenks wurde ein *Jenks Computer Algebra Prize* für außergewöhnliche Software Engineering-Beiträge in Computeralgebra ausgelobt. Vorschläge dazu sind bis zum 01.05.2004 an *Prof. B. F. Caviness, Dept. of Computer & Information Sciences, 103 Smith Hall, University of Delaware* zu richten. Weitere Informationen dazu auf `http://www.cis.udel.edu/~caviness/callForNominees.html`.

Literatur

[1] Jenks, R. D.: META/PLUS – The Syntax Extension Facility for SCRATCHPAD. Proceedings of the IFIP Congress, Ljubljana, 1971, Vol. 1, North Holland, 382-384.

[2] Jenks, R. D. (Ed.): SYMSAC '76, Proceedings of the 1976 ACM Symposium on Symbolic and Algebraic Computation (Yorktown Heights, N. Y., 1976), Assoc. Comput. Mach., New York, 1976.

[3] Griesmer, J. H., Jenks, R. D., Yun, D. Y. Y.: SCRATCHPAD User's Manual, IBM Research Publication RA70, 1975.

[4] Jenks, R. D.: MODLISP – an introduction. Proceedings of EUROSAM Marseille, June 1979. Ng, E. N. (Ed.), 466-480.

[5] Davenport, J. H., Jenks, R. D.: MODLISP. Proceedings of the LISP Conference 1980 in Stanford, California. ACM 65-74.

[6] Jenks, R. D., Sutor, R. S., Watt, St. M.: Scratchpad II: An Abstract Datatype System for Mathematical Computation. In Janßen, R. (Ed.), Trends in Computer Algebra, Proceedings of an International Symposium, Bad Neuenahr, May 1987, Lecture Notes in Computer Science, Vol. 296, Springer-Verlag, 12-37.

[7] Kaltofen, E., Watt, S.M., (Ed.): Computer and Mathematics, Springer-Verlag, New York 1989.

[8] Chudnovsky, D. V., Jenks, R. D. (Ed.): Computer Algebra. Marcel Dekker, Inc., 1989.

[9] Lambe, Larry A.: Scratchpad II as a tool for mathematical research. Notices of the Amer. Math. Soc. 36-2, 1989, 141-148.

[10] Davenport, J. H., Trager, B. M.: Scratchpad's view of algebra I: Basic commutative algebra.Technical Report 90–31, School of Mathematical Sciences, University of Bath, Bath, England, January 1990.

[11] Chudnovsky, D. V., Jenks, R. D. (Ed.): Proceedings of Computers in Mathematics. Lecture Notes in Pure and Applied Mathematics, Vol. 125, Marcel Dekker, Inc., 1990.

[12] Jenks, R. D., Sutor, R. S.: AXIOM – The Scientific Computation System. Springer-Verlag, New York and NAG, Ltd. Oxford, 1992.

[13] Lambe, Larry A.: The AXIOM system. Notices of the Amer. Math. Soc. 41-1, 1994, 14-18.

[14] Jenks, R. D., Trager, B. M.: How to Make AXIOM into a Scratchpad. ISSAC 1994, International Symposium on Symbolic and Algebraic Computation, Oxford, 1994, Giessbrecht, M. (Ed.), Proceedings ACM SIGSAM, 32-40.

[15] Fitch, J. (Ed.): EUROSAM 84, International Symposium on Symbolic and Algebraic Computation, Cambridge, England, July 9-11, 1984 Proceedings. Lecture Notes in Computer Science 174 Springer-Verlag, 1984.

[16] Jenks, R. D: A Primer: 11 Keys to New Scratchpad. EUROSAM 1984, 123-147.

[17] Sutor, R. S., Jenks, R. D.: The type inference and coercion facilities in the Scratchpad II interpreter. Proceedings of the Symposium on Interpreters and Interpretive Techniques, 1987, St. Paul Minnesota, USA, 1987, ACM SIGSAM, 56-63.

[18] Jenks, R. D.: 1962-1992: The First 30 Years of Symbolic Mathematical Programming Systems. In Janßen, R. (Ed.), Trends in Computer Algebra, Proceedings of an International Symposium, Bad Neuenahr, May 1987, Lecture Notes in Computer Science, Vol. 296

Neue Sektion „Computational Algebra“ des *Journal of Algebra*

Das *Journal of Algebra* plant eine neue Sektion „Computational Algebra“, die der konstruktiven und algorithmischen Algebra gewidmet ist. Ein „Statement of Purpose“ ist unten angehängt.

Als zuständige Herausgeber konnten Jon Carlson, Henri Cohen, John Cremona, Patrick Dehornoy, Harm Derksen, Meinolf Geck, Gerhard Hiß, Derek Holt, William Kantor, Reinhard Laubenbacher, Gunter Malle, Eamonn O'Brien, Bruno Salvy und Jean-Yves Thibon gewonnen werden.

Zunächst sollen einige Ausgaben des *Journal of Algebra* als „Special Issues“ ausschließlich Artikeln der „Computational Algebra“ vorbehalten sein. Geeignete Arbeiten können ab sofort an die folgende Adresse eingereicht werden.

Journal of Algebra/ Section on Computational Algebra
Institut Henri Poincaré
11 rue Pierre et Marie Curie
F-75005 Paris, France
e-mail: `jalgebra@ihp.jussieu.fr`

Statement of Purpose Constructive or computational methods have always been a characteristic feature of algebra. With the introduction of algebraic structures in the 19th century, non-constructive methods came into play, and for some periods in the first half of the 20th century they dominated the constructive methods. The rapid development of computer technology in the second half of that century led to a revival of constructive methods to investigate algebraic structures. This shift is reflected in the increasing number of papers submitted to the Journal of Algebra which make essential use of computer calculations or describe algorithms for computer calculations. To provide an appropriate forum for such contributions and to broaden the scope of the Journal we have introduced this new section.

An important general criterion for the publication of a paper in the new section will be its emphasis on constructive aspects in the creation or development of a theory, or the solution of a problem.

The following kinds of contributions are particularly welcome in the new section of the Journal of Algebra: Papers in which constructional (computational) methods are essential to obtain the results, papers presenting results obtained by computer calculations (to be suitable for publication such results must represent an advance of mathematics, and exhibit new methods and mathematical conclusions), papers that use computational methods to classify specific algebraic structures (in the form of tables, if appropriate), descriptions and outcomes of experiments that put forward new conjectures, support existing conjectures, or give counterexamples to existing conjectures, description and analysis of new algorithms, improvements and extensions of existing algorithms, or description of other computational methods, including practical experiments and heuristic arguments.

Contributions are welcome from all areas of *mathematics*, if the emphasis is on algebraic aspects. Contributions describing applications of algebraic results or methods, for example in *coding theory*, *cryptography*, or the algebraic theory of *differential equations*, are highly welcome.

The contributions will in general be available in print, as well as in electronic form through *ScienceDirect*©. The electronic version may contain additional material such as extensive tables or animated pictures.

Informationen über freie Stellen

Am RISC (Research Institute for Symbolic Computation) der Johannes Kepler Universität in Linz (Oberösterreich) beschäftigt sich eine Gruppe von derzeit etwa zehn Personen (Universitätsmitarbeiter, Postdocs, Studenten) mit dem Theorema-Projekt. Ziel dieses Projekts ist der Entwurf und die Implementierung eines Softwaresystems zur Unterstützung aller Phasen von mathematischer Forschung. Für dieses Projekt werden noch Mitarbeiter gesucht. Einige Stellen für wissenschaftliche Assistenten mit einer Laufzeit von drei bis vier Jahren sind noch zu besetzen. Details hierzu finden Interessenten auf den Homepages der Institutionen, `http://www.risc.uni-linz.ac.at` und `http://www.theorema.org`.

