Einleitung zur Vorlesung

Computeralgebra I

Wintersemester 2001/02

Prof. Dr. Peter Bürgisser
Martin Lotz

Diese Einleitung soll einen Überblick darüber geben, worum es in der Computeralgebra und in dieser Vorlesung geht. Weitere Hinweise, sowie der geplante Ablauf der Veranstaltung, finden sich auf der Homepage der Vorlesung, erreichbar über:

http://www-math.uni-paderborn.de/~pbuerg/

1 Ziele der Computeralgebra

Die Ziele der Computeralgebra sind vielfältig. Wir wollen die wichtigsten davon kurz auflisten.

1. **Symbolische Manipulation** von mathematischen **Objekten**, wie z.B.
   - Zahlen (ganze, rationale, algebraische, transzendente),
   - Funktionen (Polynome, rationale Funktionen, andere),
   - aus Zahlen und Funktionen zusammengesetzte Objekte, wie Vektoren oder Matrizen,
   - allgemeinere Objekte wie z.B. Mengen, Gruppen, Ringe, usw.
2. Bei Bedarf **numerische Approximation** der Lösungen von Gleichungen mit beliebiger Genauigkeit.
3. **Visualisierung** von Funktionen oder Vorgängen.

Computeralgebra Systeme gewinnen immer grössere Bedeutung in Bereichen der Naturwissenschaften und Technik. Tatsächlich kam die Motivation zur Entwicklung solcher Systeme von ausserhalb der reinen Mathematik. Aber auch in der Mathematik selber ist die Computeralgebra von Bedeutung: für didaktische Zwecke (Visualisierung), als Stütze für die Intuition sowie als Hilfe bei der Gewinnung neuer Erkenntisse (siehe das Beispiel mit π weiter unten). Im folgenden wird Computeralgebra mit CA abgekürzt.

2 Historische Entwicklung von CA-Systemen

Wir wollen nun eine kleine (und etwas vereinfachte) Darstellung zur Geschichte von CA-Systemen geben und folgen dabei dem Buch [vzGG99]. Demnach kann man grob drei zeitliche Abschnitte identifizieren.

1. Generation (ab ca. 1965)
   - MACSYMA (Joel Moses, MIT)

- SCRATCHPAD (Richard Jenks, IBM)
- REDUCE (Tony Hearn)
- SAC-I (George Collins)

Diese ersten Systeme können bereits sehr gut symbolische Operationen wie Differenzieren, Integrieren oder Faktorisieren durchführen.

2. Generation (ab ca. 1985)

   - MAPLE (Keith Geddes, Gaston Gonnet, Univ. of Waterloo, Canada)
   - MATHEMATICA (Stephen Wolfram)

   Wichtigste Neuerung der 2. Generation war die Einführung moderner Benutzerumgebungen, sowie graphische Möglichkeiten.

3. Generation (ab ca. 1990)

   - AXIOM (Nachvolger von SCRATCHPAD, von NAG vertrieben)
   - MAGMA (John Cannon, Univ. of Sydney)
   - MUPAD (Benno Fuchssteiner, Universität Paderborn)

   Diese Systeme unterscheiden sich hauptsächlich durch den inneren Aufbau, z.B. die Unterstützung abstrakter Datentypen.

Daneben gibt es zahlreiche weitere Systeme für spezielle Anwendungen, z.B. GAP (Gruppentheorie), PARI (algorithmische Zahlentheorie) oder MACAULAY (kommutative Algebra und algebraische Geometrie).

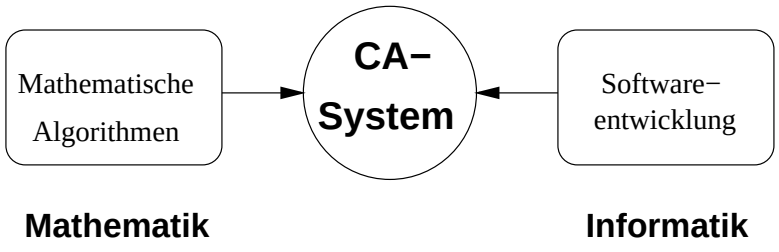
Siehe dazu das in kürze erscheinende Buch

J. Grabmeier, E. Kaltofen, V. Weispfenning,
Handbook of Computer-Algebra, Springer 2001.

für eine Übersicht. Der Fachbereich 17 hat für alle gängigen Systeme eigene Lizenzen sowie Dokumentationen, siehe dazu die Homepage zur Vorlesung.

