eCULT
eCompetence and Utilities for Learners and Teachers

Ostfalia
Hochschule für angewandte Wissenschaften

Übungsaufgaben mit Computer-Algebra-Software – Mathematik „machen“, statt nachmachen

Uta Priss

ZeLL, Ostfalia

Dezember, 2015

Salzgitter
Suderburg
Wolfenbüttel & Braunschweig
Wolfsburg

Meine Lehrerfahrung mit Mathe

- Vor 5 Jahren neuer Kurs:
  „Mathematik für Informatiker“ an einer schottischen Uni.
- Vorgabe für den Inhalt: keine.
- Studierende mit geringen Vorkenntnissen.
- Mein Ziel: auf Informatik-Anwendungen fokussieren.
  Den Studierenden die Angst vor Mathe nehmen.

Der Kurs wurde von den Studierenden als sehr gut bewertet.

Nächstes Semester: Diskrete Strukturen an der Ostfalia.

Mögliche Gründe, warum mein Kurs erfolgreich:

1. Einsatz von Computer Algebra Software (CAS) und normale Programmiersprache (Python)
   Hypothese: Zugang zur Mathematik durch Computerprogramme ist für Informatikstudierende intuitiv

2. Ed Dubinsky: APOS-Theorie

Ziel dieses Webinars

Zum Nachdenken anregen:

* über die Vielzahl der Möglichkeiten beim Einsatz von CAS

* durch Übungsaufgaben Verstehensprozesse unterstützen

→ Mathematik „machen“, statt nachmachen

Studierende sollen „Funktion“ definieren

Eine Funktion ist ...

1. Ein Algorithmus, der Eingabewerte auf Ausgabewerte abbildet.
2. Etwas, das einen Ausdruck mit x berechnet.
3. Gleichung mit Variablen; Wert der Variable wird ausgerechnet.
4. Ein Input wird bearbeitet, welches zu einer Ausgabe führt.

APOS-Theorie: Aktion und Prozess

Eine Funktion ist ...

- *Aktion:*
  „Etwas, das einen Ausdruck mit x berechnet.“
  „Gleichung mit Variablen; Wert der Variable wird ausgerechnet.“
- *Prozess:*
  „Ein Input wird bearbeitet, welches zu einer Ausgabe führt.“
  „Ein Algorithmus, der Eingabewerte auf Ausgabewerte abbildet.“
- *Objekt:*
  Eine Funktion kann selbst Eingabewert für eine andere Funktion sein.

APOS-Theorie: Verstehensebenen

- *Aktion:* Regeln ausführen
- *Prozess:* abstrakteres Verständnis, zugrundeliegendes Prinzip
- *Objekt:* Reifikation, auf höherer Ebene als Ganzes betrachtet
- *Schema:* komplexes Zusammenspiel von Aktionen, Prozessen, Objekten und Schemata

Jede Ebene muss einzeln gelernt werden.
Nicht immer in dieser Reihenfolge.

(Literatur: Leron & Dubinsky. "An abstract algebra story." AMM 1995, http://www.math.kent.edu/∼edd/AlgbraStory.pdf)

APOS-Theorie: Beispiel Quadratfunktion

- *Aktion:*
  eine Zahl mit sich selbst multiplizieren
- *Prozess:*
  $f(x) = x^2$, $\sqrt{x}$, $(a+b)^2$
- *Objekt:*
  x^2 als Parameter für andere Funktionen
- *Schema:*
  Kurvendiskussion

Objektverständnis ist schwierig

Komplexe Strukturen sind einfache Elemente auf einer höheren Abstraktionsebene. Beispiele:

* *Verkettung von Funktionen: Funktion wird Element* $g(f(x))$ *ist ok,* $g \circ f$ *ist schwierig*

* *Matrizen: Multiplikation von Zahlen ist kommutativ aber Matrixmultiplikation ist nicht kommutativ*

* *Rechnen mit Unendlichkeit:* $\aleph_0 + \aleph_0 = \aleph_0$ *(Für Zahlen:* $a + a = 2a$*)*

(Typisch für Mathematik!)

