Organ der Gesellschaft für Informatik e.V.
und mit ihr assoziierter Organisationen

Band 42 • Heft 2 • April 2019

# Informatik Spektrum

Bildung und Informatik

Springer

Informatik Spektrum
Band 42 | Heft 2 | April 2019
Organ der Gesellschaft für Informatik e.V. und mit ihr assoziierter Organisationen



**79**

**Illustration des Wachstums natürlicher Parkettierung**

Springer

# Informatik Spektrum

Organ der Gesellschaft für Informatik e.V. und mit ihr assoziierter Organisationen

Hauptaufgabe dieser Zeitschrift ist die Weiterbildung aller Informatiker durch Veröffentlichung aktueller, praktisch verwertbarer Informationen über technische und wissenschaftliche Fortschritte aus allen Bereichen der Informatik und ihrer Anwendungen. Dies soll erreicht werden durch Veröffentlichung von Übersichtsartikeln und einführenden Darstellungen sowie Berichten über Projekte und Fallstudien, die zukünftige Trends aufzeigen.

Es sollen damit unter anderem jene Leser angesprochen werden, die sich in neue Sachgebiete der Informatik einarbeiten, sich weiterbilden, sich einen Überblick verschaffen wollen, denen aber das Studium der Originalliteratur zu zeitraubend oder die Beschaffung solcher Veröffentlichungen nicht möglich ist. Damit kommt als Leser nicht nur der ausgebildete Informatikspezialist in Betracht, sondern vor allem der Praktiker, der aus seiner Tagesarbeit heraus Anschluss an die wissenschaftliche Entwicklung der Informatik sucht, aber auch der Studierende an einer Fachhochschule oder Universität, der sich Einblick in Aufgaben und Probleme der Praxis verschaffen möchte.

Durch Auswahl der Autoren und der Themen sowie durch Einflussnahme auf Inhalt und Darstellung – die Beiträge werden von mehreren Herausgebern referiert – soll erreicht werden, dass möglichst jeder Beitrag dem größten Teil der Leser verständlich und lesenswert erscheint. So soll diese Zeitschrift das gesamte Spektrum der Informatik umfassen, aber nicht in getrennte Sparten mit verschiedenen Leserkreisen zerfallen. Da die Informatik eine sich auch weiterhin stark entwickelnde anwendungsorientierte Wissenschaft ist, die ihre eigenen wissenschaftlichen und theoretischen Grundlagen zu einem großen Teil selbst entwickeln muss, will die Zeitschrift sich an den Problemen der Praxis orientieren, ohne die Aufgabe zu vergessen, ein solides wissenschaftliches Fundament zu erarbeiten. Zur Anwendungsorientierung gehört auch die Beschäftigung mit den Problemen der Auswirkung der Informatikanwendungen auf den Einzelnen, den Staat und die Gesellschaft sowie mit Fragen der Informatik-Berufe einschließlich der Ausbildungsrichtlinien und der Bedarfsschätzungen.

## Urheberrecht

Mit der Annahme eines Beitrags überträgt der Autor Springer (bzw. dem Eigentümer der Zeitschrift, sofern Springer nicht selbst Eigentümer ist) das ausschließliche Recht zur Vervielfältigung durch Druck, Nachdruck und beliebige sonstige Verfahren das Recht zur Übersetzung für alle Sprachen und Länder.

Die Zeitschrift sowie alle in ihr enthaltenen einzelnen Beiträge und Abbildungen sind urheberrechtlich geschützt. Jede Verwertung, die nicht ausdrücklich vom Urheberrechtsgesetz zugelassen ist, bedarf der vorherigen schriftlichen Zustimmung des Eigentümers. Das gilt insbesondere für Vervielfältigungen, Bearbeitungen, Übersetzungen, Mikroverfilmungen und die Einspeicherung und Verarbeitung in elektronischen Systemen. Jeder Autor, der Deutscher ist oder ständig in der Bundesrepublik Deutschland lebt oder Bürger Österreichs, der Schweiz oder eines Staates der Europäischen Gemeinschaft ist, kann unter bestimmten Voraussetzungen an der Ausschüttung der Bibliotheks- und Fotokopietantiemen teilnehmen. Nähere Einzelheiten können direkt von der Verwertungsgesellschaft WORT, Abteilung Wissenschaft, Goethestr. 49, 80336 München, eingeholt werden.

Die Wiedergabe von Gebrauchsnamen, Handelsnamen, Warenbezeichnungen usw. in dieser Zeitschrift berechtigt auch ohne besondere Kennzeichnung nicht zu der Annahme, dass solche Namen im Sinne der Warenzeichen- und Markenschutz-Gesetzgebung als frei zu betrachten wären und daher von jedermann benutzt werden dürfen.

## Vertrieb, Abonnement, Versand

Papierausgabe: ISSN 0170-6012
elektronische Ausgabe: ISSN 1432-122X
Erscheinungsweise: zweimonatlich

Den Bezugspreis können Sie beim Customer Service erfragen: customerservice@springer.com Die Lieferung der Zeitschrift läuft weiter, wenn sie nicht bis zum 30.9. eines Jahres abbestellt wird. Mitglieder der Gesellschaft für Informatik und der Schweizer Informatiker Gesellschaft erhalten die Zeitschrift im Rahmen ihrer Mitgliedschaft.

Bestellungen oder Rückfragen nimmt jede Buchhandlung oder der Verlag entgegen.
Springer, Kundenservice Zeitschriften,
Tiergartenstr. 15, 69121 Heidelberg, Germany
Tel. +49-6221-345-0, Fax: +49-6221-345-4229,
e-mail: customerservice@springer.com
Geschäftszeiten: Montag bis Freitag 8–20 h.

Bei **Adressänderungen** muss neben dem Titel der Zeitschrift die neue und die alte Adresse angegeben werden. Adressänderungen sollten mindestens 6 Wochen vor Gültigkeit gemeldet werden. **Hinweis gemäß §4 Abs. 3 der Postdienst-Datenschutzverordnung:** Bei Anschriftenänderung des Beziehers kann die Deutsche Post AG dem Verlag die neue Anschrift auch dann mitteilen, wenn kein Nachsendeauftrag gestellt ist. Hiergegen kann der Bezieher innerhalb von 14 Tagen nach Erscheinen dieses Heftes bei unserer Abonnementsbetreuung widersprechen.

## Elektronische Version

springerlink.com

## Hinweise für Autoren

http://springer.com/journal/00287

## Hauptherausgeber

Prof. Dr. Dr. h. c.mult. Wilfried Brauer (1978–1998)
Prof. Dr. Arndt Bode,
Technische Universität München (seit 1999)
Prof. Dr. T. Ludwig,
Deutsches Klimarechenzentrum GmbH, Hamburg
(seit 2019)

## Herausgeber

Prof. Dr. S. Albers, TU München
Prof. A. Bernstein, Ph. D., Universität Zürich
Prof. Dr. T. Braun, Universität Bern
Prof. Dr. O. Deussen, Universität Konstanz
Prof. Dr. H. Federrath, Universität Hamburg
Prof. Dr. G. Goos, KIT Karlsruhe
Prof. O. Günther, Ph. D., Universität Potsdam
Prof. Dr. D. Herrmann,
Otto-Friedrich-Universität Bamberg
Prof. Dr. W. Hesse, Universität Marburg
Dr. Agnes Koschmider, KIT Karlsruhe
Dr.-Ing. C. Leng, Google
Prof. Dr. F. Mattern, ETH Zürich
Prof. Dr. K.-R. Müller, TU Berlin
Prof. Dr. W. Nagel, TU Dresden
Prof. Dr. J. Nievergelt, ETH Zürich
Prof. Dr. E. Portmann, Universität Fribourg
Prof. Dr. F. Puppe, Universität Würzburg
Prof. Dr. R.H. Reussner, Universität Karlsruhe
Prof. Dr. S. Rinderle-Ma, Universität Wien
Prof. Dr. O. Spaniol, RWTH Aachen
Dr. D. Taubner, msg systems ag, München
Sven Tissot, Iteratec GmbH, Hamburg
Prof. Dr. Herbert Weber, TU Berlin

## Impressum

**Verlag:**
Springer, Tiergartenstraße 17,
69121 Heidelberg

**Redaktion:**
Peter Pagel, Sybille Thelen
Tel.: +49 611 787 8329
e-mail: Peter.Pagel@springer.com

**Herstellung:**
Philipp Kammerer,
e-mail: Philipp.Kammerer@springer.com

**Redaktion GI-Mitteilungen:**
Cornelia Winter
Gesellschaft für Informatik e.V. (GI)
Wissenschaftszentrum,
Ahrstraße 45, D-53175 Bonn,
Tel.: +49 228-302-145, Fax: +49 228-302-167,
Internet: http://www. gi.de,
e-mail: gs@gi.de

**Wissenschaftliche Kommunikation:**
Anzeigen: Eva Hanenberg
Abraham-Lincoln-Straße 46
65189 Wiesbaden
Tel.: +49 (0)611/78 78-226
Fax: +49 (0)611/78 78-430
eva.hanenberg@springer.com

**Satz:**
le-tex publishing services GmbH, Leipzig

**Druck:**
Printforce,
The Netherlands

springer.com

Eigentümer und Copyright
© Springer-Verlag GmbH Deutschland,
ein Teil von Springer Nature, 2019

***Juraj Hromkovič,***
***ETH Zürich,***
***E-Mail:***
***juraj.hromkovic@inf.ethz.ch***

# Bilden wir die Erfinderinnen, Gestalter und Entwicklerinnen digitaler Technologie aus und nicht nur ihre Konsumenten!

*Kaum ein anderes Fach erlebte und erlebt einen so turbulenten Einzug in das allgemeine Bildungssystem wie die Informatik. Jedes Land schreibt seine eigene Geschichte und diese könnten kaum unterschiedlicher sein. Der erfolgreichste und am besten definierte Einzug der Informatik in das Bildungssystem erfolgte vor fast 50 Jahren in Russland und den meisten osteuropäischen Ländern. Ich selbst genoss als Gymnasiast einen soliden Informatikunterricht für die Dauer von vier Jahren (1973–1977), jeweils vier Stunden wöchentlich. Drei Programmiersprachen, Datenverwaltung mit Cobol und Algorithmen für das Sortieren, die Suche, lineare Algebra und numerische Mathematik standen damals im Mittelpunkt.*

*Während die Informatik im Osten ganz selbstverständlich als ein normales Fach etabliert wurde, gab es in den achtziger Jahren – zumeist über das Programmieren – in den USA und in Westeuropa erste Versuche, die Informatik in den Bildungssystemen zu verankern. Mitte der neunziger Jahre kam in diesen Ländern der Rückschlag aufgrund der naiven Ideologie, dass das Allerwichtigste das Erlernen des Umgangs mit dem Computer und mit der vorhandenen Software sei und dass ein eigenes Fach Informatik nur für Fachspezialisten geeignet wäre. Die Reduktion der Informatik auf oberflächliche Betriebsanweisungen und Computerführerscheine beschädigte das Image des jungen Faches enorm und somit auch das Innovationspotenzial der Wirtschaft. Es folgte eine vergleichbar starke Herabstufung der Informatik, als würde man den Unterricht in Physik und dem Fach Maschinenbau auf den Erwerb eines Führerscheins reduzieren. Der sogenannte Informations- und Kommunikationstechnik-Unterricht (IKT/ICT) wurde zum einzigen Schulfach, für das man keine Lehrerausbildung brauchte, es reichten wenige Kurswochen.*

*Rund zehn Jahre danach haben die meisten Länder diesen groben Fehler erkannt. Die USA, Großbritannien, Frankreich und Italien führten zum Beispiel schrittweise einen obligatorischen Informatikunterricht ein, und zwar mit klarer Trennung von der Vermittlung von Anwendungskompetenzen. Nur im deutschsprachigen Raum – mit kleinen Ausnahmen wie Bayern – tut man sich damit schwer. Die Ideologie des „Computerführerscheins" wurde durch die Ideologie der „Medienkunde" ersetzt. Zum Allerwichtigsten wurden dabei die Reflexionen über die Inhalte, die mittels der Informationstechnologie verbreitet werden. Statt diese Kommunikationsaspekte und die damit verbundenen ethischen Fragen mit den Sprachwissenschaften und der Ethik zu verbinden, bemühte man sich um die „Zwangsheirat" mit der Informatik. Als Reaktion auf das Manifest der deutschsprachigen Informatikerinnen und Informatiker für ein eigenständiges Fach Informatik ohne ICT und Medien aus Dagstuhl vom Winter 2015 folgte das „Dagstuhl-Dreieck" der Medienwissenschaftler. Es betonte die Einheit der Medienwissenschaft, ICT und Informatik und schrieb der Medienkunde die führende Rolle zu. Das „Dagstuhl-Dreieck" reduzierte die Informatik auf das Reflektieren über die Technologie; die Gestaltung und die Entwicklung der Informatik kommen dabei nicht vor. Überall, wo die Ideologie des „Dagstuhl-Dreiecks" zur Anwendung kam, ist es zu keinem nennenswerten Informatikunterricht gekommen. Das betrifft die Lehrmittel sowie die Aus- und Weiterbildung von Lehrpersonen.*

*Deswegen droht zum Beispiel in der Schweiz, der erneute Versuch, Informatik im Rahmen des Faches „Medien und Informatik" in der obligatorischen Schule einzuführen, zu scheitern. Die Informatik braucht wie alle etablierten Schulfächer ein Spiralcurriculum. Man muss ein gewisses Wissen und gewisse Kompetenzen zuerst festigen als Vorstufe für eine Vertiefung des Wissens. Anders als in der Medienkunde, die ein unabhängiges Thema nach dem anderen bearbeiten kann, ist dies in der Informatik nicht möglich, weil die einzelnen Wissensschritte aufeinander aufbauen.*

https://doi.org/10.1007/s00287-019-01174-1

*Dieses Sonderheft kommt deswegen heraus. Wir stellen uns die wichtigsten Fragen für den Einzug der Informatik in die allgemeine Bildung und versuchen, sie zu beantworten: „Was ist Informatik?", „Warum ist sie allgemeinbildend?", „Welchen Mehrwert bringt die Informatik für die Schule?", „Wie fördert ein guter Informatikunterricht das Erreichen der grundlegenden Kompetenzen in anderen Fächern?", „Wie trägt der Informatikunterricht zum Verständnis der Welt und zur Vorbereitung auf die Berufe der Zukunft bei?", „Wie vermittelt man die wichtigsten Konzepte der Informatik in unterschiedlichen Altersstufen?" und „Wie stärkt der Informatikunterricht das Innovationspotenzial der Gesellschaft?"*

*Ich wünsche allen, insbesondere den Verantwortlichen und den Entscheidungsträgern und Entscheidungsträgerinnen, viel Vergnügen bei der Lektüre und lade alle ein, für das Schulfach Informatik die Lanze zu brechen.*

*Illustration of the growth of natural tessellations (an alternative numerical model to Voronoi diagrams on surfaces) on the Earhart's Flight Suit (10M facets mesh, courtesy of the Smithsonian Institution) initialized with 10,000 random seeds, their corresponding cells are visualized in different colors. The close-up on the right side of the chest (top-row) reveals the small scale geometric complexities (pocket fold, button fold) that can successfully be processed with this computationally and memory efficient approach fully implemented on graphics hardware (GPU).*

*Rhaleb Zayer, Daniel Mlakar, Markus Steinberger, and Hans-Peter Seidel. 2018. Layered fields for natural tessellations on surfaces. ACM Trans. Graph. 37(6), Article 264 (December 2018).*

# *Informatik im Kontext der allgemeinen Bildung*

**Juraj Hromkovič**

## Was ist Informatik?

Obwohl die Informatik als eine relativ junge wissenschaftliche Disziplin bekannt ist, ist die informatische Denkweise seit jeher ein Teil der menschlichen Kultur. Die Informatik hat drei Wurzeln: die *digitale Informationsdarstellung*, die *Automatisierung* von menschlichen Tätigkeiten mit Algorithmen sowie die *Entwicklung der Computertechnologie*.

Der Ursprung der ersten Wurzel ist die Erfindung der ersten Schrift durch die Sumerer vor etwa 5400 Jahren. Dieser Schritt entstand aufgrund der häufigen Konfrontation mit einer typischen Informatikaufgabe: Die Daten von rund einer Million Einwohner Mesopotamiens, insbesondere Steuerschulden und -einnahmen, galt es außerhalb des menschlichen Gehirns abzuspeichern sowie effizient und übersichtlich zu verwalten. Im Grunde kann man hier von der ersten Big-Data-Krise der Geschichte sprechen. Big Data bedeutet ja nicht, eine gewisse große Datenmenge zu haben. Es bedeutet, so viele Daten zu haben, dass man es nicht schafft, sie mit der vorhandenen Technologie zu bearbeiten. Mit der *Verschriftlichung* von Steuerdaten wurde vor 5400 Jahren in Mesopotamien der erste große Schritt in Richtung *Digitalisierung* gemacht. Die digitale Darstellung ist nämlich nichts anderes als die Darstellung von Informationen durch eine Folge von Symbolen.

Mit der Abspeicherung von Informationen außerhalb des menschlichen Gehirns wurden Daten zu einem wertvollen Gut, das man bewahren und transportieren konnte. Somit entstand das Problem des *Datenschutzes*. Bereits vor rund 3500 Jahren schützte man gewisse Daten durch gut entwickelte *Geheimschriften*. Es sind Geheimschriften aus Hochkulturen in Ägypten, Palästina, Griechenland, Indien und China bekannt. Einige der damals entwickelten Konzepte verwenden wir bis heute. Im Laufe der Zeit wurde die Entwicklung von Schriften für unterschiedliche Zwecke – unter anderem Geheimschriften – zu einer Grundkompetenz der Informatik. Das Komprimieren, um möglichst kurze Informationsdarstellung zu erreichen, oder Information so zu kodieren, dass man Beschädigungen Ihrer Darstellungen vollständig reparieren kann, sind andere Beispiele. Die häufigste Aufgabenstellung der Informationsdarstellung ist es, Darstellungen zu entwickeln, die einen effizienten Umgang mit Daten ermöglichen. Das bekannteste und älteste Beispiel ist die Entwicklung von Zahlendarstellungen so, dass man mit den Zahlen schnell rechnen kann.

Die *Automatisierung* als zweite Wurzel der Informatik ist noch älter als die Datendarstellung. Die menschliche Kultur entstand dank der Fähigkeit, Wissen zu erzeugen und dieses zum Erreichen unterschiedlicher Zielsetzungen anzuwenden. Die informatische Komponente dabei war damals wie auch heute, *Vorgehensweisen zu entwickeln*, die zum gewünschten Ziel führen, zum Beispiel der Herstellung eines Werkzeugs. Statt „Vorgehensweisen" sagen wir heute „Algorithmen".

Es war die erste Form der Automatisierung, die der Menschheit eine unglaubliche Effizienz brachte. Es war viel einfacher, ein Experte für die Durchführung einer bereits entwickelten Vorgehensweise zu

https://doi.org/10.1007/s00287-019-01158-1
© Springer-Verlag Berlin Heidelberg 2019

Juraj Hromkovič
ETH Zürich,
Zürich, Schweiz
E-Mail: juraj.hromkovic@inf.ethz.ch

## Zusammenfassung

Dieser Artikel wurde für Lehrpersonen geschrieben, die in ihrer schulischen Ausbildung bzw. in der Lehramtsausbildung niemals mit Informatik in Berührung gekommen sind und dennoch vor der Aufgabe stehen, sie mindestens teilweise zu unterrichten. Unsere Ausführung ist bemüht, die Informatik im Kontext der ganzen Wissenschaft vorzustellen und somit Informatikkonzepte mit bekannten Themen zu verbinden. Die wichtigsten Nebenziele sind die Auflösung von Ängsten vor dem Unbekannten sowie die Vermeidung eines Wettbewerbs mit der Klasse über die Kenntnisse der neuesten Apps und der Hardware.

Die Message ist klar: Die Informatik ist ein Fach wie alle anderen Fächer mit etablierten Grundkonzepten, die man schrittweise in einem Spiralcurriculum vermitteln muss. In dieser Form trägt sie nicht nur bei zum Verständnis und zur Möglichkeit der Mitgestaltung unserer Umwelt, sondern auch zum Erreichen von Grundkompetenzen in anderen Fächern, insbesondere in der Mathematik und den Sprachen. Sie ist eine wichtige Vorbereitung auf die unbekannten Berufe der Zukunft, die wir zwar nicht kennen, die aber sicher durch die Automatisierung geprägt sein werden.


sein, als das dazu notwendige Wissen neu zu erzeugen. Eines der schönsten Beispiele aus der Antike ist der Satz des Pythagoras, der eine Technologie zum Bau von rechtwinkligen Ecken bot. Weil $3^2 + 4^2 = 5^2$ ist, reichte es aus, ein Dreieck mit Seitenlängen 3, 4 und 5 aufzuspannen, um einen rechten Winkel zu erzeugen. Zwischen der intellektuellen Fähigkeit, ein solches Dreieck aufzuspannen, und der, den Satz des Pythagoras zu beweisen, liegen Welten. Durch die reine Durchführung einer bereits entwickelten Vorgehensweise konnten allerdings die meisten Menschen in den Genuss unterschiedlicher Technologien kommen, ohne den anspruchsvollen und mühsamen Weg ihrer Entdeckung gehen zu müssen.

Auch die dritte Wurzel der Informatik, die *Computertechnologie*, entstand nicht erst im 20. Jahrhundert. Als Ursprung der Entwicklung der Computertechnologie gilt die mechanische Rechenmaschine von Gottfried Wilhelm Leibniz (1646–1716). Die erste universelle, programmierbare Maschine hat Charles Babbage im 19. Jahrhundert entworfen. Der erste Mensch, der programmierte und somit eine Maschine steuerte, war Lady Ada Lovelace (1815–1852).

Die Geschichte der Kommunikationsnetzwerke ist noch viel älter. Rauchzeichen beispielsweise wurden schon vor tausenden von Jahren als Technologie verwendet, um über weite Distanzen zu kommunizieren. Das erste moderne Networking verdanken wir dem Telegraphen (1837).

Die *Informatik als eigenständige Disziplin* entstand, als die folgenden zwei Voraussetzungen erfüllt waren und ein größerer Bedarf nach Fachleuten mit informatischen Kenntnissen aufkam:

- Viele Vorgehensweisen wurden bis ins Detail verstanden und so genau beschrieben, dass man keine Improvisationsfähigkeiten und somit kein großes intellektuelles Potenzial mehr brauchte, um sie durchzuführen. Ihre Durchführung war eher eine Routinearbeit.
- Es wurden Technologien entwickelt, welche die automatische Ausführung von Algorithmen (mathematisch präzise beschriebene Vorgehensweisen) ermöglichten.

Somit hat die *Informatik heute* die folgenden zwei Komponenten, die man nicht vollständig voneinander trennen kann:

- Die Informatik erforscht in Zusammenarbeit mit anderen wissenschaftlichen Disziplinen Sachverhalte und Prozesse in allen Bereichen der Forschung und der menschlichen Aktivitäten. Dank des daraus gewonnenen Verständnisses entwickelt die Informatik *Algorithmen zur Automatisierung* der untersuchten Tätigkeiten.
- Die Informatik entwickelt *Hardware- und Softwaretechnologien*, um immer mehr Verfahren automatisieren zu können. Dazu gehört auch die Entwicklung von Programmiersprachen, welche es dem Menschen ermöglichen, mit den Maschinen zu kommunizieren und sie mit der Durchführung von unterschiedlichen Tätigkeiten zu beauftragen.

## Beitrag der Informatik zur allgemeinen Bildung

Um *Zielsetzungen für ein Informatikfach* in der Schule zu formulieren, muss man die oben erwähnte Rolle der Informatik in der Entwicklung unserer

Zivilisation beachten. Man darf sich nicht auf das kurzlebige Know-how zu den neuesten Software- und Hardwareprodukten mit ihren Applikationen fokussieren. Erstens werden sich diese in ein bis zwei Jahren wieder verändern und zweitens schafft es die Lehrperson (LP) nur sehr schwer, vor technologiebegeisterten Schülerinnen und Schülern (SuS) einen Vorsprung in Sachen Anwendungswissen zu halten. Der wichtigste Grund, den Unterricht nicht auf die reinen Anwendungskompetenzen zu beschränken, liegt allerdings tiefer: Die Hauptaufgabe der Schule besteht darin, nachhaltigen Wissenstransfer zu ermöglichen und das intellektuelle Potenzial der jungen Generation zu entfalten. Die Bedienung von fertigen technologischen Produkten trägt sehr wenig zum wirksamen Training des Gehirns bei. Das reine Knöpfedrücken, der gleichzeitige Umgang mit vielen unwichtigen Details der Applikationen und ständiges Multitasking resultieren eher in fast garantierter Konzentrationsschwäche, welche die intellektuelle Unterentwicklung der Betroffenen zur Folge hat.

Wenn man Informatik unterrichtet, muss man auf die Vermittlung der informatischen Denkweise fokussieren. Diese können die SuS nur aus historischer Perspektive und Schritt für Schritt erwerben. Zur Illustration greifen wir ein ähnliches Beispiel aus der Physik auf: Den fundamentalen Physikunterricht der Volksschule würde niemand mit den neusten Errungenschaften der Physik und großen Theorien wie Quantenmechanik oder Relativitätstheorie beginnen. Stattdessen verfolgen wir zunächst die Entwicklung der physikalischen Denkweise von der Antike bis hin zur Newton'schen Physik. Gemäß diesem Prinzip werden die SuS auch die Automatisierungstechnologie nur dann verstehen, wenn sie deren Entwicklungsschritte nachvollziehen können.

Weil die Entwicklung der informatischen Denkweise stark mit der Entwicklung der Mathematik, der Sprache und der Technik verbunden ist, sehen die *globalen Zielsetzungen dieses Lehrmittels* wie folgt aus:

1. Sie sollten zum Verständnis der vom Menschen kreierten technischen Welt beitragen und zu deren aktiver Mitgestaltung anregen. Die SuS sollen zu Erfindern und Produzenten statt nur zu Konsumenten erzogen werden.
2. Durch Informatik sollten die Grundkompetenzen im Bereich Mathematik und Sprachen gestärkt werden.
3. Die konstruktive Denkweise der technischen Disziplinen sollte in die allgemeine Bildung eingeführt werden.

### Erfinder und Produzenten statt Konsumenten erziehen

Die Menschen erwerben Wissen nicht nur mit dem Ziel, die Welt zu verstehen, sondern auch, um eigene produktive Ziele zu erreichen. Die dabei entstehenden Technologien (z. B. Feuer, Bogen, Rad, Boot, Schrift, Hausbau, Buchdruck, mechanische Werkzeuge, Maschinen, Computer, Internet) geben starke Impulse zur immer schnelleren Entwicklung der menschlichen Zivilisation. Alle Schulfächer sollten zum Verständnis der vom Menschen erzeugten Welt beitragen. Die besondere Aufgabe des Informatikunterrichts bezieht sich auf die Gestaltung und die Steuerung der technischen Welt. Der Informatikunterricht verfolgt ein hohes Ziel: Wir wünschen uns als Schulabsolventen begeisterte und kreative Erfinder und Produzenten, nicht nur Konsumenten der digitalen Technologie.

Mit dem Informatikunterricht entzaubern wir die technische Welt, die auf Knopfdruck funktioniert. Wir zeigen altersgerecht, wie technische Systeme funktionieren und wie man sie bauen und steuern kann. Ausserdem bringen wir den SuS bei, dies selbstständig zu tun. Dabei entwickeln sie die Motivation und die Fähigkeit, immer komplexere Tätigkeiten automatisieren zu können.

### Grundkompetenzen der Mathematik und Sprache fördern

Dass der Informatikunterricht auch die Grundkompetenzen anderer Fächer stärkt, ist für die *Mathematik* naheliegend, weil die Informatik und die Mathematik stark verzahnt sind. Schon bei der Informationsdarstellung und der Beschreibung der Aufgabenstellungen im Schulbuch kommt die exakte Sprache der Mathematik vor.

Der Schwerpunkt eines guten Lehrmittels liegt aber in erster Linie in der *Förderung der Fähigkeit, Probleme zu lösen*. Im Unterschied zum klassischen Mathematikunterricht geht es im Informatikunterricht nicht darum, eine vorgegebene Lösungsmethode als fertiges Produkt der Wissenschaft zu erlernen. Beim Programmieren gibt es viele Lösungswege und es geht darum, einen davon selbstständig zu entdecken und seine Funktionalität auf Korrektheit zu überprüfen. In der Fachspra-

che der Informatik besteht ein Problem aus einer Vielzahl von Problemfällen, auch genannt Probleminstanzen. Somit besteht zum Beispiel das Problem des Lösens quadratischer Gleichungen aus unendlich vielen Problemfällen – aus allen konkreten quadratischen Gleichungen. In der Algorithmik sammelt man Erfahrungen beim Lösen konkreter Problemfälle, bis man eine allgemein funktionierende Strategie entwickeln kann. Dabei ist es nicht das Ziel, zu derjenigen Strategie zu gelangen, die man in der Wissenschaft nach langjähriger Optimierung erreicht hat. Vielmehr geht es um den Prozess der Entdeckung eines geeigneten Algorithmus. Gegebenenfalls kann ein Teil dieses Prozesses auch aus der Optimierung der vorhandenen Algorithmen nach gewissen Kriterien wie die verständliche Beschreibung oder Effizienz bestehen. Im Fokus steht also nicht nur das Kennenlernen von Produkten der Wissenschaft und deren Anwendungen, sondern vielmehr das Erlernen der Prozesse der Wissenserzeugung und der kreativen Gestaltung. Dabei gehen wir jeweils von einer anschaulichen Motivation aus und betrachten den ganzen Weg der Entwicklung bis hin zur Überprüfung und zur Korrektur eigener Vorstellungen.

Wenn man die Mathematik als eine Sprache versteht, die einerseits genaue Beschreibungen von Objekten und ihren Beziehungen ermöglicht und andererseits überprüfbare Argumentationen und Untersuchungsmethoden für das Erforschen der Realität bietet, dann fördert Informatikunterricht in gleichem Maß diese beiden allgemeinen Kompetenzen, die auch im Mathematikunterricht im Vordergrund stehen sollten.

Der Beitrag des Informatikunterrichts zur *Stärkung der Sprachkompetenzen* mag auf den ersten Blick überraschend sein und braucht somit eine weiter ausgreifende Erklärung. Wir unterscheiden hier drei Ebenen, die aber nicht ganz voneinander zu trennen sind. Die erste Ebene betrifft die Entwicklung der Schrift. Die zweite Ebene ist die Entwicklung der Sprache und die dritte Ebene betrifft deren Anwendung in einer verständlichen Kommunikation.

In „Einfach Informatik" folgen wir der Geschichte der Entstehung von Schriften, weil die ganze Informatik darauf basiert, dass man alle Informationen zunächst als Folgen von Symbolen darstellen muss. Dafür wählt man ein Alphabet als eine Menge von geeigneten Symbolen und baut dann Wörter mit gegebener Bedeutung als Folgen von Symbolen. Das Kreative an diesem Prozess ist, dass man im Informatikunterricht unterschiedliche Schriften konstruiert, abhängig davon, welche Ziele man verfolgt. Es gibt eine Vielfalt von Schriften für bestimmte Zwecke: Geheimschriften für den Datenschutz, spezielle fehlerresistente Schriften für den Datentransport sowie Schriften für Zahlendarstellungen (die effizientes Rechnen ermöglichen), zur kürzest möglichen Darstellung von Objekten (Komprimierung) oder zur Datendarstellung, die eine schnelle Suche nach gewünschten Informationen ermöglicht.

Durch die *Entwicklung eigener Schriften* verstehen die SuS den Prozess der Entstehung von Schriften nicht nur allgemein, sondern ahmen den Entstehungsprozess und seine Optimierung auch selbst nach.

Die SuS sehen Sprachen oft als fertiges Produkt, das man lernen muss. Beim Programmierunterricht lernen die SuS eine Sprache – genannt Programmiersprache – die der Computer versteht und mit der sie den Computer steuern können. Die Grundbausteine einer Programmiersprache sind (ebenso wie bei anderen Sprachen) Wörter, die eine bestimmte Bedeutung haben. Das Programm ist ein Text, durch den man dem Computer die Tätigkeit erklärt, die er ausüben soll. Ein guter Programmierunterricht vermittelt nicht eine Programmiersprache als fertiges Softwareprodukt. Stattdessen startet die LP mit einem sehr kleinen Wortschatz und die SuS entdecken dann den Bedarf nach neuen Wörtern, die es einem ermöglichen, sich einfacher auszudrücken. Die LP zeigt den SuS, wie sie dem Computer neue Wörter beibringen können, um in der zukünftigen Kommunikation effizienter zu werden. Dabei können die SuS Parallelen zur Entwicklung natürlicher Sprachen ziehen. Das Wichtigste ist, dass jede Schülerin und jeder Schüler die Möglichkeit erhält, eine eigene Version des Wortschatzes der Programmiersprache zu entwickeln und zu nutzen.

Auf der dritten Ebene – der Anwendung der Sprache in einer verständlichen Kommunikation – fördert die Programmierung die *Fähigkeit, sich genau auszudrücken*. Die SuS lernen, Vorgehensweisen unmissverständlich in natürlicher Sprache zu kommunizieren und letztlich in der Programmiersprache mit mathematischer Präzision zu formulieren. Diese Fähigkeit, sich unmissverständlich auszudrücken, ist außerdem hilfreich für die Beschreibung von Problemstellungen, für die Darstellung von Daten

mittels selbst ausgewählter Alphabete und allgemein für die abstrakte Darstellung von Objekten, Situationen und Beziehungen mittels Symbolen (Texten) und Graphen.

### Die konstruktive Denkweise in der allgemeinen Bildung stärken

Im Verlauf ihrer Entwicklung hat die Menschheit ihre Fortschritte durch eine Kombination von Wissenserzeugung und kreativen sowie gestalterischen Tätigkeiten erreicht. Alle neuen Technologien entstehen dank der intelligenten Anwendung des erzeugten Wissens und dank der Erfahrung, die auf Experimentieren und gestalterischem Probieren basiert. Das Wissen als Formulierung des Verstandenen sowie das Experten-Know-how, das man noch nicht genau ausformulieren oder zusammenfassen konnte, waren in unterschiedlichen Zusammensetzungen das Maß des Erfolgs. Es ist gut nachvollziehbar, dass in Zeiten der technischen Revolution die Fähigkeit, das Wissen exakt sprachlich zu formulieren, in der Schule Einzug gehalten hat. Dies ermöglicht klare Zielsetzungen und die Erarbeitung von Messinstrumenten für die im Unterricht erzielten Fortschritte (Prüfungsbewertung). Nicht greifbares Know-how eines Experten ist im Rahmen des Schulunterrichts schwierig zu vermitteln und deshalb ungeeignet.

Heute sind wir aber in einer anderen Situation. Die meisten einfachen Anwendungen der in der Schule vermittelten Produkte der Wissenschaft (z. B. das Lösen von Systemen linearer Gleichungen) sind automatisierbar. Im Berufsleben sind Fachleute, die zwar komplexe, aber trotzdem automatisierbare Tätigkeiten ausüben, immer weniger gefragt. Gesucht werden *handlungsfähige Experten*, die kreative Tätigkeiten ausüben. Wir können sie gezielt erziehen, indem wir sie Erfahrungen sammeln lassen. Auch dann, wenn der erreichte Fortschritt nicht so leicht zu messen ist.

Zusätzlich hat die Informatik vieles von dem schwer begreifbaren Erfahrungswissen der technischen Disziplinen präzise formuliert und gewissermassen systematisiert. Das ermöglicht einen anschaulichen und verständlichen Einzug der *Denkweise der technischen Disziplinen* in die Schule. Die angestrebten Ziele sind die folgenden:

- Durch Probieren und Experimentieren Lösungswege entdecken.
- Die Lösungswege mittels Software (Programmieren) und Hardware umsetzen.
- Selbstständig eigene Produkte auf Funktionalität, Effizienz und Benutzerfreundlichkeit testen und dementsprechend bewerten.
- Eigene Produkte optimieren und für neue Funktionalitäten erweitern.

Im guten Programmierunterricht werden zu einem gewissen Grad alle diese Zielsetzungen angesprochen. Auf sie zu verzichten ist nicht möglich, weil sie seit jeher für den Fortschritt der menschlichen Gesellschaft unvermeidbar waren und heute immer wichtiger werden. In diesem Lehrmittel steht der modulare Entwurf im Vordergrund, der all die genannten Zielsetzungen erfüllt. Die SuS lösen einfache Aufgabenstellungen und bauen kleine Bausteine (Programme), deren Funktionalität leicht zu überprüfen ist. Aus diesen Bausteinen bauen sie komplexere Programme, die wiederum Bausteine für noch komplexere Systeme werden usw. Wie weit man schon mit Anfängerinnen und Anfängern mithilfe dieser modularen Vorgehensweise kommen kann, könnte sogar erfahrene LP überraschen.

## Fachdidaktische Konzepte und Methodik

In diesem Teil stellen wir die Strategie der Implementierung der oben genannten Konzepte in unserer Lehrmittelreihe „Einfach Informatik“ vor, die in 18 Bänden ein Spiralcurriculum des Informatikunterrichts vom Kindergarten bis zum Abitur entwickelt. „Einfach Informatik“ baut auf die folgenden vier grundlegenden Konzepte auf, welche sich gegenseitig stark unterstützen und zusammen die Entwicklung der individuellen Persönlichkeit der SuS fördern.

1. Förderung des kritischen Denkens
2. Individuelle Förderung/Differenzierung durch selbstständiges Lernen
3. Entdeckendes Lernen
4. Lernen durch aktives Handeln

### Förderung des kritischen Denkens

Statt auf die Vermittlung fertiger Produkte der Wissenschaft (Fakten, Methoden, Definitionen, Anwendungsanleitungen) fokussieren wir auf die Prozesse der Wissenserzeugung und der selbständigen Gestaltung aus historischer Perspektive.

Detailliert bedeutet dies:

- Wir wecken in den SuS das Bedürfnis, die Welt verstehen und mitgestalten zu wollen, Problemstellungen zu formulieren und zu lösen.
- Wir unterstützen die SuS darin, Erfahrungen durch Beobachten und Probieren zu sammeln und somit eigene Vorstellungen über die untersuchte Materie aufzubauen.
- Wir leiten die SuS an, eigene Vorstellungen auf die Korrektheit oder eigene Produkte auf die korrekte Funktionalität zu überprüfen.
- Wir ermutigen die SuS, Instrumente zur Erforschung sowie zur Mitgestaltung der vom Menschen erzeugten Welt zu entwickeln oder vorhandene Instrumente zu verbessern.

Im Einzelnen geht es tatsächlich darum, nicht die fertigen Produkte der Wissenschaft zu lernen, sondern ihre Entdeckungen und Entwicklungen selbst erleben zu dürfen. Die Welt ändert sich schnell und es ist unvorhersehbar, welche Berufe es in Zukunft geben wird. Deswegen sollte die Bildung in erster Linie auf die Kreativität und Selbständigkeit ausgerichtet sein. Die SuS müssen Entdeckungen für sich selbst neu entdecken, gebaute Werke selber nachbauen und nach Bedarf verbessern. Nur wer solche Prozesse übt, kann später etwas Neues kreativ hervorbringen. Routinetätigkeiten werden automatisiert und das Wissen ist heute schon so umfangreich, dass man nicht einmal einen winzigen Bruchteil davon erlernen kann. Wenn die SuS aber in unterschiedlichen Bereichen lernen, wie Forschungsinstrumente gebaut und welche Errungenschaften dank Entdeckungen und Erfindungen möglich wurden, sind sie dazu fähig, selbstständig etwas Neues zu entdecken oder weiterzuentwickeln.

Eine derartige, auf eigener Erfahrung beruhende Bildung produziert einerseits kreative Menschen und andererseits mündige Bürgerinnen und Bürger, die Meinungen und Expertenempfehlungen nicht als vertrauenswürdige Tatsachen hinnehmen und alles auf Glaubwürdigkeit hin selbständig überprüfen wollen und können. Im Bereich der Technologie bedeutet dies nicht nur die Fähigkeit, die Funktionalität der entwickelten Systeme zu überprüfen und die Qualität zu beurteilen, sondern auch, die Möglichkeiten von Verbesserungen oder von Weiterentwicklungen zu sehen und sie umsetzen zu können.

## Individuelle Förderung/Differenzierung durch selbstständiges Lernen

„Einfach Informatik" ist selbsterklärend und gut geeignet, den Stoff selbständig zu erlernen. Dies ermöglicht nicht nur, das in der Schule nicht Verstandene nochmals vollständig zu repetieren, sondern auch im eigenen Tempo zu arbeiten und konkrete Themen nach individuellem Bedarf wiederholend anzugehen, bis man sie auch tatsächlich nachvollziehen kann.

Dank der Möglichkeit, mit dem Schulbuch selbstständig zu arbeiten, unterstützt das Lehrmittel die individuelle Förderung. Zum Beispiel kann die LP stärkere SuS selbstständig an vertiefenden Themen arbeiten lassen, während sie die schwächeren SuS intensiver und individueller fördern kann, um die Lernziele mit allen SuS zu erreichen. Die LP ist nicht darauf angewiesen, dass alle SuS das gleiche Vorwissen mitbringen und dass alle SuS gleichzeitig an denselben Themen arbeiten.

Das Lehrmittel ist als Spiralcurriculum aufgebaut und ermöglicht es, unterschiedliche Stufen der Vertiefung in allen Themenbereichen anzugehen. In diesem Begleitband vermitteln wir der LP, wie man mit diesen Möglichkeiten umgehen kann. Mit „Einfach Informatik" kann die LP den individuellen Begabungen, Fähigkeiten, Neigungen und Motivationen einzelner SuS im Unterricht wirklich gerecht werden. Neben den unterschiedlichen Schwierigkeitsstufen der Aufgabenstellungen und möglichen Vertiefungen im Schulbuch bietet dieser Begleitband zusätzliche Vertiefungsmöglichkeiten.

Insgesamt fördert das Schulbuch für alle SuS das Erreichen von hohen Kompetenzstufen. Dies wird einerseits möglich dank der *Zerlegung von komplexeren Themen* in eine Folge von sehr kleinen Schritten, welche die LP einen nach dem anderen aufbauen kann. Im Begleitband weisen wir die LP andererseits auf das Entstehen möglicher *Misskonzepte* hin. Zusammen mit den Hinweisen bieten wir der LP auch Ideen, wie sie gewisse Misskonzepte vermeiden oder notfalls rechtzeitig aufheben kann.

In Bezug auf die individuelle Förderung weisen wir die LP auch auf Erfahrungswerte aus dem Projektunterricht an mehr als 200 Schweizer Schulen hin. Immer wieder erleben wir, dass SuS, die in anderen Fächern eher als schwach eingestuft werden, in der Informatik auf einmal zu den stärksten SuS zählen und eine führende Rolle übernehmen. Die konstruktive Tätigkeit, verbunden mit intensivem

Probieren, Testen und Verbessern sowie die diskrete Darstellung der Welt scheinen *neue Dimensionen* zu sein, die in anderen Fächern nicht in dem Maß wie in der Informatik angesprochen werden. Mit der Informatik fördern wir also eine Gruppe von SuS, die sonst diese Bestätigung nicht erfahren würde.