3 Ziele der Vorlesung

Die Entwicklung eines CA-Systems ist ein riesiges Softwareentwicklungsprojekt.



Die Fähigkeiten eines CA-Systems spiegeln unser algorithmisches Wissen der Mathematik wieder; ohne effiziente Algorithmen gibt es keine Computeralgebra! In dieser Vorlesung wird es ausschliesslich darum gehen, die mathematischen und algorithmischen Konzepte herzuleiten, auf denen die existierenden CA-Systeme basieren. Der Aspekt der Softwareentwicklung ist nicht Thema dieser Veranstaltung.

Die Vorlesung besteht aus drei Schwerpunkten:

- algorithmische Ideen,
- Korrektheitsbeweise für Algorithmen
- asymptotische Laufzeitanalyse.

In den Übungen sollen neben theoretischen Aufgaben kleine "Projekte" durchgeführt werden, wo der praktische Einsatz von MAPLE trainiert wird. Natürlich darf dabei auch ein anderes System, wie MUPAD, benutzt werden. Wir werden nun ein Beispiel eines solchen Projekts besprechen.

4 Formeln für π via Gitter Reduktion

Mitte der 90er Jahre wurde mit Hilfe der Computeralgebra eine neue Formel für π gefunden, die es erlaubt jede beliebige hexadezimale Stelle von π in praktisch linearer Zeit und logarithmischem Platz zu berechnen:

$$\pi = \sum_{i=0}^{\infty} \Big(\frac{1}{16}\Big)^i \Big(\frac{4}{8i+1} - \frac{2}{8i+4} - \frac{1}{8i+5} - \frac{1}{8i+6}\Big) \tag{1}$$

Die schnelle Berechnung einer Stelle von π mit dieser Formel benötigt schnelle Arithmetik, wie sie im ersten Kapitel der Vorlesung besprochen wird. Siehe dazu die Arbeit [BBP97] von Bailey, Borwein und Plouffe.

Wie kann man die Formel (1) beweisen? Die Idee beruht auf der Verwendung einer geometrischen Reihe:

$$\begin{aligned}
\xi_k := \int_0^1 \frac{y^{k-1}}{1-\frac{y^8}{16}} dy &= \int_0^1 \sum_{i=0}^{\infty} y^{k-1} \Big(\frac{y^8}{16}\Big)^i dy \\
&= \sum_{i=0}^{\infty} \Big(\frac{1}{16}\Big)^i \int_0^1 y^{8i+k-1} dy \\
&= \sum_{i=0}^{\infty} \frac{1}{16^i(8i+k)}
\end{aligned}$$

Somit ist (1) äquivalent zu

$$\begin{aligned}
\pi &= 4\xi_1 - 2\xi_4 - \xi_5 - \xi_6 \\
&= \int_0^1 \frac{4 - 2y^3 - y^4 - y^5}{1-\frac{y^8}{16}} dy.
\end{aligned}$$

Dieses Integral wird von MAPLE im Bruchteil einer Sekunde berechnet und gibt tatsächlich π! Dieser Beweis sagt allerdings noch nichts darüber aus, wie man auf so eine Formel kommt. Die Entdecker meinten dazu nur: "... by a combination of inspired guessing and extensive searching using the PSLQ integer relation algorithm" ([BBP97], Seite 3). Wir wollen uns nun fragen, ob es weitere ganzzahlige Beziehungen

$$-m_9\pi = \sum_{k=1}^{8} m_k \xi_k$$

mit möglichst "kleinen" Werten $m_k \in \mathbb{Z}$ und $m_9 \neq 0$ gibt.