ACE - Lernzyklus

Als Lehrender sollte man die Aktionen, Prozesse und Objekte eines Themas identifizieren.

Für wichtige Prozesse und Objekte benötigt man dann (eventuell für jede Verstehensebene):

1. einführende Aktivität: die Studierenden experimentieren, beobachten, definieren, erfahren und gestalten lassen
2. Class discussion: zusammenfassen, verstehen, Fragen stellen, Zusammenhänge erklären
3. Exercises: einüben

Übung: Grundlagen der Mengenlehre

Geben Sie je ein Beispiel für die Verständnisebenen:
Aktion, Prozess und Objekt.

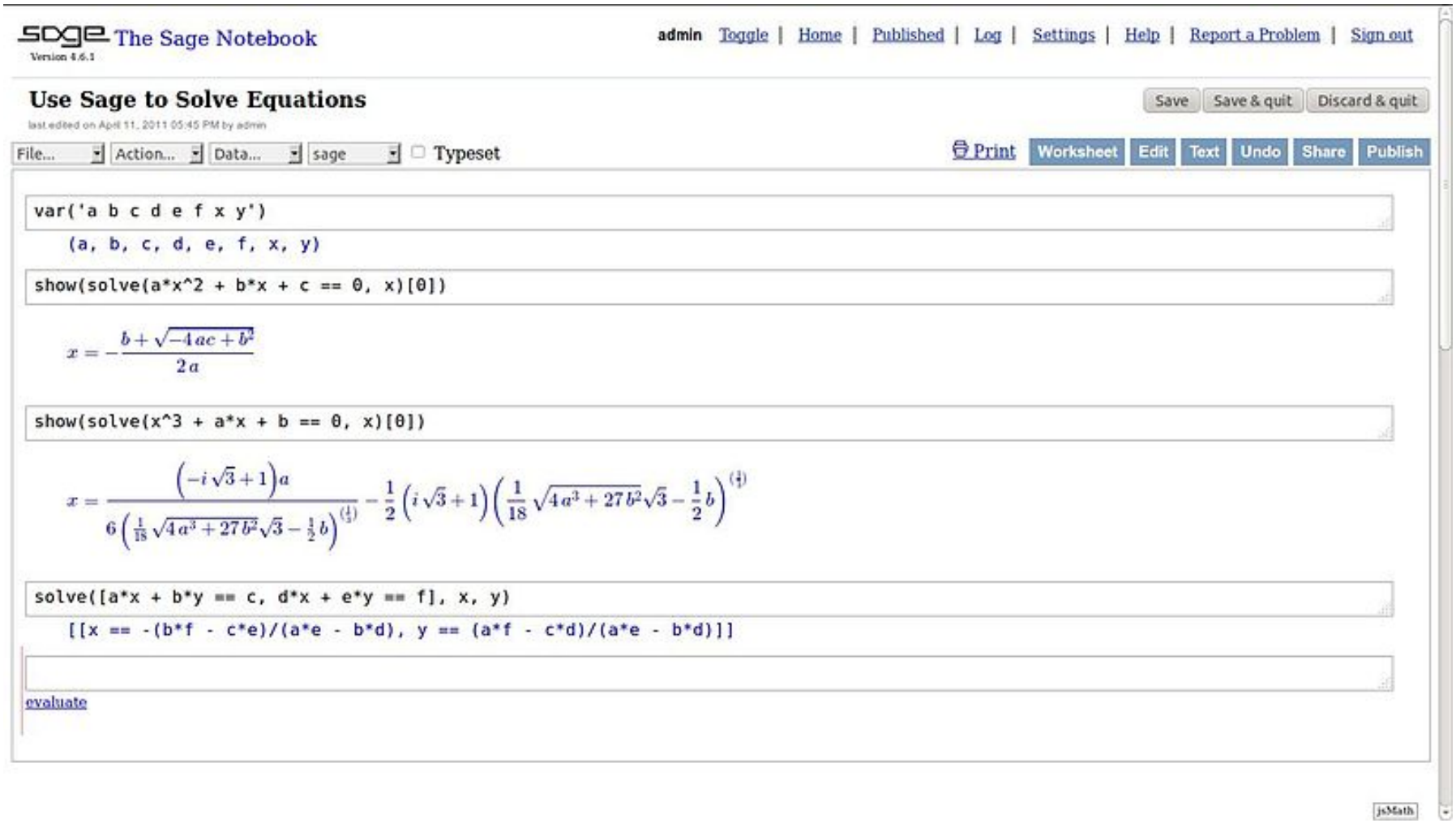
Verwenden Sie schon CAS?
Wenn ja, welche?