Mathematica beim Forum der Lehre 2004 am 02.04.2004 in Deggendorf

Das Zentrum für Hochschuldidaktik der bayerischen Fachhochschulen (DiZ) führt sein jährliches Forum der Lehre 2004 an der FH Deggendorf durch. Auf der Tagesordnung steht auch ein „Erlebnisraum“ mit dem Titel „Die ganze Welt ist Mathematica“, die von Prof. Dr. Niall Palfreyman von der FH Weihenstephan gestaltet wird. Weitere Informationen und Anmeldung unter `www.diz-bayern.de`.

Publikationen über Computeralgebra

- Buhler, J. P., Stevenhagen, P. (Eds.), *Algorithmic Number Theory*, Cambridge University Press, Cambridge, 2002, ISBN 0-521-80854-5, € 50,00.
- Justesen, J., Hoholdt, P. T., *A Course in Error-Correcting Codes*, American Mathematical Society, 2004, 192 Seiten, ISBN 3-03719-001-9.
- Green, D. J., *Gröbner Bases and the Computation of Group Cohomology*, Springer Verlag, Berlin, Heidelberg, New York, 2003, 138 Seiten, ISBN 3-540-20339-7, € 29,95.
- Joswig, M., Takayama, N., *Algebra, Geometry and Software Systems*, Springer Verlag, Berlin, Heidelberg, New York, 2003, 330 Seiten, ISBN 3-540-00256-1, € 74,85.
- Ling, S., Xing, C., *Coding theory - A first course*, Cambridge University Press, Cambridge, 2004, 234 Seiten, ISBN 0-521-52923-9, € 39,95.
- Pemmaraju, S., Skiena, S., *Computational Discrete Mathematics : Combinatorics and Graph Theory with Mathematica*, Cambridge University Press, Cambridge, 2003, 512 Seiten, ISBN 0-521-80686-0, $ 49,95.
- Schenck, H., *Computational Algebraic Geometry*, London Mathematical Society, 2003, ISBN 0-521-53650-2, $ 70,00.
- Winkler, F., Langer, U. (Eds.), *Symbolic and Numerical Scientific Computation*, Springer Verlag, Berlin, Heidelberg, New York, 2003, ? Seiten, ISBN 3-540-40554-2, € 39,95.

Besprechungen zu Büchern der Computeralgebra

H. Derksen, G. Kemper
Computational Invariant Theory

Springer Verlag, New York, Berlin, Heidelberg, 2002, ISBN 3-540-43476-3, 268 Seiten, € 90,00.

Dieses Buch behandelt Invariantentheorie, d. h. den Ring der Polynome, die unter einer Gruppe invariant sind. Wie der Titel des Buches schon sagt, liegt ein Schwerpunkt auf den algorithmischen Aspekten zur Berechnung von Erzeugendensystemen des Invariantenringes. Dadurch hebt es sich deutlich von theoretischen Büchern zu diesem Thema ab. Das Buch von B. Sturmfels von 1993 zur algorithmischen Invariantentheorie hat das Interesse an diesem Gebiet stark geprägt.

Die Themenauswahl des Buches ist gekennzeichnet durch die wissenschaftlichen Arbeiten der Autoren zu diesem Thema. Der Algorithmus zur Berechnung eines Erzeugendensystems für linear reduktive algebraische Gruppen hat die Fachwelt bei seiner Entdeckung überrascht und begeistert. Der Algorithmus basiert auf Ideen von Hilbert, für die Hilbert seinerzeit kritisiert wurde, weil sie nicht konstruktiv sind. Hundert Jahre später wissen wir es besser. Hoffentlich teilt jeder Leser meinen Enthusiasmus für diesen Algorithmus.

Ein weiterer Schwerpunkt liegt auf der modularen Invariantentheorie. Hierbei teilt die Charakteristik des Körpers die Gruppenordnung, was fatale Konsequenzen für die algebraischen Strukturen hat, die für Algorithmen für Körper der Charakteristik 0 verwendet werden. Gerade deshalb ist dieses Thema für Algebraiker so interessant.

Ein besonderer Vorteil dieses Buches ist das reichhaltige Kapitel über Anwendungen. Es gibt Abschnitte, die in die Grundideen und die Literatur zur Lösung von polynomiellen Gleichungssystemen, Graphentheorie, Kombinatorik, Kodierungstheorie, dynamische Systeme, Computer Vision einführen. Die potentielle Leserschaft des Buches beschränkt sich deshalb nicht nur auf Algebraiker, die auf dem Gebiet der Computeralgebra und der Invariantentheorie arbeiten, sondern sollte auch algebraisch (vor)gebildete Anwender ansprechen.

Wie die Autoren formulieren, wendet sich das Buch an Forscher im Gebiet der Geometrie, Computeralgebra und der Invariantentheorie. Es kann aber sicherlich auch für ein Seminar verwendet werden, das auf eine einführende Vorlesung zur Computeralgebra aufbaut.

Da die Algorithmen zur Berechnung von fundamentalen Invarianten Gröbnerbasen und algorithmische kommutative Algebra verwenden, behandelt das erste

Kapitel dieses Thema. Schon hier kann man bewundern, wie knapp, präzise und auf den Punkt genau die Autoren formulieren. Neben einer knappen Einführung in die Standardkonzepte rund um Gröbnerbasen ist der Algorithmus von de Jong zur Normalisierung enthalten.

Insgesamt gesehen ist dies ein sehr schönes Buch. Einige der Resultate sind nie zuvor in Buchform erschienen. Für weitere Ergebnisse, die nach dem Erscheinen dieses Buches erzielt wurden, siehe den Übersichtsartikel von Gregor Kemper auf Seite 7 in diesem Rundbrief. Den einzigen Nachteil, den ich an diesem Buch finden kann, ist folgender: Ich hätte das Buch gern einige Jahre früher zum Lesen gehabt.

Karin Gatermann (Berlin)

W. Lütkebohmert
Codierungstheorie

Vieweg Verlag, Braunschweig, Wiesbaden, 2002, ISBN 3-528-03197-2, € 29,90.

Vorliegendes Buch entstand aus einer einsemestrigen Vorlesung des Autors an der Universität Ulm. Sie war an Hörer mit Grundkenntnissen in Algebra gerichtet. Ziel der Vorlesung war es, den Studenten einen Rundblick über algebraisch konstruierte lineare Codes zu vermitteln. Das aus der Vorlesung mit demselben Ziel entstandene Buch wurde dann noch durch einige für die geometrischen Goppa-Codes notwendige Grundlagen aus der kommutativen Algebra und der Geometrie algebraischer Kurven angereichert.

Das Buch besteht aus drei Teilen. Der erste Teil mit den Kapiteln 1 bis 5 enthält die Elementare Codierungstheorie. Er umfasst allgemeine Standardresultate über lineare Codes inklusive der Behandlung einiger wichtiger Codeklassen wie Hamming- und Golaycodes, zyklische Codes mit BCH-Codes sowie Reed-Solomon-Codes, eingerahmt von Codekonstruktionen wie Spreizung und Verkettung sowie den klassischen Schrankensätzen: Singleton-Schranke, Plotkin-Schranke, Hamming- und Eliasschranke sowie als untere Schranke die Gilbert-Varshamov-Schranke. Wohl auf Grund fehlender Vorkenntnisse der Hörer auf dem Gebiet der Elementaren Zahlentheorie wurde auf die Behandlung der Quadratische-Reste-Codes verzichtet.

Der zweite Teil des Buches umfasst die für die Behandlung der geometrischen Goppa-Codes notwendigen geometrischen Grundlagen und besteht aus Kapitel 6.1, 8 und 7. Dabei enthält Abschnitt 6.1 eine kurze Zusammenfassung (ohne Beweise) der für die Einführung der Goppa-Codes notwendigen Sätze wie Satz von Riemann-Roch, Residuensatz und Hurwitzsche Relativgeschlechtsformel. Kapitel 8 enthält die dazugehörigen Beweise und die Konstruktion singularitätenfreier Modelle von Kurven. In Kapitel 7 werden speziell Kurven über endlichen Körpern studiert und die Riemannsche Vermutung für die zugehörigen Zetafunktionen bewiesen, was schließlich auf die für das Studium der Goppa-Codes unverzichtbaren Schrankensätze für die Anzahl der rationalen Punkte führt.