### Entdeckendes Lernen

Einer der häufigsten Fehler bei der Einführung des Informatikunterrichts ist es, komplexen IT-Systemen so schnell wie möglich zu begegnen und damit zu arbeiten. Dabei können die SuS leider nichts Wertvolles entdecken, weil die Komplexität alle überfordert. Somit bleiben die SuS auf der Ebene der Benutzerkompetenzen ohne jedes tiefere Verständnis.

In „Einfach Informatik“ bauen wir auf entdeckendes Lernen. Wir folgen der historischen Entwicklung der Ideen, der Informatikkonzepte und der IT-Technologie. Wir schaffen zuerst die Motivationen, die ursprünglich in der Geschichte der Wissenschaft entstanden und die gut nachvollziehbar sind. Mit diesen gut nachvollziehbaren Zielsetzungen beginnen die SuS, ihre ersten Erfahrungen zu sammeln. Sie versuchen, *selbstständig Lösungsvorschläge zu erarbeiten* und beurteilen dann, inwieweit die gesteckten Ziele erreicht wurden.

Sehr markant tritt das beim Programmierunterricht zutage, wo es nicht die eine Lösungsmethode gibt, die es zu meistern gilt. Es gibt viele Lösungswege und das Ziel ist es, die SuS selbstständig einen funktionsfähigen Weg entdecken zu lassen und dessen korrekte Funktionsfähigkeit in der Anwendung zu überprüfen. Außerhalb des Programmierunterrichts stellen wir meistens historische Beispiele von Lösungsversuchen vor und lassen die SuS diese testen und bearbeiten, um selbst verbesserte Lösungen zu entwickeln. Dank der Zerlegung des Stoffes in sehr kleine Schritte ermöglichen wir den SuS einen *hohen Grad an Selbstständigkeit*, indem sie in kleinen Schritten voranschreiten und in angemessener Geschwindigkeit durch den Stoff geführt werden. In einer anderen Form kommt die Führung durch das Einbeziehen von historischen Beispielen zustande. Auf diese Art und Weise erreichen die SuS schrittweise den Wissensstand und die Expertenkompetenzen der Menschen desjenigen Zeitalters, in dem die Ideen entstanden. Dies ist für die SuS in der Regel sehr motivierend, insbesondere wenn sie es zusätzlich schaffen, die damals entstandenen Konzepte zu verbessern.

### Lernen durch aktives Handeln

Hier folgen wir der konstruktivistischen Methode von Jean Piaget, welche Seymour Papert als bekanntester Schüler von Jean Piaget mit seinem Konstruktionismus beim Programmieren und bei der Suche nach Lösungsstrategien verfeinert hat. Seymour Papert fokussierte mit seiner Programmiersprache LOGO sehr stark auf das Programmieren. Programme schreiben bedeutet, konstruktiv Konzepte zu entwerfen und Produkte zu bauen, ihre Funktionalität und ihre Relevanz zu überprüfen und sie nach Bedarf iterativ zu korrigieren, zu modifizieren oder weiter zu entwickeln. Wir beschränken uns hier aber nicht auf die Entwicklung und das Testen von Programmen. Durch unsere Vorgehensweise, welche es ermöglicht, die historische Entwicklung der fundamentalsten Konzepte der Wissenschaft zu verfolgen, ermöglichen wir den SuS, diese Entwicklung konstruktiv neu zu erleben. Es geht dabei um mehr als um das bloße selbständige Entdecken einiger Konzepte. Wir ermöglichen den SuS, mit dem Wissen des entsprechenden Zeitalters ähnliche oder sogar bessere Konzepte und Produkte konstruktiv zu entwickeln sowie ihre Stärken und Schwächen zu studieren und gegenseitig zu vergleichen. Die Ideen, auf die wir bauen, sind folgende:

- Das Herstellen von Produkten und die konstruktive Entwicklung von Konzepten ist lernfördernd und sehr nachhaltig, da die SuS aktiv sind und durch ihre Aktivität ihre Wissensstrukturen im Gehirn aufbauen und vernetzen.
- Auch unzugängliche Abstraktionen werden durch eigene Aktivitäten konkret fassbar.
- Die historische Perspektive der Suche nach ständigen Verbesserungen kombiniert mit persönlichem aktivem Handeln erzeugt eine hohe Motivation.

## Zusammenfassung

Die Schulinformatik bietet große Chancen. Einerseits für die Entfaltung des kreativen Potentials von Kindern und Jugendlichen, bezüglich eines tieferen Verständnisses unserer Welt und der Entwicklung wichtiger Kulturtechniken. Anderseits zur Verbesserung von Unterrichtskonzepten und Didaktik; weil

konkret durch „making things work“ die gesamten Prozesse der Wissenserzeugung und der Anwendung des Entdeckten zur konstruktiven Entwicklung von eigenen Produkten und der Überprüfung von deren Funktionalität erlernt werden. Dies führt zur Erziehung mündiger Bürgerinnen und Bürgern, die in der Lage sind, Behauptungen kritisch zu überprüfen. Beides gemeinsam ist der beste Treibstoff für den Innovationsmotor der Gesellschaft und somit eine wertvolle und unersetzbare Investition in die Zukunft.

Gehen wir weg vom oberflächlichen Nutzen der digitalen Technologie. Machen wir uns nicht abhängig von der Funktion von Produkten anderer. Nehmen wir die Steuerung, die Gestaltung und die Weiterentwicklung der digitalen Technik und somit die Gestaltung unserer Zukunft in die eigenen Hände.

# *Ohne Informatik keine Allgemeinbildung*

Jens Gallenbacher

## Einleitung

In einem Vortrag verblüffte der John-von-Neumann-Medaillen-Preisträger Christos Papadimitriou zunächst mit der initialen Frage „Was ist Informatik?“, vor allem durch seine (scherzhafte) Antwort „Informatik ist die einzige Wissenschaft, die man nicht mit einem Satz beschreiben kann“. Die Verblüffung weicht jedoch einer milden Bestürzung, wenn man die gleiche Frage Schülerinnen und Schülern, Lehrerinnen und Lehrern oder auch einfach Menschen im Einkaufszentrum stellt. Sogar bei Studierenden, die sich für einen Studiengang Informatik eingeschrieben haben, ist das Antwortbild sehr divergent. Zugegeben: Auch die Frage nach Physik ergibt nicht einheitlich genau die Antwort „Lehre von Materie und Energie“, aber meistens ist doch eine Vorstellung in diese Richtung erkennbar.

Im Bildungsfernsehen – privat und auch öffentlich-rechtlich – sind Biologen und Physiker immer diejenigen, die freundlich, mit einfachen Worten, aber sehr seriös über ihre Forschung oder wissenschaftliche Erkenntnisse Auskunft geben. Informatiker werden dagegen oft als Clowns mit Headsets dargestellt, die mit Trial-and-Error-Methoden Roboter gegen Wände fahren lassen oder Datenpakete mit witzigen Antennen auf dem Kopf durch die Gegend schicken [16]. Klar sind die Beiträge meist ironisch gemeint, festigen aber implizit leider ein Bild, das bei den Zuschauern sehr verbreitet ist. Auch in anderen bekannten Formaten wie der Sitcom „Big Bang Theory“ werden naturwissenschaftliche Erkenntnisse als theoretisch zu untermauern dargestellt, während auch komplizierteste Software selbstverständlich „nebenbei“ geschrieben wird.

Bei unserer Befragung von 55 Schülerinnen und Schülern der Jahrgangsstufen 8 bis 12 verschiedener Gymnasien waren als Antwort auf die Frage „Nenne ein Beispiel, das für dein Technikverständnis typisch ist“ hauptsächlich Geräte „Handy, XBOX, Auto, …“, „Dass man mit PCs arbeitet“, „etwas programmieren“ die Nennungen. Nur ganz wenige möchten hier etwas „entwickeln“. Das Zerlegen eines alten PCs wurde mit großer Mehrheit stark mit Technik verbunden, während das Entwickeln einer Geheimschrift bei den allermeisten Schülerinnen und Schülern gar nicht mit Technik in Zusammenhang gebracht wurde [10].

Daher: Danke, Christos, fürs Wachrütteln! Ich möchte mich in diesem Artikel den Ursachen für die Vorstellungen und möglichen Lösungen innerhalb und außerhalb des Systems Schule beschäftigen.

## Die Vorstellung des „Wissen-schaffens“

In der Geschichte bis zum Zeitalter der Aufklärung hatte man eine genaue Struktur für das Gewinnen neuer Erkenntnisse: Vor allem stand das Studium. Das bedeutete das Studium der wissenschaftlichen (und religiösen) Literatur. Wesentliche Wege zu neuen Erkenntnissen waren die Deduktion – der Schluss vom Allgemeinen auf das Spezielle, sowie die Induktion – der Schluss vom Speziellen auf das Allgemeine. Da man die Schlussfolgerungen strikt auf bereits vorhandenes Wissen beziehen musste, waren

https://doi.org/10.1007/s00287-019-01159-0
© Springer-Verlag Berlin Heidelberg 2019

Jens Gallenbacher
Technische Universität Darmstadt,
Johannes-Gutenberg-Universität Mainz
E-Mail: jg@di.tu-darmstadt.de

**Zusammenfassung**
Unsere Lebenswelt ist heute bereits von Informatiksystemen durchdrungen, insbesondere aber gestaltet nach Prinzipien, die in der Informatik und den Ingenieurwissenschaften entwickelt wurden. Das gilt sicher umso mehr für die Lebenswelt heutiger Schülerinnen und Schüler, auf die wir unsere Kinder durch Bildung vorbereiten möchten. Die Realität ist jedoch, dass über alle Generationen hinweg viele Menschen eine sehr unterschiedliche, oft sehr einseitige oder sogar fehlleitende Vorstellung davon haben, was Informatik überhaupt ist. Die Lehrerbildung kann und muss Konzepte einer modernen Lebenswelt berücksichtigen. Um dieses Ziel zu erreichen, ist etwa eine Lehrveranstaltung für zentrale Ideen und Werkzeuge für alle Lehrämter sinnvoll, wie sie im Rahmen der Qualitätsoffensive Lehrerbildung an der TU Darmstadt eingeführt wurde.

neue Ideen quasi nur über Umwege einführbar. Auch die Induktion galt bereits als sehr umstritten, weil man natürlich streng genommen auch bei sehr vielen speziellen Indikatoren für eine allgemeine Regel nicht ausschließen kann, dass sie in Ausnahmefällen nicht gilt.

Naturwissenschaftler waren noch nicht als solche etabliert. Galileo Galilei formulierte zwar bereits Anfang des 17. Jahrhunderts Prinzipien für naturwissenschaftliche Forschung durch Experimente und Beobachtung, war damit aber seiner Zeit weit voraus: Isaac Newton wurde auch ein Jahrhundert später noch „nur“ als Philosoph anerkannt, nicht als Naturwissenschaftler. Die Lebenswelt war damals geprägt vom Glauben. Gestaltung bezog sich weitgehend auf Kunst und Architektur, weniger auf profane, alltägliche Dinge.

Die Industrialisierung schaffte im 19. Jahrhundert eine neue Sicht auf die Lebenswelt: Dampfmaschinen machten unabhängig von Flüssen als Antrieb von Wasserrädern, Kunstdünger revolutionierte den Ackerbau und reduzierte auf diese Weise ganz konkret Hungersnöte. Das Verständnis der Wirkprinzipien der Natur war nun wichtige Grundlage der Gesellschaft. Das „Entdecken“ neuer physikalischer oder chemischer Zusammenhänge wurde nun zunehmend als sinnvoll angesehen.

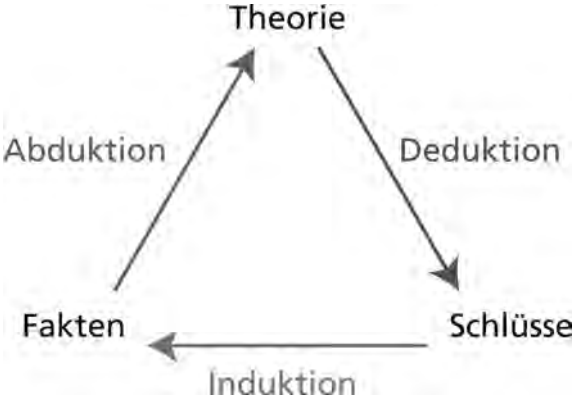



***Abb. 1 Diagramm der Peirce'schen inferentiellen Lerntheorie in einer abgewandelten Version nach [12]***

Die Beobachtung und die Formulierung von daraus abgeleiteten, neuen Ideen – den Hypothesen – ist keine direkte Schlussfolgerung aus irgendeiner Richtung, gleichwohl trotzdem sehr relevant. Charles Sanders Peirce führte daher etwa 1900 ein formales Modell für Erkenntnisgewinnung ein, das zur Induktion und Deduktion die Abduktion berücksichtigt [14]. Von einer Hypothese kann man dort deduktiv mögliche Konsequenzen ableiten, um diese danach, in Experimenten induktiv, forschend zu verifizieren oder zu falsifizieren. Die Abduktion stellt damit keine revolutionäre Idee dar, sondern ist nur die Anerkennung eines wissenschaftlichen Prinzips, das die Menschheit praktisch „schon immer„ angewandt hat, um neue Erkenntnisse über die sie umgebende Natur zu gewinnen. Mit dieser Anerkennung hielten die Naturwissenschaften dann auch Einzug in das Bildungssystem. An den Universitäten wurden naturwissenschaftliche Fakultäten etabliert und in Disziplinen unterteilt. So emanzipierte sich im 19. Jahrhundert vielerorts die Chemie als eigenständige Disziplin von der Medizin.

Auch das System Schule kam aufgrund der Notwendigkeiten der Industrialisierung nicht mehr um einen entsprechenden naturwissenschaftlichen Unterricht herum, dieser war mit zwei von 20 Stunden in den gymnasialen Stundentafeln des 19. und beginnenden 20. Jahrhunderts jedoch noch sehr untergeordnet ausgewiesen. Einige Realschulen oder auch Realgymnasien maßen dem Fach allerdings schon damals deutlich mehr Bedeutung und Raum zu. Der Übergang zu den ausdifferenzierten naturwissenschaftlichen Fächern erfolgte an den Gymnasien je

nach Gebiet erst etwa ein Jahrhundert nach dem entsprechenden Vorgang an den Universitäten. Der Anteil an der gesamten Stundentafel ist mit 12,5 % (Land Hessen, G8, Stand 2018) allerdings nicht wesentlich gesteigert worden.

Abduktion als Möglichkeit, erklärend, ableitend Hypothesen über bereits vorhandene materielle oder immaterielle Artefakte zu bilden, ist damit aber im Curriculum der Schulen vorgesehen, wenn auch meistens nicht unter dieser Bezeichnung. In den Fachprofilen Lehrerbildung der KMK [11] ist etwa von „hypothesengeleitetem Experimentieren, Modellieren und Vergleichen“ (Biologie) die Rede oder „Hypothesen- und Modellbildung“ (Sozialkunde/Politik/Wirtschaft) oder „Hypothesenbildung und -überprüfung“ (Sachunterricht in der Grundschule).

Bis Ende des letzten Jahrhunderts war diese naturwissenschaftlich erklärende Herangehensweise auch weitgehend konsistent mit der Lebenswelt der Schülerinnen und Schüler. Auch wenn die Umwelt immer mehr durch von Menschen geschaffene Artefakte beeinflusst wurde, so galten auch für sie die Naturgesetze entsprechend. Eulers Ableitung des zweiten Newton'schen Gesetzes „Kraft gleich Masse mal Beschleunigung“ ist eben auch für ein Auto gültig, für das man einschätzen muss, ob man davor gefahrlos die Straße queren kann. Gestaltete Elemente, die mit den einfachen Regeln der schulischen Erklärmodelle nicht mehr begreifbar waren, waren meistens den „Experten“ vorbehalten. Das galt etwa für Kernkraftwerke genauso wie für komplexe Computersimulationen.

## Die gestaltete Lebenswelt des 21. Jahrhunderts

Mit zunehmender Virtualisierung, Verlagerung menschlicher Kommunikation in soziale Netzwerke, der „Jeder-Jederzeit-Überall-Erreichbarkeit“ und weitere Faktoren wird die Lebenswelt aller Menschen zunehmend dominiert von menschlich gestalteten Artefakten, die natürlichen Gesetzen nicht unbedingt folgen.

Selbstverständlich kann man solche Artefakte zunächst in herkömmlicher Art und Weise betrachten und analysieren. Neben der Manganknolle, die vor einem Vulkan gefunden wurde und der Gewässerprobe der lokalen Quelle, wird im Unterricht zum Beispiel ein PC auseinandergenommen und analysiert. Der Zirkel aus Abduktion, Deduktion und


***Abb. 2 Fiktive Schlagzeile***

Induktion funktioniert auch damit und die Schülerinnen und Schüler „lernen“, ihre Lebenswelt zu verstehen. Ein höchst relevantes Element fehlt jedoch, quasi der Schlüssel zur vollwertigen Teilhabe an dieser modernen Lebenswelt: Der Impuls und das Selbstvertrauen, diese aktiv und passiv zu gestalten.

Das sei illustriert anhand eines plakativen Beispiels: Die fiktive Schlagzeile ist nicht bei den „Fake News“ einzuordnen, sondern beruht durchaus auf einer Studie vom Mai 2011 [15], in der 721 Millionen Menschen mit 69 Milliarden Freundbeziehungen untersucht wurden – eine Größenordnung, die für empirische Studien äußerst bemerkenswert erscheint. Eine zugrunde liegende Hypothese war, dass die meisten Menschen weniger Freunde haben als Freundesfreunde.

Einbezogen wurden alle Teilnehmerinnen und Teilnehmer des Dienstes Facebook, die länger als vier Wochen angemeldet waren und mindestens eine Freundbeziehung hatten. Im Ergebnis wurde tatsächlich festgestellt, dass ca. 84 % der „Studienteilnehmerinnen und Studienteilnehmer“ weniger Freunde hatten als der Durchschnitt ihrer Freunde.

Eine wichtige Frage ist allerdings, welche tatsächliche Relevanz diese Erkenntnis hat. Wir Informatiker kennen für Problemlösung und Problemanalyse das geistige Werkzeug der Modellbildung und wenn wir dieses für die Fragestellung an einem Beispiel anwenden, kommen wir etwa für eine „normale“ Situation mit neun Freunden zu einem Graph wie in der nächsten Abbildung: Knoten repräsentieren die Personen, Kanten die Freundschaftsbeziehungen. Ein sehr kontaktfreudiger Mensch steht hier im Mittelpunkt und hat recht

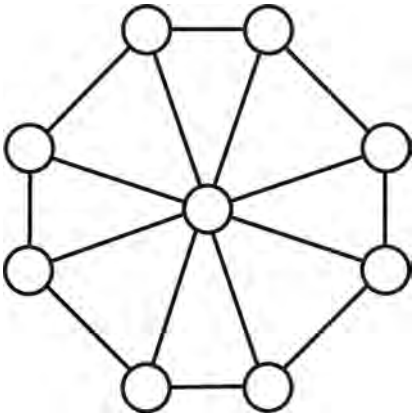


*Abb. 3 Freunde-Graph*

viele Freundschaften. Um ihn herum gibt es mehrere Cliquen, die neben ihm nur zwei weitere Freunde kennen.

Damit haben wir eine Person mit acht Freunden und durchschnittlich drei Freundesfreunden sowie acht Personen mit drei Freunden, aber durchschnittlich 4,67 Freundesfreunden. So haben „nachweislich" acht von neun Personen (also sogar knapp 89 %) weniger Freunde als der Durchschnitt ihrer Freunde. Das Beispiel bestätigt also einerseits das Ergebnis der Studie, andererseits kann man daran auch relativ leicht erahnen, dass Menschen mit vielen Freunden immer häufiger in die Freundesfreundestatistik eingehen als solche mit wenigen Freunden und damit das Facebook-Resultat nicht mehr sonderlich überraschend ist.

Zu dem gleichen Ergebnis kommt auch tatsächlich zwanzig Jahre vor der Ugander-Studie Scott Lauren Feld, der dies als „Freundschaftsparadox" bezeichnet [4]: Wenn in einer Gruppe mit Freundschaftsbeziehungen auch nur zwei Personen unterschiedlich viele Freunde haben, hat immer die Mehrheit von Personen weniger Freunde als der Durchschnitt ihrer Freunde.

Hier haben wir nun zwei unterschiedliche Betrachtungsweisen: Die Sicht von außen, die mit dem Peirce'schen Dreieck analysiert wird. Allerdings ist das Artefakt der Analyse ein soziales Netzwerk: etwas von Menschen Gestaltetes. Während der Graph Ende des letzten Jahrhunderts noch weitgehend ein Modellierungswerkzeug war, um einen Ausschnitt der Realität zu beschreiben, muss man sich in der Generation Facebook immer mehr fragen, ob der – von Menschen gestaltete und in einem Informatiksystem manifestierte – Graph nicht den Ausgangspunkt darstellt. Manche Stimmen prognostizieren gar, dass die Entsprechungen in der Zukunft einmal den maßgeblichen Teil der Welt darstellen werden.

Egal, wie man darüber denkt: Menschliche Gestaltung dominiert schon faktisch die Lebenswelt der Schülerinnen und Schüler und damit funktionieren die Peirce'schen Erklärmodelle nur noch teilweise: Gestaltung muss prominent einbezogen werden!

## Mit Peirce ins 21. Jahrhundert: Die Emanzipation der Informatik

Diese menschliche Gestaltung in unserer Lebenswelt wird nicht nur bei sozialen Netzwerken sichtbar: Mitfahr-Apps mischen den Fahrgastmarkt auf, Industrie 4.0 verspricht, Zusammenarbeit ganz neu zu denken, Versicherungen können über „Big Data" deutlich individuellere Angebote machen, Deep Learning präsentiert uns Ergebnisse, die wir inzwischen tatsächlich mit „künstlicher Intelligenz" verbinden.

Die Vordenker dieser Gestaltung sind an den Universitäten meistens im Bereich der Ingenieurwissenschaften und der Informatik zu finden. Bereiche, die sich oft Mitte des 20. Jahrhunderts von den Naturwissenschaften und der Mathematik emanzipiert haben. Die von ihnen generierten Erkenntnisse lassen sich mit dem Peirce'schen Dreieck zwar analysieren, aber nicht in ihrer Entstehung erklären. Es handelt sich um mehr als Hypothesen: Durch menschliche Gestaltung wird unsere Lebenswelt dauerhaft bereichert und erweitert. Die Abbildung zeigt einen Vorschlag, Konstruktion in das Peirce'sche Modell aufzunehmen.

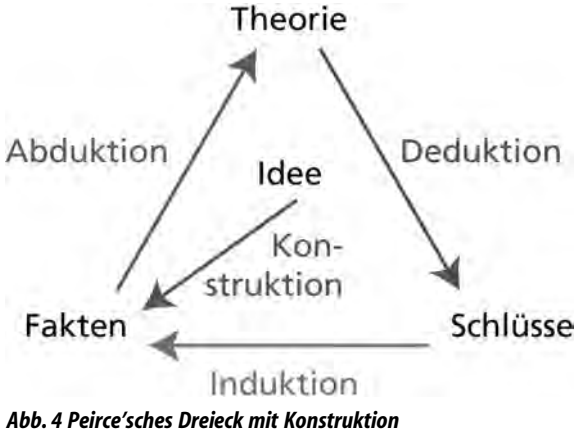



*Abb. 4 Peirce'sches Dreieck mit Konstruktion*

Durch Konstruktion werden neue Fakten geschaffen, die dann – wie bereits am Beispiel gezeigt – durch Abduktion, Deduktion und Induktion analysierbar sind.

Konstruktion ist im System Schule nicht unbekannt! Sie findet sich in klassischen Fächern wie Kunst oder Deutsch. Niemand würde erwarten, dass eine Schülerin und ein Schüler ein Gemälde von Picasso analysiert, ohne sich vorher selbst einmal mit Stift und Pinsel bemüht zu haben. Eine Interpretation des Gesamtwerks von Goethe erfolgt erst, nachdem eigene Erfahrungen mit dem Schreiben einfacher Aufsätze gesammelt wurden.

Im Bereich der Artefakte, die unsere Lebenswelt wirklich nachhaltig prägen, wird jedoch fast ausschließlich analysiert: Medienpädagogische Artikel erläutern eloquent und unter Einbeziehung quantitativer Empirie die Folgen sozialer Netzwerke. Der Beitrag „Big Data" der Bundeszentrale für politische Bildung, Quelle vieler Schulen für Unterrichtsmaterial, besteht aus sechs Artikeln von Journalisten, Juristen, Soziologen, Politologen, Medizinern, die das „Phänomen" von außen betrachten und analysieren. Ein Informatiker kommt nicht zu Wort [3].

Woran liegt das? Ist nicht Informatik als Schulfach in den Bundesländern angekommen – direkt unter dem Namen oder zumindest als Curriculum für die „klassischen" Fächer? Schon – aber betrachten wir die Sozialisation der Lehrerinnen und Lehrer in den Universitäten. Die Informatik – insbesondere die Fachdidaktik als Wissenschaft zwischen Fach und Grundwissenschaften – ist geprägt von einer Kultur, die Gestaltung zwar als sinnvolles „Handwerk" betrachtet, die lange Zeit jedoch die formale wissenschaftliche Erkenntnis und damit zum Beispiel wichtiges Kriterium für die Validität einer Dissertation, ausschließlich in den klassischen Methoden gesehen hat.

Mit „Unterrichtsentwicklungsforschung" emanzipiert sich die Fachdidaktik allerdings zunehmend von diesem Gedanken und lässt für sich selbst zu, dass Gestaltung per se Wissen schafft, indem dadurch neue Elemente in die Welt gebracht werden. Freilich – und das geht auch aus dem erweiterten Peirce'schen Modell hervor – müssen die neuen Artefakte immer noch mit Deduktion, Induktion und Abduktion in Bezug zu den gesicherten Erkenntnissen verankert werden! Verändert wird allerdings die Betrachtung, was für die Erkenntnis elementar ist.

Hieran wird die Notwendigkeit des Umdenkens in der gesamten deutschsprachigen Gesellschaft sichtbar: In einem humanistisch geprägten Umfeld sind Innovationen so lange anzuzweifeln, bis sie (unter Verwendung möglichst vieler Fremdworte und Literaturreferenzen) „bewiesen" wurden. Man tut sich traditionell viel schwerer als in anderen Kulturkreisen, einfach die Faszination der Technik auf sich wirken zu lassen. Damit wird aber die Anerkennung eher den Analysten zuteil und nicht den Gestaltern. Dadurch werden die wenigen „Gestalter", die andernorts – sicher in dieser extremen Ausprägung ebenfalls zu Unrecht – als Messias gefeiert werden, eher als zweifelhafte Personen angesehen: Die Subkultur der „Nerds" ist geboren, die „Anderen", die irgendetwas machen, das die „Normalen" nicht verstehen. Auf diese Weise werden auch die „Normalen" erfolgreich davon abgehalten, Gestaltung verknüpft mit gesellschaftlicher Verantwortung als die „eigene Mission" zu betrachten.

Wie kann und muss man diese Einsicht für das System Schule denken? Im Mittelpunkt steht hier der Auftrag der Allgemeinbildung! Man kommt nicht umhin, Konstruktion als ein Element anzuerkennen, das unsere Lebenswelt in massiver, in einigen Szenarien entscheidender Weise prägt. Schülerinnen und Schüler von heute sind nicht nur gefragt, passiv entsprechende Artefakte zu analysieren oder – ebenfalls passiv – die daraus resultierenden Systeme zu bedienen. Schülerinnen und Schüler von heute werden die Artefakte und Systeme der zukünftigen Lebenswelt gestalten! Allgemeinbildung bedeutet dabei einerseits, dass alle die Chance haben sollten, an dieser Gestaltung teilzuhaben – aktiv, indem sie selbst konstruieren, oder auch passiv, indem sie auf Basis ihres tieferen Verständnisses der zugrunde liegenden Wirkprinzipien und Modelle an Entscheidungen in Bezug auf den Einsatz teilhaben. Allgemeinbildung bedeutet andererseits, die kulturelle Kohärenz zu erhalten und schafft damit den Auftrag, zwei parallele Gesellschaften zu verhindern: die der „Nerds", die Systeme gestalten können und die der „Anderen", die die Produkte der Nerds wie Naturphänomene betrachten und mit ihnen gleich unabänderlichen Fakten umgehen.

Allgemeinbildung wurde in treffender Weise von Hans Werner Heymann auf sieben Aufgaben der Schule zurückgeführt [9]. Die bereits erwähnte kulturelle Kohärenz ist eine davon, aber auch Lebensvorbereitung, Einübung in Verständigung

und Kooperation, Weltorientierung, kritischer Vernunftgebrauch, Entfaltung von Verantwortungsbereitschaft und Stärkung des Schüler-Ichs – legen nahe, Konstruktion an den Schulen ernst zu nehmen [6].

## Strukturelle Chancen der Lehrerbildung

Bei Modellen der Erkenntnisgewinnung handelt es sich um zentrale Kompetenzen aller Fachdisziplinen, genau wie zum Beispiel bei der Betrachtung allgemeinbildender Aufgaben oder wissenschaftlicher Arbeitstechniken. Der Wissenschaftsrat forderte 2001 bereits für die Ausbildung von Lehrerinnen und Lehrern: „Hochschulausbildung soll die Haltung forschenden Lernens einüben und fördern, um die zukünftigen Lehrer zu befähigen, ihr Theoriewissen für die Analyse und Gestaltung des Berufsfeldes nutzbar zu machen und auf diese Weise ihre Lehrtätigkeit nicht wissenschaftsfern, sondern in einer forschenden Grundhaltung auszuüben. Der Erwerb dieser Kompetenz zur Vermittlung aktuellen disziplinären Wissens, verbunden mit reflexivem Berufswissen, soll in fachwissenschaftlichen, erziehungswissenschaftlichen und didaktisch-methodischen Studien erreicht werden."

Im Rahmen der Qualitätsoffensive Lehrerbildung hat die Technische Universität Darmstadt daher für alle Gymnasiallehramtsstudierenden einen Vernetzungsbereich eingerichtet, in dem unter anderem über die integrierte Veranstaltung „Zentrale Ideen und Werkzeuge" entsprechende Kompetenzen interdisziplinär erarbeitet werden. Forschendes, projektorientiertes Lernen ist zentral im Modul „Zentrale Ideen und Werkzeuge (ZIW)". Dieses Modul kann für Fächerkombinationen mit MINT-Fach im Vernetzungsbereich gewählt werden und ist für Lehramtsstudierende, die kein MINT-Fach studieren, obligatorisch [7].

Nach kurzen Impulsen in Form einer Vorlesung erarbeiten Projektteams aus vier oder fünf Studierenden unterschiedlicher Fächer eigenständig Unterrichtsskizzen zu einem sehr weit gefassten Thema, das jeweils dem gesamten Jahrgang vorgegeben ist. Thema der Pilotierung im Sommersemester 2017 war zum Beispiel „Der intelligente Kühlschrank", bei dem über Nachhaltigkeit, Funktionsprinzipien, Kommunikationsfähigkeit eines Haushaltsgeräts aus der Lebenswelt der Schülerinnen und Schüler nachgedacht werden sollte. Thema im Sommersemester 2018 war „To pixel or not to pixel" – nach einem Vortrag des Mathematikers Sir Michael Francis Atiyah, der 2017 über diskrete und stetige Modelle unserer Welt referiert hat [1].

Hauptaufgabe der Studierenden ist es, anhand der Themenvorgabe vor allem zentrale Ideen und Werkzeuge in den schulischen Kontext zu bringen und mit der Lebenswelt der Schülerinnen und Schüler zu verknüpfen. Die Vorgehensweise ist dabei an das forschungsorientierte Lernen nach Huber angelehnt. Die besten Ergebnisse werden im Rahmen einer (internen) Konferenz durch einen Peer-Review-Prozess ausgesucht und vorgestellt.

Ziel ist also nicht die einseitige Ausrichtung von Lehrpersonen auf Konstruktion und damit ausschließlich ingenieurmäßiges Denken, sondern eine Einbindung der hier geschilderten, neuen Betrachtung von Erkenntnisgewinnung in den bisherigen Kontext, verbunden mit der Chance, dieses zu nutzen, um den Zusammenhalt und Zusammenhang der unterschiedlichen Fachdisziplinen zu stärken. Das studentische Feedback zur Pilotierung war, von Bemerkungen zu organisatorischen Anlaufschwierigkeiten abgesehen, durchweg positiv, der selbst eingeschätzte Kompetenzgewinn hoch.

Diese Veranstaltung wird vom Fachbereich Informatik verantwortet! Das ist wichtig, denn es impliziert, dass Informatik nicht nur ein Selbstverständnis dafür hat, die Lebenswelt mit Informatiksystemen zu bestücken, sondern insbesondere informatische Denkweisen, informatische Modellierung, informatische Gestaltung und selbstverständlich auch informatische Verantwortung in alle Bereiche einzubringen. Die Lehrerbildung stellt hier einen ganz entscheidenden Faktor dar, weil sie Multiplikatoren hervorbringt!

Die Botschaft muss sein: „Die Systeme sind zwar kompliziert, aber nicht nur die Bedienung ist einfach, sondern auch die dahintersteckenden Konstruktionsprinzipien! Gestaltung ist nicht nur die Sache von Vielen, sondern auch die Möglichkeit!".

## Curricular-inhaltliche Implikationen

An einem Beispiel möchte ich schließlich noch illustrieren, welche Implikationen die erweiterte Sicht auf Konstruktion in Bezug auf Lehrinhalte, vor allem aber auf die spezifischen Lernziele haben kann.

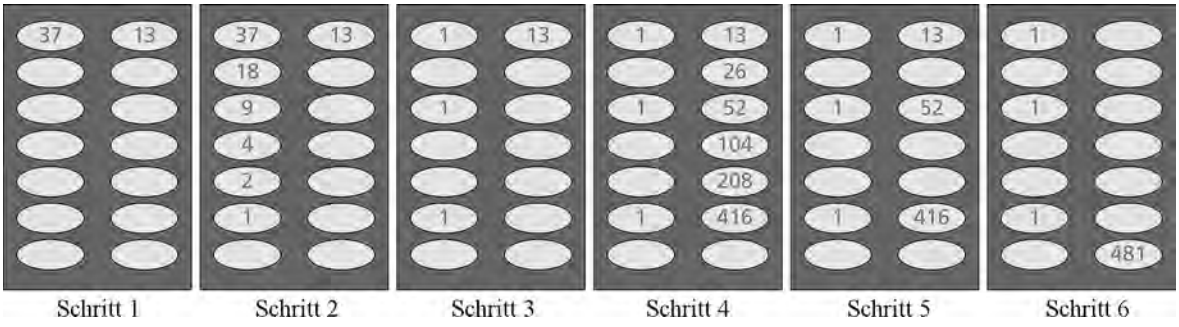



***Abb. 5 Ägyptische Multiplikation mit den Multiplikanden 37 und 13***

Das binäre Zahlensystem wird heute zweifelsohne fest mit dem Computer und der Informatik verknüpft und ist daher auch in nahezu allen Curricula als Fachinhalt enthalten. Man kann es sehr gut etwa anhand von Binärkarten oder einer Kombination aus Binäruhr und Binärkarten erklären [8], es gibt vielfältige Aktivitäten zu Binärzahlen, etwa auch in [2].

Wie motivieren wir aber unterrichtlich die Beschäftigung mit diesem doch auf den ersten Blick unhandlicheren Zahlensystem? Standardmäßig werden hier oft Erklärungen verwendet, die den folgenden zwei Ansätzen zuzuordnen sind:

1. „In Rechnersystemen müssen Signale übertragen werden und diese Signalübertragung ist bei zwei Zuständen sicherer, der Signal-Rausch-Abstand entsprechend höher."

Ja – das mag so sein. Trotzdem wird in modernen Datenübertragungsprotokollen mit deutlich mehr Zuständen gearbeitet, um die Effizienz zu steigern. Bei VDSL sind es zum Beispiel 256. Das Erklärmodell ist daher nicht stichhaltig.

2. „Zuse hatte Relais zur Verfügung und die haben zwei Schaltzustände."

Auch wenn Konrad Zuse genau genommen für die Z1 gar keine Relais, sondern selbstgefeilte Logikeinheiten aus Metallplättchen verwendet hat, bestanden die ersten voll funktionsfähigen Modelle freilich aus Relais. Allerdings hätte Zuse auch die damals in der Telefontechnik üblichen Drehwähler zur Verfügung gehabt – quasi Relais mit 10 Schaltzuständen. Auch diese Erklärung kann so also nicht stimmen.

Um einer sinnvollen Begründung auf die Spur zu kommen, folgen wir einem historisch-genetischen Ansatz [13] und vollziehen eine überlieferte Methode aus dem antiken Ägypten nach, wobei die Methode auch als „Äthiopische Multiplikation" und „Russische Bauernmultiplikation" bekannt ist. Bereits damals brauchte man eine vertrauenswürdige Methode, zum Beispiel den Gesamtpreis für den Handel über 37 Amphoren zu 13 Dinar zu bestimmen, ohne dass Verkäufer und Käufer rechnen konnten.

Bestimmt wurde das Ergebnis mithilfe zweier Spalten von Kuhlen im Sand und einer Menge kleiner Steinchen – hier repräsentiert durch die Anzahlen. Dazu musste stur der folgende Algorithmus durchgeführt werden:

1. Füllen der obersten beiden Kuhlen mit den Multiplikanden.
2. In der linken Spalte: In jede der noch leeren Kuhlen wird von oben nach unten jeweils die Hälfte (abgerundet) des Inhalts der Kuhle darüber gelegt.
3. In der linken Spalte: In jeder Kuhle werden – solange das geht – paarweise Steinchen entfernt.
4. In der rechten Spalte: In jede der noch leeren Kuhlen wird von oben nach unten jeweils das Doppelte des Inhalts der Kuhle darüber gelegt.
5. In der rechten Spalte: Jede Kuhle, die sich neben einer leeren Kuhle (der linken Spalte) befindet, wird geleert.
6. In der rechten Spalte: Alle verbleibenden Steinchen zusammenzählen. Das ist das Ergebnis.

Die Abbildung zeigt das Beispiel mit 37 und 13. Bei der Ägyptischen Multiplikation handelt es sich um ein Verfahren, das eine recht komplizierte Operation – die Multiplikation zweier prinzipiell beliebig großer Zahlen – auf eine immer gleichbleibende Folge einfacher Handlungsschritte zurückführt. In diesem Fall hauptsächlich „Halbieren", „Verdoppeln" und „Zusammenzählen". Das nennt man „strukturierte Zerlegung" und es ist die Grundidee eines Algorithmus und damit eine fundamentale Idee der Informatik.

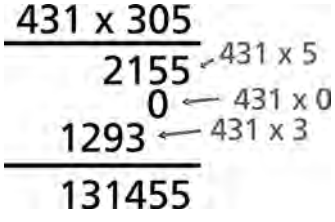



***Abb. 6 Schriftliche Multiplikation***

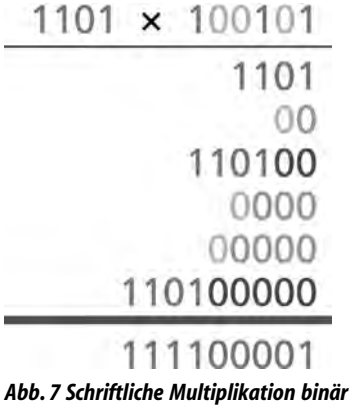



***Abb. 7 Schriftliche Multiplikation binär***

Um zu ergründen, was das mit dem Binärsystem zu tun hat, machen wir einen weiteren Schlenker und schauen uns die schriftliche Multiplikation an, wie wir sie in der Grundschule erlernen. Im Beispiel sehen Sie das für $431 \times 305$. Auch hierbei führen wir etwas Komplizierteres auf einfachere Berechnungen zurück – in diesem Fall auf das kleine Einmaleins und schriftliches Addieren, um die einzelnen Ergebnisse zusammenzuzählen.

Nun führen wir genau dieses Verfahren mit Zahlen in binärer Schreibweise durch. Die Lösung für $1101 \times 100101$ sehen Sie in Abb. 7 dargestellt. Zur Verdeutlichung sind die Zahlen nicht nur eingerückt, sondern auch um die korrekte Anzahl Nullen ergänzt.

Bereits jetzt sollten die Schülerinnen und Schüler merken, dass die schriftliche Multiplikation im Binärsystem recht einfach war. Wenn wir nun noch den Zusammenhang mit der Ägyptischen Multiplikation herstellen und zeigen, dass das Verfahren prinzipiell identisch mit unserer schriftlichen Multiplikation mit Binärzahlen ist, wird das Prinzip noch offensichtlicher. Die Abbildung zeigt das Verfahren mit der binären Repräsentation von 37 und 13. Eine ausführliche Anleitung findet sich in [8] und [5].

Prinzipiell haben bereits die alten Ägypter genauso multipliziert, wie das heute noch jeder Computer macht: Auf Basis des binären Zahlensystems. Ob das den Mathematikern der Pharaonen genauso bewusst war, ist dabei unerheblich. Wichtig ist, dass die Begründung für das Vorgehen damals wie heute in gleicher Weise zutrifft: Weil es wunderbar einfach ist! Basis ist quasi die Multiplika-

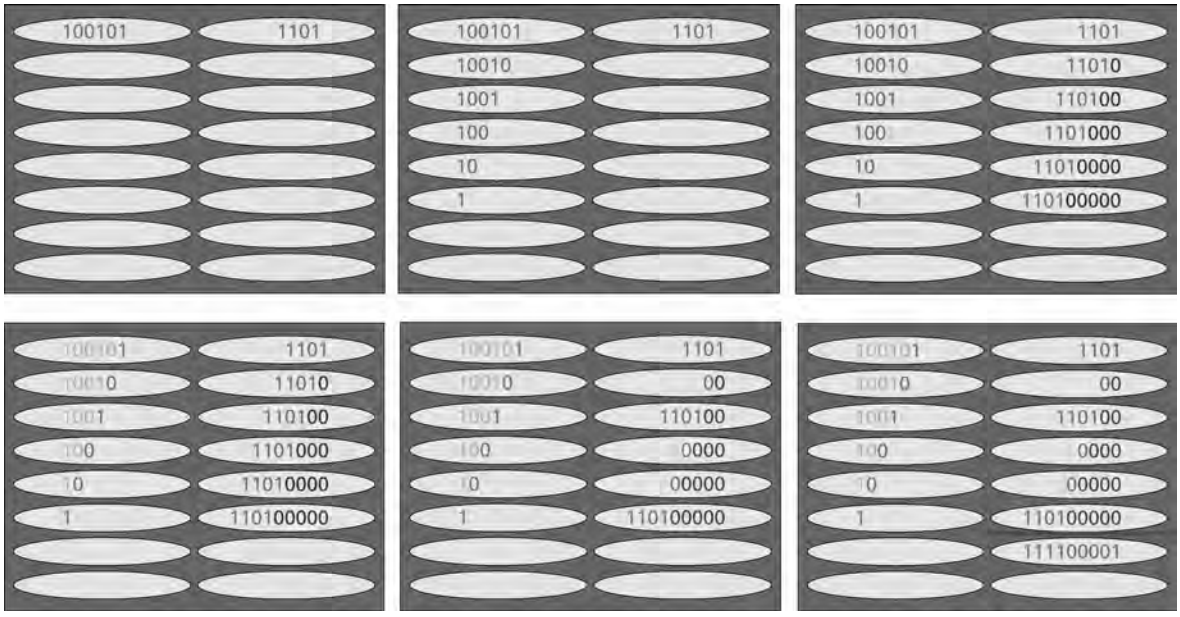



***Abb. 8 Ägyptische Multiplikation mit binärer Repräsentation***

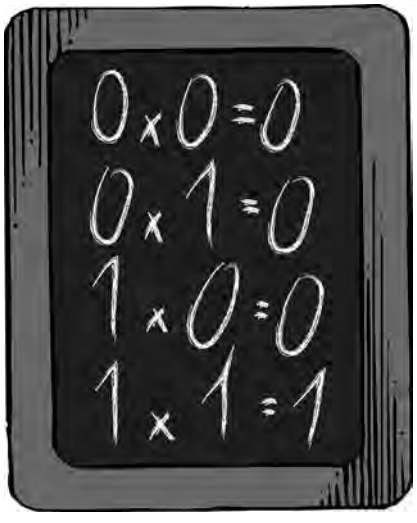



***Abb. 9 Kleines*** $1 \times 1$ ***des Binärsystems***

tionstabelle des „ganz kleinen" Einmaleins. Wie viel einfacher wäre die Grundschule für uns alle gewesen, wenn wir nur dieses hätten lernen müssen …

Basis unserer gesamten heutigen Computertechnik ist also nicht Komplexität, sondern im Gegenteil: Einfachheit. Zusammen mit der strukturierten Zerlegung lassen sich nach diesem Prinzip die allermeisten Aufgabenstellungen gut lösen.