SageMath - Open-Source Mathematical Software System

- http://www.sagemath.org/de/
- Programmiersprache Python
- Interface: Web oder Kommandozeile
- Maxima, SymPy, NumPy, SciPy, R, Mengen, Logik, Graphentheorie, graphische Darstellung

Anbindung an Programmiersprache ist vorhanden.

SageMath Notizbuch



SageMath Web-Interface

Type some Sage code below and press Evaluate.

```
1 from sympy import *
2 x = Symbol('x')
3 y = Symbol('y')
4 ((x+y)**2).expand()
```

Evaluate

Language: Sage

Share

```
x**2 + 2*x*y + y**2
```

CAS und SageMath

Haben Sie Fragen, Kommentare oder Anregungen bezüglich CAS und SageMath?

Traditionelle Lehrveranstaltung

1. Vorlesung: vormachen
2. Übung: nachmachen

→ Mathematik nachmachen

ACE - Lernzyklus (konstruktivistisch)

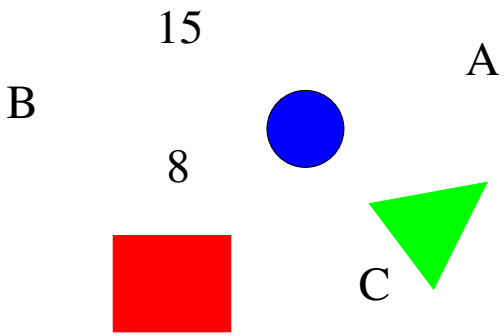
1. einführende Aktivität: die Studierenden experimentieren, beobachten, definieren, erfahren und gestalten lassen
2. Class discussion: zusammenfassen, verstehen, Fragen stellen, Zusammenhänge erklären
3. Exercises: einüben

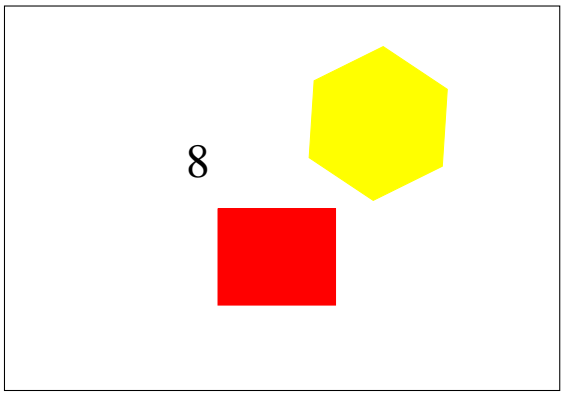
→ Mathematik (wenigstens zum Teil selber) „machen“

Was könnte eine einführende Aktivität für den Begriff „Menge“ sein?

Insbesondere auch: wie kann man dabei CAS einsetzen?

Beispiel: Mengenlehre (konstruktivistisch)





Was ist eine Menge?

Welche Beziehungen zwischen Mengen könnte man definieren?

Schreiben Sie ein Computerprogramm, das zwei Mengen einliest und deren Schnittmenge ausgibt.

Vorteil vom Einsatz von CAS

Übungen mit Papier und Bleistift:

Lernende müssen selbst entscheiden, was richtig und falsch ist. (Erfordert metakognitive Kompetenz oder Überprüfung durch Lehrenden.)

Übungen mit dem Computer:

Die mathematische Realität wird simuliert.
Lernende können direkt mit abstrakten Strukturen experimentieren und Hypothesen überprüfen.

Ziel ist trotzdem Verständnis, aber in kleineren Schritten.

(Peschek & Schneider, "CAS and Communication with Experts")

Abstrakte Inhalte mit CAS programmieren

Zum Beispiel: mathematische Definitionen mit CAS formulieren.