Der dritte Teil, bestehend aus den Kapiteln 6 und 9, enthält die angestrebten Resultate über geometrische Goppa-Codes. Nach der allgemeinen Einführung konzentriert sich der Autor auf die Konstruktion langer Codes, zuvor behandelt er aber noch zum Vergleich die klassischen Goppa-Codes, die auch schon im Rahmen der elementaren Theorie hätten vorgestellt werden können. Darauf folgt – und das ist sicher ein Höhepunkt dieses Buches – eine geometrische Konstruktion der 1996 von Garcia und Stichtenoth gefundenen Artin-Schreier-Türme, deren Punktanzahl die Drinfeld-Vladut-Schranke erreichen. Dies führt schließlich zu einem elementaren Beweis des Satzes von Tsfasman-Vladut-Zink, der besagt, dass man mit geometrischen Goppa-Codes die Gilbert-Varshamov-Schranke übertreffen kann. In meiner Vorlesung Galoistheorie im vergangenen Wintersemester habe ich diesen Teil noch durch einige allgemeine Strukturaussagen über selbstduale Codes und Automorphismen sowie durch die Untersuchung spezieller Codeklassen wie elliptische und hermitesche Codes ergänzt. (Entsprechende Resultate sind in den Büchern von Tsfasman-Vladut: *Algebraic-Geometric Codes* bzw. Stepanov: *Codes on Algebraic Curves* zu finden.) Schließlich enthält das letzte Kapitel 9 noch die Standardalgorithmen zur Codierung und Decodierung geometrischer Goppa-Codes mit praktischen Hinweisen zur Implementierung.

Am Ende des Buches hat der Autor in zwei Anhängen noch einige benötigte Resultate aus der Kommutativen Algebra und der Algebraischen Geometrie zusammengestellt, die zumeist in weiterführenden Algebra-Vorlesungen behandelt werden.

Insgesamt ist das Buch als Grundlage für Vorlesungen und auch zum Selbststudium geeignet, wünschenswerte Ergänzungen sind in der Besprechung der einzelnen Teile angemerkt. Es ist ein geometrisches Pendant zu dem inzwischen über 10 Jahre alten bewährten Buch von Stichtenoth: Algebraic Function Fields and Codes, das geometrische Goppa-Codes in der zahlentheoretischen Sprache algebraischer Funktionskörper behandelt (und ganz auf die Elementare Codierungstheorie verzichtet, wofür aber z.B. das Büchlein von Willems: *Codierungstheorie* herangezogen werden kann). Ein Plus-

punkt der vorliegenden Ausgabe ist der auch für Studierende erschwingliche Preis. Bei den konkurrierenden Büchern von Tsfasman-Vladut und Stepanov vom Kluwer-Verlag stellt der Preis selbst für Fachbibliotheken gelegentlich ein Anschaffungshindernis dar.

Bernd Heinrich Matzat (Heidelberg)

A. Werner
Elliptische Kurven in der Kryptographie

Springer Verlag, New York, Berlin, Heidelberg, 2002, ISBN 3-540-42518-7, € 22,95.

Das vorliegende Buch entstand aus einer zweisemestrigen Vorlesung über Kryptographie und richtet sich an Mathematik- und Informatikstudenten ab dem fünften Semester. Der Text beginnt mit einem einleitenden Kapitel über RSA und das Problem des diskreten Logarithmus (DL). Im zweiten Abschnitt werden affine und projektive Kurven eingeführt und der Begriff der Singularität erklärt. Anschließend werden elliptische Kurven, deren Normalformen in verschiedenen Charakteristiken und die Addition von Punkten behandelt. Das dritte Kapitel umfaßt den Schoof-Algorithmus zur Bestimmung der Anzahl von Punkten elliptischer Kurven über endlichen Körpern sowie einen Abschnitt über supersinguläre elliptische Kurven. In den nächsten beiden Kapiteln geht es um allgemeine und spezielle Verfahren für die Lösung des DL-Problems sowie um praktische Konsequenzen. Dort findet sich außerdem die Behandlung digitaler Signaturen. In einem Anhang werden die benötigten Vorkenntnisse aus der Algebra und Zahlentheorie zusammengestellt.

Bereits im Vorwort wird darauf hingewiesen, daß gelegentlich Resultate ohne Beweis zitiert werden, um den Text für Studenten mit Grundkenntnissen in linearer Algebra und Algebra zugänglich zu machen. Diese Lücken finden sich vor allem im dritten und vierten Kapitel bei der Behandlung des Schoof-Algorithmus und der speziellen DL-Lösungsverfahren (für supersinguläre und anomale Kurven).

Wir haben im vergangenen Semester ein Proseminar zum Thema Kryptographie angeboten und dabei zunächst die Texte von Buchmann, *Einführung in die Kryptographie* und Werner als Literatur benutzt. Der Inhalt des einleitenden Kapitels und die allgemeinen Methoden zum DL-Problem sowie der Indexkalkül bilden die Schnittmenge der beiden Texte. Auf Wunsch der Studenten haben wir bei der Vorbereitung der Vorträge über elliptische Kurven andere Quellen hinzugezogen, um einige der Stellen zu ergänzen, an denen in dem vorliegenden Text nur Skizzen oder Ideen gegeben werden (soweit das im Rahmen eines Proseminares möglich war).

Meiner Meinung nach geht das Konzept der Autorin nicht ganz auf - wenn zu vieles nur oberflächlich behandelt werden kann, ist es schwierig, Verständnis für die Thematik zu gewinnen. Für eine Vorlesung (bei der den Studenten die Möglichkeit zum Nachfragen gegeben ist) eignet sich dieser Stil womöglich besser als für ein Buch, das auch zum Selbststudium genutzt werden soll.

Dennoch: Für alle, die zunächst nur einen Überblick suchen und sich an den oben genannten Lücken nicht stören, ist der Text durchaus nützlich, da die behandelte Aspekte ausführlich und leicht verständlich erklärt werden.

Julia Hartmann (Heidelberg)

Berichte von Konferenzen

1. **Introduction to Algebraic Control Theory: From finite to infinite-dimensional systems**

   Otzenhausen, 14.09. – 19.09.2003

   The annual summer school of the Graduiertenkolleg "Hierachie und Symmetrie in mathematischen Modellen" (RWTH Aachen) this year focused on algebraic aspects of control theory. The largest part of the participants consisted of PhD students from Aachen, mainly members of the Graduiertenkolleg. The rest were mathematicians from Aachen, Kaiserslautern, Portugal and France.

   Most of the participants are working in different fields of mathematics not directly related to control theory. Most of the participants, especially the PhD students, have been asked to prepare a one hour talk from the material provided by the experts for this purpose. The participants found appropriate assistance in the preparation of their talks. The invited experts contributed valuable remarks and were open to questions from the participants, before, during and after the school.

   The invited experts were (in alphabetical order): Jean-Jacques Loiseau (CNRS, IRCCyN, Nantes, France), Silviu-Iulian Niculescu (CNRS, HEUDIASYC, Compiègne, France) and Alban Quadrat (INRIA Sophia Antipolis, France).

   Also Eva Zerz (Kaiserslautern) who was one of the invited experts of last year's summer school, attended this year and

gave two talks on “Introduction to state space approach” and “An introduction to the behavioural approach to multidimensional systems”.

Every day of the five days of the summer school was centered around a different topic. The first two topics originated from engineering sciences. Jean-Jacques Loiseau, who has a strong engineering background, gave several talks about the historical emergence of these approaches in engineering sciences.