In der hier gezeigten Skizze findet freilich noch keine Konstruktion statt – vielmehr wird diese nur nachvollzogen. Mit der abgeleiteten Kompetenz zur strukturierten Zerlegung lassen sich in der Folge jedoch sehr viele Prinzipien neu entdecken und neu ableiten, etwa die binäre Suche oder Sortierverfahren, aber auch außerhalb der Informatik, etwa modulare Bauweisen.

## Fazit

Informatik ist allgemeinbildend! Diese These ist in etlichen Aufsätzen der Vergangenheit begründet und nachgewiesen worden (zum Beispiel in [17]). Der hier geschilderte Paradigmenwechsel der Lebenswelt und die damit verbundene Notwendigkeit, Konstruktion als wichtiges Element der Erkenntnisgewinnung anzuerkennen und in der Schule zu vermitteln, impliziert jedoch eine Verschärfung der These:

Allgemeinbildende Schule ist heute ohne ein für alle Schülerinnen und Schüler verbindliches Pflichtfach mit angemessenem Stundenumfang, das lebensweltlich relevante Konstruktion in den Mittelpunkt stellt, nicht möglich! Das einzige etablierte Schulfach, das dies zu leisten imstande wäre, ist Informatik.

## Literatur

1. Atiyah Sir MF (2017) The Discrete and the Continuous from James Clerk Maxwell to Alan Turing. Online available http://www.heidelberg-laureate-forum.org/blog/video/lecture-thursday-september-29-2017-sir-michael-francis-atiyah/, last access: 29.3.2019
2. Bell T, Witten IH, Fellows M (2015) CS Unplugged, 2015 Revision. Online available https://classic.csunplugged.org/wp-content/uploads/2015/03/CSUnplugged_OS_2015_v3.1.pdf, last access: 29.3.2019
3. Bundeszentrale für politische Bildung „Aus Politik und Zeitgeschichte: Big Data" 65. Jahrgang 11–12/2015, 9. März 2015
4. Feld SL (1991) Why do your friends have more friends than you do? Am J Sociol 96(6)
5. Gallenbacher J (2016) IT-Grundlagen für Kinder. Mini-Serie in c't Magazin, 7/2016 S 144, 10/2016 S. 158, 12/2016 S 158
6. Gallenbacher J (2017a) Allgemeinbildung in der digitalen, gestalteten Lebenswelt. In: Diethelm I (Hrsg): Informatische Bildung zum Verstehen und Gestalten der digitalen Welt. Lecture Notes in Informatics (LNI). Gesellschaft für Informatik, S 19–28
7. Gallenbacher J (2017b) Zentrale Ideen und Werkzeuge MINTplus. In: TU Darmstadt (Hrsg): MINTplus – Systematischer und vernetzter Kompetenzaufbau in der Lehrerbildung, S 22–23, online unter http://www.zfl.tu-darm-stadt.de/media/zfl/alle_medien_in_struktur/projekte_medien/projekte_mintplus/projek-te_mintplus_anmeldung_zur_tagung_mintplus_am_5__oktober_2017/MintPlus_Broschure.pdf
8. Gallenbacher J (2017c) Abenteuer Informatik – IT zum Anfassen für alle von 9 bis 99. 4. Auflage. Springer, Berlin Heidelberg
9. Heymann H-W (2013) Allgemeinbildung und Mathematik. Studien zur Schulpädagogik und Didaktik, Bd. 13. Beltz, Weinheim Basel
10. Heun D, Gallenbacher J (2014) Technik=Technologie? Eine Untersuchung der Vorstellungen von Lernenden über Technik. Bericht auf http://www.abenteuer-informatik.de/PDF/technikbefragung2014.pdf
11. Kultusministerkonferenz (2015) Ländergemeinsame inhaltliche Anforderungen für die Fachwissenschaften und Fachdidaktiken in der Lehrerbildung. Beschluss der deutschen Kultusministerkonferenz vom 16.10.2008 i. d. F. vom 10.9.2015
12. Minnameier G (2015) Tightening the Peirce-Strings Forms of Abduction in the Context of an Inferential Taxonomy. In: Magnani L, Bertolotti T (eds) Springer Handbook of Modelbased Science. Springer, Berlin
13. Möller K (2001) Genetisches Lehren und Lernen – Facetten eines Begriffs. In: Cech D, Feige B, Kahlert J, Löffler G, Schreier H, Schwier H-J, Stoltenberg U (Hrsg) Die Aktualität der Pädagogik Martin Wagenscheins für den Sachunterricht. Klinkhardt, Bad Heilbrunn, S 15–30
14. Peirce CS (1994) The Collected Papers of Charles Sanders Peirce. Vols. I–VI ed. by Hartshorne C, Weiss P (Harvard University Press, Cambridge, MA, 1931–1935), Vols. VII–VIII ed. by Burks AW (same publisher, 1958) in der Electronic Edition 1994
15. Ugander J, Karrer B, Backstrom L, Marlow C (2011) The Anatomy of the Facebook Social Graph. eprint arXiv:1111.4503, 11/2011
16. „Sendung mit der Maus"-Beitrag (2018) „Lachgeschichte: Der Muffin Code" https://kinder.wdr.de/tv/die-sendung-mit-der-maus/av/video-lachgeschichte-der-muffin-code-100.html
17. Witten H (2003) Allgemeinbildender Informatikunterricht? Ein neuer Blick auf H. W. Heymanns Aufgaben allgemeinbildender Schulen in Informatische Fachkonzepte im Unterricht, INFOS 2003, 10. GI-Fachtagung Informatik und Schule, 17.–19. September 2003 in Garching bei München, S 59–75

# *Informatik am Gymnasium*

***Ein Klärungsversuch mit Vorschlägen***

**Michael Barot · Daniel Baumgartner**

## Die Welt im Wandel

Die menschlichen Gesellschaften wurden in ihrer Entwicklung von den frühen Anfängen bis zur modernen Zivilisation immer wieder durch Neuerungen und Umwälzungen herausgefordert. In der heutigen Zeit erleben wir jenen grossen Wandel, welcher durch den vermehrten Einsatz technischer Geräte ausgelöst wurde. Immer komplexere Aufgaben werden in zunehmend kürzerer Zeit mit stets kleineren und billigeren elektronischen Apparaten bewältigt. Viele Daten liegen digitalisiert vor und sind so der elektronischen Verarbeitung zugänglich. In steigendem Ausmass werden ehemals menschliche Tätigkeiten neu von Computern und Robotern automatisiert erledigt. Diese Entwicklung führt zu einschneidenden Veränderungen in unserer Kultur und Gesellschaft, was sich unter anderem in der Arbeitswelt und der sozialen Umgebung jedes Einzelnen manifestiert.

Die politischen Entscheidungsträger versuchen durch Eingriffe im Bildungswesen der gesellschaftlichen Entwicklung gerecht zu werden. Hierbei soll für die Lehrpersonen ein Umfeld geschaffen werden, welches ihnen ermöglicht, die Jugendlichen von morgen adäquat auszurüsten, damit sie sich einen Zugang zu dieser neuen und im Wandel befindlichen Gesellschaft erschliessen können. Bei Reden, Diskussionen und Berichterstattungen fällt auf, dass in diesem Zusammenhang zwar des Öfteren Begriffe wie „Informatik“ oder „Digitalisierung“ fallen, diese aber ohne klare Vorstellung ihrer eigentlichen Bedeutung verwendet werden. Mit dem vorliegenden Artikel wird der Versuch einer Begriffsklärung unternommen, um daraus vernünftige Folgerungen für das Gymnasium abzuleiten. Hierzu wird es sich als nützlich erweisen, zuerst die historische Entwicklung kurz nachzuzeichnen und die Informatik als Wissenschaft in ihrer Aktualität zu skizzieren. Auf diese Weise wird eine gute Orientierung möglich.

## Die Automatisierung im historischen Kontext

Die Entwicklung der Feinmechanik ermöglichte im 17. Jahrhundert den Bau einer Rechenmaschine, die Additionen mechanisch ausführen konnte. Mitte des 18. Jahrhunderts wurden zur maschinellen Steuerung von Webstühlen Lochkarten eingesetzt. Diese zwei Errungenschaften, welche einerseits die automatisierte Berechnung und andererseits die mechanische Speicherung und Steuerung ermöglichten, gelten als wichtige Wegbereiter für die Entwicklung des Computers. Der Einsatz elektronischer Bauteile verkürzte die Rechenzeit erheblich, und ermöglichte so eine enorme Effizienzsteigerung.

Die Bauteile, und mit ihnen die Computer selbst, wurden zunehmend billiger, sodass in den 80er-Jahren des 20. Jahrhunderts die Computer für den Privatgebrauch erschwinglich wurden. Universitäten und Firmen begannen pionierhaft ihre Rechenmaschinen zu lokalen Netzen zu verknüpfen. Später entwickelte man Standards, welche das heutige Internet mit den Webseiten ermöglichten. Von grosser Bedeutung ist die mit der gesamten Entwicklung

https://doi.org/10.1007/s00287-019-01168-z
© Springer-Verlag Berlin Heidelberg 2019

Michael Barot · Daniel Baumgartner
Informatik, Mathematik, Kantonsschule Schaffhausen
Pestalozzistrasse 20, 8200 Schaffhausen, Schweiz
E-Mail: {michael.barot, daniel.baumgartner}@kanti.sh.ch

**Zusammenfassung**
Im Artikel wird die Informatik als wissenschaftliche Disziplin an Gymnasien mit dem Fokus auf deren allgemeinbildenden Wert thematisiert. Insbesondere wird auf die wichtige Bedeutung der Bildung in Informatik im Hinblick auf eine umfassende Sicht auf unsere Gesellschaft und Kultur eingegangen. Mit stichhaltigen Argumenten wird begründet, weshalb zur Erreichung dieses Bildungsziels ein gutes Verständnis in Algorithmik von zentraler Bedeutung ist.

einhergehende Digitalisierung. Darunter versteht man den Prozess, wenn Informationen von ihrer analogen Form in eine digitale Form übersetzt werden. Zum Beispiel findet eine Digitalisierung statt, wenn die in analoger Form codierte Musik einer Schallplatte in die digitale Form einer CD oder MP4-Datei übersetzt wird. Und genau dies ist die Bedeutung des Begriffs der *Digitalisierung*: die Speicherung von Daten in digitaler Form. Digitalisiert wird also eine alte Fotografie oder die Aufzeichnung einer Sonate. Und nicht etwa die Gesellschaft, wie man häufig da und dort liest oder hört. Ebenso sind Bildung und Unterricht oder dergleichen nicht digitalisierbar.

Mit der Digitalisierung werden Daten einer Weiterverarbeitung zugänglich gemacht, die zugleich komplex und schnell ablaufen kann, weil sie von Computern ausgeführt wird. Wir können in Textdateien nach Begriffen suchen; die digitalisierten Fahrplandaten erlauben, die schnellste uns nützliche Zugverbindung zu finden; Daten werden verschlüsselt, um sie vor unerwünschtem Zugriff zu schützen, oder sie werden gegen Datenverlust bei der Übertragung oder Speicherung geschützt.

Die ersten wissenschaftlichen Untersuchungen zu den Möglichkeiten und Grenzen digitaler Berechnung und Verarbeitung gehörten damals noch dem Gebiet der Mathematik an. Bahnbrechend war die Arbeit des Mathematikers Alan Turing. Dort beschreibt er – noch vor dem Bau eines ersten funktionstüchtigen Computers – das bis heute gültige Modell eines Computers, das zu seinen Ehren *Turing-Maschine* genannt wird. Innerhalb weniger Jahrzehnte erwuchs daraus eine separate, und mit der Mathematik eng verwandte Wissenschaft, die man heute die *Informatik* nennt. Gemäss Duden studiert sie die „systematische Darstellung, Speicherung, Verarbeitung und Übertragung von Informationen, besonders der automatischen Verarbeitung mithilfe von Digitalrechnern".

## Die Teildisziplinen der Informatik

Die Informatikwissenschaft wird von vier tragenden Säulen gestützt. Es sind dies die *praktische*, die *technische*, die *theoretische* und die *angewandte* Informatik.

Bei einer algorithmischen Problemstellung geht es im Grunde genommen darum, aus einer gegebenen Datenmenge eine genau bestimmte darin vorhandene Information zu extrahieren. Damit ein Computer eine Aufgabe korrekt ausführt, muss eine genaue Anleitung formuliert werden, wie dies mit einer Abfolge von einfachen und eindeutigen Anweisungen erreicht werden kann. Eine derartige Anleitung nennt man einen *Algorithmus*. Eine algorithmische Problemlösung ist stets mit einem gewissen Rechenaufwand verbunden, welchen man in der Informatik als sogenannte Berechnungskomplexität bezeichnet. Wichtige Fragen der praktischen Informatik sind: Wie codiert man ein gegebenes Problem geeignet und wie löst man das Problem durch einen Algorithmus? Wie kann man die Rechenzeit so gering wie möglich halten? Im Zentrum steht die effiziente Software.

Der Bau moderner Computer beruht nach wie vor auf Ideen von John von Neumann aus den frühen Anfängen der Informatik. Im Jahre 1945 beschrieb er ihre Architektur kurz umrissen wie folgt: Der Speicher enthält sowohl Daten als auch Algorithmen, welche durch ein Rechenwerk – ein sogenannter Prozessor – ausgeführt werden. Dieses fundamentale Prinzip hat weiterhin Gültigkeit, ungeachtet der wachsenden Komplexität heutiger Computer! Bei der technischen Informatik spielen Fragestellungen rund um den Bau von Mikroprozessoren, um Rechnerarchitekturen sowie um Schalt- und Netzwerke eine wesentliche Rolle. Zentrale Fragen sind: Wie baut man robuste und schnelle Hardware?

In der theoretischen Informatik stehen andere Fragen im Vordergrund, zum Beispiel: Wie kann man den Begriff des Algorithmus mathematisch präzise erfassen? Gibt es Grenzen für die Berechenbarkeit? Diese beiden Fragen wurden in der ersten Hälfte des zwanzigsten Jahrhunderts erfolgreich und abschliessend geklärt. Wesentlich ist auch die Frage nach der Effizienz bzw. Berechnungskomple-

xität: In welcher Zeit und mit welchem Speicherplatz lässt sich ein gegebenes Problem lösen? Diese sehr schwierige und noch nicht vollständig beantwortete Fragestellung ist Gegenstand der aktuellen Forschung.

Die angewandte Informatik nutzt die Erkenntnisse der praktischen und technischen Informatik für Forschungen in nahezu allen anderen Wissenschaftsgebieten. Im Sinne einer kleinen Auswahl seien beispielhaft die Biochemie, Meteorologie, Astronomie, Medizin und Linguistik erwähnt. Zum Beispiel können neuerdings grosse organische Moleküle dreidimensional modelliert, und so deren Eigenschaften am Computer mittels Simulationen analysiert werden. Für ein tieferes Verständnis des Materieaufbaus unseres Kosmos sind die Experimente der Elementarteilchenphysik am CERN von Bedeutung, welche ohne Erkenntnisse der Informatikwissenschaft undenkbar wären. Ebenfalls von der angewandten Informatik durchdrungen sind die Ingenieurwissenschaften. Unvorstellbar ohne Informatik sind zum Beispiel die Raumfahrt, die modernen Navigationssysteme oder in der Medizin die Computertomografie. Die Anwendungsmöglichkeiten, insbesondere diejenigen unter Einbezug von Supercomputern, sind noch längst nicht ausgeschöpft.

## Die häufigsten Missverständnisse

Der aufmerksame Leser wird bemerkt haben, dass in keiner der vier Teildisziplinen der Informatik das Erlernen der Handhabung von vorgefertigten Softwareprodukten im Vordergrund steht. Es geht um viel Grundsätzlicheres! Der Unterschied zwischen der Informatik und der Benutzung eines Textverarbeitungsprogramms oder eines Mobiltelefons ist vergleichbar mit dem Unterschied zwischen der Chemie und der korrekten Verwendung von Benzin oder Backpulver: Beides sind chemische Fertigprodukte, die in unserem Alltag verwendet werden. Jedoch käme kaum jemand auf die Idee, die Chemie mit dem Erlernen der korrekten Verwendung von Benzin oder Backpulver zu verwechseln. Bei der Informatik ist ein derartiges Missverständnis erstaunlicherweise leider weit verbreitet.

Ebenfalls ist häufig zu hören, der Unterricht der Informatik solle sich um die Auswirkungen der fortschreitenden Automatisierung auf unsere Gesellschaft kümmern. Vergleichen wir diesen Anspruch wieder mit der Chemie: Die Auswirkung der Düngung in der Landwirtschaft auf die Umwelt oder die Akkumulation von Plastikabfällen in den Weltmeeren, also die Auswirkung des Einsatzes von chemischen Stoffen auf unsere Umwelt und Gesellschaft, ist in selbstverständlicher Weise nicht zentraler Inhalt des Chemieunterrichts. So wichtig die Diskussionen um diese Problematiken sind, gehören sie dem Gebiet der Umweltwissenschaft an. Genauso verhält es sich in der Informatikwissenschaft. Die Diskussionen rund um die Auswirkungen des zunehmenden Einsatzes technischer Geräte auf die Gesellschaft sollten wohl eher in der Soziologie oder einer noch zu schaffenden interdisziplinären Wissenschaft angesiedelt werden.

## Informatik auf gymnasialer Stufe

Die Bildungsaufgabe des Gymnasiums hat sich trotz des angesprochenen rasanten gesellschaftlichen Wandels nicht geändert. Und das ist auch richtig so. Nach wie vor ist stets das zentrale Bildungsziel vor Augen zu halten, nämlich die Vermittlung von breit abgestütztem und vertieftem Wissen, unter gleichzeitiger Entfaltung eines weiten kulturellen Horizonts. Zu den vordringlichsten Aufgaben gehört damit die Erarbeitung des Verständnisses für die *zeitlosen* universellen kulturellen Leistungen des Menschen. Den momentanen kurzlebigen Bedürfnissen, Modeströmungen und dergleichen sollte so wenig Platz wie möglich eingeräumt werden, damit Raum fürs Wesentliche bleibt.

In Anbetracht der oben skizzierten Teildisziplinen der Informatik ergibt sich daraus, dass ein Informatikunterricht am Gymnasium hauptsächlich die folgenden Fragen klären muss: Warum sind Algorithmen wichtig und nützlich? Welches sind die elementaren Bauteile eines Algorithmus? Gibt es Grenzen für die Algorithmisierbarkeit? Welches Architekturprinzip liegt einem Computer zugrunde? Was genau versteht man unter einer Automatisierung eines Prozesses?

## Konkrete Unterrichtsinhalte

Um in aller Klarheit zu erkennen, was unter einer Automatisierung zu verstehen ist, erweist es sich als äußerst nützlich, ja sogar unumgänglich, dass Gymnasiasten lernen Algorithmen für konkrete Problemstellungen zu entwerfen und diese dann zu programmieren. Programmieren im engeren Sinne meint die Formulierung eines Algorithmus in einer für die Maschine verständlichen Sprache.

Dazu eignen sich Programmierumgebungen, die übersichtlich gestaltet und nicht durch unnötige Funktionalitäten überladen sind. Zu bevorzugen sind solche, welche eine unmittelbare grafische Rückmeldung ermöglichen. Ziel des Programmierunterrichts muss sein, den Schülern eine klare Vorstellung der zwei wesentlichen Ablaufstrukturen, nämlich der Wiederholung und der Verzweigung, zu vermitteln. Denn dies sind die beiden grundlegenden Bestandteile, auf denen alle automatisierten Abläufe aufbauen. Auch sollten sie mit der Modularität, also von der Möglichkeit der Erledigung von wiederkehrenden Teilaufgaben durch Unterprogramme, vertraut werden. Ebenso von Bedeutung ist der Umgang mit Variablen, welche die Steuerungen von Prozessen erheblich erleichtern und verallgemeinern. Sobald die Gymnasiasten erkennen, dass für die Realisierung von mechanisierbaren Aufgaben *wenige fundamentale Prinzipien* genügen, ist ein erstes wichtiges Bildungsziel erreicht.

Darauf aufbauend können dann konkrete, mehrschichtige Aufgaben angegangen werden, wie zum Beispiel das Sortieren einer Liste oder das Absuchen einer Baumstruktur. Bei Problemstellungen dieser Art bietet sich die wunderbare Gelegenheit, verschiedene Lösungsansätze im Hinblick auf ihre Effizienz beziehungsweise Berechnungskomplexität hin zu vergleichen. Diese Effizienz kann dabei zunächst durch die Messung der Rechenzeit abgeschätzt werden. Einerseits bezüglich identischer Eingaben bei verschiedenen Algorithmen und andererseits mit zunehmend grösserer Datenmenge bei ein und demselben Algorithmus. Ebenso bietet sich hier an, die Nützlichkeit von Flussdiagrammen zu besprechen und diese auch konsequent einzusetzen.

Die Themen rund um Zufallszahlen und Simulationen bieten eine Fülle an Möglichkeiten und Variationen. Wie ist der Begriff des Algorithmus mit dem Begriff des Zufalls in Einklang zu bringen? Dies scheint auf den ersten Blick ein unmögliches Unterfangen zu sein, weil ein Algorithmus ja aus einer Abfolge von genau vorgegeben und unmissverständlichen Anweisungen besteht. Hier kommt der faszinierende Aspekt des sogenannten Pseudozufallsgenerators ins Spiel. Interessanterweise ist es möglich, mit einem Algorithmus gewisse Folgen von Zahlen zu berechnen, die sich recht gut wie zufällig „gewürfelte" Zahlenfolgen verhalten. Für einen Aussenstehenden ist es nahezu unmöglich ohne Kenntnis des Programmcodes die weitere Entwicklung der Zahlenfolge vorauszusagen. Es handelt sich zwar nicht um einen „echten" Zufall, aber Zufallsprozesse lassen sich damit erstaunlich gut simulieren. Auch bei dieser Thematik kann wieder jeweils von einfachen bis hin zu anspruchsvolleren Problemstellungen vorgegangen werden. Anwendungen gibt es derer viele. Genannt seien beispielhaft die Berechnungen von Wahrscheinlichkeiten im Zusammenhang mit Pokerspielen sowie der Randomwalk für die Simulation einer Brownschen Bewegung von Molekülen. Obendrein regt die Thematik zu philosophischen Diskussionen mit den Gymnasiasten über Zufall und Determinismus an.

Die Informatik bietet ein weites Feld von interessanten Themen für den Unterricht. Die Auswahl ist natürlich der Stundendotation anzupassen. Hier soll aber noch ein weiterer wichtiger Aspekt erwähnt werden, nämlich derjenige der sogenannten Rekursion. Es ist sicherlich lohnenswert diese Thematik zur Sprache zu bringen, weil hier die Selbstbezüglichkeit ins Spiel kommt. Bei einer Rekursion ruft sich ein Algorithmus selber auf. Diese Strategie ermöglicht in gewissen Situationen eine raffinierte Eleganz und Effizienz bei der Programmierung.

Mit geistig reiferen Gymnasiasten sollten unbedingt theoretische Fragestellungen untersucht und besprochen werden. Der fundamentale Satz von Turing ist für sie gut zugänglich. Er besagt, dass nicht jede definierbare Folge von Zahlen algorithmisch berechnet werden kann. Damit ist man bereits an den Grenzen der Algorithmisierbarkeit angelangt. Die Heranführung an diese erkenntnistheoretisch wichtige Aussage erfolgt wohl am besten mit dem Konzept der Turingmaschine. Für das Verständnis ihrer Funktionsweise eignen sich Applikationen, welche deren Simulation erlauben. Ist dieses Modell gut verstanden, so sind im Wesentlichen die Grundlagen zum Verständnis des Satzes von Turing erarbeitet. Im Übrigen kann diese Thematik durchaus zu philosophischen Diskussionen rund um die der Logik inhärenten Grenzen anregen.

## Schlussbemerkung

Im vorangegangenen Abschnitt haben wir nur eine kleine Auswahl aus dem riesigen Fundus an interessanten und relevanten Problem- und Fragestellungen der Informatikwissenschaft erwähnt, welche für den gymnasialen Unterricht geeignet sind. Sie führen alle zu einem besseren, und vor allem tieferen Verständnis unserer im Wandel der

Zeit befindlichen Gesellschaft und Kultur. Man muss sich stets vor Augen halten, die Bildung in Informatik insbesondere auch für diejenigen Gymnasiasten interessant und horizonterweiternd zu gestalten, welche später mit der Informatikwissenschaft nicht oder nur am Rande in Berührung kommen werden. Denn im Sinne einer fundierten Allgemeinbildung gilt es, dass alle eine umfassende Weltsicht erlangen. Und dies ist ja gerade die zentrale Grundaufgabe des Gymnasiums.

# *Programmierunterricht von Kindergarten bis zur Matura in einem Spiralcurriculum*

**Jacqueline Staub · Michelle Barnett Nicole Trachsler**

## Einführung

In der Schule lernen Kinder nebst lesen und schreiben nun auch zu programmieren. Dabei werden Fähigkeiten wie Kreativität und konstruktives Problemlösen gefördert. Diese sind wichtige Voraussetzungen, um sich in den noch ungewissen Berufen der Zukunft behaupten zu können. Wer schon in der Schule übt, allgemeine Probleme zu lösen und Lösungswege exakt zu beschreiben (so exakt, dass selbst intellektfreie Maschinen sie für uns ausführen können), ist gut vorbereitet, die noch unbekannten Herausforderungen der Zukunft bewältigen zu können. Guter Programmierunterricht erlaubt es, die Lernenden gezielt darauf vorzubereiten.

### Was bedeutet programmieren?

Programmieren bedeutet, im engeren Sinne, mit einer Maschine zu kommunizieren. Dazu müssen wir eine Sprache verwenden, die die Maschine versteht: eine **Programmiersprache**. Ähnlich wie natürliche Sprachen (Deutsch, Französisch oder Englisch) besteht auch eine Programmiersprache aus Wörtern. Diese nennt man in der Informatik **Befehle**. Durch die grammatikalisch korrekte Aneinanderreihung von Befehlen bilden sich größere Texte, die wir als **Programme** bezeichnen. Programmieren verlangt einen hohen Grad an Präzision: Um den Computer anzuleiten, eine Tätigkeit auszuführen, müssen wir Schritt für Schritt detailliert beschreiben, was zu tun ist. Der Computer besitzt keinerlei Improvisationsfähigkeit, weshalb die Beschreibung vollständig und eindeutig interpretierbar sein muss. So lernen Kinder, sich formal und logisch korrekt auszudrücken.

### Wieso programmieren wir?

Programmieren ist eine inhärent konstruktive Tätigkeit: Lösungen werden konstruiert, modular kombiniert und erweitert. Im Programmierunterricht wird die konstruktive Denkweise der technischen Disziplinen gefördert. Die folgenden drei Punkte erachten wir als zentrale Ziele des Programmierunterrichts [4]:

1. Mittels Probieren und Experimentieren Lösungswege für gegebene Aufgabestellungen entdecken
2. Gefundene Lösungswege mittels Programmieren umsetzen und funktionsfähige Programme daraus entwickeln
3. Die korrekte Funktionalität durch Ausführen des Programms am Computer testen und nach Bedarf das Programm korrigieren oder es für zusätzliche Aktivitäten erweitern

Die Kinder lernen im Unterricht selbstständig Neues zu entdecken, ihre eigenen sowie gegebene Vorgehensweisen kritisch zu hinterfragen und gegebenenfalls zu verbessern. Sie werden zu kreativen Produzenten, die nach eigenem Empfinden neue und innovative Lösungen entwickeln.

### Unser Spiralcurriculum

Zwischen Kindergarten und Maturitätsreife wächst die Bandbreite der erlangten Kompetenzen stetig,

https://doi.org/10.1007/s00287-019-01161-6
© Springer-Verlag Berlin Heidelberg 2019

Jacqueline Staub · Michelle Barnett · Nicole Trachsler
ETH Zürich
E-Mail: {staubj, barnettm, tnicole}@ethz.ch

Jacqueline Staub
PH Graubünden

## Zusammenfassung

In diesem Artikel stellen wir drei Programmierumgebungen vor, welche an der ETH Zürich entwickelt wurden und in einem einheitlichen Spiralcurriculum für den Programmierunterricht vom Kindergarten bis zur Maturität verwendet werden können. Der Fokus liegt auf der selbstständigen Entwicklung funktionsfähiger Programme. Ausgehend von einer Aufgabenstellung beginnt der Prozess der Erkenntnisgewinnung mittels Ausprobierens, Entwickeln einer Lösungsstrategie und deren Umsetzung in einem vollständigen Programm. Dieses wird schließlich getestet, indem es am Computer ausgeführt und nach Bedarf verbessert oder erweitert wird. Unsere Programmierumgebungen unterstützen hohe Selbstständigkeit der Kinder und Jugendlichen in allen Phasen der Herstellung eines Produktes im Sinne eines funktionierenden Programmes.


vom Lesen und Schreiben bis hin zu abstraktem Denken. Unsere Materialien (das Curriculum ebenso wie die Lernumgebungen) berücksichtigen diese kognitive Entwicklung und knüpfen gezielt an den Wissensstand der Lernenden an.

In den folgenden Kapiteln stellen wir diese kognitive Entwicklung vor. Wir illustrieren zentrale Herausforderungen, die in den jeweiligen Stufen auftreten, und erklären, wie unsere Umgebungen die Lernenden darin unterstützen, diese Herausforderungen zu meistern.

## Programmiere die Biene

Schon ab dem Kindergarten kann das Lösen einfacher Problemstellungen mittels Programmieren gefördert werden. Dazu stehen diverse kleine, robuste und einfach zu bedienende Roboter (wie z. B. der BeeBot) zur Verfügung, die sich auf Befehl hin steuern lassen. Auf Bodenmatten (wie in Abb. 1) lösen Kinder Navigationsaufgaben. Spielerisch lernen sie dabei ihre erste Programmiersprache. Um den Roboter zu steuern, stehen vier einfache Befehle zur Verfügung: Diese navigieren den Roboter ein Feld vor (respektive zurück) oder drehen ihn am Ort um 90 Grad nach rechts (respektive links).

## Blockbasiertes Programmieren am Computer

Anknüpfend an die Erfahrungen mit dem Bienenroboter steht die von uns entwickelte Programmierumgebung XLogoBlocks (https://xlogo.inf.ethz.ch/1) zur Verfügung. Darin steuern die Lernenden am Bildschirm eine Schildkröte, die eine Spur hinterlässt und so farbige Muster auf der Zeichenfläche erzeugt (s. Abb. 2).

Für Anfänger allgemein, aber insbesondere für Anfänger dieser Altersstufe, gilt es, die verwendete Programmiersprache und deren zugrunde liegenden Befehle sorgfältig auszuwählen unter Rücksichtnahme auf die verfolgten didaktischen Ziele. In dieser Stufe wollen wir das planende Denken sowie die räumliche Vorstellungskraft der Lernenden stärken. Aus diesem Grund offeriert die Umgebung bewusst nur wenige Befehle, die vom BeeBot mehrheitlich bereits bekannt sind.

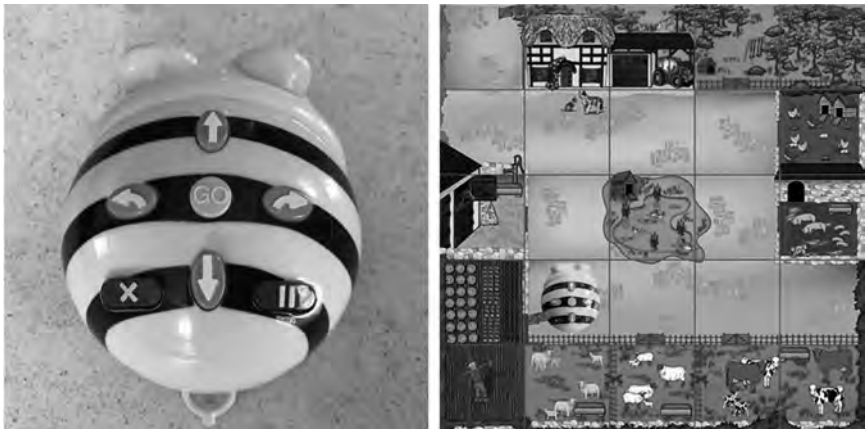



*Abb. 1 BeeBot und Matte [8]*

*Abb. 2 XLogoBlocks*

Sämtliche Befehle werden durch Blöcke dargestellt. Das **blockbasierte Programmieren** eignet sich besonders für junge Kinder, deren sprachliche und feinmotorische Fähigkeiten noch nicht ausgereift genug sind, um auf einer Tastatur zu arbeiten. Dank symbolischer Darstellung der Befehle werden weder Lese- noch Schreibfähigkeiten vorausgesetzt. Tippfehler sind ausgeschlossen und die Kinder konzentrieren sich auf das Erstellen logisch korrekter Programme.

Dazu werden die Befehle wie Puzzleteile mit der Maus zum Startblock gezogen und dort auf Knopfdruck ausgeführt. Bereits ausgeführte Befehle oder Befehlssequenzen werden gesammelt und in einer Übersicht veranschaulicht (s. Abb. 3). Diese zeigt den bisherigen Lösungsweg als Programm und entspricht dem von der Schildkröte gelaufenen Muster. Dank Übersicht kann der Lösungsweg nachvollzogen, diskutiert und nach Bedarf für Korrekturen verwendet werden.

Korrekte Programme zu schreiben, ist nicht einfach. Denn die Kinder vollziehen beim Programmieren einen ungewohnten Perspektivenwechsel: Sie versetzen sich in die Lage der Schildkröte. Dass deren Perspektive nicht immer mit der eigenen übereinstimmt, stellt viele Lernende vor eine kognitive Herausforderung. Wir unterstützen sie durch eine visuelle Hilfestellung: Die Arme der Schildkröte sind unterschiedlich eingefärbt – in derselben Farbe wie die entsprechenden Befehle *rechts* und *links* (s. Abb. 4).

Kontinuierlich lernen die Kinder, sich in einer neuen Sprache auszudrücken: Sie lesen Programme (i. e. prognostizieren deren Effekt) und beschreiben eigene Lösungswege als Sequenzen von Befehlen. Anfangs werden Lösungen Schritt für Schritt konstruiert. Später werden auch längere Befehlssequenzen geschrieben und ausgeführt.

Die Komplexität der Aufgaben kann beliebig angepasst werden. Abbildung 5 zeigt zwei Programme, die je ein Quadrat in unterschiedlicher Größe auf den Bildschirm zeichnen.

Jeder Schritt der Schildkröte hat eine konstante Länge. Um längere Linien zu zeichnen, müssen meh-

*Abb. 3 Bereits ausgeführte Befehle*

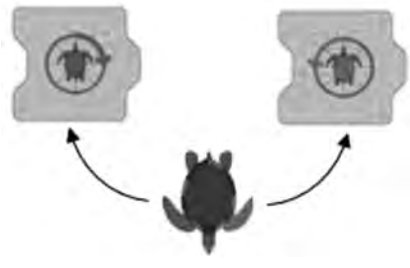


*Abb. 4 Perspektivenwechsel*

*Abb. 5 Zwei Programme, welche Quadrate verschiedener Größen zeichnen*


*Abb. 6 Der neue Befehl Repeat*

rere Schrittbefehle aneinandergereiht werden (wie im unteren Programm der Abb. 5). Mit dem Befehl *Repeat* ermöglichen wir den Kindern, lange Linien mit nur wenigen Befehlen zu zeichnen. Zur Veranschaulichung: Das Programm in Abb. 6 lässt die Schildkröte fünf Schritte vorwärtsgehen, benötigt dazu jedoch nur zwei Befehle.

Kinder im Alter zwischen fünf und acht Jahren machen gerade erste Erfahrungen mit Zahlen. Durch die Einführung des Befehls *Repeat* erhalten die Kinder die Möglichkeit, die Schildkröte mehr als nur einen Schritt aufs Mal vorwärtsgehen zu lassen. Die Kinder zählen die Schritte der Schildkröte und vertiefen so indirekt ihr Verständnis für Zahlen.

## Variable Distanzen und Winkel

Mit zunehmendem Fortschritt erlangen Kinder etwa ab der dritten Klasse ein intuitives Verständnis für den Zahlenraum bis 1000 und die grundlegenden Rechenoperationen darin.

An dieser Stelle verallgemeinern wir die verfügbaren Befehle (Abb. 7) in XLogoBlocks (https://xlogo.inf.ethz.ch/2). Neu können die Befehle durch sogenannte **Parameter** modifiziert werden. Die Kinder geben bei Bewegungsbefehlen an, wie lange eine zu zeichnende Linie sein soll. Bei Rotationsbefehlen wird entsprechend angegeben, um welchen Winkel sich die Schildkröte drehen soll.

Diese Änderung erlaubt es den Kindern eine Vielzahl neuer Muster zu zeichnen. Sämtliche Quadrate (unabhängig von deren Größe), können mittels


*Abb. 7 Neue Befehle in XLogoBlocks*

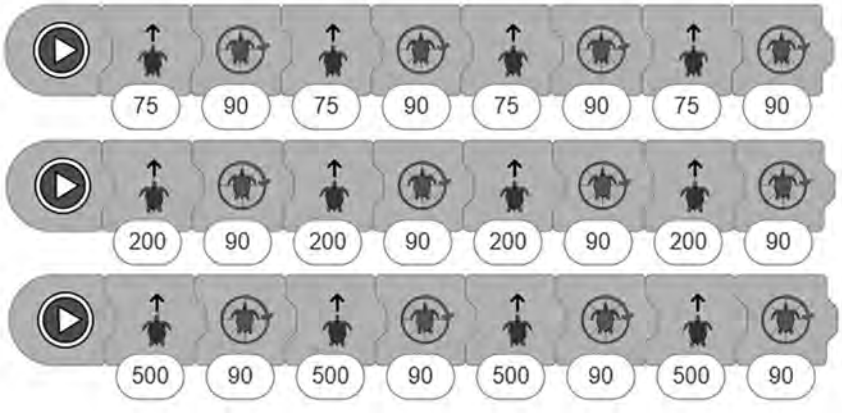



*Abb. 8 Quadrate verschiedener Größen*

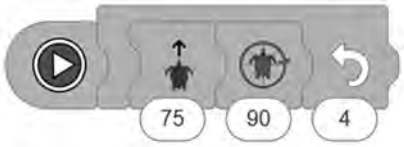



*Abb. 9 Quadrat mit Repeat*

Parameter durch nur acht Befehle dargestellt werden (siehe Abb. 8).

Mit steigender Erfahrung erkennen Kinder bald wiederkehrende Programmsequenzen in ihren Lösungswegen. Zum Zug kommt erneut der Befehl *Repeat*, mit welchem die Kinder lernen, lange und unübersichtliche Programme kürzer zu beschreiben. Anders als in der vorgehenden Stufe, werden hier nicht nur einzelne Befehle, sondern ganze Befehlssequenzen wiederholt. Dazu müssen die Lernenden wiederkehrende Muster in ihren Programmen identifizieren und dieses ausformulieren. Das Zeichnen eines Quadrats besteht z. B. aus vier identischen Seiten: Eine Linie und eine Drehung um 90°. Mit dem Befehle *Repeat* lässt sich das kurz beschreiben (Abb. 9).

Das Konzept der Repetition stellt ein mächtiges Werkzeug dar, um Arbeit durch den Computer zu automatisieren. Es schafft die Ausgangslage um in der nächsten Stufe den Mechanismus der Modularität zu verstehen.

## Textbasiertes Programmieren und Modularität

Geschriebener Text hat sich über Jahrtausende als einer der beliebtesten Informationsträger etabliert. Auch im professionellen Umfeld werden Programme stets in Textform verfasst. Dies verlangt jedoch Präzision, denn Tippfehler werden von unserem intellektfreien Kommunikationspartner nicht verstanden. Circa ab der fünften Klasse haben die Kinder ausreichende Feinmotorik- und Sprachkompetenzen entwickelt, um diese Herausforderung zu bewältigen und dabei zu lernen, sich nicht nur logisch, sondern auch formal präzise auszudrücken.

Wir verwenden die Programmiersprache Logo, welche 1967 vom Mathematiker und Psychologen Seymour Papert und seinem Team entwickelt wurde. Logo umfasst sämtliche der bisher vorgestellten Befehle. In der folgenden Liste zeigen wir auf, wie die bereits bekannten Befehle in Logo geschrieben werden (Abb. 10).

Die Programmierumgebung XLogoOnline (https://xlogo.inf.ethz.ch/3, siehe Abb. 11) ermöglicht den Einstieg ins textbasierte Programmieren.

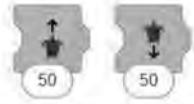
| Blockbasierter Befehl | Textbasierter Befehl (Kurzform) |
|---|---|
|   | forward 50 (fd 50)<br>back 50 (bk 50) |
|   | right 90 (rt 90)<br>left 90 (lt 90) |
|   | repeat 4[fd 50 rt 90] |

*Abb. 10 Befehle in Logo*

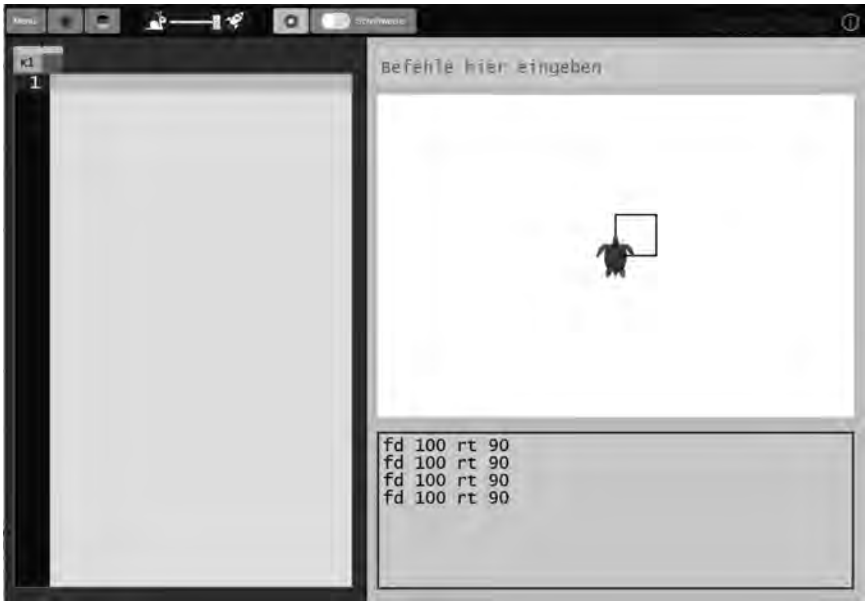



*Abb. 11 XLogoOnline*

Die Umgebung ist auf Programmieranfänger ausgerichtet und unterstützt ihren Lernprozess mittels gezielter Hilfestellung bei Fehlern, was sie von anderen Programmierumgebungen abhebt [7].

