Ulrich Kortenkamp, Schwäbisch Gmünd

CAS und DGS im Dialog – oder: Wieviel CAS braucht der Mensch?

Computeralgebra ist wunderbar; dass, was Taschenrechner für das Rechnen getan haben, können Computeralgebra-Systeme für das symbolische Arbeiten tun. Aber brauchen wir das wirklich? Oder kommen wir mit einem komfortablen makrofähigen programmierbaren Taschenrechner aus? Die Diskussion ist alt und vielfältig (siehe z.B. [1, 2]), und durch die Einführung des Zentralabiturs in immer mehr Bundesländern aktueller denn je. Zudem sind Computer immer stärker im Mathematikunterricht vertreten, und die traditionelle Dreiteilung der Mathematiksoftware (CAS, DGS, TK) verwischt durch den Trend hin zu „Mathematikwerkzeugen“ immer mehr.

Integration von CAS und DGS

Es gibt immer mehr Ansätze, Computeralgebra- und Dynamische Geometrie-Systeme miteinander zu kombinieren, und das Thema wird vielfältig diskutiert [3, 5]. Dabei werden zum Teil erheblich unterschiedliche Ansätze verfolgt. In *GeoGebra*[1] erfolgt die Kombination durch den gleichzeitigen algebraischen und geometrischen „Blick“ auf eine Konstruktion. Beide Seiten könnten interaktiv bearbeitet werden, und Terme können für Konstruktionen herangezogen werden. Das Programm beherrscht einfache symbolische Verfahren (z.b. Differenziation), aber bietet kein echtes CAS. *Geometry Expressions*[2] ist ein *constraint*-basiertes Geometriesystem, welches den konstruktiven Aspekt von DGS abschwächt, dafür aber über ein großes Potenzial im analytisch-algebraischen Bereich verfügt. Das besonders interessante System *Feli-X*[3] integriert eine DGS-Komponente in ein CAS und beschreitet dadurch neue Wege. Die große Flexibilität des Systems macht aber gleichzeitig den Unterrichtseinsatz schwierig, da die fachliche und didaktische Reduktion eines Themas stark durch die Lehrkraft gesteuert werden muss. *Cinderella*[4] in-

[1] Markus Hohenwarter et al.,`http://www.geogebra.at`
[2] Phil Todd, Saltire Software, `http://www.saltire.com`
[3] Reinhard Oldenburg, PH Heidelberg
[4] Jürgen Richter-Gebert u. Ulrich Kortenkamp, `http://cinderella.de`

tegriert in der Version 2.0 eine funktionale Programmiersprache, die zwar nicht (direkt) die symbolische Manipulation von Funktionen erlaubt, dafür aber eine Programmierumgebung darstellt, in der mathematische Sachverhalte einfach ausgedrückt werden können. Berechnungen können nicht nur auf die Daten der Konstruktion zurückgreifen, sondern sie auch selbst wieder beeinflussen.
Die *Programmierfähigkeit* eines Systems stellt eine große Errungenschaft dar, wie im folgenden dargestellt werden soll. Sie bedient zwei Bedürfnisse: Sobald Terme zur Verfügung stehen sollen (und diese sind heute eine Grundanforderung auch an DGS, insbesondere bei der Visualisierung von funktionalen Zusammenhängen), die nicht nur aus den Basisfunktionen bestehen, müssen Mechanismen für Entscheidungen (`if`) und Wiederholungen/Schleifen (`repeat`/`while`/`for`) bereitstehen. Weiterhin wird von Schülerinnen und Schülern sowie Lehrkräften der Wunsch geäußert, dass sie Visualisierungen gezielt beeinflussen möchten – dies geht in der notwendigen Eindeutigkeit *und* Flexibilität nur über eine Programmiersprache.[5]
Zu der durch CAS möglichen symbolischen Manipulation von Termen soll hier noch erwähnt werden, dass diese Funktion für das mathematische Durchdringen eines Sachverhaltes auch hinderlich sein kann; das Lesen in Formeln (schon von Plücker gefordert) kann nicht stattfinden, wenn jede Formel automatisch in eine kanonische Form gebracht wird. Konkrete Beispiele sind Teleskopsummen, die nur „ausgeschrieben“ einleuchten, oder das Verfahren des doppelten Abzählens in der diskreten Mathematik.