The last three days were concentrated on the algebraic topics in control theory. Linear control systems are viewed as modules over certain rings. The rings are dictated by the type and the module by the details of the control problem. Algebraic notions then start to dominate the study of such systems: torsion submodules, torsion freeness, reflexivity, projectivity. These properties are tested by usual constructions from homological algebra: Ext and Tor functors. Supplementary introduction to basic notions on module theory and commutative algebra was provided in several night sessions. The algebraic approach leads to concrete computational algorithms that were implemented in different packages by mathematicians from Aachen who gave an introduction to these packages in one of the night sessions.

Mohamed Barakat (Aachen)

2. Differential Equations and Galois Groups

Oberflockenbach, 02. – 04.10.2003

Die Arbeitsgruppe *Algorithmische Algebra* des IWR veranstaltete im Gästehaus der Universität Heidelberg in Oberflockenbach (Odenwald) im Rahmen des europäischen Netzwerks *Galois Theory and Effective Methods* (GTEM) einen Workshop über die Galois-Theorie von Differentialgleichungen mit 25 Teilnehmern. Da auch einige (angehende) Doktoranden teilnahmen, gab es mehrere einführende Vorträge. Das Programm war wie folgt: Marius van der Put, *Introduction to Differential Galois Theory*; Claudine Mitschi, *An Overview on Analytic Aspects of Differential Galois Theory*; Jacques-Arthur Weil, *Reduction mod p of Differential Systems in Practice*; Frits Beukers, *Introduction to Hypergeometric Functions in One Variable*; Michael Dettweiler, *Riemann-Hilbert Correspondence for Rigid Local Systems*; Werner M. Seiler, *Vessiot Theory of Partial Differential Equations*; Peter Müller, *Partial Differential Equations of Krull Dimension Zero*; Thomas Oberlies, *Embedding Problems and Inverse Problems*; Maint Berkenbosch, *Families of Differential Equations*; Felix Ulmer, *Liouvillian Solutions of Linear Differential Equations*; Julia Hartmann, *Hrushovski's Algorithm*; B. Heinrich Matzat, *p-adic versus Iterative Differential Equations*.

Werner M. Seiler (Heidelberg)

3. 6. Mitteldeutscher Computeralgebra-Tag

Halle, 10.10.2003

Mit der inzwischen sechsten Auflage des Mitteldeutschen Computeralgebra-Tags (MCAT) setzen wir eine Tradition fort, einmal im Jahr die über ganz Mitteldeutschland verstreuten Computeralgebraiker zu einem Tagesseminar zusammenzuführen. Traditionell stehen dabei Fachthemen, neue Entwicklungen bei den großen CAS und das Thema „CAS in der Schule“ auf der Tagesordnung. Die Vorbereitung lag wie immer in den Händen von H.-G. Gräbe (Uni Leipzig), P. Schenzel (Uni Halle) und T. Buchanan (FH Merseburg), die lokale Organisation bei den Hallenser Kollegen. In diesem Jahr wurden folgende Vorträge gehalten:

R. Achilles (Uni Bologna), Zu Schnittzahlen von Varietäten; S. Graubner (Ostwald-Gymnasium Leipzig), Über ein Problem aus der nichtlinearen Optimierung; C. Franke (FH Merseburg), Projektive Rekonstruktion mittels Invarianten von sechs Punkten aus drei unkalibrierten Bildern; S. Graubner (Ostwald-Gymnasium Leipzig), Neues in Maple 9; A. Brenner (Albert-Schweitzer-Gymnasium Erfurt), Probleme beim Problemlösen im Mathematikunterricht mit CAS; T. Pries (MLU Halle), Zur Parallelisierung der Faktorisierung mit ECM; H.-G. Gräbe (Uni Leipzig), Primes ist in P – Aktuelle Entwicklungen.

Im Vorfeld dieses CA-Tages wurde außerdem eine Mailingliste aufgesetzt, die diese Aktivitäten begleitet. Interessenten können sich unter `http://ais.informatik.uni-leipzig.de/mailman/listinfo/compalg` subskribieren.

Hans-Gert Gräbe (Leipzig)

4. ITMC 2003 – Innovative Teaching of Mathematics

Kyoto, Japan, 20. – 22.11. 2003

The conference enjoyed the hospitality of the Research Institute for Mathematical Sciences (RIMS), Kyoto University in the very beautiful autumn season. The historic Kyoto as the ancient capital of Japan was enjoyed by all participants. There were around 40-50 participants from the US, Germany, The Netherlands, Italy, Spain, Australia and most notably Japan. The attendants came from universities, schools and private companies.

The conference had a double theme:

1) For the first time ever Clifford geometric algebra for teaching was investigated in detail. M. Mori gave an enthusiastic greeting address. Mathematical foundations were explained by D. Hestenes (Arizona), H. Ishi and S. Gomyo (Yokohama), Y. Nagatomo (Fukuoka). C. Perwass (Kiel) and L. Dorst (Amsterdam) showed how to use well-tailored software (CLUCalc and GAViewer) for teaching geometric algebra. R. Gonzalez-Calvet (Barcelona) reported from his rich experience with geometric algebra summer schools for teachers. G. Sommer (Kiel) gave a wide-ranging survey of applications in robotic vision. S. Sato (Schlumberger) explained why he would like to see geometric algebra literature in Japanese.

2) The second focus was on new technology and software for teaching mathematics. H. Makishita (Tsukuba) introduced the use of Cabri to strengthen student anticipation and U. Kortenkamp (Berlin) introduced the interactive geometry software Cinderella. E. Brehm (Berlin) gave examples of visualizing algorithms using dynamic geometry (with Cinderella). E. Hitzer explained how to use geometric algebra and Cinderella for studying conic sections. H. Uno (Nara), responsible for Personal Digital Assistant (PDA) development at Sharp, introduced the latest state of PDA technology and the wide open opportunities to apply PDAs in learning. An example is the Cinderella implementation on a Zaurus Linux PDA. The audience expressed the hope for future PDA school hardware - robust and affordable for the average student.

Special time was dedicated to tutorials for teachers and to a free software demo session. This conference would not have been possible without the dedicated hard work of Prof.

R. Nagaoka (UAJ) and Dr. H. Ishi (Yokohama CU). One of their great achievements was to organize financial support. Strong financial support came also from the RIMS (tokubetsu keikaku) and Prof. Okamoto (RIMS). In this context special thanks goes also to Prof. Saito (RIMS). Many more people helped during the actual conference.

The full timetable with abstracts and related links is available at the conference homepage `http://sinai.mech.fukui-u.ac.jp/ITM2003/index.html`. The RIMS will soon print a proceedings volume, information on this will also be available from the conference homepage.

E. Hitzer (Fukui, Japan)

Hinweise auf Konferenzen

1. GAMM Jahrestagung

Dresden, 21. – 27.03.2004

Die Gesellschaft für Angewandte Mathematik und Mechanik e.V. (GAMM) lädt Sie ein zur Teilnahme an der Wissenschaftlichen Jahreskonferenz 2004 in Dresden vom 21. bis zum 27. März. Hauptvorträge: DAE methods for constrained and coupled differential equations in technical simulation (Martin Arnold, Martin-Luther-Universität Halle-Wittenberg), Inverse Probleme (Andreas Kirsch, Universität Karlsruhe), Sequential quadratic pogramming methods for the optimization of disributed parameter systems (Matthias Heinkenschloss, Rice University), Gebietszerlegungsmethoden, Parallelisierung, FEM-BEM-Kopplung, Präkonditionierung linearer Gleichungssysteme und Softwareentwicklung im Bereich FEM und BEM (Ulrich Langer, Johannes Kepler Universität Linz), On numerical stability in large scalelinear algebraic computations (Zdenek Strakos, Academy of Sciences of the Czech Republic), Macroscopic response of materials with multiwell energies (Antonio DeSimone, Max-Planck-Institut für Mathematik in den Naturwissenschaften), Herausforderung komplexe molekulare Systeme (Christof Schütte, Freie Universität Berlin).