Programmieren in geschriebener Sprache erlaubt es den Kindern, beliebig große und komplexe Muster zu erzeugen. Gleichzeitig wird der Prozess jedoch auch anfälliger für Fehler: Jeder Tippfehler, sei er noch so unauffällig und klein, führt dazu, dass der Computer die Ausführung des Programms verweigert. XLogoOnline unterstützt die Programmierenden gezielt darin, mit dieser Herausforderung umzugehen und die Fehler selbstständig zu beheben. Die Umgebung bietet zwei Arten von Hilfestellungen. Diese betreffen zwei Typen von Fehlern: **syntaktische Fehler** und **logische Fehler**.

1. Enthält das Programm syntaktische Fehler, wird es vom Computer nicht verstanden. Vergisst

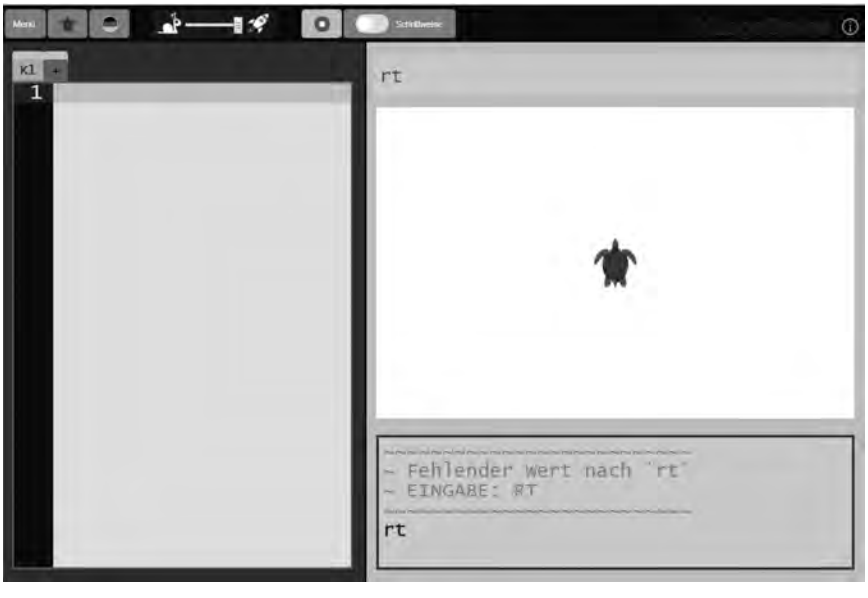



*Abb. 12 Fehlermeldungen*

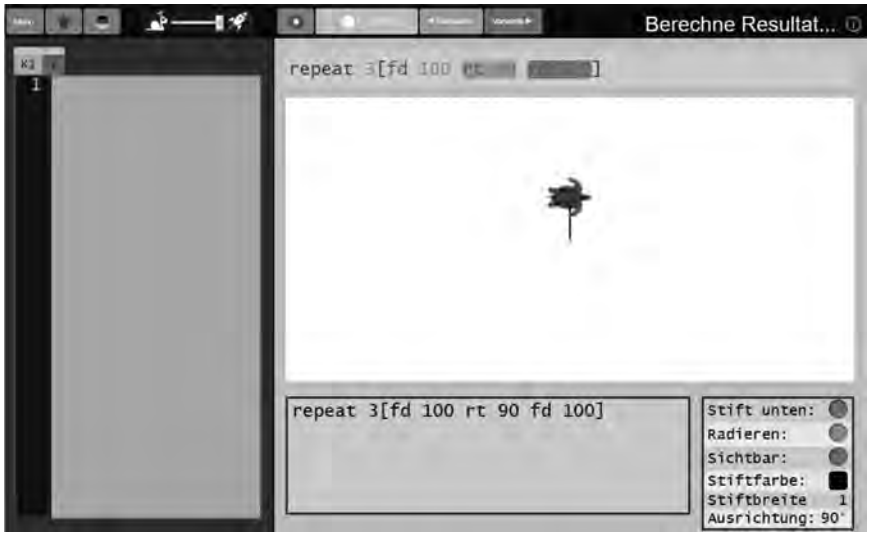



*Abb. 13 Schritt für Schritt wird das Programm nachvollzogen*


*Abb. 14 In Orange der zuletzt ausgeführte und in Blau der nächste auszuführende Befehl*

man beispielsweise (wie in Abb. 12) anzugeben, um welchen Winkel sich die Schildkröte drehen soll, ist der Computer nicht fähig den Befehl auszuführen. XLogoOnline prüft das Geschriebene kontinuierlich und weist mit einer kurzen Erklärung auf den Fehler hin. So werden sich Kinder ihrer Fehler bewusst und können diese korrigieren [2].

2. Logische Fehler (im Gegensatz zu syntaktischen Fehlern) führen nicht dazu, dass ein Programm nicht ausgeführt werden kann; das Programm wird erfolgreich in ein Bild umgewandelt, allerdings entspricht dieses nicht den Erwartungen. Vor diesen Fehlern kann uns der Computer nicht schützen. Stattdessen bietet XLogoOnline einen Mechanismus, um fehlerhafte Programme Schritt für Schritt auszuführen, sodass die Fehlerstelle einfacher gefunden und der Fehler behoben werden kann (siehe Abb. 13 und 14). Dabei steuern die Kinder die Ausführung des Programms manuell und beobachten Befehl um Befehl die unmittelbare Auswirkung. Schwierige Stellen können beliebig oft durchlaufen werden mittels gezieltem Vor- und Zurückspulen.

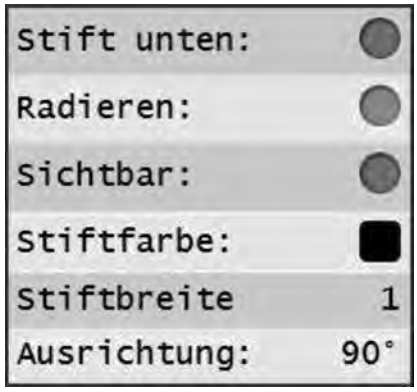



*Abb. 15 Aktive Effekte der Schildkröte*

In einer Tabelle (siehe Abb. 15) werden darüber hinaus diejenigen Effekte sichtbar, die nicht direkt visualisiert sind: Hat man beispielsweise vergessen die Stiftfarbe korrekt zu setzen, wird der Fehler hier ersichtlich.

Mittels textbasierter Programmierung können größere und auch komplexere Programme konstruiert werden, als dies zuvor der Fall war. Wie in natürlichen Sprachen kann sich auch Logo dynamisch entwickeln und erlaubt neue Wörter (also Befehle) zu definieren und diese anschließend zu verwenden. So können bereits geschriebene Programme einfach wiederverwendet werden, um immer größere und komplexere Konstrukte zu erzeugen. Betrachten wir dazu ein Beispiel, in welchem ein Muster aus drei Quadraten gezeichnet werden soll (siehe Abb. 16).


*Abb. 16 Drei Quadrate*

Das wiederkehrende Muster ist ein Quadrat der Seitenlänge 100. Wir erweitern die Sprache mit einem neuen Befehl `quadrat100`.

```
TO quadrat100
repeat 4[ fd 100 rt 90]
END
```

Dieser kann anschließend verwendet werden, um das Muster zu zeichnen:

```
repeat 3[ quadrat100 fd 100]
```

Wir führen neue Befehle ein, um jegliche Konstrukte übersichtlich und kurz mit einer Handvoll Befehlen zu beschreiben. Verschachtelungen dieser Art nennt man **modulares Programmieren**.

In einem nächsten Schritt erfahren Kinder, wie sie Programme schreiben, die zu mehr als nur einem Zweck eingesetzt werden können. Dazu verwenden sie das Konzept der **Parametrisierung**, welches bereits in Kapitel „Variable Distanzen und Winkel“ ein erstes Mal unbewusst eingesetzt wurde. Analog zum Befehl fd, dessen Schrittlänge beliebig parametrisiert werden kann, kann auch das Programm quadrat soweit parametrisiert werden, dass dessen Seitenlänge frei bestimmt werden kann.

```
TO quadrat : laenge
repeat 4[ fd : laenge rt 90]
END
```

Mit dem modularen Entwurf lernen die Kinder eine wichtige Methode, um Komplexität zu meistern. Sie lernen, wie sie Probleme in kleinere Teilprobleme zerlegen, welche sie bereits gelöst und deren Funktionalität sie getestet haben. Erst wenn die zu verwendenden Module korrekt funktionieren, fahren sie fort, um daraus ein komplexeres Muster zu erzeugen. Mittels Parametrisierung lernen sie einen Mechanismus kennen, der es ihnen erlaubt, eine Vielzahl an verschiedenen Mustern in nur einem einzigen neuen Programm zu vereinen. An dieser Stelle werden sich die Kinder bewusst, wie die verwendete Sprache der vorherigen beiden Zyklen intern funktioniert und sind fähig die verwendete Programmiersprache selbst sinnvoll zu erweitern.

## Ein Schritt in die Professionalisierung

Mit dem Übertritt in die Sekundarstufe I sind Schüler kognitiv in der Lage, auch abstraktere Konzepte wie Variablen, Bedingungen und Listen erfolgreich zu meistern. Mit Python führen wir eine Programmiersprache ein, die selbst im professionellen Umfeld weit verbreitet ist und unseren didaktischen Ansprüchen genügt.

Wir entwickelten die beiden Programmierumgebungen TigerJython [1] und WebTiger-Jython (https://webtigerjython.ethz.ch, s. Abb. 17), die für Tablets, Laptops und Computer verfügbar sind. Als Lernumgebung offerieren wir den Schülern auch hier bewusst nur die Funktionalitäten, welche diese für den Unterricht benötigen. Das Kernstück der Umgebungen ist ein Mechanismus, der Schülerprogramme analysiert und im Falle von Fehlern eine automatische Meldung produziert. Diese sind in leicht verständlicher Sprache formuliert, weisen gezielt auf den Fehler hin und erlauben es den Lernenden, selbstständig ihre Programme zu korrigieren.

Eine Aufgabenstellung dieser Zielstufe lautet, Quadrate zunehmender Größe in einem Programm zu vereinen.

Die Schülerinnen wissen aus früheren Erfahrungen, wie sie ein parametrisiertes Programm *quadrat(groesse)* schreiben, um Quadrate beliebiger Größen zu zeichnen. Bei dieser Aufgabe stoßen die Jugendlichen jedoch an eine Grenze: Um das gegebene Bild zu zeichnen, müsste der Befehl 50-mal in Folge mit verschiedenen Parametern geschrieben werden.

```
quadrat (10)
quadrat (20)
quadrat (30)
...
quadrat (500)
```

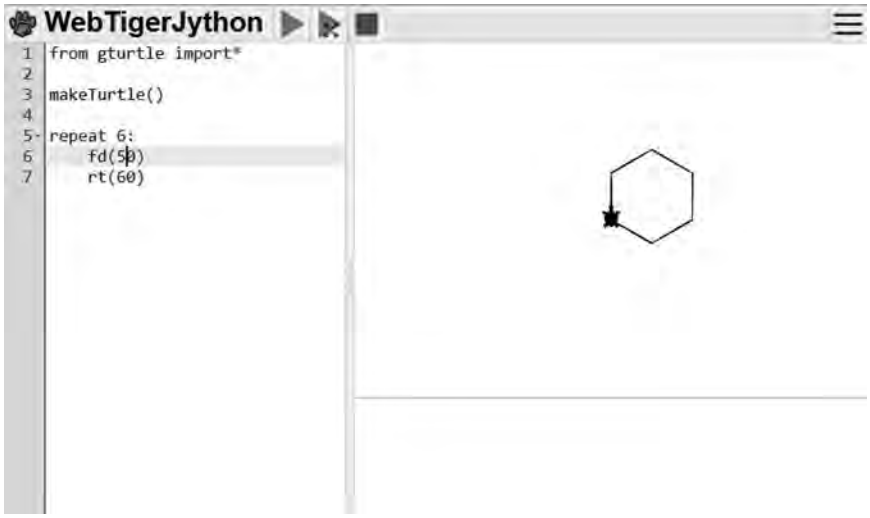



*Abb. 17 WebTigerJython*

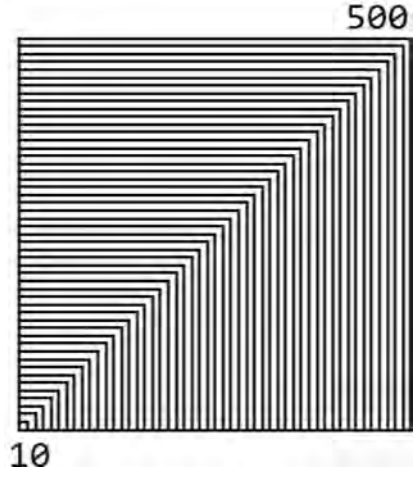



*Abb. 18 Beispielaufgabe mit Variablen*

An dieser Stelle offeriert sich das Konzept der Variable als hilfreiches Werkzeug für die Programmierenden: Anstatt dieselbe Zeile immer und immer wieder zu schreiben, führen sie eine Variable ein und lassen den Befehl *quadrat(groesse)* wiederholt ausführen. Die Variable „groesse“ wird in jedem Durchgang entsprechend angepasst:

```
groesse = 10
repeat 50:
    quadrat ( groesse )
    groesse = groesse + 10
```

Das Konzept einer Variable zu verstehen, ist ohne eine solide mathematische Verankerung kaum möglich. Daher empfehlen wir, dieses schwierige Konzept erst an dieser Stelle einzuführen. Dieser Schritt stellt eine wichtige Grundlage dar, um weiterführende Programmierkonzepte wie Bedingungen oder Listen angehen zu können.

## Implikationen

Unser Ziel ist es, mit dem Programmieren in der Schule das kritische Denken der Schülerinnen und Schüler durch entdeckendes und selbstständiges Lernen zu fördern. Dabei wollen wir nicht eine zukünftige Generation von Informatikern erziehen, vielmehr sollen die Kinder für die digitale Zukunft vorbereitet werden, indem sie deren Entwicklung nachvollziehen. Dies ermächtigt sie dazu, selbst mitzubestimmen, wie sich die digitale Welt weiterentwickeln soll. Sie lernen an komplexe Aufgabenstellungen heranzugehen und Lösungsstrategien zu finden. Das Arbeiten mit einer Programmiersprache und das Erweitern der Sprache mit eigenen Befehlen, bietet ein universelles Werkzeug, um zukünftige Probleme anzugehen. Auf diese Weise werden die Lernenden zu Problemlösern, Schaffern und Erfindern und konsumieren nicht nur die Erfindungen anderer.

## Literatur

1. Arnold J, Kohn T, Plüss A. http://www.tigerjython.ch, letzter Zugriff: 20.1.2019
2. Forster M et al. (2018) Autonomous recovery from programming errors made by primary school children. International Conference on Informatics in Schools: Situation, Evolution, and Perspectives. Springer, Cham
3. Hromkovič J (2017) Einfach Informatik 7/9 – Programmieren. Klett und Balmer AG, Zug
4. Hromkovič J (2017) Einfach Informatik 7/9 – Programmieren (Begleitband). Klett und Balmer AG, Zug
5. Hromkovič J, Lacher R (2017) The computer science way of thinking in human history and consequences for the design of computer science curricula. In: Proc. of ISSEP 2017. LNCS 10696:3–11

6. Hromkovič J, Kohn T, Komm D, Serafini G (2016) Combining the power of python with the simplicity of logo for a sustainable computer science education. In: 9th International Conference on Informatics in Schools: Situation, Evolution, and Perspectives (ISSEP 2016), Münster, Germany, October 13–15, 2016. Springer, Cham, pp 155–166
7. Hromkovič J, Serafini G, Staub J (2017) XLogoOnline: a single-page, browser-based programming environment for schools aiming at reducing cognitive load on pupils. In: International Conference on Informatics in Schools: Situation, Evolution, and Perspectives. Springer, Cham
8. krgarden.ca und tts-international.com, letzter Zugriff: 22.2.2019

# *Denn sie wissen nicht, was sie programmieren*

**Tobias Kohn · Dennis Komm**

Was ist ein Computer? Eine unschuldige Frage, die zu beantworten jedoch gar nicht so einfach ist. Wir könnten etwa entgegnen, dass ein Computer eine Rechenmaschine ist, die Anweisungen ausführt. Dann stellt sich allerdings unmittelbar die Frage, was es denn bedeutet zu rechnen, und welche Anweisungen ein Computer ausführen kann. Muss er beispielsweise die Multiplikation beherrschen? Kann ein Computer die Anweisung ausführen, alle perfekten Zahlen auszugeben?

Alan Turing hat Mitte der Dreißigerjahre des zwanzigsten Jahrhunderts das Konzept einer universellen Rechenmaschine vorgestellt, die einigen wenigen Bedingungen genügen muss [10]. Wenn eine Maschine diese Grundbedingungen erfüllt, dann kann sie alles berechnen, was überhaupt berechnet werden kann (zumindest gehen wir stark davon aus). Fassen wir einen Computer als Umsetzung einer solchen universellen Rechenmaschine auf, dann muss er die Multiplikation nicht unbedingt vom Design her kennen und kann trotzdem beliebige Zahlen miteinander multiplizieren. Für einen konkreten Prozessor bedeutet das, dass wir uns die Multiplikation mit einem Programm aus der Addition und Subtraktion selbst bauen können.

Ist das nicht bemerkenswert? Aus einigen wenigen Grundoperationen können wir stückweise eine Maschine bauen, die prinzipiell alles Mögliche berechnen kann. Genau diese Idee wollen wir natürlich auch in den Informatikunterricht hineinbringen: Mit den richtigen Grundbausteinen lässt sich alles bauen.

Didaktisch stehen wir aber noch immer vor der ursprünglichen Frage: Was ist ein Computer? Schülerinnen und Schüler haben am Anfang ihrer Karriere keine Antwort hierauf. Insbesondere fehlt Ihnen eine Vorstellung von den konkreten Grundbausteinen der Maschine und mit welchen Anweisungen sie diese Grundbausteine kombinieren und anwenden können. Schon vor über dreißig Jahren hat sich gezeigt, dass Lernende annehmen, dass eine Variable mit dem Namen „max" automatisch das Maximum aller Werte enthalten wird [9]. Dem Computer wird ein Verständnis des Programmcodes angedichtet; die Fähigkeit, dem Code nicht nur buchstabengetreu zu folgen, sondern den Sinn dahinter zu verstehen und das Programm damit korrekt zu interpretieren und umzusetzen. Sie kommunizieren mit dem Computer wie mit einem Menschen [8]. Mit den Sprachassistenten wie „Alexa" und „Siri" hat sich die Illusion des Computers, der uns versteht, weiter verstärkt.

Diese Vorstellung eines selbstständig agierenden Wesens des Computers wird immer wieder in der verwendeten Sprache deutlich. Wenn wir angeben, „dem Computer beizubringen", wie er eine Aufgabe löst, dann unterstellen wir ihm eine Lernkapazität. Sprachlich wird der Computer zu einem Wesen, das unseren Anweisungen Folge leistet und neue Tricks lernt. Dieses implizite Verständnis wird auch in Schülerfragen deutlich, etwa „Wie sage ich dem Computer, was er tun soll?" Eigentlich ist das

https://doi.org/10.1007/s00287-019-01157-2
© Springer-Verlag Berlin Heidelberg 2019

Tobias Kohn
University of Cambridge,
Cambridge, Großbritannien
E-Mail: tk534@cam.ac.uk

Dennis Komm
ETH Zürich und PH Graubünden,
Zürich, Schweiz
E-Mail: dennis.komm@inf.ethz.ch

Computerprogramm nur eine Maschine, die wir laufen lassen; eine Art virtuelle Kugelbahn mit vorgegebenen Abläufen. Der Computer als universelle Rechenmaschine hat genauso wenig Verständnis für unser Programm, wenn er es simuliert und ausführt, wie die Murmel, die einer vorgegebenen Bahn entlang der Schwerkraft folgt. Im Unterschied zur Murmel, die wir bei ihrem Lauf beobachten können, bleibt der Computer aber eine Blackbox [1]. Die Ausführung passiert irgendwo im Verborgenen – viel zu klein, zu komplex und zu schnell für menschliche Beobachtung. Darin liegt das große Dilemma des Programmierunterrichts: Wie versteht man eine Maschine, die man nicht sehen, anfassen oder begreifen kann?

Als einen möglichen Ausweg verwenden wir Analogien. Besonders oft wird der Computer mit einem denkenden Prozessorkern und einem Gedächtnis im Speicher dargestellt. Variablen sind Orte im Speicher, die Werte aufnehmen können. Der Computer lädt dann die Werte aus den Variablen in den Prozessor, rechnet mit ihnen und speichert die Resultate wieder in Variablen. Für das Verständnis bringt eine solche Analogie allerdings wenig. Der Computer wird hier mittels statischer Bauteile beschrieben, die bestimmte Aufgaben übernehmen. Die Dynamik, der Ablauf innerhalb und zwischen diesen Teilen bleibt aber weiterhin magisch. In erster Linie bleiben wir damit die Antwort schuldig, wer oder was die Aktionen ausführt.

Von technischer Seite stellt sich das Problem, dass ein Programm nur indirekt vom physischen Computer ausgeführt wird. Wenn wir ein Python-Programm auf einem üblichen Desktop-Computer ausführen, dann emuliert der eigentliche Prozessor einen Intel-Prozessor, auf dem der Python-Interpreter einen virtuellen Python-Computer emuliert, der dann das eigentliche Programm ausführt. Die verschiedenen Stufen der Emulation in dieser Pyramide sind hochkomplex und ihr Verständnis erfordert ein mehrjähriges Studium. Zwischen dem Python-Programm einer Schülerin oder eines Schülers und den physisch im Computer ablaufenden Prozessen gibt es eine so große konzeptuelle Differenz, dass eine Erklärung auf Stufe der Hardware wenig zum Verständnis des eigentlichen Programmablaufs beiträgt.

Lassen wir ein Programm schrittweise ausführen, so erhalten wir wiederum nur statische Bilder. Als wollten wir das Konzept der Bewegung verstehen, indem wir die einzelnen Bilder eines Films betrachten. Bereits im antiken Griechenland erkannte Zenon, dass diese Weltsicht zu Problemen und Paradoxien führt: Bewegung kann nicht über statische Bilder begriffen werden. Für den Informatikunterricht haben sich deshalb all jene Systeme bewährt, die dem Computer einen begreifbaren Körper geben. Das kann eine Schildkröte sein, die auf den Bildschirm zeichnet, eine Katze, die einem Weg folgt, oder ein Roboter, der die Anweisungen in physisch sichtbare Bewegung umwandelt. All diese Systeme haben gemein, dass das Grundvokabular der Maschine genauso sichtbar wird, wie die Ausführung – selbst wenn diese wiederum simuliert ist. Der Computer ist nicht mehr eine mysteriöse Blackbox, sondern wird durch eine konkrete Maschine ersetzt. Es sind die Schildkröte und der Roboter, die die Anweisungen ausführen. Welche Anweisungen sie verstehen wird direkt durch die physische Form sichtbar. Allerdings verfehlt ein Roboter mit Spracherkennung dieses Ziel dann bereits wieder. Damit suggerieren wir nämlich wiederum die Fähigkeit eines Verständnisses und die Maschine erhält das Wesen zurück, welches unsere Wünsche interpretiert, anstatt dem Lauf eines Programms zu folgen. Ferner sind diese konkreten Maschinenmodelle letztlich in ihrer Anwendung beschränkt. Die universelle Rechenmaschine hat einen größeren Funktionsumfang als durch eine Schildkröte oder einen sich physisch bewegenden Roboter umgesetzt werden kann.

Die Fehlvorstellungen, die die Schülerinnen und Schüler entwickeln, zeigen sich auf sehr vielfältige Weise. Ein Schüler fragt etwa: „Wenn ich ‚3 + 4' in einer Variablen speichere, wie sage ich dann dem Computer, wann er die Rechnung ‚3 + 4' und wann er das Resultat ‚7' ausgeben soll?" (siehe Programm 1).

**Programm 1:** Ausgabe einer Variablen *x* in Python mit zwei unterschiedlichen, gewünschten Resultaten.

```
x = 3 + 4
print("Die Rechnung", x, "ergibt", x)
```

Damit wird bereits deutlich, wie schwierig das Konzept von Variablen in der Informatik ist [3]. In kaum einer Programmiersprache steht eine Variable für den ganzen arithmetischen Ausdruck wie hier ‚3 + 4' und Ausdrücke werden immer gleich ausgewertet

– also noch vor der Zuweisung zu einer Variablen. Das gleiche Phänomen tritt auf, wenn wir den Schülerinnen und Schülern Programm 2 vorlegen und danach fragen, was ausgegeben wird. Rund ein Drittel entscheidet sich für die Antwort ‚25' anstelle der korrekten Antwort ‚9'.

**Programm 2:** Eine Variable $y$ in Python, deren Wert von einer weiteren Variablen $x$ abhängt, die wiederum unterschiedliche Werte annimmt.

```
x = 3
y = x * x
x = 5
print(y)
```

Aus mathematischer Sicht ergibt diese falsche Antwort allerdings durchaus Sinn. Schließlich ist das Symbol $y$ im Programmcode als $x \cdot x$ definiert; und zum Zeitpunkt der Ausgabe hat $x$ den Wert 5. In der Programmierung funktioniert diese Argumentation aber nicht: Die Zuweisung `y = x * x` definiert eben keinen festen Zusammenhang zwischen den Symbolen $x$ und $y$ wie in der Mathematik.

Als erfahrene Informatikerin bzw. erfahrener Informatiker mit einer festen Vorstellung von Variablen sind wir versucht, das obige Fehlverständnis dahingehend zu interpretieren, dass die Schülerinnen und Schüler glauben, dass ganze Ausdrücke oder Formeln in einer Variablen gespeichert werden, anstelle von einzelnen Werten. Den Variablen wird offenbar ein funktionaler Aspekt angedichtet, der so nicht existiert. Der Ausweg scheint damit klar gegeben: Wir müssen im Unterricht das Konzept der Datentypen klar betonen und darauf hinweisen, dass eine Variable zunächst nur einen einzelnen Wert (mit gegebenem Datentyp) speichern kann. Wenn die Schülerinnen und Schüler erst einmal verstehen, dass sowohl $x$ als auch $y$ in den obigen Beispielen ganze Zahlen sein müssen, dann sind die Fehler nur zu offensichtlich.

Bei dieser Herangehensweise verkennen wir aber, dass die Schülerinnen und Schüler am Anfang ihres Programmierunterrichts einen völlig anderen Erfahrungshintergrund besitzen als wir als Informatikerinnen und Informatiker. Die meisten von ihnen werden bereits im Mathematikunterricht ähnlich wirkende Problemstellungen gesehen haben. Auch die Mathematik kennt das Prinzip von Datentypen mit den ganzen und rationalen Zahlen etc. Es ist vor diesem Hintergrund auch kein Widerspruch zu fordern, dass $x \cdot x$ eine ganze Zahl sein muss – tatsächlich ist diese Forderung in der Mathematik völlig orthogonal zu der Idee, dass der Ausdruck sofort ausgewertet werden soll. Solange Schülerinnen und Schüler also von der Mathematik her denken, hat die Idee, dass eine Variable oder ein Wert einem Datentyp entsprechen müssen, nichts mit deren Auswertung zu tun. Unsere didaktisch so schön aufgebaute Intervention läuft einfach ins Leere.

Da der Computer aber letztlich eine Rechenmaschine ist, ist es da nicht natürlich, mit der Vorstellung an ihn heranzugehen, dass er gleich rechnet wie wir das in der Mathematik lernen? Dass auch der Computer Ausdrücke nach algebraischen Regeln umformt, vereinfacht und dann schließlich vom Ende her Werte für die Variablen einsetzt? Der effektive Wert einer Variablen ist ja oft erst am Ende bekannt, sodass es sinnvoll ist, bis zum Schluss mit dem Einsetzen von Variablenwerten zu warten.

Die Mathematik baut in ihrem Umgang mit Symbolen primär auf dem Substitutionsprinzip auf: Wenn Symbol und Symbolfolgen (Ausdrücke) äquivalent sind, dann dürfen wir sie in jedem weiteren Ausdruck nach Belieben ersetzen (das geht in beide Richtungen). Konkret lernen die Schülerinnen und Schüler das Konzept einer Variablen an Aufgaben wie „Gegeben sei $17 = 5 \cdot x + 2$, finde $x$". Eine Lösung erfolgt in mehreren Schritten durch Äquivalenzumformungen:

$$\begin{aligned} & 17 = 5 \cdot x + 2 \\ \Leftrightarrow\ & 15 = 5 \cdot x \\ \Leftrightarrow\ & 3 \ \ = x \end{aligned}$$

Die letzte Zeile sagt uns, dass wir $x$ und 3 im Kontext dieser Aufgabe gleichsetzen können. Dies gilt somit für alle der obigen drei Zeilen. Insbesondere überprüfen die Schülerinnen und Schüler durch Einsetzen in die erste Zeile, ob die Aufgabe korrekt gelöst wurde.

Demgegenüber verwendet die Programmierung ein ganz anderes Konzept der „Variablen". Eine Variable ist ein Symbol, das für einen konkreten, zur Laufzeit existierenden Wert steht. Eine Variable ohne Wert ist tatsächlich ein Fehler und führt zum Abbruch der Programmausführung. Computer rechnen nicht mit Symbolen, sondern mit Werten.

Das Zuweisen eines Wertes ist ein weiterer Fallstrick, da dies in vielen Programmiersprachen, wie

beispielsweise auch Python, über das Gleichheitszeichen geschieht (eine bekannte Ausnahme ist z. B. Pascal mit ‚:=‘). Eine anfängliche Zuweisung wie `x=2` kann intuitiv schnell verstanden werden, aber das Inkrementieren einer Variablen mit `x=x+1` ist, wenn wir das Gleichheitszeichen so verstehen, wie wir es aus dem Mathematikunterricht kennen, extrem verwirrend; $x$ ist also eine Zahl, die gleich sich selber plus 1 ist? Hier ist es unabdingbar, genau zu erklären, wie der Computer ‚=‘ versteht, insbesondere, dass wir die linke und die rechte Seite nicht vertauschen können. Wenn wir den Sinn hinter `x=x+1` verstanden haben, mag es dennoch irritieren, dass `x+1=x` zu einem Fehler führen wird.

Insbesondere bei jungen Schülerinnen und Schülern müssen wir mit der Einführung von Variablen also enorm vorsichtig sein, da es hier aus diversen Gründen, beispielsweise, wenn der Begriff analog zum Mathematikunterricht verstanden oder dem Computer ein Intellekt zugesprochen wird, zu Fehlvorstellungen kommen kann. Allerdings kommen wir natürlich nicht ohne Variablen aus, wenn wir seriösen Programmierunterricht zum Ziel haben.

An dieser Stelle sei noch auf einen besonderen Punkt aufmerksam gemacht. Das im Folgenden thematisierte Konzept der Schleifen scheint ohne Variablen z. B. schwer umzusetzen zu sein, weswegen letztere meist vor ersteren im Unterricht vorgestellt werden. Allerdings bieten einige Programmiersprachen wie LOGO [7] mit der **`repeat`**-Schleife Alternativen hierzu. Diese erlaubt, Anweisungen eine feste Anzahl von Malen zu wiederholen und benötigt deswegen keine Variablen, kann aber im weiteren Verlauf des Unterrichts mit diesen kombiniert werden; das Konzept wird beispielsweise auch in Varianten von Python verwendet [2, 6].

Probleme im Zusammenhang mit Schleifen ergeben sich spätestens, sobald wir über das Abbrechen einer Wiederholung bzw. das Verlassen einer Schleife sprechen. Hier offenbart sich nämlich, wie schwierig das Konzept der sequenziellen Ausführung von Rechenschritten ist.

Programm 3 zeigt eine einfache **`while`**-Schleife zur Ausgabe von Quadratzahlen. Welches ist die letzte Quadratzahl, die hier ausgegeben wird? Mitunter argumentieren Schülerinnen und Schüler hier folgendermaßen, dass die letzte ausgegebene Zahl 1 sein muss (obwohl 0 die korrekte Antwort wäre): „Im letzten Durchgang der Schleife hat $x$ zunächst den Wert 1. In Zeile 3 wird dieser Wert jedoch zu 0, sodass die Bedingung der Schleife ($x > 0$) nicht mehr erfüllt ist, und die Schleife damit abbricht noch bevor die **`print()`**-Anweisung ausgeführt würde.“

**Programm 3:** Eine **`while`**-Schleife in Python zur Ausgabe von Quadratzahlen.

```
x = 5
while x > 0:
  x -= 1
  print(x * x)
```

Wir entdecken in dieser Fehlvorstellung eine ähnliche Idee wie bereits zuvor mit dem schrittweisen Auswerten von Variablen und Ausdrücken. Dem Computer wird eine Übersicht über das gesamte Programm zugestanden. Es wird erwartet, dass er das **`while`** `x > 0` wirklich als ein „solange $x$ größer ist als 0“ übersetzt und diese Bedingung während der Ausführung des Schleifenkörpers „im Hinterkopf“ behält. Schon aus unserer Sprache wird deutlich: Der Computer wird hier zu einem selbstständig agierenden Wesen, das den Programmcode „versteht“.

Auf der anderen Seite zeigt Programm 4 ein Beispiel, wo das Abbrechen der Schleife ganz anders verstanden wird. Das Stoppen der Wiederholung mittels **`break`** soll zwar die Schleife beenden, damit aber noch kein Herausspringen aus dem Schleifenkörper bewirken. In der Realität wird die Ausgabeanweisung in Zeile 5 aber natürlich nie ausgeführt. Bitten wir die Schülerin bzw. den Schüler, ihren bzw. seinen Code in Programm 4 zu beschreiben, so erhalten wir an der kritischen Stelle: „Wenn $x$ durch $t$ teilbar ist, dann beende die Schleife und gib den Teiler $t$ aus.“

**Programm 4:** Eine `while`-Schleife in Python zum Testen auf Teilbarkeit einer Zahl $x$.

```
t = 2
while t < x:
  if x % t == 0:
    break
    print(x, "ist teilbar durch", t)
  else:
    t += 1
```

In beiden Fällen ist es sicherlich eine gute Idee, die Schleife mit den Schülerinnen und Schülern zusammen zu expandieren und die sequenzielle Arbeitsweise des Computers somit explizit zu machen und zu untersuchen. Während dies im zweiten Fall den Fehler schnell finden lässt, ist es für den ersten aber unabdingbar, den Prozess der Schleifenausführung genauer zu betrachten und insbesondere darauf aufmerksam zu machen, dass der Computer nicht mitdenkt; die Bedingung $x > 0$ wird nur genau dort getestet, wo sie auch explizit im Programmcode erscheint und als Teil der sequenziellen Abarbeitung ausgeführt wird.

Auf den ersten Blick scheinen Computer-Algebra-Systeme durchaus in der Lage zu sein, symbolisch zu rechnen. Zusammen mit ihren Möglichkeiten zur Programmierung scheinen solche Systeme dem Denken der Schülerinnen und Schüler näher zu sein. Warum also nicht einfach anstelle einer imperativen Programmiersprache wie Python ein Mathematiksystem verwenden?

Zunächst einmal sollten wir uns bewusst sein, dass all diese Fehlvorstellungen immer nur einen Teil der Schülerinnen und Schüler betreffen, der in der Regel unter der Hälfte liegt. Mehr noch wollen wir mit dem Informatikunterricht ja bewusst etwas anderes als nur mathematische Konzepte auf den Computer zu übertragen. In der Informatik spielt die sequenzielle und zeitliche Abfolge der einzelnen (Rechen-)Schritte eine zentrale Rolle. Das verursacht beim Lernen bisweilen Schwierigkeiten, eröffnet aber auch eine völlig neue Sicht auf die Welt.

Nehmen wir als Beispiel eine große Zahl $z$ und fragen uns, ob diese Zahl $z$ eine Primzahl ist. Aus Sicht der Mathematik liegt die Antwort schon von vornherein fest. Die Eigenschaft, eine Primzahl zu sein, wohnt der Zahl $z$ intrinsisch inne oder eben nicht. Aus Sicht der Informatik müssen wir die Antwort aber durch einen Algorithmus bestimmen; somit wird eine existenzielle Aussage („Es existiert ein nichttrivialer Teiler von $z$ oder nicht“) zu einer konstruktiven Suche nach einem solchen Teiler. Durch die zeitliche Abfolge der einzelnen Schritte im Algorithmus ergibt sich dadurch natürlicherweise die Frage danach, wie viel Zeit der ganze Algorithmus benötigt und welches die kleinste Anzahl der Schritte ist, die gebraucht werden, um die Frage korrekt zu beantworten.

Damit sind wir bereits mitten im Kern der Informatik. Über die Anzahl der Lösungsschritte können wir die Effizienz von Algorithmen und Lösungsverfahren vergleichen und damit Probleme gemäß ihrer Schwierigkeit klassifizieren. Gerade im Zusammenhang mit der Teilbarkeit gegebener Zahlen können wir hier einfache Beispiele finden, um das Konzept der Zeitkomplexität zu erläutern [4]. „Ist die Zahl $z$ durch 2 teilbar?“ kann beispielsweise einfach beantwortet werden, indem wir auf die letzte Stelle von $z$ schauen. Diese letzte Stelle anzugucken, egal ob $z$ als Dezimal- oder Binärzahl dargestellt wird, bedeutet für jedes $z$ denselben und somit einen konstanten Aufwand. „Ist $z$ durch 5 teilbar?“ lässt sich ebenfalls mit konstantem Zeitaufwand beantworten. „Ist $z$ durch 3 teilbar“ kann allerdings nicht so einfach beantwortet werden. Hier brauchen wir einen Algorithmus, der mehr Arbeit investiert und beispielsweise die Quersumme von $z$ berechnet. Auch ohne diese Strategie im Detail zu implementieren, sehen die Schülerinnen und Schüler, dass der Zeitaufwand hier nicht mehr konstant ist, sondern mit der Anzahl der Dezimalstellen von $z$ zusammenhängt.

Die obige Frage „Hat $z$ einen nichttrivialen Teiler?“ können wir nur mit noch mehr Zeitaufwand beantworten. Auf diesem Weg können wir eine erste, gute Intuition für unterschiedliche Komplexitätsklassen aufbauen, ohne allzu formal zu werden [4].

Natürlich hängt die Anzahl der Schritte von der Maschine ab, die den Algorithmus ausführt. Ein einfacher Prozessor muss die Multiplikation vielleicht über einen längeren Algorithmus ausführen, während größere Prozessoren das in einem Schritt erledigen. Um also überhaupt die Frage sinnvoll stellen zu können, wie viele Schritte ein Algorithmus benötigt, müssen wir die ausführende Maschine bestimmen. Wir müssen klären, was ein Computer ist. Welche bessere Möglichkeit gäbe es da, als den Computer durch die Programmierung zu erforschen?

Im Allgemeinen sollten Programmieraufgaben immer im Kontext mit einem konkreten Maschinenmodell gestellt werden [5]. Implizit passiert dies in beinahe allen Fällen. Was hilft es schon, einen Sortieralgorithmus zu implementieren, indem man Pythons `sorted()`-Funktion verwendet? Man spricht der Rechenmaschine diese Fähigkeit also ab und erlaubt nur das Verwenden von „Grundopera-

tionen“ wie einfachen arithmetischen Operationen, Vergleichen, Schleifen etc. Auch sollte eine `min()`-Funktion offensichtlich nicht verwendet werden, wenn beispielsweise Bubblesort implementiert wird; dadurch ginge nämlich der Kern des Algorithmus verloren und ein wesentlicher Teil der Komplexität wird vor der Programmierin bzw. dem Programmierer versteckt. Wir schränken die Fähigkeiten der Rechenmaschine also ein und das aus gutem Grund. Steht das Sortieren allerdings nicht im Vordergrund, sondern soll beispielsweise ein Greedy-Algorithmus für ein gegebenes Problem implementiert werden, so scheint das Verwenden von `sorted()` durchaus legitim.

Eine tiefe Einsicht und gute Intuition für den Komplexitätsbegriff kann vermittelt werden, wenn wir Turings anfangs erwähnte Idee einer universellen Rechenmaschine, die nur wenige Operationen beherrscht, nachempfinden. Vorausgesetzt wird eine Maschine, mit der lediglich zwei Ziffern addiert werden können und die Vergleiche, Variablen und Schleifen beherrscht. Hieraus bauen wir zunächst eine Funktion, mit der zwei natürliche Zahlen mit beliebig vielen Stellen addiert werden können – das ist schwerer als man zunächst denken mag, denn natürlich kann ein Übertrag entstehen, der korrekt verrechnet werden muss.

Anschließend verwenden wir diese neue Funktion, um eine weitere Funktion zu schreiben, die zwei beliebige natürliche Zahlen multipliziert. Für zwei Zahlen $x$ und $y$ kann dies korrekt gelöst werden, indem wir $x$ genau $y - 1$ Mal zu sich selbst addieren. Allerdings ist der entsprechende Algorithmus viel zu langsam (er besitzt eine Laufzeit, die exponentiell in der Eingabelänge ist); eine clevere Alternative bietet der Schulalgorithmus.

Nach dessen Implementierung können wir diesen wiederum verwenden, um zu exponenzieren. Auch hier ist der naive Ansatz allerdings zu langsam und die Methode des wiederholten Quadrierens kann vorgestellt, analysiert und implementiert werden. Stück für Stück machen wir unsere Maschine immer mächtiger und können auf dem Weg viele spannende Konzepte rund um Algorithmen und ihre Komplexität aufzeigen und diskutieren.

Das Modell der ausführenden Maschine mit ihren Fähigkeiten begleitet uns in der Informatik also von den ersten Schritten im Programmieren bis hin zu den Fragestellungen der theoretischen Informatik über Effizienz und Berechenbarkeit.

Zusammenfassend können wir sagen, dass viele Fehlvorstellungen und daraus resultierende Fehler im Programmierunterricht daraus entstehen, dass das zugrundeliegende Maschinenmodell, das den Code ausführt, nicht ausreichend verstanden wird. Als Folge hieraus werden dem Computer Fähigkeiten angedichtet, die er nicht besitzt; diese haben ihren Ursprung beispielsweise in der Verwendung ähnlicher aber eben nicht gleicher Konzepte aus dem Mathematikunterricht. Ferner unterstellen unerfahrene Programmiererinnen und Programmierer dem Computer häufig einen Intellekt, der ihn befähigt, Tätigkeiten zu einem gewissen Zeitpunkt auszuführen, auch wenn sie nicht explizit im Code zu finden sind.