Funktionen schreiben, die Definitionen (Axiome) überprüfen.

Beispiel: Gruppentheorie (Leron & Dubinsky)

Interaction with the computer	Interaction within the team
> Z12 := {0..11};	Why doesn't Z12 have 12 in it?
> a12 := |x,y -> (x+y) mod 12|;	What's the difference between Z12 and a12?
> is_group(Z12,a12); true;	What are the inputs to is_group?
> identity(Z12,a12); 0;	What's happening here? I thought it would be true or false? What is *identity* supposed to return?
> is_group(Z12-{0},a12); false;	What's wrong? Let's check the group properties.
> is_closed(Z12-{0},a12); false;	Aha! So it's not closed, but I can't see why. Let's try some numbers.

Beispiel: Typische Fehlvorstellungen vom Funktionsbegriff

- immer eine Gleichung mit Variablen
- Etwas zum Ausrechnen oder Einsetzen
- nur durch eine Formel darstellbar
- Wohldefiniertheit, Eindeutigkeit und Umkehrbarkeit verwechselt
- kausaler Zusammenhang zwischen Argumenten und Wert
- Begriff der Umkehrfunktion ist nicht klar

(Breidenbach et al. “Development of the process conception of function.” ESIM 23.3 (1992))

Der Funktionsbegriff mit CAS

Verschiedene Darstellungsformen:

- Graph
- CAS Formel
- in Programmiersprachen: iterativ, rekursiv
- nicht-mathematische Funktionen (String.length)

Window-Shuttle-Technik (Heugl, 2006; Dörfler, 1991):
Gleichzeitige Darstellung von Formel und Graph

Objektverständnis von Funktionen

Abstrakte Funktionsdarstellung in modernen Programmiersprachen.
(Rekursion, Lambda, anonyme Funktion, Callback, ...)

```
def f(x) : return x+1
def g(x) : return x*2
f(g(3))
```

Zusammenhänge Mathematik/Informatik (CAS)

Mathematischer Begriff	Relevanz für die Informatik
Menge	alternative Datenstruktur zu Arrays
Primzahlen	Kryptographie
Mathematische Unendlichkeit	Zahlendarstellung im Computer
logische Operatoren	logische Programmiersprachen
Funktionen in der Mathematik	Funktionen in Programmiersprachen
Graphen	Graph-Software benutzen
Relationen	Datenbanktabellen

Fragen und Diskussion

Können Sie sich vorstellen CAS auch in Ihrer Lehrveranstaltung im Sinne von APOS-Theorie einzusetzen?

Setzen Sie selbst schon CAS für einführende Aktivitäten ein?

Haben Sie andere Fragen, Anregungen oder Kommentare?

Literatur und Links

SageMath: http://www.sagemath.org/de/

APOS-Theorie (Einführung und Bibliographie):
Dubinsky & McDonald. "APOS: A constructivist theory of learning in undergraduate mathematics education research." ICMI-Studies Series, Springer, 2002. http://www.math.kent.edu/∼edd/ICMIPaper.pdf

APOS-Theorie Beispiele:
Leron & Dubinsky. "An abstract algebra story." AMM 1995, http://www.math.kent.edu/∼edd/AlgbraStory.pdf

Untersuchung CAS und APOS:

Wolfgang Lindner: Wissenskonstruktion mit Computeralgebrasystemen in der Linearen Algebra/Geometrie der Sekundarstufe II (Dissertation)

http://duepublico.uni-duisburg-essen.de/servlets/DocumentServlet?id=28602

Das eCULT-Projekt an der Ostfalia:
http://ostfalia.de/cms/de/ecult/

Das Gesamt-eCULT-Projekt:
http://www.ecult-niedersachsen.de/

GEFÖRDERT VOM
Bundesministerium für Bildung und Forschung

1

[1] Dieses Vorhaben wird aus Mitteln des Bundesministeriums für Bildung und Forschung unter dem Förderkennzeichen 01PL11066H gefördert. Die Verantwortung für den Inhalt dieser Veröffentlichung liegt beim Autor.