Sektion: Computeralgebra und Computeranalysis

Sektionsleitung:
Walter Gander (Zürich), Karin Gatermann (Berlin)

Weitere Informationen:
`http://www.math.tu-dresden.de/gamm2004`

2. 9th Rhine Workshop on Computer Algebra

Nimwegen, Niederlande, 25. – 26.03.2004

This will be the ninth edition of a workshop initiated in Strasbourg in 1988 and held every second year since. To avoid competition with well-established conferences in the field, the workshop is kept as informal as possible. Its two main purposes are to offer an opportunity for newcomers in the field to present their work and to be a forum aimed at (but not restricted to) European researchers.

Organization:
Wieb Bosma (University of Nijmegen, `bosma@math.kun.nl`, Workshop Chair), Arjeh Cohen (University of Eindhoven, `amc@win.tue.nl`, Program Committee Chair

Topics:
The topics of the workshop include all aspects of computer algebra, from theory to applications and systems.

Further Information:
`http://www-math.sci.kun.nl/~bosma/RWCA04`

3. Computeralgebra in Lehre, Ausbildung und Weiterbildung IV: Konsequenzen aus PISA

Haus Schönenberg bei Ellwangen, 13. – 16.04.2004

Diese Tagung wird von der Fachgruppe Computeralgebra in Kooperation mit der MNU, der Fachgruppe Didaktik der Mathematik der DMV sowie der GDM veranstaltet. Sie setzt die bisherigen Tagungen in Thurnau und Schöntal fort und findet in der Zeit von Dienstag, 13.04.2004 bis Freitag, 16.04.2004 im Haus Schönenberg bei Ellwangen statt. (Näheres siehe Seite 6.)

Weitere Informationen:
`http://www.fachgruppe-computeralgebra.de/CLAW/Schoenenberg2004`

4. ECCAD 2004 – East Coast Computer Algebra Day 2004

Waterloo, Ontario, Canada, 08.05.2004

East Coast Computer Algebra Day is an annual one-day conference that provides opportunities to learn and share new developments and to present research results in the area of symbolic computation. ECCAD 2004 is hosted by the Computer Algebra Research Group, Department of Physics and Computer Science, Wilfried Laurier University, Waterloo, Ontario, Canada. Invited speakers include Jonathan Borwein (Simon Fraser University), Daniel Lazard (Paris VI), Bernd Sturmfels (UC Berkeley).

Organization:
I. S. Kotsireas (Waterloo)

Weitere Informationen:
`http://www.cargo.wlu.ca/eccad2004/index.php`

5. Algorithms and Number Theory

Schloss Dagstuhl, 16.05. – 21.05.2004

Seminars on Algorithmische Zahlentheorie and Algorithms and Number Theory were already held at the IBFI in the years 1992, 1994, 1998 and 2001. The corresponding seminar reports document the success of these meetings. The area of Algorithmic Number Theory is on the borderline between Mathematics (Number Theory) and Computer Science (including Complexity Theory). It has developed rapidly in the

last 20 years and important results were obtained. Number theoretical algorithms have become fundamental for many applications in Cryptography, Coding Theory and also for Computer Algebra Systems.

The central topics of this seminar will be classical computational number theory, algorithmic aspects of elliptic curves and curves of higher genus, and their applications. Additionally, we plan to discuss new areas in which there have been important developments recently.

The proposed new seminar about Algorithmic Number Theory is planned for an exchange of ideas between mathematicians and computer scientists. By the connection of methods from number theory with those from complexity theory and the theory of algorithms an area of research has been established which has a variety of important applications. Consequently, we plan to emphasize those subjects with a potential for applications in Cryptography and Coding Theory.

Organizers:
J. Buhler (Reed College, Portland), J. Cremona (University of Nottingham), M. E. Pohst (TU Berlin).

Further Information:
`http://www.dagstuhl.de/04211`

6. USACAS 2004

Glenview, Illinois, 19.06. – 20.06.2004

Computer Algebra Systems have the potential to revolutionize mathematics education at the secondary level. They do for algebra and calculus what calculators do for arithmetic: simplify expressions, solving equations, factoring, taking derivatives, and much more. The second U.S. conference on computer algebra systems in secondary mathematics takes place at Glenbrook South High School, 4000 West Lake Avenue, Glenview, Illinois.

Organizers:
James E. Schultz, Robert L. Morton (Ohio University), Natalie Jakucyn (Mathematics Department, Glenbrook South High School).

Registration:
By May 15, 2004: $ 185, late registration: $ 225. (Fee includes continental breakfast, box lunch, afternoon snack and shirt.)

Further Information:
`http://www4.glenbrook.k12.il.us/usacas/2004.html`

7. ISSAC 2004 – International Symposium on Symbolic and Algebraic Computation

Santander, Spain, 04. – 07.07.2004

ISSAC is the yearly premier international symposium in Symbolic and Algebraic Computation that provides an opportunity to learn of new developments and to present original research results in all areas of symbolic mathematical computation. In 2004, ISSAC is hosted by the University of Cantabria, in the city of Santander, Spain, July 4-7. Invited speakers are Pablo Parrilo, Francisco Santos, Jan Verschelde.

Organization:
Josef Schicho (General Chair)

Further Information:
`http://www.risc.uni-linz.ac.at/issac2004/index.html`

8. CASC 2004 – The 7th International Workshop on Computer Algebra in Scientific Computing

Sankt Petersburg, 04. – 07.07.2004

CASC 2004 will be held at the Euler International Mathematical Institute, Saint Petersburg, Russia in July 12-19, 2004. The methods of Scientific Computing play an important role in research and engineering applications in the natural and the engineering sciences. The significance and impact of computer algebra methods and computer algebra systems for scientific computing has increased considerably in recent times. Nowadays, such general-purpose computer algebra systems as Mathematica, Maple, MuPAD and others enable their users to solve the following three important tasks within a uniform framework: symbolic manipulation, numerical computation, visualization.

Organizers:
Workshop general organizing committee: Werner Meixner (Munich, chair), Annelies Schmidt (secretary) Local organizing committee chair: Nikolay Vassiliev (St. Petersburg)

Topics:
The topics addressed in the workshop cover all the basic areas of scientific computing provided by application of computer algebra methods and software.

Important Dates:
April 4, 2004: Submission of full paper (up to 15 pages), via email to `casc2004@in.tum.de`
May 11, 2004: Notification of acceptance
May 31, 2004: Camera-ready papers must be received
June 27, 2004: Deadline for advance registration

Further Information:
`http://wwwmayr.informatik.tu-muenchen.de/konferenzen/CASC2004`

9. From Arithmetic to Cryptology – Conference on the occasion of Gerhard Frey's 60th birthday

Essen, 08. – 10.07.2004

From Thursday July 8 to Saturday July 10 2004, we want to celebrate Gerhard Frey's 60th birthday with an international research conference at the University of Duisburg-Essen, Essen Campus. The following topics will be of premier importance in the conference: Diophantine equations, curves and fundamental groups, abelian varieties, modular forms and modular curves, application of the above to cryptology.

We have organized an allotment of rooms in two hotels. Please register for the conference by June 1, as otherwise we cannot garuantee for a hotel room. Please note that there is a conference fee of 60 € which includes a banquet.

Young researchers are encouraged to participate in the conference; some funds are available to cover the conference fee and part of the travel costs.