## Literatur


1. du Boulay B, O'Shea T, Monk J (1981) The black box inside the glass box: presenting computing concepts to novices. Int J Man-Machine Stud 14(3):237–249
2. Hromkovič J, Kohn T (2018) Einfach Informatik – Programmieren. Klett & Balmer, Baar
3. Kohn T (2017) Variable Evaluation: An Exploration of Novice Programmers' Understanding and Common Misconceptions. In: Proc SIGCSE 2017, pp 345–350
4. Komm D, Kohn T (2017) An introduction to running time analysis for an SOI workshop. Olymp Inform 11:77–86
5. Komm D, Kohn T (2018) Teaching Programming and Algorithmic Complexity with Tangible Machines. In: Proc ISSEP 2018, LNCS 11169. Springer, Cham, pp 68–83
6. Hromkovič J, Kohn T, Komm D, Serafini G (2016) Combining the Power of Python with the Simplicity of Logo for a Sustainable Computer Science Education. In: Proc ISSEP 2016, LNCS 9973. Springer, Cham, pp 155–166
7. Papert SA (1993) Mindstorms: Children, Computers, and Powerful Ideas, 2nd edn. Basic Books, New York
8. Pea RD (1986) Language-independent conceptual "bugs" in novice programming. J Edu Comput Res 2(1):25–36
9. Putnam RT, Sleeman D, Baxter JA, Kuspa LK (1986) A summary of misconceptions of high school Basic programmers. J Edu Comput Res 2(4):459–472
10. Turing AM (1936) On computable numbers, with an application to the Entscheidungsproblem. P Lond Math Soc 42(2):230–265

# *Wie Mathematik und Informatik im Unterricht voneinander profitieren können*

***Teil 1: Abstraktionsfähigkeit***

**Urs Hauser · Dennis Komm**
**Giovanni Serafini**

## Einleitung

Die Mathematik entstand aus der historischen Notwendigkeit, Güter bzw. Objekte jeglicher Art zu zählen und ihre Werte, zunächst für wirtschaftliche und anschließend für wissenschaftliche Zwecke, zu messen und zu vergleichen. Die Entwicklung möglichst allgemeiner abstrakter Notationen und formaler Methoden, welche den quantitativen Umgang mit Objekten aller Arten ermöglichen, ist ihre primäre und inhärente Aufgabe [2]. Bereits 1623 merkte Galileo Galilei an, dass die Mathematik eine Sprache ist, die den Menschen die Möglichkeit eröffnet, die Welt und die Naturgesetze zu untersuchen und zu beschreiben: „The great book of nature can be read only by those who know the language in which it was written. And this language is mathematics." Lernen, mathematisch zu denken, ist eine der grundlegenden Kompetenzen, die Gymnasiastinnen und Gymnasiasten am Ende ihrer schulischen Laufbahn erreichen sollen.

Die Informatik hat sich wiederum erst im Verlauf der zweiten Hälfte des 20. Jahrhunderts als eigenständige wissenschaftliche Disziplin etabliert. Ihre Wurzeln reichen jedoch weit zurück in die Geschichte der menschlichen Kultur. Die Methoden der Informatik begleiteten und ergänzten die Entstehung und die Entwicklung der Mathematik und die Informatik bediente sich aus den abstrakten, formalen Konzepten und methodischen Ansätzen der Mathematik, um ihren Zweck zu verfolgen, nämlich das Zählen, das Messen und das Vergleichen von Objekten zu automatisieren. Die Durchführung automatisierter (algorithmischer) Berechnungen setzt voraus, dass die zu bearbeitenden Informationen digitalisiert, d. h. als endliche Folgen von Symbolen über einem Alphabet dargestellt werden. Die Informatik stellt heute den Kern der sogenannten Digitalisierung dar. Das allgemeinbildende Schulfach Informatik soll demnach Schülerinnen und Schülern ermöglichen, zu lernen, wie die von Menschen entwickelte technische Welt verstanden und gesteuert, aber auch mitgestaltet werden kann [6].

Die Schulinformatik soll den Schülerinnen und Schülern in einem Spiralcurriculum beibringen, konkrete Problemsituationen zu analysieren, wesentliche Informationen zu identifizieren und für die Lösung des Problems abstrakte und dennoch aussagekräftige Daten formal zu beschreiben. Der iterative Entwurf automatischer (algorithmischer) Lösungswege und die Frage nach der Qualität der Algorithmen bilden den konstruktiven, gestalterischen Kern des Unterrichts. Die in der Schule zu erwerbende Kompetenz wird mit dem Konzept des algorithmischen Denkens sinngemäß beschrieben und beinhaltet alle Denkprozesse, welche zur Entwicklung automatisierter Lösungen in Form von Algorithmen führen [1].

https://doi.org/10.1007/s00287-019-01165-2
© Springer-Verlag Berlin Heidelberg 2019

Urs Hauser
ETH Zürich und PH Luzern,
Zürich, Schweiz
E-Mail: urs.hauser@inf.ethz.ch

Dennis Komm
ETH Zürich und PH Graubünden,
Zürich, Schweiz
E-Mail: dennis.komm@inf.ethz.ch

Giovanni Serafini
ETH Zürich,
Zürich, Schweiz
E-Mail: giovanni.serafini@inf.ethz.ch

Dieser zweigeteilte Beitrag befasst sich mit fächerübergreifenden Kompetenzen des Informatik- und Mathematikunterrichts: Aus schulischer Perspektive bildet der Prozess der mathematischen Modellierung einer Problemstellung das Bindeglied zwischen den beiden Fächern. Die Förderung der Problemlösefähigkeit und des abstrakten Denkens sind zwei fundamentale und fächerübergreifende Kompetenzen des Informatik- und Mathematikunterrichts. Wir stellen die Schnittstellen der beiden Fächer an konkreten Beispielen dar und erläutern, wie das Problemlösen und das abstrakte Denken gefördert werden können.

## Problemlösestrategien

Unter Problemen versteht man Aufgabenanforderungen, für deren Bewältigung man keine bereits verfügbare Lösungsstrategie abrufen kann, sondern diese erst entwickeln muss. Zur Lösung eines Problems muss auf der Grundlage bestehenden Wissens neues Wissen generiert werden. Dafür gibt es verschiedene Strategien, wie zum Beispiel induktives Schlussfolgern oder Analogieschlüsse. „Problemlösekompetenz" bedeutet damit die Fähigkeit, neue Situationen mit einem fachlich angemessenen Repertoire an Methoden und Konzepten zu bewältigen [9].

Der Problemlöseprozess kann in seiner Grundstruktur als Transformation eines Anfangszustandes in einen gewünschten Endzustand beschrieben werden, wobei eine *Barriere* zu überwinden ist [3]. Es handelt sich um einen strukturierten Prozess. Nach George Pólya [8] fördert der Mathematikunterricht die Entwicklung von Heurismen des Problemlösens. Er teilt den Problemlöseprozess in vier Phasen ein: die Analyse und das Verstehen des Problems, den Entwurf eines Plans, seine Implementierung und schließlich die Überprüfung (siehe Abb. 1).

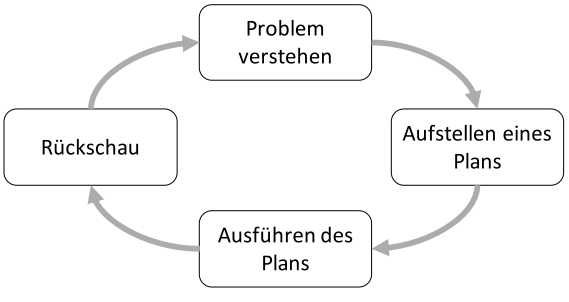



***Abb. 1 Problemlöseprozess nach Pólya***

Um Probleme zu lösen, benötigt man Problemlösestrategien (heuristische Strategien), die wir als allgemeingültige Verarbeitungsvorschriften für eine Klasse von Problemen definieren. Wir fokussieren auf die folgende Auswahl an Heurismen:

1. *Die Variation der Darstellung.*
   Ein Repräsentationswechsel eines Problems durch eine Zeichnung, eine Handlung oder Konstruktion. Ein wesentlicher Prozess der Darstellung eines Problems in der formalen Sprache der Mathematik besteht in der *Abstraktion*.
2. *Die Variation der Problemstellung.*
   Durch eine *Umformulierung* des Problems, zum Beispiel durch eine Variation der Anordnung der Daten, des Anspruchs an die Exaktheit der Lösung oder des Allgemeinheitsgrades, lässt sich allenfalls ein Anknüpfen an bisheriges Wissen bilden. Bei der *Analogiebildung* kann die Lösungsstrategie, die man durch ähnliche Problemstellungen entwickelt hat, auf das aktuelle Problem angewendet werden.
3. *Modularisierung.*
   Zerlegung eines Problems in (einfachere) Teilprobleme, deren Bewältigung die Lösung des Gesamtproblems schrittweise ermöglicht.

Dieser Artikel stellt den ersten von zweien dar, die sich mit der Frage beschäftigen, wie Informatik und Mathematik voneinander profitieren können. Hier fokussieren wir auf den ersten der beiden oberen Punkte, im zweiten Teil gehen wir auf die anderen beiden ein.

## Abstraktion

**Beispiel (Wahrscheinlichkeitsrechnung).** Wie groß ist die Wahrscheinlichkeit, dass beim zweimaligen Werfen eines regulären Würfels die zweite Augenzahl größer ist als die erste?

Gemäß Pólya besteht der erste Schritt bei der Lösung eines Problems in der Analyse und dem Verstehen der Aufgabenstellung. Um die gesuchte Wahrscheinlichkeit berechnen zu können, benötigen wir die Anzahl aller möglichen Wurfergebnisse und aller günstigen Wurfergebnisse (jene, bei denen der Würfel im zweiten Wurf eine größere Zahl zeigt). Es geht also um den in der Einleitung erwähnten ursprünglichen Prozess des *Zählens*. Um effizient zählen zu können, brauchen wir eine möglichst systematische Darstellung der Ergebnisse (Daten). Die Wurfereignisse kann man als zweistellige Zahlen

**Darstellung als Tabelle**

Tabelle 1

| AZ1 \ AZ2 | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| 1 | 11 | 12 | 13 | 14 | 15 | 16 |
| 2 | 21 | 22 | 23 | 24 | 25 | 26 |
| 3 | 31 | 32 | 33 | 34 | 35 | 36 |
| 4 | 41 | 42 | 43 | 44 | 45 | 46 |
| 5 | 51 | 52 | 53 | 54 | 55 | 56 |
| 6 | 61 | 62 | 63 | 64 | 65 | 66 |

(für die Augenzahlen der Würfe) modellieren und als sortierte Liste notieren:

12, 13, 14, …, 45, 46, …, 56, …, 66 .

Die Lernenden sind hier mit zentralen Konzepten der Informatik konfrontiert: der *Darstellung*, der *Suche* und dem *Sortieren* von Daten. Man kann hier den Zusammenhang zum allgemeinen Begriff der digitalen Informationsdarstellung als Folge von Symbolen (hier Ziffern bzw. Augenzahlen der Würfel) aufzeigen. Die vorliegenden Zahlen sind eigentlich nur eine Folge von zwei verbundenen Ziffern (Elementarereignissen), nicht zu verwechseln mit der Augensumme. In der Informatik nennt man das ein *Wort* der Länge 2 über dem *Alphabet* $\{1, 2, 3, 4, 5, 6\}$.

Zurück zur Darstellung der Daten. Eine weit effizientere Variante, Objekte zu zählen, verwendet eine Tabelle (siehe Tab. 1). Mathematisch entspricht sie dem Konzept des *kartesischen Produkts* $A \times A$ der Menge aller möglichen Wurfereignisse $A = \{1, 2, 3, 4, 5, 6\}$. Die Elemente der neuen Menge sind geordnete Paare $(x, y)$ mit $x, y \in A$. Die günstigen Ergebnisse lassen sich in Tabellen besonders anschaulich darstellen und abzählen. Die Wahrscheinlichkeit als Quotient aller günstigen durch alle möglichen Ergebnisse beträgt

$$P(\mathrm{AZ1} < \mathrm{AZ2}) = \frac{15}{36} .$$

Die Tabellendarstellung liefert wichtige Erkenntnisse der abzählenden Kombinatorik, zum Beispiel die Anzahl aller Möglichkeiten, 2 Elemente aus $n$ Elementen unter Beachtung der Reihenfolge und mit möglicher Wiederholung auszuwählen (Variationen mit Wiederholung) $6^2$.

Wenn der Ergebnisraum größer wird (mehrere Würfe), so lassen sich die Ergebnisse nicht mehr in einer einzelnen Tabelle darstellen. Hier kann man zwar den Ansatz der *modularen Methode* verfolgen, der darin besteht, die 36 bisherigen Lösungen der Tab. 1 in einer zweiten Tabelle mit je 36 Zeilen und Spalten mit allen Konkenationen aufzulisten. Die quadratische Form der Tabelle führt uns dann zu $36^2 = 6^4$ Fällen und allgemein für Wörter der Länge $k$ über einem Alphabet mit $n$ Elementen zu $n^k$ Fällen.

Die Darstellung mit Tabellen ist bei mehreren Würfen nicht mehr praktikabel und eine Variation der Darstellung ist notwendig. Hier hilft uns ein fundamentales Konzept, welches die Informatik zur Auflistung von Objekten mit gewissen Eigenschaften liefert: *Baumdiagramme*. Insbesondere bieten sie den Vorteil, nicht mehr alle, sondern nur die uns interessierenden Objekte aufzulisten. In Abb. 2 sind lediglich alle günstigen Lösungen unserer Problemstellung abgebildet. Falls die Problemstellung geändert und zum Beispiel dreimal gewürfelt würde, so müsste der Baum lediglich um eine Stufe erweitert werden. Die *Baumtiefe* entspricht der Länge des Wortes.

**Beispiel (Graphen als Abstraktion).** Im folgenden Beispiel gehen wir näher auf den Prozess der *Abstraktion* ein. Im preussischen Königsberg des 18. Jahrhunderts führten sieben Brücken über den Fluss Pregel (siehe Abb. 3). Beim *Königsberger-Brücken-Problem* geht es um die Fragestellung, ob es einen Rundweg gibt, bei dem man jede der Brücken genau einmal überqueren kann.

Es handelt sich um ein topologisches Problem, bei dem es nicht auf die Lage der Brücken ankommt, sondern nur darauf, welche Brücke welche Stadt-

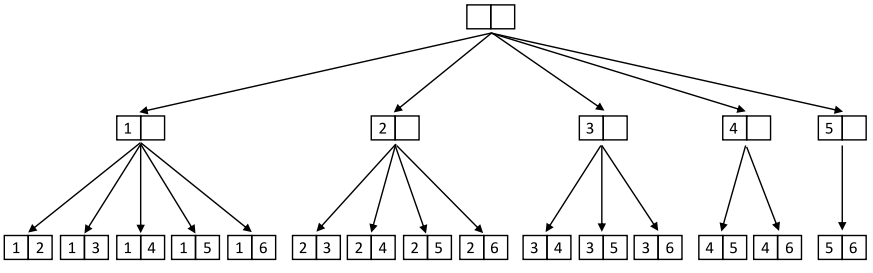



***Abb. 2 Teillösungen im Baumdiagramm***


***Abb. 3 Königsberg und seine Brücken***

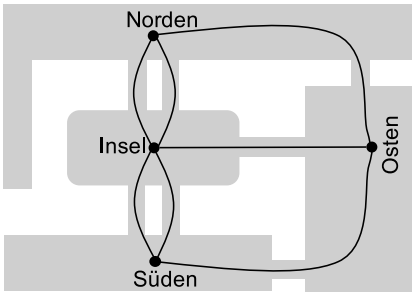



***Abb. 4 Graph als Abstraktion der Brücken***

teile verbindet. Um ein besseres Verständnis für die Problemstellung zu erlangen, abstrahieren wir die schriftlich und grafisch präsentierte Fragestellung, indem wir sie auf das Wesentliche reduzieren. Dazu modellieren wir den Rundgang als einen (*Multi-*) *Graphen*, wobei wir die einzelnen Stadtgebiete (oder Ufer) als *Knoten* und die Brücken als *Kanten* darstellen (siehe Abb. 4). Der *Grad* eines Knotens bezeichnet die Anzahl der Kanten, die in einem Knoten zusammenlaufen. In unserem Beispiel wäre dies die Anzahl der Brücken, durch die die Stadtteile miteinander verbunden sind.

Mithilfe der Abstraktion als Graph kann man nun das Problem kontextfrei und fokussiert auf die Fragestellung angehen. Damit lässt sich die gleiche Problemstellung auch allgemein lösen; egal, ob es sich bei den Kanten um Brücken, Straßen, Flüsse und bei den Knoten um Stadtteile, Personen oder andere Objekte handelt. Man erreicht mit der Abstraktion eine Vereinfachung und gute Veranschaulichung der Problemdarstellung und damit eine Reduktion der Komplexität.

Der Rundgang in Königsberg wird dann auf einen sogenannten *Eulerkreis* reduziert, der zusammenhängend ist und bei dem jede Kante genau einmal besucht wird.

Durch Probieren stellt man schnell die Unlösbarkeit der Aufgabe fest. Das Problem wurde von Leonhard Euler für andere Stadtpläne bzw. Graphen verallgemeinert. „*In einem zusammenhängenden Graphen existiert genau dann ein Eulerkreis, wenn alle Knoten geraden Grad besitzen.*“ In unserem Beispiel haben offensichtlich alle Knoten einen ungeraden Grad.

**Beispiel (Zahlen als Abstraktion).** „Zählen“ ist eine der fundamentalen Handlungen unserer Zivilisation, wie frühe Funde von gekerbten Knochen aus dem Spätpaläolithikum (30.000 bis 20.000 Jahre v. Chr.) belegen. Durch das Einkerben von Holz oder Knochen oder das Anlegen von Steinhäufchen konnten Gegenstände gezählt und ihre Anzahl festgehalten und sogar einfache Rechnungen ausgeführt werden [7]. Ein Hirte war damit in der Lage, den Bestand seiner Herde festzustellen, ohne über einen abstrakten Zahlenbegriff zu verfügen.

Der Übergang vom Zählen zum abstrakten *Zahlenbegriff* als Quantifizierung dauerte allerdings sehr lange. Die Entwicklung von Handwerk und Handel trug wesentlich dazu bei. Besonders hervorzuheben ist das Niveau der Mathematik in Mesopotamien, ca. 5 000 v. Chr. Wir sehen hier den *Ursprung der Digitalisierung*, denn schon damals wurde eine eigene Schrift als Folge von Symbolen und ein eigenes Zahlensystem entwickelt und verwendet. Zahlen waren also ein grundlegendes Mittel zur Beschreibung des (damaligen) Lebens und zur langfristigen Speicherung der Steuereinträge und Buchaltung des Landes. Zahlen standen von Anfang an im Fokus der symbolischen Darstellung und Schriftentwicklung. Die Entwicklung von Symbolen zur Darstellung von Zahlen und anderen Objekten ist ein Abstraktionsschritt, mit dem man die Welt beschreiben und modellieren kann.

Auf den Informatikunterricht trifft dies zu, weil es zu den Grundkompetenzen von Informatikerinnen und Informatikern gehört, unterschiedliche Objekte, Beziehungen und Informationen als Folge von Symbolen darzustellen, um sie abspeichern und bei Bedarf mit dem Computer bearbeiten zu können [5]. Als Informatikerin bzw. Informatiker achtet man hierbei darauf, dass die digitale Informationsdarstellung eine effiziente Durchführung der gewünschten Berechnungen (der

Datenverarbeitung) ermöglicht. Die Entwicklung der Zahlendarstellung ist ein schönes Beispiel. Die adischen Systeme der Zahlendarstellung haben sich gegen die Konkurrenz durchgesetzt, weil sie eine effiziente Durchführung der Multiplikation ermöglichen.

**Beispiel (Kombinatorik).** Die folgende Aufgabe stammt aus einer für Primarschüler (6. Schuljahr) obligatorischen Mathematik-Aufnahmeprüfung an das Gymnasium Glarus aus dem Jahr 2008. Es handelt sich um eine für die Lernenden ungewohnte und anspruchsvolle Problemstellung. Je ein Drittel der Prüflinge hat bei der Aufgabe die volle oder die halbe Punktzahl oder sogar keinen Punkt erzielt. Die unterschiedlichen Problemstrategien wurden sehr gut sichtbar und zeigten auch die Fehlvorstellungen bei der rechnerischen Bestimmung der Anzahl der Möglichkeiten. Die Darstellung der Situation als abstrahierter Graph machte keinerlei Probleme.

Abbildung 5 zeigt alle möglichen Skiabfahrten von der Bergstation $A$ zur Talstation $D$ eines Skigebiets, dargestellt als gerichteter Multigraph.

(a) Wie viele verschiedene Abfahrtsrouten gibt es, welche von $A$ über $B$ nach $D$ verlaufen?
(b) Fridolin möchte auf möglichst vielen verschiedenen Routen von $A$ nach $D$ fahren. Wie viele Möglichkeiten hat er?

Der am häufigsten aufgetretene Fehler bei den Schülerlösungen war das falsche Berechnen der möglichen Routen zwischen zwei Knoten. Hier wurde einfach gezählt und aufsummiert. Zwischen $A$ und $D$ über $B$ wurden $2 + 3$ anstatt $2 \cdot 3$ Wege berechnet. Eine systematische Darstellung als Baumdiagramm wie in Abb. 6 schafft Klarheit und hilft beim Verständnis der *Produktregel*. Die Anzahl möglicher Wege von $A$ nach $D$ ist $2 \cdot 3 + 2 \cdot 2 \cdot 2 + 1 \cdot 2 = 16$.

In höheren Klassen (ab dem 10. Schuljahr) kann man im Rahmen des Programmierunterrichts überlegen, wie sich eine solche Problemstellung mit dem Computer lösen lässt. Wie kann man einen Graphen mathematisch darstellen und in einem Programm abspeichern? Ein Graph $G = (V, E)$ besteht aus einer Menge von Knoten $V$ und einer Menge von Kanten $E$, wobei die Knoten durchgehend von 1 bis $n$ nummeriert sind. Eine Möglichkeit, einen Graphen mathematisch darzustellen, ist die Adja-

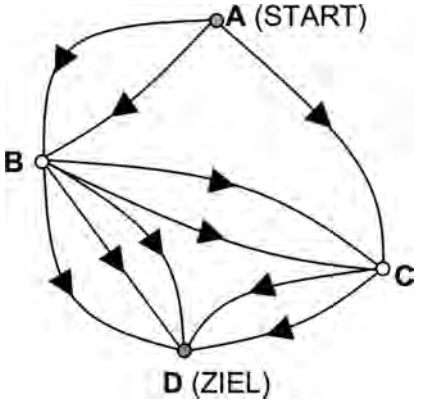



*Abb. 5 Skigebiet als Multigraph*

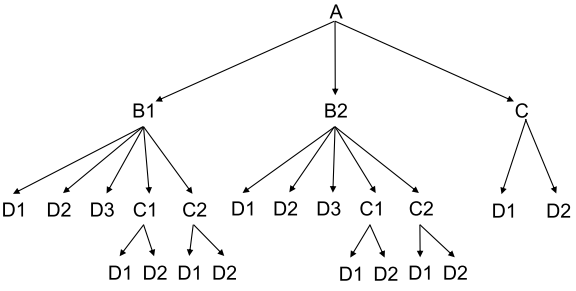



*Abb. 6 Skipisten als Baum*

zenzmatrix (Nachbarschaftsmatrix), deren Einträge $a_{ij}$ der Anzahl der Kanten vom $i$-ten Knoten zum $j$-ten Knoten entsprechen. In unserem Beispiel entspricht $a_{12} = |(1, 2)| = |(A, B)| = 2$ der Anzahl der Kanten von $A$ nach $B$. Die ganze Matrix für das Skigebiet ist

$$M = \begin{pmatrix} 0 & 2 & 1 & 0 \\ 0 & 0 & 2 & 3 \\ 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & 0 \end{pmatrix}.$$

Hier kann man im Mathematikunterricht an die *Matrizenrechnung* anknüpfen, zum Beispiel bei der Berechnung von Übergangswahrscheinlichkeiten mit stochastischen Matrizen für Markov-Ketten.

Eine weitere Speichermöglichkeit ist die Adjazenzliste, in der die Nachbarknoten jedes Knotens gespeichert werden, also `liste=[[1,1,2],[3,3,3,2,2],[3,3]]`.

Im zweiten Teil des Artikels werden wir Beispiele für die oben genannten Heurismen *Varia-*

*tion der Problemstellung* und *Modularisierung* aufzeigen.

## Literatur


1. Aho AV (2012) Computation and computational thinking. Comput J 55(7): 832–835
2. Devlin K (2012) Introduction to Mathematical Thinking. K. Devlin, Petaluma
3. Heinze A (2007) Problemlösen im mathematischen und außermathematischen Kontext. Journal der Mathematik-Didaktik 3–30
4. Hromkovič J (2018) Einfach Informatik – Strategien. Klett & Balmer Verlag, Baar
5. Hromkovič J (2018) Einfach Informatik – Daten darstellen, verschlüsseln, komprimieren. Klett & Balmer Verlag, Baar
6. Hromkovič J, Lacher R (2017) The Computer Science Way of Thinking in Human History and Consequences for the Design of Computer Science Curricula. In: Proc. of ISSEP 2017, LNCS 10696. Springer, pp 3–11
7. Ifrah G (1991) Universalgeschichte der Zahlen. Campus Verlag, Frankfurt New York
8. Pólya G (1949) Schule des Denkens. A. Francke, Bern
9. Stern E, Hardy I (2005) Anspruchsvolle Lernaufgaben. Handbuch Grundschulpädagogik und Grundschuldidaktik, S 396–402

# *Wie Mathematik und Informatik im Unterricht voneinander profitieren können*

***Teil 2: Variation der Problemstellung und Modularisierung***

**Urs Hauser · Dennis Komm**
**Giovanni Serafini**

Nachdem wir im ersten Teil des Beitrags die *Abstraktion* bzw. die *Variation der Darstellung* besprochen haben, fokussieren wir im zweiten Teil auf die beiden Heurismen *Variation der Problemstellung* und *Modularisierung*.

## Variation der Problemstellung

**Beispiel (Kombinatorik).** In diesem Beispiel legen wir den Fokus auf die *Analogiebildung* als übergeordneten Problemlöseprozess. Der Grundgedanke bei der Analogiebildung besteht darin, eine unbekannte Situation zu analysieren und mit bereits Bekanntem zu vergleichen, um aufgrund gemeinsamer Strukturen vertraute Methoden und Strategien auf die unbekannte Situation zu übertragen.

Ein schachbrettartiges Straßennetz einer Stadt wird als Graph (Quadratgitter) modelliert. Die Knoten sind die Kreuzungen und die Kanten die Straßenabschnitte. Wir suchen alle Routen von $A$ nach $B$, die keine Umwege beinhalten. Die Fragestellung können wir wie folgt umformulieren: *Wie viele Wege der Länge* 8 *von A nach B gibt es, die keine Umwege beinhalten?*

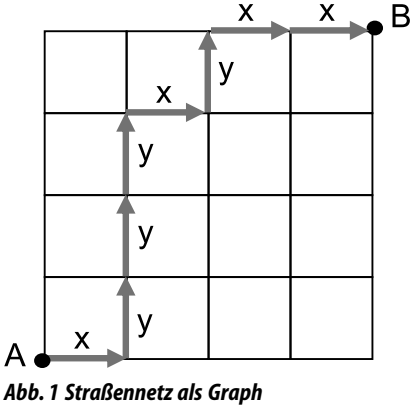



***Abb. 1 Straßennetz als Graph***

Wir legen das Gitter in ein Koordinatensystem und stellen einen Weg als Folge der Zeichen $x$ für einen Schritt in die $x$-Richtung und $y$ für einen Schritt in die $y$-Richtung dar. Der eingezeichnete Weg in Abb. 1 entspricht somit dem Wort

*xyyyxyxx*.

Das Problem der Berechnung aller Wege führen wir also auf das Problem der Berechnung aller Wörter der Länge 8 über dem Alphabet $\{x, y\}$ mit je vier $x$ und vier $y$ zurück. Die Analogie zur Kombinatorik besteht darin, solche Problemstellungen als *Permutationen mit Wiederholungen* zu betrachten. Die Wörter entsprechen den Permutationen der acht Buchstaben (dies sind 8!), wobei $x$ und $y$ aber nur je viermal auftreten können. Durch die Permutation der vier identischen Buchstaben $x$ bzw. $y$ ändert sich das Wort jeweils nicht. Deshalb müssen wir noch durch je 4! Wiederholungen dividieren und erhalten

$$\text{Anzahl Wege} = \frac{8!}{4! \cdot 4!} = 70.$$

https://doi.org/10.1007/s00287-019-01167-0
© Springer-Verlag Berlin Heidelberg 2019

Urs Hauser
ETH Zürich und PH Luzern,
Zürich, Schweiz
E-Mail: urs.hauser@inf.ethz.ch

Dennis Komm
ETH Zürich und PH Graubünden,
Zürich, Schweiz
E-Mail: dennis.komm@inf.ethz.ch

Giovanni Serafini
ETH Zürich,
Zürich, Schweiz
E-Mail: giovanni.serafini@inf.ethz.ch

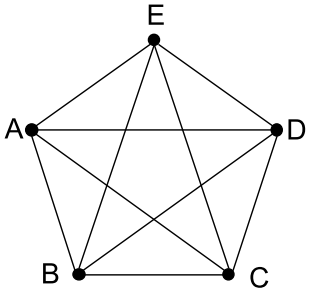



***Abb. 2 Geometrische Abstraktion des Händeschüttelns***

**Beispiel (Kombinatorik).** Es befinden sich $n = 5$ Personen auf einer Party, die sich alle zur Begrüßung die Hände schütteln. Wie oft werden die Hände geschüttelt?

Diese Fragestellung bietet sich zur Förderung der Problemlösefähigkeit an. Je nach Vorwissen kann das Problem unterschiedlich gelöst werden.

Man kann das Problem geometrisch abstrahieren und die Teilnehmerinnen und Teilnehmer als Punkte und das Händeschütteln als Verbindungsstrecken von je zwei Punkten darstellen. Übersichtshalber ordnen wir die Punkte in Form eines konvexen Fünfecks an (siehe Abb. 2).

Die Fragestellung reduziert sich nun auf die Frage, wie viele unterschiedliche Verbindungsstrecken die fünf Punkte haben können. Jeder Punkt besitzt vier Verbindungsstrecken, nämlich jeweils eine zu jedem anderen Punkt des Fünfecks. Zeichnet man die Verbindungsstrecken für jeden der Fünfeckpunkte ein, so erhält man $4 \cdot 5$ Strecken. Da wir jede Strecke doppelt gezählt haben, müssen wir die Anzahl noch halbieren und erhalten

$$\frac{4 \cdot 5}{2} = 10.$$

Man kann die obige Frage auch umformulieren. Eine Gerade ist durch zwei disjunkte Punkte eindeutig bestimmt. Wenn wir nach der Lösung suchen, müssen wir die Anzahl der Möglichkeiten bestimmen, aus fünf gegebenen Punkten je eine Zweiergruppe auszuwählen, was auf

$$\binom{5}{2} = \frac{5!}{3!2!} = 10$$

Arten möglich ist.

Man kann ein Händeschütteln auch als Wort der Länge 2 darstellen (zum Beispiel $AB$ oder $CA$). Alle möglichen Wörter kann man mit einem Baumdiagramm darstellen (siehe Abb. 3). Weil der erste Knoten den Grad 5 und jeder zweite Knoten den Grad 4 hat, erhalten wir für alle Möglichkeiten $5 \cdot 4$, was man durch Zählen der Blätter einfach nachprüfen kann. Da $AB$ und $BA$ der gleiche Händedruck ist, können wir alle symmetrischen Lösungen streichen, und das ist genau die Hälfte. Ein Algorithmus, der die erwähnten Fälle zählt, ist leicht zu programmieren.

## Modularisierung

Der modulare Entwurf ist eines der wichtigsten Konzepte in der Informatik. Bei der Entwicklung von komplexen Programmen lässt sich die Komplexität mithilfe einer Zerlegung in einzelne Module (Befehle), die individuell verifiziert werden können, verringern. Das Programm wird dadurch übersichtlicher und strukturierter.

Die Schülerinnen und Schüler können zum Beispiel erleben, wie komplexe Figuren aus einfacheren Bausteinen zusammengesetzt werden können. Sie lernen, solche Figuren zu analysieren und diese in einfache Bausteine zu zerlegen.

**Beispiel (Planimetrie).** Wir wollen den Flächeninhalt eines regulären $n$-Ecks mit Seitenlänge $s$ bestimmen.

Das Problem wird in Teilprobleme zerlegt. Da ein reguläres $n$-Eck aus $n$ kongruenten gleichschenkligen

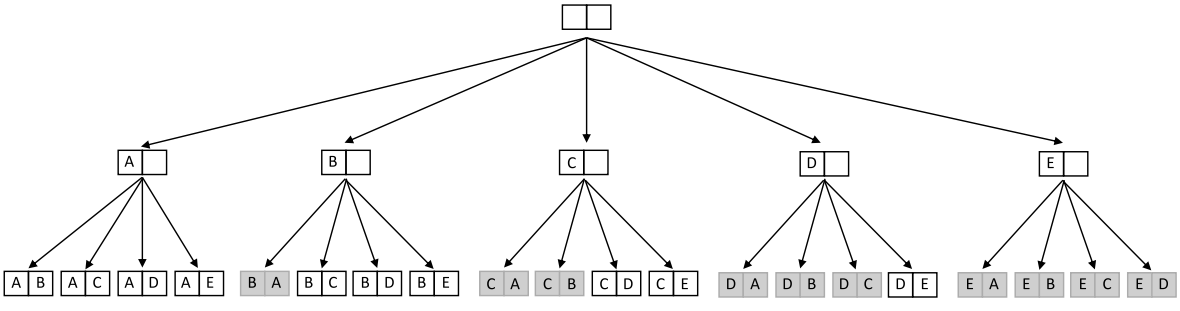



***Abb. 3 Anzahl Händeschütteln als Baum***

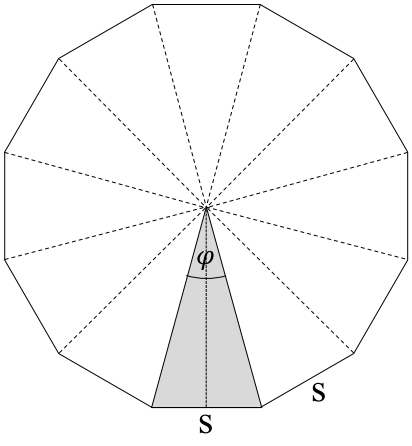



***Abb. 4 Bestimmungsdreieck als Teil eines regulären $n$-Ecks***

Dreiecken besteht, können wir die Problemstellung auf die Frage reduzieren, wie groß der Flächeninhalt des gleichschenkligen Bestimmungsdreiecks mit Basisseitenlänge $s$ und dem spitzen Winkel $\varphi = 360°/n$ ist (Abb. 4). Diese Teilaufgabe können Lernende mit trigonometrischen Kenntnissen (ab dem 10. Schuljahr) lösen. Das Erstellen eines Netzes von Dreiecken für Flächen im 3D-Raum (Triangulation) kommt bei der computergestützten geometrischen Modellierung (CAGD) zur Anwendung.

**Beispiel (Programmieren).** Das Programmieren bietet eine besonders nachhaltige Möglichkeit, den modularen Entwurf zu erlernen. Im folgenden Beispiel verwenden wir TigerJython und die Turtlegrafik [2, 3].

Die Schülerinnen und Schüler sollen ein Programm schreiben, mit dem ein Stern gemäß Abb. 5 gezeichnet wird.

Die Idee beim modularen Entwurf ist, dass die Lernenden die komplexe Figur des Sterns in einfachere Bausteine zerlegen. Dazu entwerfen sie eigene Befehle, mit denen die Figur schrittweise aufgebaut

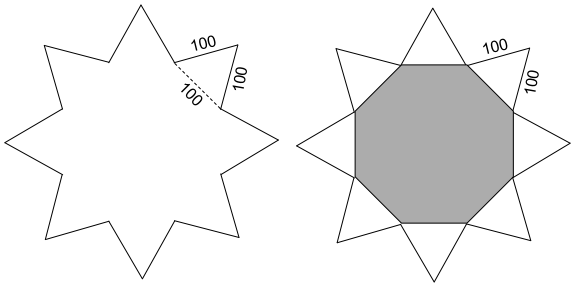



***Abb. 5 Stern mit acht Zacken***

und gezeichnet wird. Der Stern besteht aus acht Zacken, für die man einen eigenen Befehl `zacke()` erstellen kann.

Um den Stern zeichnen zu können, ist es wichtig, sich über die Position der Turtle vor der Ausführung des Befehls klar zu werden. Dieser führt jedes Mal exakt die gleichen Anweisungen aus. In der Ausrichtung der Turtle liegt die Herausforderung. Wir können uns die Bewegungsrichtung der Turtle im Inneren des Sterns auf einem regulären Achteck vorstellen, bei dem die Drehwinkel bekannt sind. Wir richten die Turtle nach jeder Zacke am Achteck aus. Schließlich können die Lernenden einen eigenen Befehl `stern()` kreieren, der es ihnen ermöglicht, mehrere Sterne zeichnen zu lassen (siehe Programm 1).

Programm 1. Stern mit acht Zacken

```
from gturtle import*
makeTurtle()

def zacke():
    left(60)
    forward(50)
    right(120)
    forward(50)
    left(15)

def stern():
    repeat 8:
        zacke()

stern()
```

Die Schülerinnen und Schüler haben also die Möglichkeit, die *Sprache der Turtle* mit eigenen Befehlen selbstgestaltend und kreativ zu erweitern. Diese Befehle können auch wieder innerhalb anderer Befehle verwendet werden. Sie lernen so intuitiv, was es bedeutet, strukturiert und modular zu programmieren.

In einem nächsten Schritt kann die Aufgabenstellung erweitert werden: *Zeichne einen Sternenhimmel, der aus Sternen wie in Abb. 6 in verschiedenen Größen besteht.*

Mit der Einführung von *Parametern* erreichen die Lernenden eine große Flexibilisierung, indem sie einen Befehl für verschiedene Ausgaben, abhängig vom Parameterwert, verwenden können. Die Lernenden setzen sich mit Parametern und Variablen sowohl im Informatik- wie auch im Mathematikunterricht

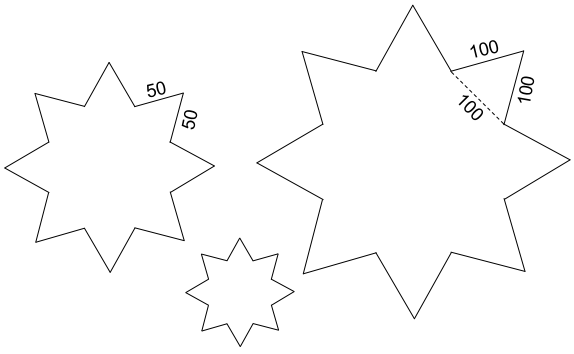



***Abb. 6 Ein „Sternenhimmel“***

intensiv auseinander. Die Thematisierung der Gemeinsamkeiten und Unterschiede kann in beiden Fächern zu einem tieferen Verständnis der Konzepte beitragen.

**Beispiel (Folgen und Reihen).** Aus Karten soll ein Kartenhaus gemäß Abb. 7 gebaut werden, wobei wir die Etagen von oben nach unten nummerieren. Wie viele Karten benötigt man für ein Kartenhaus mit $n$ Etagen?

Die Lernenden können zunächst eine Skizze erstellen und die Karten zählen, wobei zu beachten ist, dass es auf der untersten Etage keine Bodenkarten gibt.

Wir möchten aber ein allgemeines Verfahren finden. Bezeichnen wir die Anzahl der Karten eines Kartenhauses mit $n-1$ Etagen mit $a(n)$ und beginnen mit der ersten Etage $a(1)=2$. Für ein Kartenhaus mit zwei Etagen benötigen wir die Karten der ersten Etage plus jene der zweiten Etage, also $a(1)+2\cdot 3-1$ Karten. Für ein Kartenhaus mit vier Etagen benötigt man die Karten der ersten drei Etagen plus jene der vierten Etage, also $a(3)+4\cdot 3-1$. Dabei bestimmen wir die Anzahl der Karten in der vierten Etage, indem wir von den vier vollständigen Dreiecken ($4\cdot 3$) eine Karte subtrahieren (siehe Abb. 8). Zusammenfassend erhalten wir

$$
\begin{aligned}
a(2) &= a(1)+2\cdot 3-1\\
a(3) &= a(2)+3\cdot 3-1\\
a(4) &= a(3)+4\cdot 3-1\\
&\vdots\\
a(n) &= a(n-1)+n\cdot 3-1
\end{aligned}
$$

als Kartenanzahlen.

Ohne Kenntnis der vorangegangenen Kartenzahlen lässt sich die Summe bei diesem Ansatz nicht

**Kartenhäuser und die benötigten Karten** Tabelle 1

| Etagen | Anzahl Karten |
|---|---|
| 1 | 2 |
| 2 | 7 |
| 3 | 15 |
| 4 | 26 |

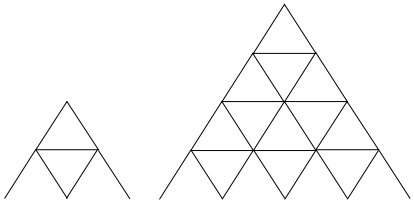


***Abb. 7 Kartenhaus mit $n=2$ und $n=4$ Etagen***

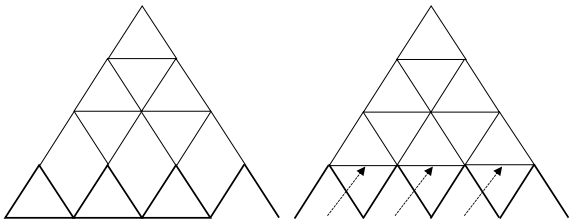


***Abb. 8 Berechnung der Kartenzahl der untersten Etage***

bilden. Das Problem wird also zurückgeführt auf die Berechnung der Anzahl der Karten eines Hauses mit $n-1$ Etagen etc. Das Konzept nennt man *Rekursion* und es stellt eine Möglichkeit dar, diese Aufgabe zu lösen.

Zur Rekursionsvorschrift $a(n-1)+n\cdot 3-1$ gehört zwingend die Angabe des Anfangswerts $a(1)=2$, damit die Aufgabe eindeutig lösbar ist. Wenn man die Rekursion im Programmierunterricht thematisiert, so lassen sich, neben der konkreten Programmierung, spannende Überlegungen zur Speicherorganisation machen. Programm 2 zeigt, wie die rekursive Berechnung in TigerJython umgesetzt werden kann.

Programm 2. Rekursive Funktion in TigerJyton

```
def a(n):
    if n>1:
        return a(n-1)+3*n-1
    else:
        return 2
```

Es gibt eine zweite Möglichkeit, die Folge der Kartenzahlen *explizit* zu bestimmen. Eine Etage entspricht

der Anzahl ihrer Dreiecke und wir zählen wiederum je drei Karten pro Dreieck. Für zwei Etagen sind dies $3 \cdot (2 + 1) - 2 = 7$ Karten, für vier Etagen $3 \cdot (4 + 3 + 2 + 1) - 4 = 26$ Karten.