MUT - Unit Tests im Mathematikunterricht

Eine neue Sichtweise auf CAS bieten sogenannte Math Unit Tests (analog zu Unit Tests in der Softwareentwicklung nach Beck/Cunningham, siehe dazu auch [4]): Anstatt direkt zu versuchen, die *Lösung* eines Problems mit CAS zu finden, werden zunächst in der CAS-Umgebung *Anforderungen an eine Lösung* formuliert, die automatisiert überprüft werden können sobald eine mögliche Lösung vorliegt.
Die Aufgabe „Finde den Schnitt der Geraden $y = 2x + 3$ und $y = 8 - 3x$“ ist

[5] Insbesondere wurde dieser Wunsch in Unterrichtsversuchen mit der Erweiterung Visage (`http://cinderella.de/visage`) des DFG-Forschungszentrum MATHEON geäußert.

mit dem `solve`-Befehl eines CAS trivial – wenn man die Aufgabe verstanden hat: gesucht sind Punkte, die auf beiden Geraden liegen, dazu müssen die Koordinaten der Punkte beide Gleichungen erfüllen! Eine mögliche MUT-Formulierung der Aufgabe wäre dann zum Beispiel:

> Formuliere Tests, die folgendes überprüfen können:
>
> - $(x; y) = (4; 11)$ ist keine Lösung
> - $(x; y) = (2; 2)$ ist keine Lösung
> - $(x; y)$ ist keine Lösung

Didaktische Motivation

Wie im Beispiel demonstriert ermöglicht das CAS die Formulierung der Tests *vor* der eigentlichen Lösung der Aufgabe. Damit stärken wir die traditionelle (und bewährte) Probe. Das gebetsmühlenartige „$13 - x = 5 \ldots\ x = 9 \ldots$ Probe: $5 + 9 = 14 \ldots$ stimmt“ wird dadurch verhindert: Die Probe lautet „$5 + x = 14$“, ob die gefundene Lösung $x = 9$ dann wirklich korrekt ist (ist sie nicht!), entscheidet der Rechner.
Durch das Voranstellen der Tests wird das Denken erzwungen, und Schülerinnen und Schüler beschreiben die Anforderungen an ihre Lösung selbst!

Die Zukunft des CAS in Unterricht und Prüfung

Wichtige Aufgaben, die ein CAS bedienen kann, sind das *Programmieren und Automatisieren*, um komplexe Visualisierungen zu ermöglichen und komplexe Aufgaben in überschaubare Teile zu unterteilen; das *Abkürzen* von Routineaufgaben; die *Ergebnissicherung*, insbesondere durch die in CAS enthaltenen Editoren für sog. *notebooks*; hinzu kommt – neu – die oben kurz beschriebene automatisierte Verifikation/Probe, kurz MUT. Gerade für die ergiebigen Modellierungsaufgaben hilft hier das CAS eine Situation zu mathematisieren und diese Mathematisierung kritisch zu prüfen und hinterfragen.
Als wichtigste Forschungsfrage zum Unterrichts- und Prüfungseinsatz von CAS sehe ich daher, ob und wie die automatisierte Probe (MUT) in den regulären Unterricht eingebaut werden kann. Ist es möglich, die Formulierung

von solchen Tests bereits frühzeitig (d.h. sogar in der Grundschule) zu kultivieren, und kann dadurch die mathematische Bildung der Schülerinnen und Schüler verbessert werden?
Auch ohne die Beantwortung dieser Frage sehe ich aber die Notwendigkeit zu einem veränderten Bewußtsein im Umgang mit CAS. Wird es nur als Rechenwerkzeug verwendet (der Status Quo!), dann laufen wir Gefahr, dass der Mathematikunterricht sich in routinierter Abarbeitung von erlernten Algorithmen erschöpft. Benutzen wir aber das CAS als Formulierungshilfe für mathematisches Verständnis, welche keine Ergebnisse liefert, sondern Ergebnisse überprüft, dann erschließt sich ein großes Potenzial. Auch das drängende Problem des Einsatzes von Computeralgebrasystemen in Prüfungen (damit verbunden die Frage „Wann enthält ein Taschenrechner ein CAS?“, siehe dazu den Artikel von Andreas Pallack in diesem Band) ist damit entschärft.

Literatur

[1] H. Hischer (Hrsg.): Wieviel Termumformung braucht der Mensch? – Fragen zu Zielen und Inhalten eines Mathematikunterrichts angesichts der Verfügbarkeit informatischer Methoden, Tagungsband der Herbsttagung 1993 des AK Mathematikunterricht und Informatik. Franzbecker, Hildesheim 1993.

[2] W. Herget, H. Heugl, B. Kutzler u. E. Lehmann: Welche handwerklichen Rechenkompetenzen sind im CAS-Zeitalter unverzichtbar? In: der mathematische und naturwissenschaftliche Unterricht 54, S. 458–464, 2001.

[3] Ulrich Kortenkamp: Combing CAS and DGS – Towards Algorithmic Thinking. Proceedings of SCE 2006, World Scientific, to appear 2007.

[4] Ulrich Kortenkamp: MUNIT - Mathematical Unit Tests, erscheint in: Kortenkamp, Weigand, Weth (Hrsg.): Tagungsband der Herbsttagung 2006 des AK Mathematikunterricht und Informatik, Franzbecker, Hildesheim 2007.

[5] Reinhard Oldenburg: Bidirektionale Verknüpfung von Computeralgebra und dynamischer Geometrie, JMD 26, 249–273, 2005