Organizers:
Gebhard Böckle, Claus Diem, Hans-Georg Rück

Invited Speakers: Jannis Antoniadis (University of Crete, Greece), Eva Bayer (University of Lausanne, Switzerland), Pilar Bayer (University of Barcelona, Spain), Nigel Boston (University of Wisconsin, Madison, USA), Bas Edixhoven (University of Leiden, Netherlands), Moshe Jarden (University of Tel Aviv, Israel), Wulf-Dieter Geyer (University of Erlangen, Germany), Ernst Kani (Queen's University, Kinston, Canada), Ian Kiming (University of Copenhagen, Denmark),

Hendrik Lenstra (University of Leiden, Netherlands), Kumar Murty (University of Toronto, Canada), Claus-Günter Schmidt (University of Karlsruhe, Germany), Rene Schoof (University of Rome, Italy), Helmut Völklein (University of Florida, Gainesville, USA), Stefan Wewers (University of Bonn, Germany).

Further Information:
`http://www.exp-math.uni-essen.de/~birthday`

10. ACA'2004 – The 10th International Conference on Applications of Computer Algebra

Beaumont, Texas, 21. – 23.07.2004

The ACA conference series is dedicated to reporting serious applications of symbolic computation (a.k.a. computer algebra) theories and tools for mathematics, logic, science, engineering and education. The 10th ACA conference will be held in Beaumont, Texas, on July 21-23, 2004. Selected papers of ACA'2003 and ACA'2004 will be published in a Special Issue of Journal of Symbolic Computation.

Organizers:
General Chair: Quoc-Nam Tran (Lamar)
Organizing Committee: Larry Osborne (Lamar), Stanly Steinberg (UNM)
Program Chair: Quoc-Nam Tran (Lamar), Vladimir Gerdt (Dubna)

Further Information:
`http://buchberger.cs.lamar.edu/ACA2004/index.jsp`

11. CCCG 2004 – The 16th Canadian Conference on Computational Geometry

Montreal, Canada, 09. – 11.08.2004

The Canadian Conference on Computational Geometry (CCCG) focuses on the mathematics of discrete geometry from a computational point of view. Abstracting and studying the geometry problems that underlie important applications of computing (such as geographic information systems, computer-aided design, simulation, robotics, solid modeling, databases, and graphics) leads not only to new mathematical results, but also to improvements in these application areas. Despite its international following, CCCG maintains the informality of a smaller workshop and attracts a large number of students.

Further Information:
`http://www.cs.concordia.ca/cccg/`

12. DMV Jahrestagung 2004

Heidelberg, 13.09. – 17.09.2004

Das Präsidium der Deutschen Mathematiker-Vereinigung und die örtliche Tagungsleitung laden alle interessierten Kolleginnen und Kollegen herzlich zur Teilnahme an der Jahrestagung 2004 ein. Die Tagung findet vom 13. (Anreise) bis 18. September 2004 (Abreise) an der 617-jährigen Universität Heidelberg statt.

Das wissenschaftliche Programm beginnt am 13. September und endet am Nachmittag des 18. September 2004. Vormittags werden Plenarsitzungen mit den Hauptvorträgen abgehalten. Nachmittags finden Vorträge in folgenden Sektionen statt:

Logik, Algebra, Computeralgebra, Zahlentheorie, Komplexe Analysis, Differentialgeometrie, Differentialgleichungen, dynamische Systeme und Kontrolltheorie, Partielle Differentialgleichungen, Geometrie, Topologie, Wahrscheinlichkeitstheorie, Statistik, Diskrete Mathematik, Optimierung, Numerische Mathematik, Wissenschaftliches Rechnen, Mathematische Physik, Mathematik in den Biowissenschaften, Mathematik in den Finanz- und Wirtschaftswissenschaften, Geschichte der Mathematik, Didaktik, Funktionalanalysis.

Darüber hinaus werden Minisymposien in den Bereichen Mathematik in den Biowissenschaften bzw. Mathematik in den Finanz- und Wirtschaftswissenschaften stattfinden. Weitere Minisymposien sind möglich. Vorschläge richten Sie bitte an die zuständigen Tagungsleiter.

Folgende Hauptvortragende haben bereits zugesagt: N. Alon (Tel Aviv), L. Erdös (München), G. P. Galdi (Pittsburgh, PA), R. H. W. Hoppe (Augsburg/Houston), W. Meeks (Amherst, MA), G. Papanicolaou (Stanford, CA), F. Pop (Philadelphia,PA), S. Sauter(Zürich), R. J. Stern (Irvine, CA), M. van der Put (Groningen), H. Furstenberg (Jerusalem).

(Der Vortrag von Herrn van der Put findet auf Vorschlag der Fachgruppe Computeralgebra statt.) Während der Jahrestagung werden die ordentliche Mitgliederversammlung der DMV sowie Sitzungen der Fachgruppen einberufen. Vom 13. bis 15. September findet parallel zur Tagung die traditionelle Studierendenkonferenz Mathematik sowie am 16. September ein Schüler-und Lehrertag statt. Ferner wird die Wanderausstellung „Mathematik zum Anfassen" vom 14.09.2004 bis 28.09.2004 in Heidelberg gastieren.

Die Tagungsgebühren bitten wir der folgenden Aufstellung zu entnehmen: Mitglieder der DMV, ÖMG: 65 €, Nichtmitglieder : 90 €, Studenten: 20 €, Begleitpersonen: 30 €. Bei Anmeldungen nach dem 15. Juli 2004 werden auf Grund des zusätzlichen Verwaltungsaufwandes erhöhte Gebühren von resp. 80 €, 110 €, 30 €, 40 € berechnet.

Anmeldung:
Die Tagungsanmeldung kann schriftlich (Prof. Dr. R. Weissauer, Ruprecht-Karls-Universität Heidelberg, Mathematisches Institut INF 288, 69120 Heidelberg, Fax: 06221 - 548312) oder per e-mail erfolgen.

Weitere Informationen:
`http://www.dmv2004.uni-hd.de`

13. INFORMATIK 2004 – 34. Jahrestagung der Gesellschaft für Informatik

Ulm, 22. – 24.09.2004

Die jährlich stattfindende Jahrestagung der Gesellschaft für Informatik, INFORMATIK 2004, präsentiert traditionell ein breites Spektrum an relevanten Themen der Informatik. In eingeladenen Vorträgen und ausgewählten Workshops werden aktuelle Trends und Entwicklungen beleuchtet und diskutiert. Darüber hinaus finden Fachleute aus Wissenschaft, Wirtschaft und Praxis auf der INFORMATIK 2004 ein Forum für den Austausch mit Gleichgesinnten.

Die INFORMATIK 2004 steht unter dem Motto „Informatik verbindet". Sowohl in den Hauptvorträgen als auch in den Workshops wird dieses Motto unter verschiedenen Gesichtspunkten behandelt werden.

Die Tagung besteht aus dem Tag der Informatik und Workshops. Der Tag der Informatik findet am Mittwoch, den 22.09.2004, statt. An diesem Tag werden renommierte Fachleute aus Wissenschaft und Praxis in Hauptvorträgen auf wichtige Entwicklungstrends in der Informatik eingehen und ihre Visionen über die Zukunft der Informatik mit uns teilen.

Am Mittwoch finden außerdem die Mitgliederversammlung der Gesellschaft für Informatik sowie das Konferenzbankett statt.

Die Workshops finden am Montag und Dienstag (20./21.09.2004) sowie am Donnerstag und Freitag (23./24.09.2004) statt. Sie werden in der Regel von GI-Fachgruppen durchgeführt, welche dort Highlights aus ihren Forschungs- und Anwendungsgebieten vorstellen oder Diskussions- und Informationsveranstaltungen zu aktuellen Themen durchführen.

Organisation:
Peter Dadam (Ulm)

Weitere Informationen:
`http://www.informatik2004.de`

14. 7th International Conference on Artificial Intelligence and Symbolic Computation

RISC (Research Institute for Symbolic Computation), Castle of Hagenberg, Austria, 22. – 24.09.2004

Artificial Intelligence and Symbolic Computation are two views and approaches for automating problem solving, in particular mathematical problem solving. The two approaches are based on heuristics and on mathematical algorithmics, respectively. Artificial Intelligence can be applied to Symbolic Computation and Symbolic Computation can be applied to Artificial Intelligence. Hence, a wealth of challenges, ideas, theoretical insights and results, methods and algorithms arise in the interaction of the two fields and research communities.