Bei $n$ Etagen erhält man also

$$3 \cdot \frac{n \cdot (n+1)}{2} - n = \frac{1}{2} \cdot n \cdot (3n+1).$$

**Beispiel (Bäume und Rekursion).** Ein Beispiel zur Anwendung einer rekursiven Funktion ist das Erzeugen von Bäumen.

Gesucht ist ein Programm zur Erzeugung $n$-stelliger Zahlen mit den Ziffern 0 und 1 (siehe Abb. 9 für $n = 3$).

Die Verzweigung in Baumdiagrammen entspricht der Reduktion einer komplexen Aufgabe zu einfacheren Teilaufgaben. Die Auflistung aller Wörter der Länge $n$ reduzieren wir auf die Auflistung der Wörter der Länge $n - 1$. Die Arbeit von rekursiven Programmen wird gerade mit Bäumen oftmals sehr anschaulich dargestellt. Die am häufigsten verwendete Struktur zur systematischen Speicherung und Verwaltung von Daten entspricht dem Baum [1]. Wie lässt sich ein Baum programmieren? In Programm 3 wird eine Liste mithilfe des rekursiven Aufrufs von `generiere(tiefe)` erzeugt [4].

Programm 3. Generieren aller 3-stelligen Binärzahlen

```
n=3
zahl=[0]*n
def generiere(tiefe):
    for i in range(2):
        zahl[tiefe]=i
        if tiefe==n-1:
            print zahl
        else:
            generiere(tiefe+1)
generiere(0)
```

Diese Aufgabe eignet sich, um das Wachstum der Lösungen mit der Stellenzahl zu thematisieren und einen Link zum Thema *Wachstumsprozesse* in der Mathematik herzustellen. Die Anzahl aller Lösungen wächst mit der Zahl $n$ exponentiell und ist ab einer gewissen Größe nicht mehr berechenbar (für $n = 80$ sind es bereits $2^{80}$, also mehr als $10^{24}$ Lösungen).

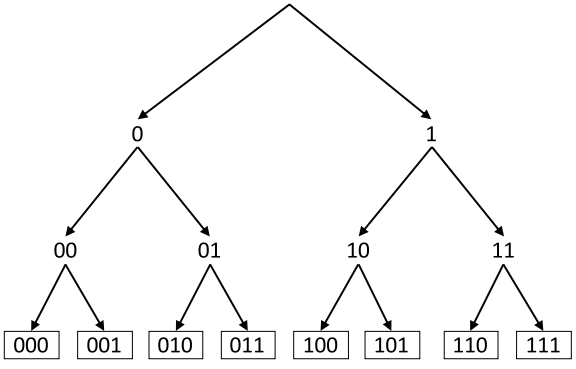



***Abb. 9 Baum zur Darstellung aller 3-stelligen Binärzahlen***

Als Anwendung der Generierung von Bäumen betrachten wir die folgende Aufgabe: *Wie viele Möglichkeiten gibt es, $k = 8$ (optisch nicht unterscheidbare) Bonbons auf $n = 4$ Kinder zu verteilen? Dabei ist es auch möglich, dass ein Kind oder mehrere Kinder kein Bonbon erhalten.* Die Lösungen sind Folgen von 4 Zahlen, die in der Summe 8 ergeben. Es geht also lediglich darum, wie viele Bonbons die Kinder erhalten. Dies ist das Gleiche wie die Anzahl der Möglichkeiten zu zählen, 8 in 4 Summanden zu zerlegen, wobei die Reihenfolge der Summanden eine Rolle spielt ($4 + 3 + 0 + 1$ ist eine andere Lösung als $1 + 3 + 0 + 4$).

Möchte man alle Lösungen mittels eines Suchbaums darstellen, hat man folgende Möglichkeiten: Weist man jedes Bonbon einem der vier Kinder zu, so entstünde ein 8-stufiger Baum, in dem jeder Knoten den Ausgangsgrad 4 hätte. Insgesamt hätte der Baum somit $4^8$ Blätter. Der Baum wächst schnell über alle Grenzen an (bei 50 Kindern und 200 Bonbons hätte der Baum mehr als $10^{339}$ Blätter).

Wir verfolgen hier eine andere Strategie und entscheiden für jedes Kind, wie viele Bonbons es erhält. Jeder Knoten des 4-stufigen Baums hat einen maximalen Ausgangsgrad von 8. Eigentlich verkleinert sich dieser bei jeder Stufe, denn die Summe aller Bonbons ist ja auf 8 begrenzt.

Wie stellen wir die Lösungen mathematisch dar?

Wir tun dies hier als 4-Tupel $(k_1, k_2, k_3, k_4)$, wobei $k_i$ der Anzahl von Bonbons entspricht, welche das $i$-te Kind erhält. Eine Möglichkeit ist $(2, 3, 2, 1)$, was wir bildlich mit kleinen Kreisen und unterteilenden Wänden darstellen (siehe Abb. 10 und Abb. 11).

Nun stellen wir $(k_1, k_2, k_3, k_4)$ wiederum als Folge von Nullen und Einsen (als Trennstriche) dar, wodurch wir beispielsweise

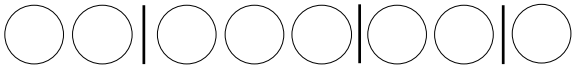


***Abb. 10 Repräsentation der Folge (2,3,2,1)***

***Abb. 11 Repräsentation der Folge (2,3,0,3)***

00100010010

erhalten, und suchen nach allen unterschiedlichen Möglichkeiten, die Zeichenfolge anordnen zu können. Für $n = 4$ Kinder und $k = 8$ Bonbons brauchen wir drei Einsen (allgemein $n - 1$) und acht Nullen. Die Anzahl der Zeichenfolgen der Länge 11 mit 8 Nullen beträgt

$$\binom{4-1+8}{8} = \binom{11}{8} = 330$$

und allgemein

$$\binom{n+k-1}{k}.$$

Ein weiteres Beispiel ist die Berechnung der Anzahl aller (maximal) 6-stelligen Zahlen (im Dezimalsystem) mit der einfachen Quersumme 9 (siehe auch Pierhöfer [4]). Hier können wir die $n = 6$ Stellen der Zahl mittels fünf Einsen trennen.

00100010001011

steht dann für die Zahl 233100 oder

11111000000000

für die Zahl 9 und wir erhalten

$$\binom{6+9-1}{9} = \binom{14}{9} = 2002.$$

Die Umsetzung in einem TigerJython-Programm erfolgt wiederum rekursiv nach dem gleichen Prinzip wie im ersten Teilbeispiel von „Bäume und Rekursion“.

Zusammenfassend können wir sagen, dass der Informatik- und Mathematikunterricht die Problemlösefähigkeit und das abstrakte Denken von Lernenden in sehr hohem Maß fördern. Man kann

Programm 4. Generieren aller 6-stelligen Zahlen mit einfacher Quersumme 9

```
n=6
zahl=[0]*n
anz=0
quers=9
d=0

def generiere(tiefe,q):
    global anz
    if tiefe==n-1:
        d=quers-q
        zahl[tiefe]=quers-q
        print zahl
        anz+=1
    else:
        d=quers-q
        for i in range(d+1):
            zahl[tiefe]=i
            generiere(tiefe+1,q+i)

generiere(0,0)
print anz
```

ferner feststellen, dass das neue Fach Informatik einen besonders großen Mehrwert im Fächerkanon darstellt.

Einerseits führt die Informatik den Lernenden die Wichtigkeit der mathematischen Methoden zur Modellierung und Verifizierung vor Augen und stärkt somit die Position der Mathematik. Zusätzlich schafft es der Informatikunterricht, neue Akzente zu setzen und den Lernenden das Lösen von Problemen als kreativen Prozess erfahrbar zu machen und ihnen die Möglichkeit zu geben, die technische Welt zu entdecken, zu verstehen und aktiv mitzugestalten.

## Literatur


1. Hromkovič J (2018) Einfach Informatik – Strategien (Begleitband). Klett & Balmer Verlag, Baar
2. Hromkovič J, Kohn T (2018) Einfach Informatik – Programmieren. Klett & Balmer Verlag, Baar
3. Hromkovič J, Kohn T, Komm D, Serafini G (2016) Combining the Power of Python with the Simplicity of Logo for a Sustainable Computer Science Education. In: Proc. of ISSEP 2016. Springer, pp 155–166
4. Pierhöfer H (2009) Bäume und Backtracking, Leitprogramm, ETH

# *IoT-basiertes Prozessmanagement*

***Mobile Benutzerführung in der digitalen Fabrik***

**Stefan Schönig · Stefan Jablonski**
**Andreas Ermer**

**Vorspann**

Der folgende Beitrag beschreibt einen Ansatz, der die „BigData"-Welt der Erfassung und Analyse von IoT- bzw. Sensordaten mit der Technologie des Prozessmanagements verbindet und die Aktivierung von Bedieneraufgaben auf Basis benutzerdefinierter Bedingungen auf IoT-Objekten ermöglicht.

## Einleitung und Motivation

Die durch die „Industrie 4.0"-Initiative geförderte Verschmelzung von Produktion und Informationstechnologie führt im zukünftigem Produktionsumfeld zu einer Allgegenwärtigkeit von rechnergestützten Informationsverarbeitungssystemen, die unter dem Begriff „Ubiquitous Computing" zusammengefasst werden. In der Produktion wird darunter auch die Vernetzung von Produktionsmaschinen mit übergeordneten Informationssystemen verstanden, sodass sämtliche produktionsbeteiligten Objekte über eine mobile Infrastruktur untereinander kommunizieren und interagieren können [9]. Prozessrelevante Informationen lassen sich so in Echtzeit erfassen, wodurch Prozesse entlang der gesamten Wertschöpfungskette effizient gestaltet werden können [3].

Durch die Möglichkeiten, die mit der digitalen Transformation einhergehen, lässt sich die Produktionsplanung und -steuerung unterstützen. Dabei ermöglichen Prozessführungs-, Prozessüberwachungs- sowie Analysedaten auf Basis von Internet-of-Things (IoT) Anwendungen eine umfassende Sicht auf Abläufe in der Produktion. Als ein mögliches Szenario wird ein Produktionsprozess betrachtet, der Material in einer Maschine bearbeitet. Für die Einhaltung der Produktqualität, welche durch Sensoren überwacht wird, sind manuelle Eingriffe und die ständige Bereitschaft von Bedienern erforderlich. Basierend auf Sensordaten und seiner Erfahrung muss der Bediener Entscheidungen treffen, um die Produktqualität durch seinen Eingriff sicher zu stellen. Ein solches Szenario wird besser beherrschbar, wenn digitale Produktions- und Maschinendaten zeitnah von menschlichen Bedienern zugegriffen und verarbeitet werden können. In der Kombination aus IoT-Anwendungen und Prozess- bzw. Produktionssteuerung ergeben sich sowohl Reaktionszeit- und Kosteneinsparungen für Unternehmen als auch Effizienz- und Qualitätssteigerungen, indem die gezielte Vergabe und Abwicklung von Arbeitsaufträgen durchgeführt und überwacht werden kann.

Der vorgestellte Ansatz (vgl. Abb. 1) verbindet die „BigData"-Welt der Erfassung und Analyse von IoT- bzw. Sensordaten [5] mit der Technologie des Prozessmanagements [1]. Dabei werden individuelle Arbeitsabläufe von Anlagenbedienern, d. h. Workflows bzw. Prozesse, in einer übersichtlichen Grafik mithilfe der Prozessstandardnotation BPMN (Business Process Modeling and Notation) [6] modelliert. Ein Prozessmodell enthält also Aufgaben, sog. Tasks, die bestimmten Personen bzw. Gruppen von Perso-

https://doi.org/10.1007/s00287-019-01140-x
© Springer-Verlag Berlin Heidelberg 2019

Stefan Schönig · Stefan Jablonski
Lehrstuhl für Datenbanken und Informationssysteme,
Universität Bayreuth,
Universitätsstraße 30, 95447 Bayreuth
E-Mail: stefan.schoenig@uni-bayreuth.de

Andreas Ermer
Maxsyma GmbH & Co. KG,
Neustädter Straße 9, 92685 Floß

## Zusammenfassung

Durch die Möglichkeiten, die mit der digitalen Transformation einhergehen, lässt sich die Produktionsplanung und -steuerung unterstützen. Dabei ermöglichen Prozessführungs-, Prozessüberwachungs- sowie Analysedaten auf Basis von Internet-of-Things(IoT)-Anwendungen eine umfassende Sicht auf Abläufe in der Produktion. In diesem Artikel wird ein Ansatz vorgestellt, der die „BigData"-Welt der Erfassung und Analyse von IoT- bzw. Sensordaten mit der Technologie des Prozessmanagements verbindet. Durch die Anbindung von Sensor- und Anlagendaten wird die selektive Aktivierung von Aufgaben auf Basis benutzerdefinierter Bedingungen ermöglicht. Ziel dieses Beitrags ist, den Nachweis zu erbringen, dass mit bestehender Technologie eine neue Qualität in die Steuerung und Überwachung von Produktionsprozessen gebracht werden kann. Der Beitrag demonstriert den Aufbau eines effektiven und effizienten Gesamtsystems im Produktionsbereich und wurde ausgiebig in der Praxis evaluiert.


nen zugeordnet sind. Durch die Anbindung jeglicher Art von Sensor- und Anlagendaten wird die selektive Aktivierung von Tasks auf Basis benutzerdefinierter Bedingungen ermöglicht. Der Ansatz ermöglicht somit die Verknüpfung von Sensoren und Steuerungen mit menschlichen Aufgaben. Modellierte Prozesse können über eine mobile Administrationsoberfläche gestartet und verwaltet werden. Der Status laufender Prozesse kann stets überwacht und nachvollzogen werden. Bereits abgeschlossene Prozesse können nachträglich analysiert und Flaschenhälse sichtbar gemacht werden. Bediener werden über aktuelle Aufgaben auf mobilen Endgeräten wie z. B. Smartwatches benachrichtigt. Hierdurch wird der Bediener bei seinen Tätigkeiten unterstützt, ohne bei der Durchführung seiner Kernaufgaben durch zusätzlichen Aufwand abgelenkt zu werden.

Ziel dieses Beitrags ist, den Nachweis zu erbringen, dass mit bestehender Technologie und mit vertretbarem Aufwand eine neue Qualität in die Steuerung und Überwachung von Produktionsprozessen gebracht werden kann. Hierbei wird auf Standardtechnologien aus den Bereichen Prozessmodellierung und -ausführung, mobile Geräte und Sensortechnik gesetzt. Der Beitrag demonstriert den Aufbau eines effektiven und effizienten Gesamtsystems im Produktionsbereich und soll aufgrund seiner kostengünstigen Umsetzung die Anwendung der oben genannten Technologien sowohl in kleinen, mittleren als auch großen Unternehmen motivieren.

### Gemeinsame Sprache und Cockpit

Eine Verknüpfung von Industrie 4.0-Konzepten und Prozessmanagement ermöglicht es, Mitarbeiter bei ihren operationalen Aufgaben zu unterstützen. Es wird dafür gesorgt, dass Daten in Echtzeit zusammenlaufen und in eine interpretierbare Darstellung übersetzt werden. Diese abstrahiert zum einen von den vielen proprietären Darstellungen von Maschinen- und Sensordaten und ist daher von der gesamten Belegschaft eines Produktions-

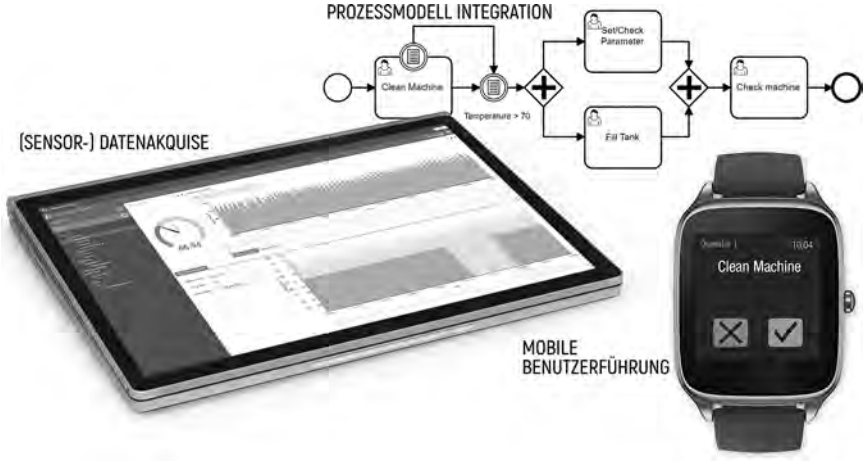



*Abb. 1 Übersicht des Ansatzes*

**Abstract**
The connection between production and information technology leads to a ubiquity of digitally supported information processing systems. Such a production system represents a typical Internet of Things (IoT) system where electronic hardware, e. g., sensors and actuators are tightly integrated with human operators. To support processes at an operational level, a Business Process Management system (BPMS) can be used. In the following article, we introduce an approach for an IoT-aware BPMS that exploits IoT data for production process support by providing IoT data in a process-aware way, considering IoT data for interaction in a predefined process model, and providing wearable user interfaces with context specific IoT data provision. The approach has been evaluated extensively in production industry.

betriebs leicht zu verstehen und zu interpretieren. Die Darstellung der aktuellen Produktionsdaten erfolgt in unserem Anwendungsfall auf Smartwatches. Prozessbeteiligte erkennen auf dem Display ihrer Smartwatch, welche Aufgaben anstehen, wie beispielsweise Wartungsarbeiten, Materialnachlieferung oder Behebung einer Maschinenstörung. Das Prozessmanagementsystem löst Aufgaben aus und teilt sie dem geeigneten Mitarbeiter zu, unabhängig davon, wo sich Maschinenbediener, Wartungs- oder Servicemitarbeiter gerade – innerhalb der Produktionshalle – aufhalten. So wissen sie stets über den Status aller Anlagen und ihrer Aufgaben Bescheid. Dies verkürzt die Reaktionszeiten der Mitarbeiter und reduziert Maschinenstillstände aufgrund verspäteter Reaktionen des Bedienerpersonals.

## Schritt für Schritt zur IoT-basierten Ablauforganisation

Im ersten Schritt werden Produktionsprozesse, Arbeitsabläufe von beteiligten Bedienern sowie notwendige Daten und Informationen aus Maschinen, Steuerungen und Sensoren in einer standardisierten, grafischen Spezifikationssprache für Abläufe (z. B. BPMN) [6] modelliert und digitalisiert. Auf Basis einer Prozessautomatisierungskomponente können die resultierenden Prozessmodelle schließlich ausgeführt werden und konkrete Aufgaben an Mitarbeiter auf Basis aktueller Anlagen- und Sensordaten verteilt werden. Die mobile Zuteilung von Aufgaben an Mitarbeiter setzt das Vorhandensein einer mobilen verteilten Netzwerkinfrastruktur voraus, d. h. beispielsweise in Form einer drahtlosen Client-Server-Architektur. Auf diese Weise können gerade notwendige Aufgaben vom Planungssystem (z. B. via WLAN) an diverse mobile Geräte (z. B. Smartwatches) gesendet werden, welche jeweils bestimmten Anlagenbedienern zugewiesen sind.

Überschreitet beispielsweise der aktuelle Wert eines Temperatursensors eine bestimmte Warngrenze, wird – wie vorher im Prozessmodell definiert sein muss – eine bestimmte Arbeitsaufgabe aktiviert und an die mobilen Geräte von Mitarbeitern einer bestimmten, vordefinierten Bedienergruppe gesendet. Das System sorgt automatisch dafür, dass Mitarbeiter mit der passenden Qualifikation den Arbeitsauftrag erhalten. Damit die unterschiedlichen Aufgaben zielgerichtet zugewiesen werden können, muss ein Unternehmen Mitarbeiter bestimmten Gruppen auf Basis von Qualifikationen oder Schichtgruppen zuteilen.

### Einbeziehung der Bediener

Mitarbeiter werden auf diese Weise mittels haptischer, akustischer oder visueller Signale auf dem dedizierten mobilen Gerät auf neue Aufgaben hingewiesen, die sie annehmen oder ablehnen können. Ist eine konkrete Aufgabe von einem Mitarbeiter angenommen, wird diese alle anderen Mitarbeitern der Gruppe entzogen. Falls ein Mitarbeiter eine angenommene Aufgabe nicht mehr erfüllen kann, kann er diese auch eigenständig wieder für alle zuvor festgelegten Mitarbeitergruppen freigeben und entsprechend selbst wieder neue Aufgaben empfangen. Auf diese Weise lassen sich alle Arbeiter mit mobilen Geräten gruppenspezifisch kontaktieren, um Aufgaben und Arbeitsanweisungen zu verteilen.

Ziel der Lösung ist dabei ein hoher Grad an Benutzerorientierung, d. h. das System und Prozessmodelle lassen sich von Mitarbeitern selbst weiterentwickeln. Viele frei erhältliche BPMN-Modellierungsplattformen (z. B. BPMN-IO) erlauben es Facharbeitern, selbst Modellierung und Anpassung von Prozessen vorzunehmen. Dies ermöglicht es, Maschinenanbindung und Ablaufsteuerung gemeinsam mit Mitarbeitern umzusetzen und stetig zu verbessern.

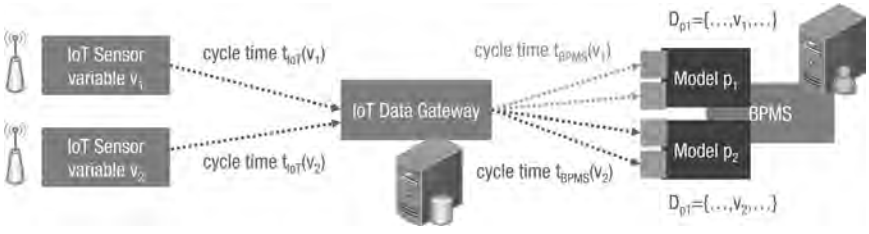



***Abb. 2 Anbindung von IoT-Daten an Prozessmanagementsystem***

## Integrierte Architektur für IoT-basiertes Prozessmanagement

In etablierten Ansätzen und Systemen wird die integrierte Nutzung von IoT-Technologie für die systemgestützte Prozessausführung insbesondere durch das Fehlen einer gemeinsamen Architektur eingeschränkt, welche die Kommunikation zwischen beiden Bereichen standardisiert und steuert. Zur Überbrückung dieser konzeptuellen Lücken wird ein Ansatz für IoT-Anwendungen zur Unterstützung mobiler Ablauforganisationen in der Produktion vorschlagen, der IoT-Technologien für das Prozessmanagement nutzt [7]. Dazu wird eine integrierte Architektur für ein IoT-fähiges Prozessmanagement vorgestellt, welche die folgenden Teilaspekte umfasst:

- *Anbindung von IoT-Daten an laufende Prozesse bzw. Prozessmodelle* – Zu erfassende und anzubindende Daten aus Sensoren und Steuerungen der Produktionsanlage müssen konfiguriert werden. Relevante Daten müssen in entsprechenden Prozessmodellen referenziert werden.
- *Aktivierung von Aufgaben auf Basis aktueller IoT-Daten* – Mithilfe einer Prozess-Engine werden Prozessmodelle ausgeführt und Daten in Echtzeit ausgewertet. Auf Basis frei definierbarer Bedingungen auf diesen Daten in Prozessmodellen werden menschliche Aufgaben aktiviert.
- *Mobile Bereitstellung von Aufgaben an Bediener* – Diese Aufgaben werden unmittelbar an verantwortliches Bedienpersonal gesendet, unabhängig von deren Position an der Anlage.

### Anbindung von IoT-Daten an Prozessmanagement-Systeme

Im ersten Schritt werden aktuelle Daten von IoT-Objekten an das Prozessmanagementsystem gesendet. Hierbei ist zu berücksichtigen, dass bestimmte IoT-Daten nicht an alle laufenden Prozesse (Prozessinstanzen) gesendet werden, sondern nur an diejenigen, welche auch eine bestimmte IoT-Variable referenzieren. Aus diesem Grund muss eine Abbildung zwischen IoT-Variablen und verwendeten Prozessmodellen etabliert werden. Betrachten wir hierzu Abb. 2. Eine Menge an Prozessmodellen $P$ wird durch das Prozessmanagementsystem ausgeführt, wobei jedes Modell $p_i$ eine Menge an IoT-Variablen $D_{p_i}$ referenziert. Jede Variable $v \in D_p$ besitzt hierbei einen eindeutigen Identifikator. Die grundlegende Annahme ist hierbei, dass jede benötigte Variable $v_i$ dabei über den gleichen Identifikator im entsprechenden Prozessmodell referenziert werden kann. Nachdem eine derartige semantische Abbildung zwischen Daten und Modell definiert worden ist, werden jeweils aktuellste Werte der referenzierten Variablen an das Prozessmanagementsystem gesendet. Die technische Umsetzung der Anbindung von Daten an laufende Prozesse wird dabei von der IoT-Infrastruktur übernommen. Zu beachten ist hierbei, dass die Abstraktionslücke zwischen teilweise hochfrequenten IoT- und Sensordaten und der Welt der menschlichen Aufgaben eines Geschäftsprozesses überbrückt werden muss. In Abb. 2 ist dies schematisch durch die Angabe von verschiedenen Sendezykluszeiten gelöst: Während IoT-Daten potenziell mehrfach pro Sekunde abgetastet und gespeichert werden ($cycle\,time_{IoT}$), werden aktuelle Werte nur jede Sekunde oder nach dem Auslösen bestimmter Bedingungen ($cycle\,time_{BPMS}$) (Stichwort *Complex Event Processing* [2]) an die Prozess-Engine gesendet.

### Referenzieren von IoT-Variablen in Prozessmodellen

Basierend auf der etalierten Echtzeitverbindung von IoT-Objekten und eines Prozessmanagementsystems können IoT-Variablen auf verschiedene Weisen in Prozessmodellen referenziert werden. Hierbei nehmen wir an, dass sämtliche Prozessmodelle konform zur Modellierungssprache BPMN abgebildet sind. In einem Prozessmodell wird spezi-

fiziert, welche (menschlichen) Aktivitäten in welcher Reihenfolge ausgeführt werden. Damit das Prozessmanagementsystem Aktivitäten auf Basis der aktuellen Werte von IoT-Variablen aktivieren oder beenden kann, muss das Modell bestimmte Informationen beinhalten. Dieser Schritt erfordert es daher, gegebene Prozessmodelle von Arbeitsabläufen zu erweitern. Dabei können IoT-Variablen lediglich lesend referenziert werden (*passiv, informierend*) als auch aktiv geschrieben werden (*aktiv, interagierend*).

**Passive Referenzierung.** Die Sprache BPMN enthält verschiedene Konstrukte, die eine Referenzieren von IoT-Variablen in Prozessmodellen erlauben. Die folgenden Sprachelemente eignen sich gut zur exemplarischen Erläuterung der Integrationskonzepte:

- IoT-basierte Conditional Events: Trigger Events können angewendet werden, um Aufgaben oder ganze Prozesse zu aktivieren, wenn eine bestimmte Bedingung auf den aktuellen Werten von IoT-Variablen erfüllt ist. Beim sog. *Intermediate Conditional Catch Event* wird die Ausführung an einem bestimmten Punkt angehalten, bis eine bestimmte Bedingung erfüllt ist. Anschließend wird der Prozess mit der nächsten Aufgabe fortgesetzt. *Boundary Events* können dazu verwendet werden, Aufgaben abzubrechen bzw. zu entziehen, wenn referenzierte IoT-Variablen eine hinterlegte Bedingung erfüllen. Ein *Conditional Start Event* startet gesamte Prozesse, z. B. das Auffüllen eines Teilelagers, auf Basis der Werte von verknüpften IoT-Werten.
- IoT-basierte Entscheidungspunkte: In BPMN werden sog. datenbasierte Gateways dazu verwendet, um Entscheidungen zu modellieren. Hier können IoT-Variablen eingesetzt werden, um den weiteren Verlauf im Prozess zu bestimmen.
- *IoT-basierte Schleifen*: Auch zur Abbildung von sich wiederholenden Aktionen können IoT-Variablen als ereignisbasierte Trigger eingesetzt werden.

**Aktive Interaktion.** Zur vollständigen bidirektionalen Integration von IoT-Objekten und Prozessmanagement ist es notwendig, auch eine aktive Interaktion, d. h. einen schreibenden Zugriff auf IoT-Objekte zu ermöglichen. Auch hierfür bietet der BPMN-Standard einige Konzepte, die zur Kommunikation verwendet werden können. *Service Tasks* werden verwendet, um (Web-)Services oder allgemein Programmcodes aufzurufen und sind daher die Lösung zur Abbildung von automatisierbaren Aufgaben. Existierende Prozessmanagementsysteme enthalten häufig bereits implementierte Konnektoren zu etablierten Web-Protokollen wie HTTP. In vielen Fällen bieten IoT-Objekte bereits HTTP-Schnittstellen an, wodurch eine direkte Schnittstelle aus dem Prozessmanagementsystem heraus möglich ist. Aber auch in Fällen, in denen IoT-Objekte nur über Fremdprotokolle angesteuert werden können, ist eine Integration über Kopplungssysteme, die meist frei erhältlich sind, möglich.

### Analyse von Prozessen

Als weiteren Schritt können auftragsrelevante Daten und bereits ausgeführte Prozesse in Form von digitalen Ereignisprotokollen weiterverarbeitet und analysiert werden (Stichwort: *Process Mining* [11]). Zu den Analyseergebnissen zählen beispielsweise durchschnittliche Bearbeitungszeiten der jeweiligen Arbeitsaufträge und -abfolgen, von der Annahme des Auftrags bis zur Rückmeldung seiner Beendigung. Daraus bekommt die Prozessplanung und -steuerung genauere Informationen und damit eine bessere Transparenz hinsichtlich firmeninterner Produktionsprozesse und -aufgaben. Hierzu werden alle bereits abgeschlossenen und auch laufenden Prozesse angezeigt und deren minimale, maximale und durchschnittliche Laufzeiten angegeben. In jeden Prozess kann man „einzoomen" und minimale, maximale und durchschnittliche Laufzeiten der untergeordneten Aktivitäten einsehen. Außerdem können umfangreiche Anfragen an das Ereignisprotokoll gestellt und prozessspezifische KPIs berechnet werden.

## Technische Umsetzung der bidirektionalen Kommunikation

Der beschriebene Ansatz wird auf Basis einer Vierschichtarchitektur umgesetzt, die in Abb. 3 dargestellt ist [8]. Diese Architektur besteht aus den folgenden Ebenen: *(i)* Datenquellen, d. h. beliebige Sensoren und Anlagensteuerungen; *(ii)* Daten- und Kommunikationsinfrastruktur; *(iii)* Prozessmanagementsystem; und *(iv)* Datensenken, d. h. mobile Geräte von Bedienern bzw. wiederum IoT-Objekten wie z. B. Robotern oder anderen Aktoren. Die Kommunikationen über Ebenengrenzen hinweg erfolgt auf Basis von

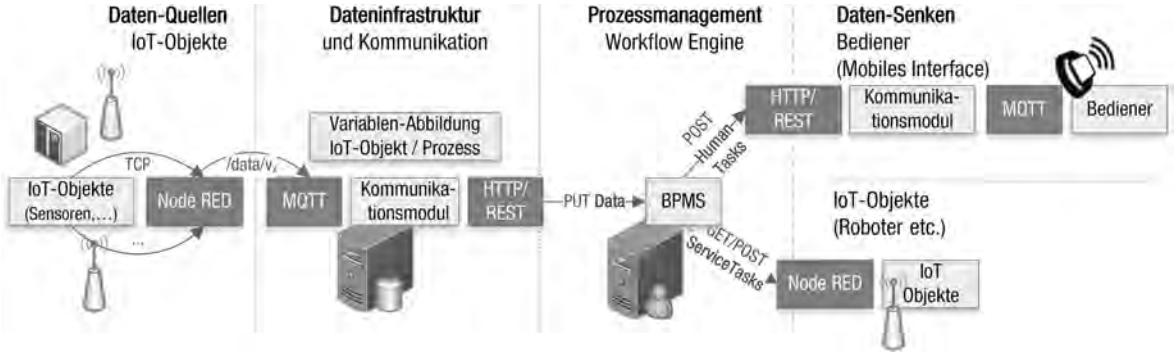



***Abb. 3 Technische Gesamtarchitektur***

Standardkommunikationsprotokollen wie dem Message-Queue-Telemetry-Transport(MQTT)-Protokoll, dem HTTP-Protokoll oder dem Constrained Application Protocol (CoAP). Zur generischen Anbindung von beliebigen IoT-Objekten wird die frei zugängliche Node-RED-Plattform von IBM verwendet, welche als Vermittlungs- und Kopplungsinstanz die Kommunikation zwischen zahlreichen IoT-Protokollen und Datenquellen wie TCP-kompatiblen Sensoren und dem ansatzspezifischen Kommunikationsprotokoll MQTT übernimmt. Wie in Abb. 3 dargestellt, werden auf diese Weise beliebige Datenvariablen $v_x$ aus Sensoren auf jeweils ein MQTT-Topic ($/data/v_x$) abgebildet, welche in der Dateninfrastruktur verarbeitet werden. Dies ermöglicht eine vollständige Unabhängigkeit des Systems von der anzubindenden IoT-Infrastruktur. Die aufgezeichneten Daten werden in einer NoSQL-Datenbank (z. B. Apache Cassandra) gespeichert. Anhand der definierten Abbildung von IoT-Variablen zu Prozessvariablen, d. h. Variablen, die in Prozessmodellen referenziert werden, werden aktuelle Datenwerte zyklisch an ein angebundenes Prozessmanagementsystem gesendet. Das IoT/Prozesskommunikationsmodul sendet dabei aktuelle Daten nur an diejenigen Prozessinstanzen, deren Prozessmodelle eine bestimmte Variable referenzieren. Auf Basis der aktuell vorliegenden IoT-Daten berechnet die Prozess-Engine neue Aufgaben, welche schließlich an Bediener verteilt werden. Für die Ausführung der BPMN-Prozessmodelle wird das frei verfügbare Prozessmanagementsystem Camunda eingesetzt [4]. Camunda bietet eine HTTP/restbasierte Kommunikationsschnittstelle, die verwendet wird, um *(i)* aktuelle Daten an das System zu senden (*PUT*) und um *(ii)* aktivierte Tasks an Bediener und andere Objekte bzw. Systeme zu verteilen (*GET* und *POST*). Die mobile Benutzerschnittstelle wird als androidbasierte Smartwatchapplikation umgesetzt, welche es Bedienern erlaubt, Aufgaben zu starten oder zu beenden und neue Prozesse zu initiieren. Die Zuordnung einer Smartwatch zu einem bestimmten Bediener erfolgt dabei über einen unveränderlichen Geräteidentifikator, der in der Administrationsoberfläche bei dem jeweiligen Benutzer gespeichert wird. Ein zentrales Kommunikationsmodul übernimmt schließlich die Abbildung der HTTP-Befehle auf MQTT-Topics, welche von der Smartwatchapplikation verarbeitet werden können. Im Fall von aktiven Interaktionen mit dem IoT-Umfeld werden BPMN-Service-Tasks eingesetzt, welche mittels eines von Camunda implementieren HTTP-Konnektors entweder direkt über das Protokoll mit IoT-Objekten kommunizieren und Variableninhalte übergeben können oder indirekt wiederum über die Node-RED-Plattform. Muss beispielsweise nach Abschluss einer manuellen Aufgabe ein Roboter zu einer bestimmten Position gerufen werden, so wird diese als Parameter eines HTTP-Aufrufs vom Prozesssystem an die Node-RED-Plattform übertragen. Hier kann die Position aus dem Aufruf, d. h. als URL-Parameter oder dem POST-Body, ausgelesen werden und der entsprechende Befehl an den Roboter in einem unterstützen Protokoll abgesetzt werden. Auf diese Weise wird von der vorgeschlagenen Lösung die vollständige bidirektionale Kommunikation zwischen IoT-Objekten (z. B. Sensoren, Smartwatches und Roboter) und Prozessmanagementsystemen realisiert.

## Anwendung in Industrie

Die vorgestellten Konzepte wurden unter anderem in Produktionsanlagen für Wellpappe umgesetzt. Das Projekt „Corrugator Wearables“

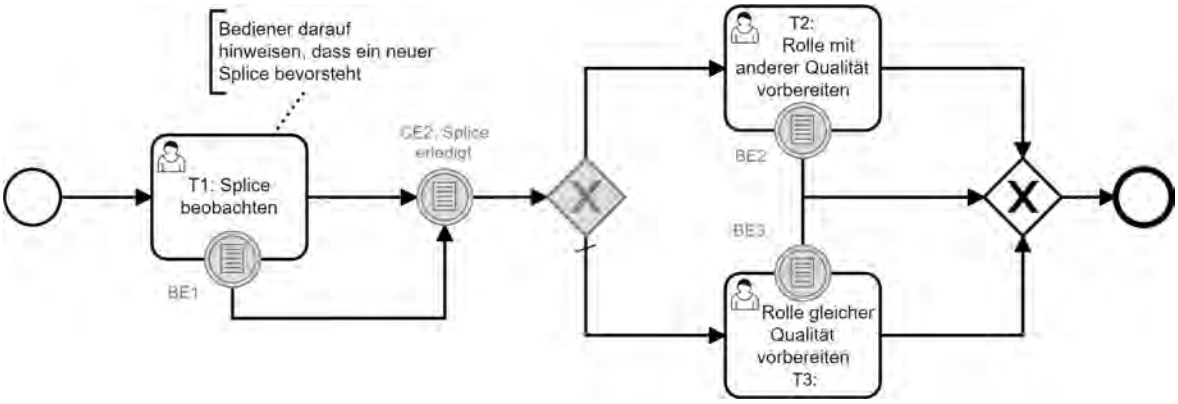



***Abb. 4 Exemplarischer Auszug aus implementiertem Produktionsprozess***

wurde im Auftrag der Firma BHS Corrugated GmbH (www.bhs-world.com) von den Autoren und der Firma Maxsyma GmbH & Co. KG (www.maxsyma.de) in einem Zeitraum von mehreren Monaten im Jahr 2018 realisiert. Hierbei wurden die vollständigen Produktionsprozesse von zwei Produktionswerken in Nordbayern und England nach dem beschriebenen Verfahren umgesetzt. Weiteres Informationsmaterial zum beschriebenen Projekt kann auf Anfrage zur Verfügung gestellt werden. Aufgrund zunehmender Automatisierung und Betriebspersonalreduktion wird eine derartige Produktionsstraße durch immer weniger Personal abgedeckt. Interaktionen mit der bis zu 140 m langen Anlage erfordern deshalb längere Wege zu Bedienterminals und führen zu verzögerten Informationsfluss. Die verlängerten Reaktionszeiten sind häufig Auslöser erhöhter Ausschussmengen.

Sämtliche Produktionsabläufe und notwendige Tätigkeiten von Bedienern werden vorab in Form von BPMN-Prozessmodellen abgebildet. Die modellierten Prozesse referenzieren dabei zahlreiche verschiedene Sensorvariablen in Form von bedingten Ereignissen (*Intermediate Conditional Catch Events*), wodurch bei bestimmten Sensorwerten neue manuelle Aufgaben ausgelöst werden oder laufende Aufgaben (*Boundary Events*) abgebrochen werden. Ein exemplarischer Auszug aus den modellierten Prozessen ist in Abb. 4 dargestellt. Das Modell enthält sowohl menschliche Aufgaben als auch IoT-basierte Ereignisse und Entscheidungspunkte, welche die Abarbeitung des Prozesses während der Ausführung steuern. Bei der Umsetzung des Konzepts wurde ein Rollenmodell implementiert, in dem jeder Bediener den von ihm zu überwachenden Bereich der Anlage definiert. Im Fall des abgebildeten


***Abb. 5 Mobile Prozesssteuerung auf der Smartwatch***

Verklebungsprozesses („Splice“) erhalten Bediener im Bereich, in dem die Papierrollen in die Anlage eingespeist werden, durch die kontinuierliche Überwachung Aufgaben zum nächsten anstehenden Rollenwechsel oder über auftretende Defekte in der Wellpappenbahn. Um Bediener in Echtzeit auf anstehende Aufgaben hinzuweisen, wurde das gesamte Personal mit Smartwatches (siehe Abb. 5) ausgestattet und entsprechenden Gruppen entsprechend des jeweiligen Tätigkeitsbereiches zugeteilt. Neben aktuell verfügbaren Aufgaben werden auch diverse relevante Kontextinformationen auf der Smartwatch angezeigt. So stehen beispielsweise dem Bediener im Trockenende – das ist der Bereich, in dem fertige Wellpappenstapel die Anlage verlassen – Kontrollelemente zur Verfügung, die über die Restlaufzeit des Produktionsauftrages, den nächsten Stapelaustransport oder die Produktionsgeschwindigkeit informieren. Anwender im Nassende – der Bereich, in dem die Papierrollen in die Anlage eingespeist werden – erhalten durch die kontinuierliche Überwachung detaillierte Informationen zum nächsten anstehenden Rollenwechsel oder zu auftretenden Defekten in der Wellpappenbahn. Durch die beschriebene Umsetzung konnten Wege- und Reaktionszeiten signifikant verkürzt, damit Ausschuss reduziert und die Qualität gesteigert werden. Auch auf die Anzahl der Anlagenstillstände wirkte sich der Ansatz positiv aus. Diese konnten deutlich früher erkannt und proaktiv behoben werden. Somit konnte eine nachhaltige Steigerung der Gesamtanlageneffektivität erzielt werden.

## Fazit und Ausblick

Der vorgestellte Ansatz verbindet die Erfassung und Analyse von IoT- bzw. Sensordaten mit der Technologie des Prozessmanagements. Dabei wird bewusst auf Standardtechnologie aus den jeweiligen Bereichen gebaut. Es wird deutlich, dass die vorgestellte Integration auf Basis bereits existierender Konzepte und technischer Lösungen möglich ist und nicht notwendigerweise neue Konzepte geschaffen werden müssen. Die Standardprozessmodellierungssprache BPMN bietet adäquate Modellierungskonstrukte zur Abbildung und Referenzierung von IoT-Objekten und der damit verbundenen Echtzeitdaten. Existierende Prozessmanagementsysteme wie Camunda implementieren mit einer HTTP-Schnittstelle bereits ein adäquates Protokoll zur Umsetzung der bidirektionalen Kommunikation zwischen IoT- und Prozesswelt.

Dennoch gibt es Verbesserungspotenziale, die zur effizienteren und sichereren Kommunikation umgesetzt werden können. Wünschenswerte wäre beispielsweise eine IoT-/Prozesskommunikation über das schlanke MQTT-Protokoll, um im Fall von größeren Anwendungsfällen mit vielen Benutzern und IoT-Objekten den auftretenden Datenverkehr zu minimieren. Auch eine Erweiterung hinsichtlich sicherer und verschlüsselter IoT-Kommunikation durch die Verwendung von Secure MQTT [10] wäre dabei denkbar.