Nehme dazu eine grosse natürliche Zahl, z.B. 10^{25}, und setze $p = \lfloor 10^{25}\pi \rfloor$, $x_k = \lfloor 10^{25}\xi_k \rfloor$. Gesucht sind dann "kleine" $(m_1, \ldots, m_9)$, so dass

$$\sum_{k=1}^{8} m_k x_k + m_9 p \tag{2}$$

möglichst klein wird. Wir suchen also eine Näherungslösung und hoffen das diese dann eine exakte Lösung liefert. Betrachte dazu die 9×10 Matrix

$$L := \begin{pmatrix} 1 & 0 & \ldots & 0 & x_1 \\ 0 & 1 & \ldots & 0 & x_2 \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ 0 & 0 & \ldots & 1 & p \end{pmatrix}$$

bestehend aus der Konkatenation der 9×9-Einheitsmatrix und dem Spaltenvektor $(x_1, \ldots, x_8, p)^\top$. Die Zeilen $\mathbf{v}_1, \ldots, \mathbf{v}_9$ erzeugen eine Untergruppe von $\mathbb{Z}^{10}$, ein sogenanntes **Gitter**. Wir suchen ein möglichst kurzes Element $\sum_{j=1}^{9} m_j \mathbf{v}_j \neq 0$ aus diesem Gitter, d.h. einen möglichst kurzen Vektor (in der Euklidischen Norm)

$$(m_1, \ldots, m_9, \sum_{k=1}^{8} m_k x_k + m_9 p) \neq 0.$$

Allein bei der Einschränkung $|m_k| \leq 10$ kommen 21^9 Möglichkeiten zum testen in Frage, falls uns nichts raffiniertes einfällt. Doch es gibt den berühmten LLL-Algorithmus von Lenstra, Lenstra & Lovász ([LLL82], siehe auch Kapitel 3 von [CCS99] oder Kapitel 16 von [vzGG99]), der es ermöglicht, "approximativ kurze" Gittervektoren schnell zu finden. Dieser Algorithmus wird im letzten Kapitel der Vorlesung besprochen. In MAPLE ist dieser implementiert und kann zur Lösung der Aufgabe angewandt werden. Der Algorithmus liefert uns folgende Vektoren (Zeilen der Matrix):

$$\begin{bmatrix} -4 & 0 & 0 & 2 & 1 & 1 & 0 & 0 & 1 & 5 \\ 0 & -8 & -4 & -4 & 0 & 0 & 1 & 0 & 2 & 5 \\ -61 & 582 & 697 & -1253 & 453 & -1003 & -347 & -396 & 10 & 559 \\ -333 & 966 & 324 & -1656 & -56 & 784 & 1131 & -351 & -27 & 255 \\ 429 & 714 & -1591 & 778 & -517 & -1215 & 598 & 362 & -87 & 398 \end{bmatrix}$$

Den MAPLE-Code dazu findet man auf der Homepage der Vorlesung. Die ersten 9 Zahlen einer Zeile repräsentieren $m_1, \dots m_9$, die letzte Zahl ist der Wert der Summe (2). Da wir an kleinen Werten interessiert sind, kommen nur die ersten beiden Zeilen in Frage. Die erste Zeile

$$(-4, 0, 0, 2, 1, 1, 0, 0, 1, 5)$$

entspricht der Lösung

$$-4\xi_1 + 2\xi_4 + \xi_5 + \xi_6 + \pi = 0,$$

welche uns bereits bekannt ist.

Die zweite Zeile entspricht dem Lösungsvorschlag

$$2\pi = 8\xi_2 + 4\xi_3 + 4\xi_4 - \xi_7.$$

Genau wie beim Beweis von (1) kann man diese Identität durch ein Integral nachprüfen. Explizit haben wir also gefunden:

$$2\pi = \sum_{i=0}^{\infty} \Big(\frac{1}{16}\Big)^i \Big(\frac{8}{8i+2} + \frac{4}{8i+3} + \frac{4}{8i+4} - \frac{1}{8i+7}\Big).$$

Es ist bemerkenswert, dass man, nachdem sich Mathematiker über tausende von Jahren damit beschäftigt haben, heute noch so einfache und leicht auszurechnende Identitäten für π finden kann.

Literatur

[BBP97] David Bailey, Peter Borwein, and Simon Plouffe. On the rapid computation of various polylogarithmic constants. *Math. Comp.*, 66(218):903–913, 1997.

[CCS99] Arjeh M. Cohen, Hans Cuypers, and Hans Sterk, editors. *Some tapas of computer algebra.* Springer-Verlag, Berlin, 1999.

[LLL82] A. K. Lenstra, H. W. Lenstra, Jr., and L. Lovász. Factoring polynomials with rational coefficients. *Math. Ann.*, 261(4):515–534, 1982.

[vzGG99] Joachim von zur Gathen and Jürgen Gerhard. *Modern computer algebra.* Cambridge University Press, New York, 1999.