Organization:
Conference Chairman: Bruno Buchberger
Program Committee Chairman: John Campbell
Local Organization: Betina Curtis

Invited speakers:
Alan Bundy (University of Edinburgh, UK), Markus Rosenkranz (University of Linz, RISC, Austria), Helmut Schwichtenberg (University of Munich, Germany), Zbigniew Stachniak (York University, Canada).

Important Dates:
May 1, 2004 : Submission of papers
June 20, 2004 : Notification of acceptance/rejection
July 31, 2004 : Submission of final camera-ready version.

The proceedings of the conference containing the refereed and accepted papers will appear as a volume of the Springer Lecture Notes.

Further Information:
`http://www.risc.uni-linz.ac.at/conferences/aisc2004/`

15. CHEP 2004

Interlaken, Schweiz, 27.09. – 01.10.2004

CHEP conferences provide an international forum to exchange information on computing experience and needs for the High Energy Physics community, and to review recent, ongoing and future activities. CHEP conferences are held every 18 months, the previous one being held in San Diego in March 2003.

Organization:
W. von Rüden, J. Harvey, A. Silverman

Further Information:
`http://www.chep2004.org`

16. ICTMT6 – 6th International Conference on Teaching Mathematics with Technology

Volos, Greece, 10.10. – 13.10.2004

The first International Conference on Technology in Mathematics Teaching was organized in 1993 at the University of Birmingham in England. Since then this conference has been organized every two years giving people working on Curriculum Development and Mathematics Education the opportunity to meet and collaborate.

Further Information:
`http://ictmt6.pre.uth.gr`

17. ATCM 2004 – The Asian Technology Conference in Mathematics

Singapur, 13. – 17.12.2004

There is little doubt that technology has made an impact on the teaching and Mathematics. In this conference, we shall go beyond justifying the use of technology in Mathematics to discuss and examine the best practices of applying technology in the teaching and learning of Mathematics and in Mathematics research. In particular, the conference will focus on how technology can be exploited to enrich and enhance Mathematics learning, teaching and research at all levels.

Topics:
The topics include, but are not limited to: Geometry Using Technology, Computer Algebra, Internet Technology for Mathematics, Graphics Calculators, Mathematical Software and Tools on WWW.

Organizers:
Local Organizing Committee Chair: Dr. Keng Cheng ANG

Further Information:
`http://www.atcminc.com/mConferences/ATCM04`

18. ACAT 2005 – 10th International Workshop On Advanced Computing And Analysis

Zeuthen, 23.05. – 27.05.2005

The main purpose of the ACAT (formerly AIHENP) series of workshops is to gather physicists (experimentalists and theorists) and computer science oriented researchers to exchange ideas, to discuss standards and to promote new technologies related to Computing intelligenceïn physics research. The applications are targeted mainly to particle and nuclear physics, astrophysics and accelerator science. Other fields (robotics, nanotechnologies, bio-computing) should not be left apart as they may have similar problems and common solution can be proposed.

Further Information:
`http://www.desy.de/acat05`

19. Tagung der Fachgruppe Computeralgebra

Kassel, 16. – 18.06.2005

Diese Tagung der Fachgruppe Computeralgebra wurde bereits auf Seite 7 angekündigt. Wir wollen auf dieser Tagung

wieder vor allem Nachwuchswissenschaftlern die Vorstellung ihrer Ergebnisse ermöglichen. Auf der anderen Seite wird in verschiedenen Übersichtsvorträgen auch zum aktuellen Stand in einigen wichtigen Gebieten der Computeralgebra berichtet sowie über in Deutschland mitentwickelte Computeralgebra-Software informiert.

Organisation:
Gunter Malle (Kassel)

Weitere Informationen:
`http://www.mathematik.uni-kassel.de/compmath/ca.htm`

Lehrveranstaltungen zu Computeralgebra im SS 2004

- **Rheinisch–Westfälische Technische Hochschule Aachen**
  Arbeitsgemeinschaft Maple, V. Dietrich, E. Görlich, S2
  Algebraisches Praktikum, U. Schoenwaelder, P2
  Algorithmische Gruppen- und Darstellungstheorie, G. Hiß, S2
  Einführung in Maple, V. Dietrich
  Proseminar zur Linearen Algebra (Kryptographie), H. Pahlings S2

- **Universität Bayreuth**
  Seminar Computeralgebra, A. Kerber, S2

- **Technische Universität Darmstadt**
  Proseminar Public Key Infrastrukturen, J. Buchmann, M. Lippert, A. Wiesmaier, PS2
  Vorlesung Algorithmen für Quantencomputer, J. Buchmann, A. Schmidt, M. Döring, V2 + Ü2
  Vorlesung VPN - Virtual Private Networks, drahtgebunden und drahtlos, W. Böhmer, V2
  Vorlesung Effiziente Kryptographie, T. Takagi, K. Schmidt-Samoa, V2 + Ü2
  Praktikum Public Key Infrastruktur und Anwendungen, J. Buchmann, M. Lippert, A. Wiesmaier, P4
  Praktikum Effiziente Kryptographie mit Java, T. Takagi, K. Wirt, E. Karatsiolis, P4
  Seminar Gitter in der Kryptographie, J. Buchmann, C. Ludwig, S2
  Seminar Kryptographie mit elliptischen Kurven, U. Vollmer, S2

- **Universität Dortmund**
  Algebra II (Computeralgebra I), M. Kreuzer, V4 + Ü2

- **Fachhochschule Flensburg**
  Analysis mit Maple, N. Pavlik, Ü1
  Lineare Algebra mit Maple, P. Thieler, Ü1
  Computational Imaging mit Maple, M. Kersken,V3 + Ü1
  Information Mining mit Maple, M. Kersken, V3 + Ü1
  Applied Logics mit Maple, P. Thieler, V3 + Ü1

- **Technische Universität Hamburg-Harburg**
  Diskrete Mathematik Ib, K.-H. Zimmermann, V2 + Ü1

- **Universität Heidelberg**
  Computeralgebra, W. M. Seiler, V4 + Ü2
  Codes und Gruppen, P. Müller, V2

- **Universität Kaiserslautern**
  Computeralgebra, T. Keilen, V4
  Seminar Singularitätentheorie und Computeralgebra, G.-M. Greuel, G. Pfister, S2
  Proseminar Kodierungstheorie und Kryptographie, C. Lossen, S2

- **Pädagogische Hochschule Karlsruhe**
  Informatik II, J. Ziegenbalg, V2
  Modellbildung in Mathematik und Informatik, J. Ziegenbalg, V2

- **Universität Kassel**
  Einführung in Computeralgebrasysteme II, R. Schaper, V2
  Computeralgebra II, W. Koepf, V4

- **Universität Linz, Research Institute for Symbolic Computation**
  Elimination Theory, D. Wang, V2
  Mathematik lernen und lehren mit Computeralgebra-Systemen, B. Kutzler, V2
  Programmieren in Mathematica, W. Windsteiger, P2
  Geometrische Grundlagen für Symbolic Computation, S. Stifter, V2 + Ü1
  Projektseminar Computeralgebra, F. Winkler, S2

- **Universität Oldenburg**
  Seminar zur Algebra und Computeralgebra, W. Schmale, S2
  Seminar Mathematische Anwendersysteme, B. von Pape, W. Schmale, S2

- **Universität Osnabrück**
  Mathematische Anwendersysteme (Mathematica), H. Spindler, V4

- **Universität Rostock**
  Symbolisches Rechnen I, K. Hantzschmann, V2

- **Universität Ulm**
  Computeralgebra, G. Baumann, V2

Aufnahmeantrag für Mitgliedschaft in der Fachgruppe Computeralgebra

(Im folgenden jeweils Zutreffendes bitte im entsprechenden Feld [ ] ankreuzen bzw. _____ ausfüllen.)