## Literatur


1. Dumas M, La Rosa M, Mendling J, Reijers HA (2013) Fundamentals of business process management. Bd 1. Springer, Heidelberg, p 2
2. Eckert M, Bry F (2009) Complex Event Processing (CEP). Informatik-Spektrum 32:163, https://doi.org/10.1007/s00287-009-0329-6
3. Günthner W, Wölfle M, Fischer R (2011) Wearable Computing und RFID in Produktion und Logistik – Ansätze zur bereichsübergreifenden Nutzung digitaler Informationen. Logistics Journal, DOI: http://doi.org/10.2195/lj_NotRev_guenther_de_201110_01
4. http://www.camunda.org
5. Meroni G, Baresi L, Montali M, Plebani P (2018) Multi-party business process compliance monitoring through IoT-enabled artifacts. Informat Syst 73:61–78
6. Object Management Group (OMG) (2011) BPMN Spezifikation. Version 2.0. Final Adopted Specification
7. Schönig S, Ackermann L, Jablonski S, Ermer A (2018) An integrated architecture for IoT-aware business process execution. In: Gulden J, Reinhartz-Berger I, Schmidt R, Guerreiro S, Guédria W, Bera P (eds) Enterprise, Business-Process and Information Systems Modeling. Springer, Cham, pp 19–34
8. Schönig S, Aires AP, Ermer A, Jablonski S (2018) Workflow support in wearable production information systems. In: Mendling J, Mouratidis H (eds) Information Systems in the Big Data Era. CAiSE 2018. Lecture Notes in Business Information Processing 317. Springer, Cham
9. Schönig S, Jablonski S, Ermer A, Aires Silva AP (2017) Digital connected production: wearable manufacturing information systems. In: Debruyne C, Panetto H, Weichhart G, Bollen P, Ciuciu I, Vidal M, Meersman R (eds) On the Move to Meaningful Internet Systems: OTM 2017 Workshops – Rhodes, Greece. Lecture Notes in Computer Science 10697. Springer, Heidelberg, pp 56–65
10. Singh M, Rajan MA, Shivraj VL, Balamuralidhar P (2015) Secure MQTT for Internet of Things (IoT). In: Communication Systems and Network Technologies (CSNT), 2015 Fifth International Conference on IEEE, pp 746–751
11. Van der Aalst WM (2016) Process Mining: Data Science in Action. Springer, Heidelberg

# *Mehrgittermethode*

## *Grundlage der computergestützten Wissenschaften*

**Ulrich Rüde**

### Einleitung

„Die Mehrgittermethode stellt einen der größten Fortschritte im Bereich der Numerik in den letzten Jahrzehnten dar." Dieser Satz stammt von Jens Volkert aus dem „aktuellen Schlagwort" des Informatikspektrums vom April 1989 [13], und er gilt unverändert auch heute, 30 Jahre später. Was sind Mehrgitterverfahren, so dass sie – ganz untypisch für die Informatik – über Jahrzehnte hinweg ein Dauerbrenner sind? Wie schon Jens Volkert vorhergesagt hatte, erklärt sich ihre Bedeutung aus der computergestützten Wissenschaft (engl. Computational Science and Engineering, CSE). Mehrgitterverfahren sind ein zentraler Bestandteil der Algorithmik, die realitätsgetreue Computersimulationen möglich macht.

Wissenschaftliche Erkenntnisse beruhen heute zunehmend auf Computermodellen. Beispiele findet man in der Physik, der Chemie, der Biomedizin, den Geowissenschaften, oder der Astronomie, aber das sind längst nicht alle. Überall helfen Computersimulationen, Prozesse und Systeme zu analysieren, die sich aufgrund der Zeit- und Raumskalen einer direkten menschlichen Beobachtung entziehen. Gleichzeitig sind Computersimulationen Grundlage in vielen Ingenieursdisziplinen, wenn z. B. neue Materialen am Computer entworfen werden oder wenn sichere und leise Flugzeuge entwickelt werden. Computersimulationen werden genutzt, um die Energieausbeute aus Windfarmen zu optimieren und um die Ausbreitung von Schadstoffen im Grundwasser vorherzusagen. Das Anwendungsspektrum ist universell. Nicht zuletzt können mit Computersimulationen belegbare, quantitative Aussagen über das zukünftige Klima gemacht werden. Nur mit Hilfe dieser Simulationsmodelle können dann die verschiedenen gesellschaftlichen und politischen Handlungsalternativen durchgespielt und bewertet werden.

Meist sind wissenschaftliche Modelle als partielle Differentialgleichungen formuliert. Diese werden z. B. mit der Methode der finiten Elemente oder mit finiten Differenzen diskretisiert, so dass große, dünn besetzte Gleichungssysteme entstehen. Die Bedeutung des Mehrgitterverfahrens liegt nun darin, dass diese Gleichungssysteme besonders effizient gelöst werden können, und dass deshalb genauere und zuverlässigere algorithmische Modelle der Realität möglich werden.

Das Mehrgitterverfahren ist dabei kein einzelner Algorithmus sondern vielmehr ein Konstruktionsprinzip für hocheffiziente Lösungsalgorithmen. Hocheffizient heißt hier, dass Mehrgitteralgorithmen asymptotisch optimale Komplexität haben. Im Idealfall benötigen sie nur $C \cdot N$ Gleitpunktoperationen (Floating Point Operations, FLOPS) zur Berechnung von $N$ Unbekannten, wobei $C$ eine Konstante ist. Diese lineare Komplexität zur Lösung von Gleichungssystemen ist dramatisch besser als z. B. das Standardverfahren mit Gauß'scher Elimination, das eine Komplexität $\frac{2}{3} N^3$ besitzt. Die asymptotisch optimale Komplexität ist auch die Grundvoraus-

https://doi.org/10.1007/s00287-019-01154-5
© Die Autoren 2019. Dieser Artikel wurde mit Open Access auf Springerlink.com veröffentlicht.

Ulrich Rüde
Lehrstuhl für Systemsimulation,
FAU Erlangen-Nürnberg,
Cauerstrasse 11, 91058 Erlangen, Deutschland
CERFACS, Avenue Gaspard Coriolis 42,
Toulouse 31057, France
E-Mail: ulrich.ruede@fau.de

Alle „Aktuellen Schlagwörter" seit 1988 finden Sie unter: http://www.is.informatik.uni-wuerzburg.de/as

setzung für skalierbare parallele Verfahren. Denn damit ein Verfahren auf einem Parallelrechner mit der Zahl der Prozessoren skaliert, darf der Aufwand nicht schneller als linear mit der Problemgröße ansteigen.

Allerdings sind Mehrgitterverfahren nicht universell für alle Gleichungssysteme einsetzbar. Mehrgitterverfahren müssen auch oft individuell für eine gegebene Problemklasse entwickelt werden, damit die spezifische inhärente Struktur des Problems ausgenützt werden kann. Das Prinzip von Mehrgitterverfahren geht auf Arbeiten von Fedorenko zurück [3], aber praktisch nutzbare Mehrgitterverfahren wurden zuerst von Brandt [1] und Hackbusch [5] vorgeschlagen und analysiert.

## Grundlagen der Mehrgitterverfahren

Mehrgitterverfahren beruhen auf stationären Iterationsverfahren. Um die Darstellung kompakt zu halten, nehmen wir an, dass eine gegebene lineare Differenzialgleichung auf einem Gitter mit der Maschenweite $h$ diskretisiert wurde und damit ein Gleichungssystem mit der Form

$$A_h u_h = f_h, \tag{1}$$

mit $N_h$ Unbekannten zu lösen ist. Der Vektor der Unbekannten ist $u_h \in \mathbb{R}^{N_h}$, die nichtsinguläre Matrix $A_h \in \mathbb{R}^{N_h \times N_h}$ ist dünn besetzt. Auch direkte Eliminationsverfahren können hierfür optimiert werden, so dass sie bessere Komplexität als das klassische Gauß-Verfahren erreichen. Jedoch kann mit direkten Verfahren in der Regel keine lineare Komplexität erreicht werden. Die Alternative sind Iterationsverfahren. Wir betrachten zunächst eine stationäre Iteration der Form, die von einem gegebenen $u_h^0$ startet und dann für $k = 1, 2, 3, \ldots$

$$u_h^{k+1} = u_h^k + \omega D_h^{-1} \left( f_h - A_h u_h^k \right) \tag{2}$$

berechnet. Beim klassischen Verfahren von Jacobi wird $D_h$ als Diagonale von $A_h$ gewählt, und $\omega$ ist ein skalarer Relaxationsparameter. Damit ist die Inverse $D_h^{-1}$ zwar trivial berechenbar und braucht auch gar nicht explizit gespeichert zu werden, aber für viele praktisch relevante Fälle konvergiert dieses Verfahren zu langsam. Dies bedeutet, dass zwar jede Ausführung von (2) nur lineare Komplexität hat, die Zahl der erforderlichen Iterationen von (2) aber mit $N_h$ anwächst. Damit ist der Gesamtaufwand schlechter als linear. Beim bekannten Gauß-Seidel-Verfahren und anderen, alternativen Relaxationsverfahren ist die Situation ähnlich.

Die langsame Konvergenz kann mathematisch präzise analysiert werden, in Spezialfällen z. B. mit Fourier-Techniken. Hier soll ein einfacheres, informatisches Argument helfen, das grundlegende Problem zu verdeutlichen. Die dünn besetzte Matrix $A_h$ kann als Graph interpretiert werden, der dem Gitter der Diskretisierung entspricht. Jede Iteration (2) transportiert nun numerische Werte entlang den Kanten dieses Graphen. Um die Information zur Berechnung der Lösung vollständig durch das gesamte Gitter zu transportieren, sind mindestens so viele Iterationsschritte nötig, wie dem Durchmesser des Gitters als Graphen entspricht. Für viele der zu lösenden Probleme, speziell, wenn die Ausgangsdifferenzialgleichung vom elliptischen Typ ist, ist dieser vollständige Austausch unabdingbar. Wir haben damit eine untere Schranke für die Iterationszahl und damit den Rechenaufwand für alle Verfahren der Bauart (2). Es sei noch angemerkt, dass die gleichen Schranken für das häufig verwendete Verfahren der konjugierten Gradienten gelten. Es ist mit dieser Überlegung offensichtlich, dass bei all diesen Verfahren die Zahl der erforderlichen Iterationen wachsen muss, wenn das Gitter für eine bessere Auflösung der Differentialgleichung verfeinert und damit der Durchmesser des Graphen größer wird. Um diese Komplexitätsschranke zu brechen, braucht es eine andere Idee, nämlich die der Mehrgitterverfahren.

Das Mehrgitterverfahren beruht auf einer Rekursion, die unterschiedlich feine Auflösungen der gegebenen Differenzialgleichungen geschickt kombiniert. Im obigen Sinne nutzen Mehrgitterverfahren eine Familie von Graphen, die zusammen einen schnelleren Transport der Information durch das Lösungsgebiet ermöglichen. Nehmen wir hierzu zunächst an, dass es zwei Gitter gibt. Neben dem Gitter mit der feinen Auflösung, $h$, sei nun auch eines mit gröberer Auflösung, $H$, gegeben, so dass die gegebene Differenzialgleichung auch durch ein kleineres Gleichungssystem der Form

$$A_H u_H = f_H, \tag{3}$$

auf einem gröberen Gitter mit $N_H$ Unbekannten approximiert werden kann. Durch Interpolation der Daten ist eine Transferabbildung $I_H^h$ vom groben Gitter $H$ auf das feine Gitter $h$ gegeben, sowie in umgekehrter Richtung die Restriktionsabbildung

$I_h^H$. Die Lösung des Gleichungssystems (3) mit der kleinen Matrix $A_H$ ist nun billiger als die Lösung mit der großen Matrix $A_h$. Weil $A_H$ eine Näherung an $A_h$ ist, liegt es nahe, diese Relation sowie (3) zu nutzen, um ein iteratives Verfahren für die Lösung von (1) zu konstruieren. Die sogenannte Grobgitterkorrektur hat die Form

$$u_h^{k+1} = u_h^k + I_H^h A_H^{-1} I_h^H \left(f_h - A_h u_h^k\right). \tag{4}$$

Dabei wird die inverse Matrix $A_H^{-1}$ natürlich nicht explizit berechnet (sie wäre dicht besetzt), sondern, dazu äquivalent – aber effizienter – das Gleichungssystem (3) mit rechter Seite $f_H = I_h^H \left(f_h - A_h u_h^k\right)$ gelöst. Es ist sofort klar, dass die exakte Lösung von (1) ein Fixpunkt der Iterationsvorschrift (4) ist. Jedoch ist diese stationäre Iteration für sich allein genommen divergent, weil $I_H^h A_H^{-1} I_h^H$ keinen vollen Rang hat, wenn $N_H < N_h$.

Um ein konvergentes Verfahren zu erhalten, muss die Grobgitterkorrektur (4) mit den Relaxationsverfahren vom Typ (2) kombiniert werden. Diese Kombination, d. h. die abwechselnde Ausführung von (2) und (4), führt zu einem sehr schnell konvergenten Verfahren, denn beide Verfahren haben komplementäre Eigenschaften. Die Relaxationen vom Typ (2) kümmern sich um feine Auflösungsdetails, die Grobgitterkorrektur (4) transportiert die Information global durch das Gitter. Dass die Grobgitterapproximation weniger genau ist, erzeugt keinen Schaden, da dies iterativ wiederholt wird. Im Mehrgitterjargon wird die Relaxation auch als *Glätter* bezeichnet, weil man analysieren kann, dass ihre Rolle die Reduktion des hochfrequenten Fehlers in einer Näherungslösung ist. Alternative Glätter sind das Gauß-Seidel-Verfahren oder unvollständige ILU-Faktorisierungen. Viele weitere Varianten sind möglich. Dabei ist nicht entscheidend, dass der Glätter (2) als Löser schnell konvergiert, nur die Eigenschaft des Glättens ist entscheidend.

Vom bisher beschriebenen Zweigitterverfahren zum Mehrgitterverfahren kommt man, wenn man die Lösung des Grobgitterproblems (2) rekursiv nach dem gleichen Schema berechnet. Diese Rekursion wird fortgesetzt, bis das Grobgitterproblem so klein wird, dass es z. B. auch mit einem direkten Verfahren schnell lösbar ist. Insgesamt erhält man damit den folgenden Grundalgorithmus:

Algorithmus
V-Zyklus = $Vcycle(u_h, h, N_h, f_h, \nu_{\text{pre}}, \nu_{\text{post}})$
- # Löse exakt:
  - If ($N_h$ == Coarsest) solve $A_h u_h = f_h$; return $u_h$
- # Glätte $\nu_{\text{pre}}$ mal:
  - $u_h = Relax(u_h, h, N_h, f_h, \nu_{\text{pre}})$
- # Berechne Residuum:
  - $r_h = f_h - A_h u_h$
- # Restringiere das Residuum zum groben Gitter:
  - $f_H = I_h^H r_h$
- # Initialisiere das Grobgitter mit 0:
  - $u_H = 0$; $N_H = N_h / 2^d$
- # Rekursion:
  - $u_H = Vcycle(u_H, H, N_H, f_H, \nu_{\text{pre}}, \nu_{\text{post}})$
- # Grobgitterkorrektur:
  - $u_h = u_h + I_H^h u_H$
- # Glätte $\nu_{\text{post}}$ mal:
  - $u_h = Relax(u_h, h, N_h, f_h, \nu_{\text{post}})$
  - return $u_h$

Die mathematische Analyse, warum und unter welchen Bedingungen das Verfahren schnell konvergiert, würde den Platz eines Schlagworts sprengen, so dass wir auf die weiterführende Literatur verweisen müssen [1, 2, 5, 6, 11].

## Komplexitätsüberlegungen

Wir können hier aber wenigstens die Kostenanalyse durchspielen. Die Kosten auf einem Gitterlevel, d. h. in einem rekursiven Aufruf von *Vcycle*, ergeben sich aus $\nu_{\text{pre}} + \nu_{\text{post}}$ Relaxationen, plus die Berechnung des Residuums und der Gittertransfers. Dabei ist entscheidend, dass $\nu_{\text{pre}}$ und $\nu_{\text{post}}$ typischerweise fixe kleine Zahlen sind, oft 1, 2, oder 3. Damit erzeugt jedes Level nur lineare Kosten mit $C_h \cdot N_h$ FLOPS. In die Rekursion geht die Annahme ein, dass jedes gröbere Level um den Faktor $2^d$ weniger Aufwand erzeugt, d. h., wir nehmen an, dass die Auflösung $h$ in jeder räumlichen Dimension ($d = 1, 2, 3$) halbiert wird. Der Gesamtaufwand kann somit mit einer geometrischen Reihe als

$$C_h \cdot N_h \left(1 + 2^{-d} + 2^{-2d} + 2^{-3d} + \ldots\right) \leq$$
$$C_h \cdot N_h \left(\frac{1}{1 - 2^{-d}}\right) = C_h \cdot N_h \left(\frac{2^d}{2^d - 1}\right) \tag{5}$$

abgeschätzt werden. Man sieht damit, dass jeder V-Zyklus des Mehrgitterverfahrens Kosten verursacht, die proportional zu $N_h$ sind. Das Verfahren hat damit asymptotisch optimale Komplexität. Ist auch die

schnelle Konvergenz des Verfahrens gegeben, wird eine Lösung mit vorgeschriebener Genauigkeit nach einer konstanten Zahl von V-Zyklen erreicht. Im Idealfall konvergieren Mehrgitterraten mit Konvergenzraten von 0,1 oder besser, so dass z. B. 5 Stellen Genauigkeit bereits nach 5 V-Zyklen erreicht sind.

Es ist sogar noch mehr möglich. Denn mit immer weiterer Verfeinerung des Gitters möchte man natürlich die Genauigkeit jeweils entsprechend erhöhen. Auch dies ist mit linearem Aufwand möglich, wenn man den V-Zyklus in eine weitere Rekursion einbettet. Dies ist dann unter den Namen „Full Multigrid" oder „Nested Iteration" in der Literatur bekannt. Zusammengefasst hat man damit ein Verfahren, bei dem die Lösung eines Gleichungssystems (1) nur um einen konstanten Faktor teurer ist als die Multiplikation eines Vektors mit der Matrix $A_h$. Für typische Modellprobleme wie das Poisson-Problem in 2D, diskretisiert mit Differenzen zweiter Ordnung, kann so eine Lösung mit weniger als $30\,N_h$ FLOPS erreicht werden [11].

## Algorithmische Varianten und Parallelisierung

Die exzellente Effizienz ist äußerst erfreulich, weil man damit auch große Probleme extrem schnell lösen kann. Dies erfordert aber auch eine effiziente Implementierung. Die Lösung einer Poisson-Gleichung in 2D auf einem Gitter von $1024 \times 1024$ benötigt nur $30 \times 10^6$ FLOPS, kann also auf einer modernen CPU oder auch mit einer GPU in nur Millisekunden erledigt werden. Umso erstaunlicher ist, dass immer wieder Algorithmen vorschlagen werden, die drei, vier oder fünf Größenordnungen weniger effizient als das Mehrgitterverfahren sind. Hier ist Vorsicht angeraten: Nicht alles, was in der theorieorientierten Literatur zu finden ist, hat auch praktische Relevanz.

Neben den geometrischen Mehrgitterverfahren, die die Struktur der partiellen Differenzialgleichung und ihrer Diskretisierung explizit nutzen um die Diskretisierungen $A_h$ und $A_H$ etc. direkt zu erzeugen, gibt es sogenannte algebraische Mehrgitterverfahren. Diese sind dadurch motiviert, dass man gerne die Aufstellung der Matrizen vom algebraischen Lösungsverfahren trennen würde. Algebraische Mehrgitterverfahren müssen deshalb versuchen, die inhärente Vergröberungsstruktur aus der Matrix $A_h$ zu bestimmen. In diesem Fall werden verschiedene Heuristiken eingesetzt um zunächst eine Interpolation $I_H^h$ zu konstruieren. Dann setzt man die Restriktion als Transponierte an $I_H^h = (I_H^h)^T$ und kann damit die sogenannte Galerkin-Approximation $A_H = I_h^H A_h I_H^h$ verwenden. In jedem Fall führt dies jedoch zu signifikanten Overheads im Vergleich zu geometrischen Mehrgitterverfahren.

In die entgegengesetzte Richtung gehen jüngere Überlegungen, dass auf modernen Rechnerarchitekturen die Laufzeit eher durch die Speicherbandbreite begrenzt wird als durch die Ausführung der FLOPS. Deshalb rücken sogenannte matrixfreie Verfahren in den Fokus. Dabei wird die Matrix $A_h$ nicht gespeichert, sondern bei Bedarf „on-the-fly" berechnet. Man erkauft sich damit eine verringerte Speicherbandbreite durch einen erhöhten FLOPS-Aufwand. Andere Optimierungsmöglichkeiten ergeben sich z. B. durch eine bessere Nutzung der Speicherhierarchie, d. h. eine bessere Nutzung der Caches, wie sie z. B. im Projekt „Data-Local Iterative Methods" (DIME) schon früh untersucht wurde. Dieses Projekt wurde von 1998–2005 gemeinsam von dem Lehrstuhl für Rechnerarchitektur der TU München (Prof. Bode) und dem Lehrstuhl für Systemsimulation der Universität Erlangen-Nürnberg durchgeführt.

Die Parallelisierung von Mehrgitterverfahren ist schon seit den 1980er-Jahren ein wichtiges Thema. Im Suprenum-Projekt wurde von schon 1985–1990 an der damaligen Gesellschaft für Mathematik und Datenverarbeitung (GMD) in Kooperation mit mehreren Universitäten eine parallele Rechnerarchitektur speziell für Mehrgitterverfahren entwickelt und realisiert. Dabei wurden viele wichtige Grundlagen für das parallele Hochleistungsrechnen gelegt, die bis heute ausstrahlen. Algorithmisch stellt sich bei den Mehrgitterverfahren das Problem, dass nicht nur sehr große, hochaufgelöste Gitter bearbeitet werden müssen, sondern eine hierarchische Folge von Gittern, die sukzessive kleiner werden. Deshalb müssen Mehrgitterverfahren besonders sorgfältig implementiert werden, damit die Schleifen-Overheads und Kommunikationslatenzen auf großen parallelen Systemen nicht zu stark zu Buche schlagen. Oft ist es so, dass gut implementierte Mehrgitterverfahren den deutlich schnellsten Löser für eine Problemklasse bereitstellen, obwohl sie im Hinblick auf maximale FLOPS-Rate oder Systemauslastung hinter anderen Verfahren zurückstehen. Es ist deshalb wichtig, sich das Problem des Datentransports klar zu machen. Mehrgitterverfahren nutzen eine Hierarchie von Gittern, um die Daten

schnell durch das Lösungsgebiet zu transportieren. Es ist unvermeidlich, dass die Systemauslastung eines großen Parallelrechners sinkt, während die groben Gitter bearbeitet werden. Der Schluss, dass man deshalb die groben Gitter vermeiden müsste, ist aber grundfalsch. Denn wenn man den essenziellen Datenaustausch auf feineren Gittern durchführt, hat man zwar womöglich eine bessere Systemauslastung und höhere FLOPS-Rate, muss aber dafür mit einer unnötig hohen Iterationszahl bezahlen. Die Lösung findet man damit sicher nicht schneller, sondern insgesamt langsamer und mit redundant erhöhtem Rechenaufwand. Forschung zu diesem Themenkomplex findet derzeit in Deutschland gebündelt im DFG-Schwerpunktprogramm „Software für Exascale Computing“ (SPPEXA) [9] statt.

## Ein Beispiel aus den Geowissenschaften

Abschließend wollen wir die Leistungsfähigkeit moderner Mehrgitterverfahren mit einem Problem aus der Geophysik illustrieren. Es geht darum, den Erdmantel zu modellieren, also die ca. 3000 km dicke Felsschicht unseres Planeten, die nach außen durch die Kontinentalplatten und die Lithosphäre begrenzt ist und nach innen durch den flüssigen Eisenkern. Auf einer Zeitskala von Jahrmillionen verhält sich der Erdmantel wie eine zähe Flüssigkeit mit Strömungsgeschwindigkeiten von wenigen cm pro Jahr. Die langsame Konvektionsströmung des Erdmantels ist Ursache der Kontinentalverschiebung; sie führt zur Formung von Gebirgen und bedingt die Entstehung von Erdbeben. Der Erdmantel hat ein Volumen von ca. $10^{12}$ km$^3$. Weil sich geologische Phänomene auf der Skala von 1 km oder weniger abspielen, würde man gerne die Simulation des Erdmantels mit einer Auflösung von 1 km oder weniger berechnen. Die Diskretisierung des Gebiets führt damit zu Systemen von ca. $10^{12}$ Gitterzellen, von denen jede durch mehrere Zustandsvariablen wie Temperatur, Geschwindigkeit, und Druck gekennzeichnet ist. In jedem Zeitschritt einer Simulation muss nun ein Gleichungssystem mit diesen $N = 10^{12}$ oder mehr Unbekannten gelöst werden. Würde man dazu unbesehen das Gauß'sche Eliminationsverfahren einsetzen, so müsste man die astronomische Zahl von $2/3\,N^3 = 2/3 \cdot 10^{36}$ FLOPS ausführen. Selbst wenn man die dünne Besetzungsstruktur der Matrix z. B. mit dem Nested-Dissection-Algorithmus für eine Reduktion des Fill-In nutzen würde, kann man den Aufwand hochrechnen und erhält immer noch mehr als $10^{24}$ FLOPS. Ein PC mit 100 GigaFLOPS Leistung würde hierfür eine Rechenzeit mehr als 100 000 Jahren benötigen. Selbst der aktuell schnellste Rechner in Deutschland, SuperMucNG [10], würde für diese Anzahl FLOPS noch länger als ein Jahr benötigen. Wohlgemerkt: für einen von vielen nötigen Zeitschritten. Dieser gigantische Rechenaufwand rechtfertigt die Klassifizierung des Problems als „Grand Challenge-Anwendung“.

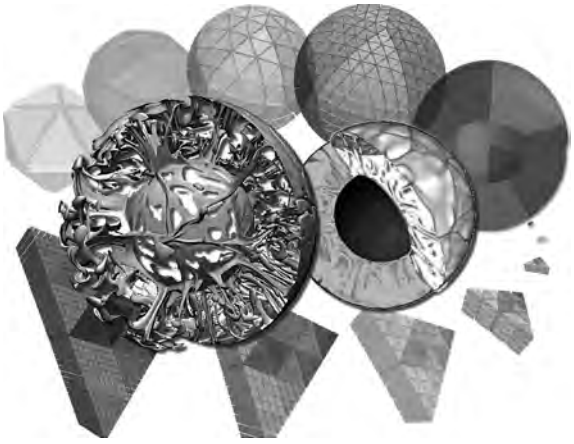


*Abb. 1 Gitterhierarchie für die Simulation der Erdmantelkonvektion mit 1 km Auflösung*

Im SPPEXA-Projekt TerraNeo wird ein neues paralleles Software-Framework für die Erdmantelkonvektion entwickelt [7]. Die dabei genutzte Mehrgitterhierarchie wird in Abb. 1 zusammen mit einer Visualisierung der auf- und absteigenden Strömungen und Temperaturfelder exemplarisch dargestellt. Für weitere Hintergründe zu den Methoden und Forschungsergebnissen verweisen wir auf die Webseite [7], die dort gelisteten Veröffentlichungen und speziell auf das Teachlet [8]. In diesem interdisziplinären Projekt, das Informatiker und Mathematiker mit Geophysikern zusammenführt, ist es gelungen, die relevanten Gleichungssysteme mit bis zu $10^{13}$ Unbekannten zu lösen. Ein einzelner Lösungsvektor benötigt dabei bereits 80 TByte Speicherplatz, so dass selbst die größten heute verfügbaren Supercomputer nur wenige von diesen Vektoren gleichzeitig speichern können. Es ist unmöglich, die dazu gehörigen Matrizen zu speichern. Es mussten deshalb neue matrixfreie Mehrgitterverfahren und ein spezieller Glätter entwickelt werden. Mit diesem Mehrgitterverfahren

und bei Verwendung eines der schnellsten Supercomputer [4] kann eine solche Lösung in ca. 13 min berechnet werden. Mehrgitterverfahren und moderne Supercomputer zusammen ermöglichen so geophysikalische Simulationen mit einer Detailtreue und einer Auflösung, die mit anderen Verfahren völlig undenkbar wäre.

## Ausblick

Wegen der rapide steigenden Bedeutung von Computersimulationen in Wissenschaft und Technik gewinnt *Computational Science and Engineering* (CSE) als neue Fachdisziplin weiter zunehmend an Bedeutung. Auf der Basis von Grundlagen aus der Mathematik und Informatik können neue Simulationsverfahren entwickelt werden. Schnelle Algorithmen, wie das Mehrgitterverfahren, sind unverzichtbar, um ausreichend genaue Berechnungen durchzuführen und damit fundamentale Fortschritte in vielen Wissenschaftsbereichen zu erzielen. Die Computermodelle des CSE beruhen auf physikalischen Grundgesetzen wie Massen-, Energie-, und Impulserhaltung und bauen darauf kausale Wirkungsketten auf. Diese können präzise formuliert, hinterfragt und überprüft werden. Darin unterscheiden sich CSE-Methoden von Big-Data-Ansätzen, die aus gegebenen Daten Korrelationen bestimmen und diese mithilfe von ausgeklügelten Interpolations- und Extrapolationstechniken nutzen, um Plausibiliätsvorhersagen zu machen. Eine große Zukunftsaufgabe wird es sein, diese beiden Ansätze sachgerecht zu verbinden.

**Open Access.** Dieser Artikel wird unter der Creative Commons Namensnennung 4.0 International Lizenz (http://creativecommons.org/licenses/by/4.0/deed.de) veröffentlicht, welche die Nutzung, Vervielfältigung, Bearbeitung, Verbreitung und Wiedergabe in jeglichem Medium und Format erlaubt, sofern Sie den/die ursprünglichen Autor(en) und die Quelle ordnungsgemäß nennen, einen Link zur Creative Commons Lizenz beifügen und angeben, ob Änderungen vorgenommen wurden.

## Literatur

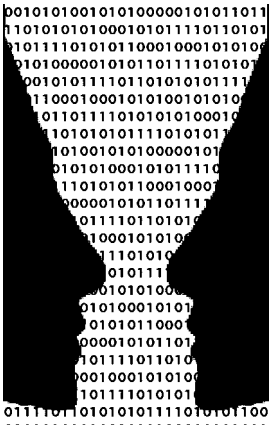
1. Brandt A (1977) Multi-level adaptive solutions to boundary-value problems. Math Comp 31:333–390
2. Brandt A, Livne OE (2011) Multigrid techniques: 1984 guide with applications to fluid dynamics. Vol. 67. SIAM
3. Fedorenko RP (1962) A relaxation method for solving elliptic difference equations. U.S.S.R. Comput Math Math Phys 1(5):1092–1096
4. Gmeiner B, Huber M, John L, Rüde U, Wohlmuth B (2016) A quantitative performance study for Stokes solvers at the extreme scale. J Comput Sci 17: 509–521
5. Hackbusch W (1976) Ein iteratives Verfahren zur schnellen Auflösung elliptischer Randwertprobleme. Report 76-12. Institut für Angewandte Mathematik, Universität Köln
6. Hackbusch W (1985) Multi-grid methods and applications. Springer Series in Computational Mathematics 4. Springer, Berlin, New York
7. http://terraneo.fau.de
8. http://terraneo.fau.de/Files/teachlet_v1.0_1080p.mp4
9. http://www.sppexa.de
10. https://doku.lrz.de/display/PUBLIC/SuperMUC-NG
11. Stüben K, Trottenberg U (1982) Multigrid methods: Fundamental algorithms, model problem analysis and applications. In: Hackbusch W, Trottenberg U (eds) Multigrid methods. Lecture notes in mathematics 960. Springer, Berlin, Heidelberg
12. Trottenberg U, Oosterlee CW, Schuller A (2000) Multigrid. Elsevier
13. Volkert J (1989) Mehrgittermethode. Informatik-Spektrum 12(2)

FORUM

# *Gewissensbits – wie würden Sie urteilen?*

*Constanze Kurz,*
*Netzpolitik.org, Berlin*
*Rainer Rehak,*
*Weizenbaum-Institut*
*für die vernetzte Gesellschaft, Berlin*

Mit den ethischen Leitlinien der GI haben wir es uns zur Aufgabe gemacht, Diskurse zu ethischen Problemen der Informatik zu initiieren und zu fördern. Mitglieder der Fachgruppe „Informatik und Ethik“ der GI stellen jeweils ein hypothetisches, aber realistisches Fallbeispiel vor, das zur Diskussion anregen soll. Die Fälle können jeweils von Interessierten im Blog der Fachgruppe auf der GI-Website https://gewissensbits.gi.de kommentiert und diskutiert werden.



## Fallbeispiel HackerZero

Elvira und Nico haben Informatik studiert und schreiben gerade ihre Dissertationen im Bereich IT-Sicherheit. Daneben lehren sie an verschiedenen Universitäten. Sie haben ein Projekt namens „HackerZero“ angeregt und konnten dafür Fördergelder gewinnen. Das Projekt bietet eine universitäre Plattform als Portal für Sicherheitsforschung an. Zehn verschiedene Universitäten sind an dem Projekt beteiligt. Elvira und Nico freuen sich über die Zusage, dass die Förderung für weitere zwei Jahre verlängert wurde.

Wer Sicherheitslücken in Software entdeckt hat, kann diese über die innovative Plattform „HackerZero“ an die betroffenen Hersteller oder Anbieter melden und muss gleichzeitig das Vorgehen und Wissen offenlegen. Da die Kommunikation über Elvira und Nico als Betreiber der Plattform läuft, braucht man sich nicht über unangenehme juristische Folgen zu sorgen, da sie die Meldungen entgegennehmen und weiterleiten. Elvira und Nico übernehmen also die Kommunikation mit den betroffenen Softwareanbietern. Als Informatikfachleute prüfen die beiden aber auch die technische Seite der gemeldeten Schwachstellen.

Um zu verhindern, dass die Sicherheitslücken ausgenutzt werden, haben die kommerziellen Partner und auch die Open-Source-Projekte nach Meldung eine gewisse Zeit zur Verfügung, in der sie das Wissen um eine Sicherheitslücke exklusiv erhalten. Aber nach spätestens drei Monaten wird das Wissen für alle Plattformmitglieder offengelegt und dann veröffentlicht. Den Firmen wird also eine Frist gesetzt, Patches für die gefundenen Probleme in ihrer Software bereitzustellen. Wenn die Firmen die Sicherheitslücken als „behoben“ markieren, so werden diese auch vor Fristablauf schon offengelegt.

Die Firmen können entscheiden, ob sie zusätzlich auch Informationen zur Lösung bekanntgeben wollen. Damit können alle Personen, die sich mit Sicherheitsfragen beschäftigen, aus diesen Fehlern – und den Lösungen – lernen.

Die Plattform „HackerZero“ wird vom Konsortium der zehn beteiligten Universitäten betrieben. Sie steht

https://doi.org/10.1007/s00287-019-01163-4

auch kommerziellen Partnern als Plattform zur Verfügung, um ihre Produkte samt Quellcode für die Sicherheitsprüfung zu hinterlegen. Open-Source-Projekte können sich ebenfalls als Partner anmelden.

„HackerZero“ bietet IT-Sicherheitsforschern für die Meldung von Softwareschwachstellen eine Aufwandsentschädigung in Form eines Preisgeldes an. Es sind keine großen Summen, die mit dem kommerziellen Markt mithalten könnten, aber damit soll ein zusätzlicher Anreiz für das Nutzen der Plattform zur Offenlegung von Problemen gesetzt werden.

In „HackerZero“ werden diese Preisgelder von den kommerziellen Partnern finanziert. Die Gelder kommen in einen gemeinsamen Topf, sodass für alle Meldungen von Sicherheitslücken Gelder zur Verfügung stehen und auch Open-Source-Projekte von der Plattform profitieren können. Sowohl Elvira als auch Nico sind davon überzeugt, dass die Offenlegung von Schwachstellen stets von sehr großem Nutzen für alle an IT-Sicherheitsforschung Interessierten sowie für die Softwarehersteller selbst ist.

Während einer Projektsitzung kommt es wegen einer Neuanmeldung zu heftigen Diskussionen: Der neue kommerzielle Partner hat sich bei „HackerZero“ angemeldet, sein Produkt nutzt jedoch selbst Schwachstellen in anderen Softwareprodukten aus. Einige der universitären Partner haben nun mit dem Ausstieg aus dem Projekt gedroht.

Elvira hatte Nico nach der Anmeldung sofort geschrieben, dass sie den neuen Partner nicht akzeptabel findet. Klar sei doch, dass der Neuzugang ein Käufer von Sicherheitslücken sei oder aber mindestens ein Interesse haben müsse, Schwachstellen in der Software möglichst lange offenzuhalten, um das eigene Produkt besser verkaufen zu können. Das widerspräche klar der Intention der ganzen „HackerZero“-Plattform. Elvira meint, dass nicht mal klar sei, ob das Produkt der potenziellen neuen Partnerfirma legal sei.

Nico hält dagegen, dass es immer gut sei, Schwachstellen aufzudecken, auch in solchen Softwareprodukten, die ihrerseits Sicherheitslücken ausnutzen. Die Firma hätte ihren Sitz in Deutschland, was ein illegales Produkt wahrscheinlich ausschließe. Außerdem sei es gerade bei solcher Software besonders wichtig, dass sie sicher und handwerklich gut programmiert sei. Schließlich sei ihnen doch beiden klar, dass die Kunden der Firma wohl vornehmlich Strafverfolgungsbehörden sein würden.

## Fragen

- Ist es sinnvoll und ethisch vertretbar, über eine universitäre Plattform Gelder für Schwachstellenmeldungen anzubieten, wenn eine Offenlegung zu einem späteren Zeitpunkt stattfindet?
- Müssen sich Elvira und Nico damit auseinandersetzen, welche Firmen und welche Software bei „HackerZero“ angemeldet sind? Wäre es notwendig gewesen, dies festzulegen, als das Projekt definiert wurde?
- Widerspricht die Aufnahme eines Partners, dessen Geschäft die Ausnutzung von Schwachstellen ist, grundsätzlich dem Projektziel? Warum? Gäbe es Gründe, die die Aufnahme rechtfertigen? Wenn ja, welche wären das?
- Wäre es vertretbar oder sogar notwendig, spezifische Regeln für den Umgang mit Vorabinformationen über Sicherheitslücken einzuführen?
- Würde sich etwas ändern, wenn eine staatliche Stelle als Partner teilnehmen wollte?
- Änderte sich dadurch etwas, wenn sich keine der beteiligten Universitäten über die neue Partnerfirma verärgert gezeigt hätte? Müssten Elvira und Nico dennoch die Art der zu untersuchenden Software diskutieren?
- Wen könnten Elvira und Nico hinzuziehen, wenn sich die beiden über die Anmeldung der neuen Partnerfirma nicht einig werden können? Sollten sie das Dilemma gar öffentlich diskutieren?

Die Fachgruppe ist unter http://www.fg-ie.gi.de/ erreichbar. Unser Buch *Gewissensbisse – Ethische Probleme der Informatik. Biometrie – Datenschutz – geistiges Eigentum* ist im Oktober 2009 im Transkript-Verlag erschienen. Ein neues Werk ist in Arbeit.

FORUM

# *E-Contracting*

*Ursula Sury*

Unter **E-Contracting** versteht man die komplette elektronische Abbildung des Vertragsprozesses. Es enthält auch alle relevanten Dokumente der Vertragsverhandlung, wie zum Beispiel die elektronischen Protokolle der Besprechungen zwischen den Vertragsparteien.

Beim E-Contracting ist es möglich, den Vertrag abzuschliessen, ohne vorher mit den Geschäftsparteien physisch oder persönlich in Kontakt zu treten. Sofern der elektronische Verhandlungsprozess erfolgreich ist, kommt der elektronische Vertrag zustande. Die Vertragsparteien werden dadurch rechtsgültig gebunden. Der elektronische Vertrag umfasst sämtliche Phasen der Geschäftstransaktion.

Wenn es um die Lieferung immaterieller Güter geht, ist es möglich, dass diese auch wieder auf elektronischem Weg ausgeliefert werden können. Somit sind auch die Fragen der Rechtzeitigkeit und der Qualität digital messbar und gegebenenfalls automatisierbar. Dies ist eben eine der Anwendungen von Smart Contracts.

Die ganze Thematik hat heute mit der Diskussion über und um **Smart Contracts** stark an Bedeutung gewonnen (siehe hierzu auch „Smart Contracts“ von Ursula Sury im Heft 4/2017, Informatik Spektrum). Grundsätzlich ist aber die Thematik nicht neu. Schon vor rund 20 Jahren hat man im Bereich EDI/EDIFACT elektronisch Verträge abgeschlossen und den Vollzug durchgeführt oder ausgelöst.

Bei den **EDIFACT/EDI** Konstruktionen wird zuerst ein Rahmenvertrag im traditionellen Sinn und auf traditionellem Weg, d. h. durch übereinstimmende gegenseitige Willensäusserung von natürlichen Personen, abgeschlossen. Darin werden sämtliche Aspekte, welche inhaltlich und technisch abgewickelt werden sollen, geregelt. Inhaltlich soll zum Beispiel vorgegeben werden, in welchem IT-System Bestellungen ausgelöst werden sollen. Dasselbe ist möglich für die Festlegung, innert welcher Fristen Lieferungen von korrespondierenden Systemen erfolgen sollen.

Dennoch können im ganzen System auch Fehler auftreten, d. h. es bestehen (rechtliche) **Risiken**, die benannt und zugeordnet werden müssen. Beispielsweise können falsche Bestellungen oder Lieferungen ausgelöst werden oder Übertragungsfehler vorliegen. Solche Risiken müssen bei der Anwendung eines solchen Systems berücksichtigt werden.

Ein wichtiger Teil der Thematik des E-Contracting ist die Frage der von den Parteien für den Vertragsabschluss und die Vertragserfüllung akzeptierten **Form** und damit verbunden die Frage, was bei Streitigkeiten als beweisgenüglich anerkannt wird.

Bestehen klassische Rahmenverträge, kann dies durchaus auch dort geregelt werden. Ansonsten ist man gut bedient, die digitalen Dokumente analog den Anforderungen der Archivierung von Buchhaltungsunterlagen mit Verschlüsselungen oder ähnlichen Methoden zu sichern.

Ein gemeinsam übereinstimmender Wille bildet die Grundlage eines Vertrages. Dieser kann in unterschiedlicher Form vorliegen. Im klassischen Vertrag kann er

https://doi.org/10.1007/s00287-019-01162-5

beispielsweise schon durch konkludentes Verhalten erkenntlich sein. Jedoch kann man auch vereinbaren, dass dieser schriftlich festgehalten wird. In der digitalen Welt wird hierfür auch das Email verwendet. Mit dem Aufkommen der Smart Contracts stellt sich jetzt jedoch die Frage, ob ein Programmcode, der einen „Willen" ausdrückt und eine Handlung auslöst, auch eine Willenserklärung darstellt. Eine spannende Frage, deren Antwort noch ausstehend ist.

## Fazit

Die Digitalisierung macht auch bei Vertragsverhandlungen und -abschlüssen keinen Halt. E-Contracting führt zu einer Vereinfachung von Prozessen, jedoch sind auch die damit verbundenen Risiken nicht ausser Acht zu lassen und richtig einzuordnen. E-Contracting ermöglicht, dass Rechtzeitigkeit und Qualität digital gemessen und automatisiert werden kann.