Name: _____ Vorname: _____

Akademischer Grad/Titel: _____

Privatadresse

Straße/Postfach: _____

PLZ/Ort: _____ Telefon: _____

e-mail: _____ Telefax: _____

Dienstanschrift

Firma/Institution: _____

Straße/Postfach: _____

PLZ/Ort: _____ Telefon: _____

e-mail: _____ Telefax: _____

Gewünschte Postanschrift: [ ] Privatadresse [ ] Dienstanschrift

1. Hiermit beantrage ich zum 1. Januar 200__ die Aufnahme als Mitglied in die Fachgruppe

   Computeralgebra (CA) (bei der GI: 0.2.1).

2. Der Jahresbeitrag beträgt €7,50 bzw. €9,00. Ich ordne mich folgender Beitragsklasse zu:

   [ ] **€7,50** für Mitglieder einer der drei Trägergesellschaften
   - [ ] GI Mitgliedsnummer: _____
   - [ ] DMV Mitgliedsnummer: _____
   - [ ] GAMM Mitgliedsnummer: _____

   Der Beitrag zur Fachgruppe Computeralgebra wird mit der Beitragsrechnung der Trägergesellschaft in Rechnung gestellt. (Bei Mitgliedschaft bei mehreren Trägergesellschaften wird dies von derjenigen durchgeführt, zu der Sie diesen Antrag schicken.) [ ] Ich habe dafür bereits eine Einzugsvollmacht erteilt. Diese wird hiermit für den Beitrag für die Fachgruppe Computeralgebra erweitert.

   [ ] **€7,50**. Ich bin aber noch nicht Mitglied einer der drei Trägergesellschaften. Deshalb beantrage ich gleichzeitig die Mitgliedschaft in der

   [ ] GI [ ] DMV [ ] GAMM.

   und bitte um Übersendung der entsprechenden Unterlagen.

   [ ] **€9,00** für Nichtmitglieder der drei Trägergesellschaften. [ ] Gleichzeitig bitte ich um Zusendung von Informationen über die Mitgliedschaft in folgenden Gesellschaften:

   [ ] GI [ ] DMV [ ] GAMM.

3. Die in dieses Formular eingetragenen Angaben werden elektronisch gespeichert. Ich bin damit einverstanden, dass meine Postanschrift durch die Trägergesellschaften oder durch Dritte nach Weitergabe durch eine Trägergesellschaft wie folgt genutzt werden kann (ist nichts angekreuzt, so wird c. angenommen).
   - [ ] a. Zusendungen aller Art mit Bezug zur Informatik, Mathematik bzw. Mechanik.
   - [ ] b. Zusendungen durch wiss. Institutionen mit Bezug zur Informatik, Mathematik bzw. Mechanik.
   - [ ] c. Nur Zusendungen interner Art von GI, DMV bzw. GAMM.

Ort, Datum: _____ Unterschrift: _____

Bitte senden Sie dieses Formular an:

Sprecher der Fachgruppe Computeralgebra
Prof. Dr. Wolfram Koepf
Fachbereich Mathematik/Informatik
Universität Kassel
Heinrich-Plett-Str. 40
34132 Kassel
0561-804-4207,-4646 (Fax)
koepf@mathematik.uni-kassel.de
http://www.mathematik.uni-kassel.de/~koepf

Fachgruppenleitung Computeralgebra 2002-2005

Sprecher:
Prof. Dr. Wolfram Koepf
Universität Kassel
Fachbereich Mathematik/Informatik
Heinrich-Plett-Str. 40
34132 Kassel
0561-804-4207,-4646 (Fax)
koepf@mathematik.uni-kassel.de
http://www.mathematik.uni-kassel.de/~koepf

Stellv. Sprecher:
Prof. Dr. H. Michael Möller
Universität Dortmund
Fachbereich Mathematik
Vogelpothsweg 87
44221 Dortmund
0231-755-3077
Moeller@math.uni-dortmund.de

Fachreferentin Chemie:
PD Dr. Karin Gatermann
Konrad-Zuse-Zentrum Berlin (ZIB)
Takustr. 7
14195 Berlin-Dahlem
030-84185-217, -107 (Fax)
gatermann@zib.de
http://www.zib.de/gatermann

Prof. Dr. Johannes Grabmeier
Fachhochschule Deggendorf
Edlmairstr. 6+8
94469 Deggendorf
0991-3615-141
johannes.grabmeier@fh-deggendorf.de
http://www.fh-deggendorf.de/home/
allgemein/professoren/grabmeier

Vertreter der GAMM,
Fachreferent Computational Engineering:
Prof. Dr. Klaus Hackl
Ruhr-Universität Bochum
Universitätsstr. 150
44780 Bochum
0234-32-26025, -14154 (Fax)
hackl@am.bi.rub.de

Fachexperte Physik:
Dr. Thomas Hahn
Max-Planck-Institut für Physik
Föhringer Ring 6
80805 München
089-32354-300, -304 (Fax)
hahn@feynarts.de
http://wwwth.mppmu.mpg.de/members/hahn/

Vertreter der GI:
Prof. Dr. Karl Hantzschmann
Universität Rostock
Fachbereich Informatik
Albert-Einstein-Str. 21
18059 Rostock
0381-498-3400,-3399 (Fax)
hantzschmann@informatik.uni-rostock.de

Fachreferent Lehre und Didaktik:
Prof. Dr. Hans-Wolfgang Henn
Universität Dortmund
Fachbereich Mathematik
Vogelpothsweg 87
44227 Dortmund
0231-755-2939, -2948 (Fax)
wolfgang.henn@mathematik.uni-dortmund.de
http://www.mathematik.uni-dortmund.de/
didaktik/personelles/people/henn.htm

Prof. Dr. Gerhard Hiß
Lehrstuhl D für Mathematik
RWTH Aachen
Templergraben 64
52062 Aachen
0241-80-94543, -92108 (Fax)
Gerhard.Hiss@Math.RWTH-Aachen.de
http://www.math.rwth-aachen.de/LDFM/homes/Gerhard.Hiss

Fachreferent Schule:
OStD. Heiko Knechtel
An der Tränke 2a
31675 Bückeburg
05722-23628
HKnechtel@aol.com

Fachexperte Mathematische Software:
Prof. Dr. Ulrich Kortenkamp
Technische Universität Berlin
Fachbereich Mathematik
Straße des 17. Juni 136
10623 Berlin
030-314-25748, -21269 (Fax)
kortenka@math.tu-berlin.de
http://www.kortenkamps.net

Vertreter der DMV:
Prof. Dr. B. Heinrich Matzat
IWR, Univ. Heidelberg,
Im Neuenheimer Feld 368
69120 Heidelberg
06221-54-8242,-8318(Sekr.),-8850 (Fax)
matzat@iwr.uni-heidelberg.de

Fachreferent Internet:
Dr. Ulrich Schwardmann
GWDG
Am Fassberg
37077 Göttingen
0551-201-1542
Ulrich.Schwardmann@gwdg.de
http://www.gwdg.de/~uschwar1

Fachexperte Industrie:
Dr. Andreas Sorgatz
SciFace Software
Technologiepark 11
33100 Paderborn
05251-690-751, -799 (Fax)
sorgatz@sciface.com
http://www.mupad.de/~andi

Fachreferent Fachhochschulen:
Prof. Dr. Wilhelm Werner
Fachhochschule Heilbronn
Max-Planck-Str. 39
74081 Heilbronn
07131-504387
werner@fh-heilbronn.de

Redakteur Rundbrief:
Dr. Markus Wessler
Universität Kassel
Fachbereich Mathematik/Informatik
Heinrich-Plett-Str. 40
34132 Kassel
0561-804-4192,-4646 (Fax)
wessler@mathematik.uni-kassel.de
http://www.mathematik.uni-kassel.de/~wessler