*Ursula Sury ist selbständige Rechtsanwältin in Luzern, Zug und Zürich (CH) und Vizedirektorin an der Hochschule Luzern – Informatik. Sie ist zudem Dozentin für Informatikrecht, Datenschutzrecht und Digitalisierungsrecht.*

FORUM

## Einsichten eines Informatikers von geringem Verstande

## Autonom gefahren

*Reinhard Wilhelm*
*Saarland Informatics Campus, Saarbrücken*

Neulich bin ich zum ersten Mal autonom gefahren, genauer autonom gefahren worden. Na ja, nicht in einem Auto, sondern in einer Zugtoilette. Gut gestalteter Raum, sehr aufgeräumt, kein störender Pedalbetrieb, wie man ihn früher hatte. Nur noch ein paar gut beleuchtete Bedienelemente, ansonsten alles rechnergesteuert. Ich hatte keine zu erledigenden Büroarbeiten dabei, könnte mir aber vorstellen, sie in entspannter Atmosphäre dort zu erledigen, Zeitung lesen sowieso. Gebremst bzw. beschleunigt wurde gänzlich ohne mein Zutun, so sanft, dass keinerlei Schwindelgefühle auftraten.

Bei der Vernetzung mit Zugtoiletten in entgegenkommenden oder überholenden Zügen ist die Bahn schon weiter als die Autohersteller. Zu Demonstrationszwecken ließ sie sogar die Nutzer diese Kommunikation verfolgen, „Hallo, Klo in Wagen 4 von RE2734, wie sieht es bei Dir mit Verspätung aus? Ich habe derzeit eine Verspätung von 27 Minuten." „Hallo, Klo in Wagen 21 von ICE2132, da bist du aber gut dran, bei mir sind es schon 47 Minuten. Noch gute Fahrt!" „Dir auch gute Fahrt!"

Der Einsatz künstlicher Intelligenz ist derzeit noch nicht ganz perfekt durchdacht, beim automatischen Spülen war wohl die durchschnittliche und nicht die maximale Entleerungsdauer gelernt worden – das ist ja durchaus ein weit verbreiteter Fehler –, so dass das automatisch ausgelöste Spülen etwas zu früh kam. Ein ähnlicher Fehler wurde bei der Verweildauer gemacht, was zu einer vorzeitigen Öffnung der Tür führte. Vielleicht sollte die Bahn mal ihre KI-Systeme aus China und nicht aus Cyber-Valley im deutschen Südwesten beziehen.

Bei allem Lob muss man allerdings sagen, dass man die Zugtoiletten erst dann als voll autonom bezeichnen kann, wenn sie auch das undress- und das dress-by-wire vollständig beherrschen. Bisher ist da noch zu viel menschlich betriebene Mechanik bzw. Hydraulik nötig. Auch da wird man ausgereifte künstliche Intelligenz brauchen. Denn das Erkennen geschlechtstypischer Bekleidungen – vom dritten Geschlecht ganz zu schweigen – sowie von modischen Verirrungen wie hinten geknöpften Hosen, geschnürten Miedern oder Ganzkörperstrumpfhosen und das Öffnen von raffinierten Gürtel- oder Hosenträgerschließen würden derzeitige Systeme noch hoffnungslos überfordern.

Es ist noch viel zu tun! KI-Forscher, an die Arbeit!

https://doi.org/10.1007/s00287-019-01160-7

FORUM

# *Achtung.Datentrickserei*

## Wahlbetrug – Wer die Wahl hat hat die Qual

*Hans-J. Lenz,*
*Freie Universität Berlin*

Man könnte meinen, Wählen wäre das Einfachste der Welt. Im Wesentlichen benötigt man nur Stimmabgabe und -auszählung und das Wahlergebnis – einfacher geht es nicht. Erstaunlich ist, welcher Millionen- bis Milliardenaufwand im politischen Bereich betrieben wird, und wie Politiker, Parteien und Dritte Wahlen mittels Pressemedien, Tweets, Internet und sozialer Netze immer dreister und skurriler Daten direkt bzw. indirekt zu manipulieren versuchen.

Auf den ersten Blick scheinen Wahlen im politischen, aber auch betrieblichen oder behördlichen Bereich einfach durchführbar zu sein. Denn drei Grundsätze liegen „fairen" Wahlen zugrunde: Sie sollen frei, gleich und geheim sein unter Beachtung des aktiven und passiven Wahlrechts. Bei Kommunal-, Landes- und Bundeswahlen in Deutschland kommt noch hinzu, wie die Stimmenanteile von Parteien auf Parlamentssitze nach jeweils geltendem Wahlgesetz umgerechnet werden. Es liegt auf der Hand, dass die vielen „Drehschrauben" bei einer Wahl viele Eingriffs- und Manipulationsmöglichkeiten bieten – man denke nur beispielsweise an die Höhe und Herkunft des Wahlbudgets eines Kandidaten bzw. einer Partei oder an die Wahrhaftigkeit der ausgezählten Stimmen. Bevor wir uns in Einzelheiten verlieren, scheint es angebracht, sich die einzelnen Phasen einer Wahl in Erinnerung zu rufen und daran entlang Manipulationsrisiken aufzuzeigen.

Auf den ersten Blick erscheint die Planungsphase unverfänglich – die Bestimmung eines Wahlleiters, seines Teams und der dafür notwendigen Ausrüstung und Organisation lassen wir einmal außen vor. Wie das Beispiel England unter Premierministerin Thatcher zeigte, muss eine Regierung sich beim Festsetzen von Neuwahlen nicht einmal strikt an die Legislaturperiode halten. Sie kann Wahltermine zugunsten der Regierungspartei verschieben, wenn es politisch und wirtschaftlich oder vielleicht auch militärisch opportun erscheint. Ähnlich gelagert ist es bei der Wahlkreisfestlegung. In der Bundesrepublik war ein Neuzuschnitt zuletzt nach dem Wendejahr 1989 angesagt. In Brandenburg hat das Verwaltungsgericht Cottbus im August 2018 entschieden, dass die Stadt Cottbus die Kommunalwahlen von 2014 wiederholen muss. Die Richter bemängelten, dass die Anzahl Bürger in den fünf Wahlkreisen zu sehr schwankte, in einem 12,6 % unterdurchschnittlich zu wenig, im anderen mit 21 % über dem Stadtmittel zu viel [2]. Aktuell besteht das Problem in den USA bei den Zwischenwahlen („Midterms") 2018. Der Zuschnitt („Gerrymandering" oder „Redistricting") von Wahlkreisen zum Vorteil der Partei, die in einem Bundesstaat an der Macht ist, führte beispielsweise in Pennsylvania dazu, dass Wahlkreise nicht mehr zwischen Demokraten und Republikanern umkämpft waren wie vorher. Das Oberste Gericht von Pennsylvania wies die Wahlkreisaufteilung aus Fairnessgründen zurück und forderte eine neue Aufteilung [8]. Die gesetzlichen Vorschriften zur Stimmenauszählung wie Verhältnis- oder Mehrheitswahlrecht mit allen

https://doi.org/10.1007/s00287-019-01164-3

Varianten können zu Ergebnissen führen, die scheinbar manipulationsbedingt sind. Eine besondere Art wird in den Sozial- und Politikwissenschaften der USA als „Partisan Symmetry" bezeichnet. G. King führt als Beispiel die Wahlen in Wisconsin an. 2012 erzielten die Republikaner dort 48 % der abgegebenen Stimmen und erhielten dafür 60 % der Parlamentssitze. 2014 gewannen dagegen die Demokraten 48 %, aber erhielten nur 36 % der Sitze – bei derselben Wahlbezirksaufteilung [9].

Wie nicht anders zu erwarten, ist auch der Wahlkampf keineswegs frei von Irregularitäten bis hin zum Wahlbetrug. Zu den mehr noch traditionellen Mitteln gehört das Entfernen von Wahlplakaten und Wahlaufrufen einer bestimmten Partei. Während derartige Aktionen im Allgemeinen lokal begrenzt sind, haben illegale Wahlkampfspenden landesweite Auswirkungen. Ein Fall von unzählig vielen in aller Welt, jedoch ein besonders fragwürdiger und aktueller ist der des undurchsichtigen Hedgefondsmilliardärs und Informatikers Robert Mercer, der seine massive finanzielle Unterstützung in Millionenhöhe für Trump bei den Wahlen in den USA 2016 geschickt verschleierte [5]. Aber auch CDU und FDP waren in der Vergangenheit in illegale Spenden und „Schwarzen Kassen" und damit zwangsweise in Datenmanipulationen verwickelt, siehe Flick-Affäre [3]. Obige Aktionen und illegale Wahlkampfspenden haben zwar „nur" indirekten Effekt auf das Wahlverhalten der Bürger, im US-Wahlkampf zeigte sich aber mit dem Skandal um „Cambridge Analytica" erstmals eine neue Qualität der Wahlmanipulation. Das Wahlkampfteam von Trump um den erzkonservativen Steve Bannon verband sich mit Robert Mercer, Breitbart News und dem auf Datenanalyse spezialisierten Unternehmen Cambridge Analytica sowie Facebook. Das Ziel war, unter Ausschluss der Öffentlichkeit und vor allem der demokratischen Partei unentschlossene Wähler auszumachen, deren Profil zu erstellen – das bei Informatikern und Marketingexperten hochgeschätzte „Profiling" – um sie dann gezielt medial zu beeinflussen. Cambridge Analytica nutzte dabei die Facebook-Daten von Millionen Nutzern, filterte außerdem leicht zugängliche persönliche Daten aus dem Internet und glich sie („Similarity Join") mit gekauften Daten von Banken, Kreditkartenunternehmen und Google und Twitter ab. Wähler, die der US National Rifle Association (NRA) angehörten, bekamen Tweets mit dem Hinweis, Clinton wolle ihnen im Gegensatz zu Trump nach der Wahl die Waffen abnehmen. Bis heute nicht vollständig aufgeklärt ist der Einfluss Russlands auf die US-Wahlen von 2016 durch medial gestreute Falschmeldungen über beide Kandidaten zugunsten von Trump und Angriffe auf das digitale Wahlsystem der USA [1].

Die Wahl selbst erscheint weniger gefährdet zu sein hinsichtlich von Wahlfälschungen und Datenmanipulationen, erfordern freie, faire und geheime Wahlen doch, dass die Wahllokale wahlplakatfrei sind, für alle Wähler und Wählerinnen gleichermaßen freier Zugang zum Wahllokal und zu den Urnen besteht und in den Wahlkabinen das Wahlgeheimnis gewahrt bleibt. Dennoch bleibt reichlich Gelegenheit, Wähler indirekt durch Drohungen, Behinderungen oder Angriffe zu beeinflussen, wie beispielsweise durch die Taliban in Afghanistan oder durch regional Aufständische in Afrika geschehen. US-Journalisten haben Indizien auf „massive Manipulation am Wahltag" in Ohio bei den US-Wahlen 2004 gefunden [7]. „So seien Hunderttausende, fast ausnahmslos als Demokraten registrierte Wähler, an den Urnen zurückgewiesen worden, weil sie angeblich im falschen Wahllokal waren, nicht auf den Wahllisten standen oder angeblich nicht wahlberechtigt waren. In von Demokraten dominierten Wahlbezirken seien außerdem zu wenige Wahllokale mit zu wenigen Helfern und Wahlmaschinen geöffnet gewesen" [7].

Am häufigsten mit Wahlbetrug assoziiert wird die Stimmenauszählung. Das gilt wohl gleichermaßen für die klassische manuelle wie maschinelle Auszählung. Bei den abgegebenen Stimmen ist noch zwischen Brief- und Wahllokalwählern zu unterscheiden. Drei Zahlen stehen dabei im Fokus: Die Anzahl der Wahlberechtigten, die der gültigen Stimmen und die absoluten Stimmenanteile, die ein Kandidat oder eine Partei erzielt. Wie wir noch sehen werden, hängen diese Größen in interessanter Weise miteinander zusammen. So ergaben die US-Wahlen im November 2018, dass bei der Wahl über einen Senatssitz und den Gouverneursposten in Florida die beiden Kandidaten mit nur einem halben Prozentpunkt so dicht beieinanderlagen, dass eine maschinelle Neuauszählung gemäß dem Wahlgesetz in Florida notwendig wurde [4]. Zweifel an US-Wahlmaschinen kamen bereits 2004 auf. Die Wahlcomputer, „die vom engen Bush-Freund Mark O'Dell produziert wurden, sollen manipuliert worden sein. Der Wähler erhält an diesen Computern keinen Papierausdruck seiner Abstimmung. O'Dells Versicherung,

dass es unmöglich sei, seine Wahlmaschinen zu manipulieren, wurde von Informatik-Professor A. Rubin postwendend widerlegt“ [7]. Bei den Parlamentswahlen 2011 in Russland wurden Wahlmanipulationen in erheblichem Umfang aufgedeckt und publik gemacht. Dazu trugen D. Kobak, S. Shpilkin und M.S. Pshenichnikov mit der Studie „Statistical fingerprints of electoral fraud?“ [6] wesentlich bei. Trägt man in einer Häufigkeitsverteilung für eine Partei wie beispielsweise Putins Partei „Einiges Russland“, auf der Abszisse zweihundert disjunkte Prozentintervalle der Länge 0,5 zwischen 0 % und 100 % ab, und auf der Ordinate die jeweils zugehörige Anzahl der Wahllokale, so müsste sich bei freier, unabhängiger Wahl und rund 95000 Wahllokalen in guter Näherung eine Glockenform – auch Gaußkurve genannt – ergeben, siehe Abb. 1.

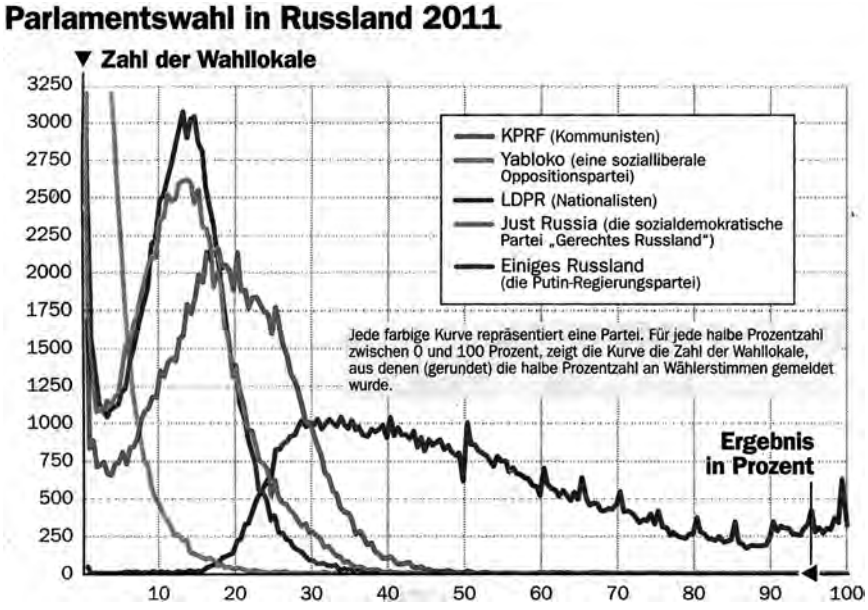


***Abb. 1 Parlamentswahl Russland 2011 [10]***

Dies lässt sich mittels eines zentralen Grenzwertsatzes der Stochastik auch theoretisch begründen. Vergleichsweise weist die Häufigkeitsverteilung der Zweitstimmen für CDU bzw. SPD bei der Bundestagswahl 2009 sehr gut eine solche Glockenform auf [10]. Die darin sich widerspiegelnde Homogenität der Wählerschaft einer Partei ist nicht zwangsläufig überall gegeben. So ist die Zweitstimmenverteilung der Partei „Die Linke“ bei der Bundestagswahl 2009 zweigipflig und erklärt sich gut aus dem unterschiedlichen Wählerverhalten in Ost- und Westdeutschland. Die Verteilung von Putins Partei dagegen ist rechtsschief, d. h. hat ab etwa 35 % ein ungewöhnlich weit ausgezogenes Auslaufstück nach rechts, also in Richtung 100 %. Wie bei einem Krokodilschwanz zeigen sich Zacken oder Spitzen bei den glatten Prozentwerten. Verdächtig sind die glatten oder runden Stimmenanteile, die überwiegend aus Wahllokalen mit hoher Wahlbeteiligung stammen. Die russischen Wissenschaftler Kobak et al. fanden weiterhin heraus, dass sich 2011 in vielen Wahllokalen das Ergebnis für Putins Partei allein anhand der Wahlbeteiligung verdächtig genau berechnen ließ. Statt Wahlzettel auszuzählen, wurde in jenen Wahllokalen der Ansatz Stimmenanteil % $= p \times$ Wahlbeteiligung % benutzt. Er beruht auf simpler Korrelation zwischen Stimmenanteil und Wahlbeteiligung: Der Stimmenanteil ist desto höher, je größer die Wahlbeteiligung ist. Hinzu kommt als wichtiger Einflussfaktor die menschliche Schwäche, ganze oder runde Zahlen bei Zahlenangaben zu bevorzugen. In den Russlandwahlen 2011 führten Korrelation und Vorliebe zu runden Zahlen zu den obigen „Häufigkeitsspitzen“. Halten wir fest, Mehrgipfligkeit, Korrelation und Glattheit sind für sich genommen kein Indiz für Wahlbetrug und induzierte Datenmanipulation oder gar -fabrikation. Zusammengenommen und mit der Erfahrung aus Wahlen im In- und Ausland erweisen sie sich als wirksames Werkzeug in der Hand des geschulten Datenanalytikers, gefälschte Wahlen als solche zu identifizieren. So gesehen verwundert es nicht, dass die russische Regierung aus dem Wahldebakel von 2011 gelernt hat. Die Stadtwahlen von Moskau zeigen ein Jahr später keine derartigen Auffälligkeiten mehr [6]. Gleiches gilt wohl für die russlandweiten Wahlen im Jahr 2018.

Gibt es so etwas wie eine Moral von der Geschichte? Die Moskauer Bevölkerung gab eine weise Antwort im Dezember 2011, als der Verdacht auf Wahlfälschung erstmals aufkam und belegbar wurde. Verantwortlich für deren korrekte Durchführung war der Wahlleiter Wladimir Tschurow. „Tschurow glauben wir nicht!“ und „Wir glauben Gauß!“ stand auf in Moskau gezeigten Plakaten [10]. Es lebe die Gaußsche Glockenkurve.

Es lebe die Wahrheit!

## Literatur


1. Ammann B (2016) Russische Wahlhilfe für Trump: Echo des Kalten Krieges. Neue Zürcher Zeitung, 26.11.2016
2. Dassler S (2018) Verwaltungsgericht erklärt Wahl für ungültig. Der Tagesspiegel, 28.8.2018, S. 11
3. Dettler M, Röbel S (2017) Das Ehrenwort. Spiegel Nr. 49/2017
4. dpa (2018) Florida zählt neu aus. Der Tagesspiegel, Nr. 23648, 12.12.2018, S. 4
5. Huchon T (2006) Fake America great again. Wie Facebook und Co. die Demokratie gefährden. ARTE, 9.10.2018, 20:15–21:15 Uhr
6. Kobak D, Shpilkin S, Pshenichnikov MS (2016) Statistical fingerprints of electoral fraud? Significance 13(4):20–23
7. o. V. (2006) Behielt Bush durch Wahlbetrug die Macht? Hamburger Abendblatt, 6.6.2006
8. Schäuble J (2018) Wo werden die Midterms entschieden? Der Tagesspiegel, 4.11.2018, S. 4
9. Tarran B (2018) Gerrymandering, redistricting and statistics: A Q&A with Vince Barabba. Significance 15(5):39
10. Ziegler GM (2013) Wir glauben Gauß. Der Tagesspiegel, 21.9.2013, S. 31

**Gesellschaft für Informatik e.V. (GI)**, Wissenschaftszentrum, Ahrstraße 45, 53175 Bonn, Tel.: 0228/302-145, Fax 0228/302-167,
e-mail: gs@gi.de, Server: http://www.gi.de
**Geschäftsführung:** Cornelia Winter, e-mail: cornelia.winter@gi.de, Tel.: -145, Daniel Krupka, e-mail: daniel.krupka@gi.de, Tel.: 030-7261566-15
**Sekretariat:** Kerstin Schimmel, e-mail: kerstin.schimmel@gi.de, Tel.: -145
**Mitgliederwerbung:** Ludger Porada, e-mail: ludger.porada@gi.de, Tel.: -146
**Finanzen/Buchhaltung:** Elena Kerkmann, e-mail: elena.kerkmann@gi.de, Tel.: -153, Swetlana Ruppel, e-mail: swetlana.ruppel@gi.de, Tel.: -152
**Mitgliederverwaltung:** Tanja Diegeler, e-mail: tanja.diegeler@gi.de, Tel.: -149, Andrea Mertins, e-mail andrea.mertins@gi.de, Tel.: -151
**ITK:** Viktor Schröder, e-mail: viktor.schroeder@gi.de, Tel.: -156, Karl-Heinz Künkel, e-mail: karl-heinz.kuenkel@gi.de, Tel.: -143

# *Aus der Geschäftsstelle*

## Ordentliche Mitgliederversammlung der GI: Termin vormerken

Am Dienstag, dem 24. September 2019 findet ab 17:30 Uhr die Ordentliche Mitgliederversammlung der Gesellschaft für Informatik an der Universität Kassel statt. Einladung und Tagesordnung werden im nächsten Informatik Spektrum und auf der GI-Webseite unter https://gi.de/aktuelles/ veröffentlicht.

## 50 Jahre GI

In diesem Jahr wird die GI 50! Am 16. September 1969 fanden sich in Bonn einige Professoren zusammen und beschlossen, eine wissenschaftliche Fachgesellschaft zu gründen. Als erster Vorsitzender fungiert Günter Hotz. Das Jubiläum werden wir in diesem Jahr feiern: mit einem Festakt am 16. September in Berlin und auf der INFORMATIK 2019 in Kassel. Informationen zu weiteren Veranstaltungen finden Sie sukzessive unter https://50jahre.gi.de/.

## GI-Junior-Fellows 2019 gesucht!

Herausragende Jungtalente verdienen es, besonders gefördert zu werden. Denn sie sind unsere Zukunft: nicht nur für Wissenschaft und Berufsfeld, sondern für unsere gesamte Gesellschaft. 2013 hat die Gesellschaft für Informatik e. V. (GI) deshalb das GI-Junior-Fellowship begründet, um junge, engagierte Talente ideell wie finanziell zu unterstützen. Für das Jahr 2019 suchen wir nun herausragende junge Informatikerinnen und Informatiker, die über den Tellerrand hinausblicken, sich für ihr Thema und auch über dessen Grenzen hinweg engagieren und interessieren und bereit sind, mit anderen GI-Junior-Fellows als Netzwerk innerhalb und außerhalb der GI zu agieren. Wir freuen uns auf spannende Köpfe! Details zu Kriterien und Procedere finden Sie unter https://gi.de/ueber-uns/personen/junior-fellows/. Nominierungsende ist der 15. Mai 2019, die persönliche Vorstellung findet auf Einladung am 12. Juli 2019 in Bonn statt.

## Wir danken unseren Fördermitgliedern

| | |
|---|---|
| Accso - Accelerated Solutions GmbH | Darmstadt |
| acocon GmbH | Bielefeld |
| ADB Solutions GmbH | Bindlach |
| Advanced Dynamics GmbH | Köln |
| AKAD Bildungsgesellschaft mbH | Stuttgart |
| AKDB - Anstalt für Kommunale Datenverarbeitung in Bayern | München |
| Alfred-Wegener-Institut | Bremerhaven |
| AlphaCarina Software GmbH | Grattersdorf |
| A · S Technology Consulting GmbH | Frankfurt am Main |
| Aschert & Bohrmann GmbH | Köln |
| ask-Innovative Visualisierungslösungen GmbH | Darmstadt |
| Bayer Business Services GmbH | Leverkusen |
| BITBW | Stuttgart |
| BSSM Becker Software System Management GmbH | Rossdorf |
| Berufsfachschule für IT-Berufe | Plattling |
| BESIT e. V. | Nürnberg |
| best-practice innovations GmbH | Köln |
| Beuth Hochschule für Technik Berlin | Berlin |
| Bildungszentrum für informationsverarbeitende Berufe e. V. | Paderborn |
| bluecue consulting GmbH & Co.KG | Bielefeld |
| Boldly Go Industries GmbH | Frankfurt am Main |
| Büren & Partner | Nürnberg |
| Bundesamt für Sicherheit in der Informationstechnik | Bonn |
| Bundesverband IT-Mittelstand e. V. (BITMi) | Aachen |
| Capgemini Deutschland GmbH | München |
| Carl-Cranz-Gesellschaft e. V. | Wessling/Obb. |
| CAST e. V. | Darmstadt |
| ckc ag | Braunschweig |
| CODE University of Applied Sciences | Berlin |
| Cognizant Technology Solutions GmbH | München |
| CommitWork GmbH | Dortmund |
| con terra GmbH | Münster |
| creatale GmbH | Ludwigsburg |
| Dataport | Altenholz |
| Der Hessische Datenschutzbeauftragte | Wiesbaden |
| DESY Deutsches Elektronen Synchroton | Hamburg |
| Deutsches Forschungszentrum für künstliche Intelligenz GmbH | Kaiserslautern |
| DHBW Heidenheim | Heidenheim |
| DLGI Dienstleistungsgesellschaft für Informatik mbH | Bonn |
| Duale Hochschule Baden-Württemberg | Stuttgart |
| Duale Hochschule Baden-Württemberg | Karlsruhe |
| Duale Hochschule Gera-Eisenach | Gera |
| edv-anwendungsberatung zühlke & bieker gmbH | Recklinghausen |
| eifel-online GmbH | Mechernich |
| Euroforum Deutschland GmbH | Düsseldorf |
| Fachbereich DCSM | Wiesbaden |
| Fachhochschule Aachen | Aachen |
| Fachhochschule für die Wirtschaft Hannover | Hannover |
| Fachhochschule Dortmund | Dortmund |
| Fachhochschule Münster | Münster |
| Fachhochschule Schmalkalden | Schmalkalden |
| Fachhochschule Wedel | Wedel |
| FINCON Unternehmensberatung GmbH | Hamburg |
| FIZ Karlsruhe GmbH | Eggenstein-Leopoldshafen |
| Forschungszentrum Informatik | Karlsruhe |
| Forschungszentrum Jülich GmbH | Jülich |
| Fraunhofer-Institut für offene Kommunikationssysteme FOKUS | Berlin |
| Fraunhofer-Gesellschaft e. V. | München |
| Fraunhofer-Gesellschaft zur Förderung der angewandten Forschung e. V. | Wachtberg |
| Fraunhofer IESE | Kaiserslautern |
| Fraunhofer ESK | München |
| Fraunhofer Institut IAO | Stuttgart |
| Fraunhofer-Institut-IPA | Stuttgart |
| Friedrich-Schiller-Universität | Jena |
| Geib IT GmbH | Saarwellingen |
| G.E.M. Team Solutions GbR | Neustadt |
| Generali Deutschland Informatik Services GmbH | Aachen |
| genua gmbh | Kirchheim |
| Georg-August-Universität Göttingen | Göttingen |
| Georg-Simon-Ohm-Berufskolleg der Stadt Köln | Köln |
| Gesamtschule Bad Lippspringe Schlangen | Bad Lippspringe |
| Gesellschaft der Förderer u. Freunde der HS Würzburg-Schweinfurt e. V. | Würzburg |
| Gesellschaft für Datenschutz und Datensicherheit e. V. | Bonn |
| GESIS Gesellschaft für Informationssysteme mbH | Salzgitter |
| GESIS - Leibniz-Institut für Sozialwissenschaften | Köln |
| GFU Cyrus AG | Köln |
| Goerigk Informationstechnologie GmbH | Bad Nauheim |
| Gottlieb-Daimler-Schule 2 | Böblingen |
| Graf-Stauffenberg-Gymnasium | Flörsheim a. M. |
| GWDG - Gesellschaft für wissenschaftliche Datenverarbeitung mbH | Göttingen |
| Hasso-Plattner-Institut für SW-Systemtechnik GmbH | Potsdam |
| HAW Hochschule Amberg-Weiden | Amberg |
| Heinrich-Heine-Gymnasium | Dortmund |
| Helmholtz-Zentrum Berlin für Materialien und Energie GmbH | Berlin |
| Helmholtz-Zentrum Geesthacht | Geesthacht |
| Helmholtz-Zentrum Potsdam | Potsdam |
| Helmut Schmidt Universität | Hamburg |
| Hessische Zentrale für Datenverarbeitung | Wiesbaden |
| HIS Hochschul-Informationssysteme eG | Hannover |
| HMS Analytical Software GmbH | Heidelberg |
| HNF Heinz Nixdorf MuseumsForum GmbH | Paderborn |
| Hochschule Aalen | Aalen |
| Hochschule Albstadt-Sigmaringen | Albstadt |
| Hochschule Anhalt | Köthen |
| Hochschule Bremerhaven | Bremerhaven |
| Hochschule Esslingen | Esslingen |
| Hochschule Fulda | Fulda |
| Hochschule Furtwangen | Villingen-Schwenningen |
| Hochschule Hannover | Hannover |
| Hochschule Karlsruhe | Karlsruhe |
| Hochschule Niederrhein | Krefeld |
| Hochschule Ruhr West | Bottrop |
| Hochschule Weserbergland | Hameln |
| Hochschule Wismar | Wismar |
| HPA Hamburg Port Authority AöR | Hamburg |
| Humboldt Universität Berlin | Berlin |
| IBM Deutschland GmbH | Ehningen |
| ICS AG | Stuttgart |
| imbus AG | Möhrendorf |
| Immanuel-Kant-Gymnasium Bad Oeynhausen | Bad Oeynhausen |
| Infodas Gesellschaft für Systementwicklung und Informationsverarbeitung mbH | Köln |
| Information und Technik NRW | Düsseldorf |
| Information Works GmbH | Köln |

https://doi.org/10.1007/s00287-019-01151-8

| | |
|---|---|
| Initiative D21 e. V. | Berlin |
| innoQ Deutschland GmbH | Monheim am Rhein |
| innovas GmbH | Hamburg |
| Institut für Ökonometrie | Bonn |
| Institutsverbund Informatik | Stuttgart |
| IT-Matic GmbH | Mannheim |
| it@M zentraler Dienstleister für Informations- und Telekommunikationstechnik der Landeshauptstadt München | München |
| IT-Forum Rhein-Neckar e. V. | Ludwigshafen |
| IT Service Omikron GmbH (ITSO) | Berlin |
| ITech Progress GmbH | Ludwigshafen |
| iteratec GmbH | Unterhaching |
| IVU Traffic Technologies AG | Berlin |
| Jakubowski Software GmbH | Korschenbroich |
| jambit GmbH | München |
| Karlsruher Institut für Technologie – KIT | Eggenstein-Leopoldshafen |
| KRIWAN Testzentrum GmbH | Forchtenberg |
| LinnB GmbH | Hamburg |
| LIQUID NEWSROOM | Bad Reichenhall |
| Login-IT GmbH & Co. KG | Kirchheimbolanden |
| ma design GmbH & Co. KG | Kiel |
| Max-Planck-Institut für Informatik | Saarbrücken |
| mehrwert intermediale kommunikation GmbH | Köln |
| MicroConsult Microelectronics Consulting & Training GmbH | München |
| MOBOTIX AG | Langmeil |
| msg systems ag | Ismaning |
| Netzkunst24 Design- und Internetagentur GmbH | Lüneburg |
| Neubert Consulting GmbH | Fürth |
| NORDAKADEMIE | Elmshorn |
| Normfall GmbH | München |
| NovaTec Consulting GmbH | Leinfelden-Echterdingen |
| OFFIS e. V. | Oldenburg |
| Opitz Müller und Partner GbR | Berlin |
| OSSENO Software GmbH | Kaiserslautern |
| Ostfalia Hochschule für angewandte Wissenschaften | Wolfenbüttel |
| Otto-Friedrich-Universität Bamberg | Bamberg |
| Otto-von-Guericke-Universität Magdeburg | Magdeburg |
| Parawana Consulting GmbH | Hof |
| Patzschke + Rasp Software GmbH | Wiesbaden |
| perbit Software GmbH | Altenberge |
| PPI AG | Hamburg |
| Private FernFachhochschule Darmstadt | Darmstadt |
| PROMATIS software GmbH | Ettlingen |
| QAware GmbH | München |
| Qualmity | Berlin |
| Rohde & Schwarz GmbH & Co. KG | München |
| Robert Bosch GmbH | Renningen |
| RWTH Aachen (IMA) | Aachen |
| SD Software-Design GmbH | Bad Krozingen |
| SAP Deutschland SE & Co. KG | Walldorf |
| Schulverein Anna-Schmidt e. V. | Frankfurt |
| Seeliger & Co. GmbH | Eichenau |
| SHD Einzelhandelssoftware GmbH | Andernach |
| SielTMCi Siebenhofer.Consulting e.U. | Steyr |
| Siemens AG | München |
| SIZ GmbH | Bonn |
| SMO GmbH | Karlsruhe |
| SoftServe UGE GmbH | Frankfurt |
| Software AG | Darmstadt |
| Softwarekontor GmbH | Ludwigshafen |
| Springer-Verlag GmbH | Heidelberg |
| SQLan Gesellschaft für Informations- und Netzwerksysteme mbH | Wiesbaden |
| SRH Hochschule Heidelberg | Heidelberg |
| SYRACOM AG | Wiesbaden |
| SYSLAB.COM GmbH | München |
| Talanx Systeme AG | Hannover |
| TBiQ Managed Testing Services GmbH | Bochholt |
| Technische Akademie Esslingen e. V. | Ostfildern |
| Technische Universität Berlin | Berlin |
| Technische Universität Chemnitz | Chemnitz |
| Technische Universität Dresden | Dresden |
| Technische Universität München | Garching |
| TH Brandenburg University of Applied Sciences | Brandenburg |
| TH Mittelhessen | Gießen |
| The MathWorks GmbH | Ismaning |
| Transaction Software GmbH | München |
| TU Berlin | Berlin |
| TU Dortmund | Dortmund |
| TÜV Informationstechnik GmbH | Essen |
| Universität Bremen | Bremen |
| Universität der Bundeswehr München | Neubiberg |
| Universität des Saarlandes | Saarbrücken |
| Universität Duisburg-Essen | Essen |
| Universität Erlangen-Nürnberg | Erlangen |
| Uni HH/MIN Fakultät/FB Informatik | Hamburg |
| Universität Hamburg | Hamburg |
| Universität Kassel | Kassel |
| Universität Koblenz-Landau | Koblenz |
| Universität Leipzig | Leipzig |
| Universität Mannheim | Mannheim |
| Universität Osnabrück | Osnabrück |
| Universität Paderborn | Paderborn |
| Universität Rostock | Rostock |
| Universitätsbibliothek Bamberg | Bamberg |
| Universität Tübingen | Tübingen |
| Verband der Vereine Creditreform e. V. | Neuss |
| VOSDAV GmbH | Berlin |
| Walter de Gruyter GmbH | München |
| WWU Münster | Münster |
| Werum IT Solutions GmbH | Lüneburg |
| Westfälische Hochschule Gelsenkirchen | Gelsenkirchen |
| WidasConcepts GmbH | Wimsheim |
| zisa consulting GmbH | Berlin |

# *Presse- und Öffentlichkeitsarbeit der GI*

## Mehr Anstrengung zur Absicherung sicherheitskritischer Infrastrukturen notwendig (21.12.2018)

Die Gesellschaft für Informatik nimmt die aktuelle Berichterstattung rund um schlecht gesicherte Wasser- und Kläranlagen zum Anlass, um auf die Notwendigkeit der Informationssicherheit bei Versorgungsunternehmen aufmerksam zu machen. Das seit 2015 gültige und von der GI ausdrücklich begrüßte „Gesetz zur Erhöhung der Sicherheit informationstechnischer Systeme“ (IT-Sicherheitsgesetz) betrifft derzeit eine absolute Minderheit von Versorgungsanlagen in Deutschland, da in der „Verordnung zur Bestimmung Kritischer Infrastrukturen nach dem BSI-Gesetz“ (BSI-Kritisverordnung) ein Schwellwert von 500.000 versorgten Einwohnern vorgesehen ist.

GI Junior-Fellow Tim Philipp Schäfers, der die Sicherheitsmängel mit aufgedeckt hat, sieht darin eine Schwachstelle der aktuellen Gesetzgebung: „Anlagen, die weniger als eine halbe Millionen Menschen versorgen, gelten nicht als kritische Infrastrukturen und müssen insoweit nicht die gesetzlichen Auflagen des IT-Sicherheitsgesetzes umsetzen, obwohl diese mitunter ebenfalls essenzielle Leistungen für die Gesellschaft erbringen. Da es bei kleineren Anbietern bisher in der Regel keine staatlich auferlegten Kontrollen gibt, haben manche Betreiber nicht einmal absolut grundlegende Sicherheitsmaßnahmen ergriffen.“ Die GI fordert aus diesem Grund eine Senkung der Schwellwerte und stärkere Kontrollen in Bezug auf die Einhaltung gesetzlicher Vorgaben durch Behörden.

Prof. Dr. Hannes Federrath, Präsident der Gesellschaft für Informatik und IT-Sicherheitsexperte kritisiert in diesem Zusammenhang auch die allgemeine Ausrichtung der aktuellen Cybersicherheitspolitik Deutschlands: „Mit seiner aktuellen Cybersicherheitspolitik – etwa dem Aufstellen des Kommando Cyber- und Informationsraum der Bundeswehr, dem Ankauf von IT-Sicherheitslücken für Nachrichtendienste und der Schaffung der Zentrale Stelle für Informationstechnik im Sicherheitsbereich (ZITiS) – betreibt die Bundesregierung eine überwiegend offensive Cybersicherheitspolitik. Diese Maßnahmen und insbesondere das Offenhalten von IT-Sicherheitslücken, etwa in weitverbreiteten Betriebssystemen, schwächen die IT-Sicherheit massiv. Während einzelne Behörden, wie etwa das Bundesamt für Sicherheit in der Informationstechnik (BSI), zu einer Stärkung der IT-Sicherheit beitragen, wird diese Leistung durch die Arbeit anderer Behörden untergraben.“

Die GI fordert deshalb eine Abkehr von der derzeitig überwiegend offensiven Cybersicherheitspolitik und der Teilnahme am „digitalem Wettrüsten“. Stattdessen sei eine Hinwendung zu einer verantwortlich-defensiven Cybersicherheitspolitik notwendig.

Bausteine einer neuen IT-Sicherheitsstrategie sollten der Aufbau eines unabhängigen Com-

puter Emergency Response Teams (CERT), die Etablierung eines behördenübergreifenden Prozesses zum Umgang mit IT-Sicherheitslücken und das Vorantreiben internationaler Standards sein. Durch entsprechende Maßnahmen könnte letztlich eine Verschwendung von Ressourcen vermieden und eine nachhaltige Basis für den sicheren Betrieb von kritischen Infrastrukturen geschaffen werden.

Die kompletten GI-Pressemitteilungen finden Sie unter https://gi.de/ueber-uns/presse/.

## *Tagungsankündigungen*

### Jahrestagung der Fachgruppe Frauen und Informatik

Vom 10.–12. Mai 2019 findet an der Hochschule Bremerhaven die Jahrestagung der Fachgruppe zum Thema „Informatik und Nachhaltigkeit" statt. Wissenschaftlerinnen aus dem Alfred-Wegener-Institut werden über die IT-Unterstützung der Arktisexpedition MOSAIC und über die Tsunami-Modellierung für das indonesische Frühwarnsystem berichten. Besuch und Vortrag im Klimahaus Bremerhaven runden das Programm ab.

Außerdem wird zum ersten Mal der Preis der Fachgruppe für die gelungene Abschlussarbeit einer Informatik-Studentin verliehen. Weitere Informationen zur Fachgruppe finden Sie unter www.frauen-informatik.de.

### SCIENCE – PEACE – SECURITY '19

Vom 26.–27.9.2019 findet im Lichtenberg-Haus in Darmstadt die Konferenz Science – Peace – Security statt. Die Konferenz hat es sich zum Ziel gesetzt, gegenwärtige und zukünftige Herausforderungen für Frieden und Sicherheit zu benennen. Das schließt naturwissenschaftlich-technische sowie interdisziplinäre Beiträge ein, mit Schwerpunkten bei internationaler Sicherheit, Friedensschaffung sowie transparenz- und vertrauensbildenden Maßnahmen, Rüstungskontrolle, Abrüstung und Konfliktmanagement und einem besonderen Fokus auf Cyber War und Cyber Peace. Weitere Informationen finden Sie unter https://sps.peasec.de. (Aus: GI Fachgruppe Mensch-Maschine-Interaktion in sicherheitskritischen Systemen)

### INFORMATIK 2019 in Kassel: Tracks und Einreichungen

Die Jahrestagung INFORMATIK 2019 findet anlässlich der GI-Gründung im Jahr 1969 vom 23.–26. September 2019 an der Universität Kassel statt. Dort wird der 50. Geburtstag mit einem attraktiven Programm für und mit „Jung und Alt" aus Wissenschaft und Praxis gewürdigt.

Das Leitthema der Tagung lautet „50 Jahre Gesellschaft für Informatik – Informatik für Gesellschaft". Es reflektiert zum einen ein zentrales Anliegen der GI, zum anderen einen Schwerpunkt der Informatik an der Universität Kassel. Gegenüber früheren Jahrestagungen ändert sich die Tagungsstruktur: Themen, die besondere Aufmerksamkeit in Wissenschaft, Praxis und Gesellschaft erfahren, werden im Rahmen vorgegebener thematischer „Tracks" behandelt, die jeweils von ausgewiesenen Fachleuten aus der Informatik-Community gestaltet werden. Die besten Beiträge der Tracks werden zur Veröffentlichung in passenden Zeitschriften weitergeleitet werden.

Die Aufrufe zur Einreichung von Beiträgen und weitere Informationen finden Sie unter www.informatik2019.de. Die Einreichungsfrist endet am 19. April 2019.

Die INFORMATIK 2019 findet zusammen mit der 33. Jahrestagung des GI-Fachausschusses Umweltinformatik (EnviroInfo 2019), der 42. Jahrestagung des GI-Fachbereichs Künstliche Intelligenz (KI 2019) und der Studierendenkonferenz SKILL 2019 statt. Zusätzlich ergänzen mehrere Workshops das wissenschaftliche Programm. Die Partnertagungen und Workshops veröffentlichen eigene Aufrufe zur Einreichung von Beiträgen.

## *Tagungsberichte*

### Fachgruppe für Informatik und soziale Entwicklung fördert IT-Wettbewerb in Kabul

Die GI-Fachgruppe Informatik und soziale Entwicklung der GI freut sich, dieses Jahr die zum drittem Mal vom Zentrum für internationale und interkulturelle Kommunikation (ZiiK) der TU Berlin veranstaltete IT-Ausstellung Kabul zu unterstützen. Vom 15. bis zum 16.12.2018 stellten Informatik-Studierende von zwölf afghanischen Universitäten 35 IT-Projekte in Gegenwart des Ministers für höhere Bildung und des Direktors des ZiiK der TU Berlin und Gründers der Fachgruppe Informatik und soziale Entwicklung, Dr. Nazir Peroz, und vielen weiteren Gästen vor.

Nach der Vorstellung der einzelnen Projekte wurden die besten von einer Jury aus Vertretern aus Wissenschaft, Wirtschaft und Politik ausgewählt und prämiert. Darunter waren gleich zwei Projekte von Studierenden der Polytechnischen Universität Kabul: Eine mobile

*Abb. 1 Der Minister für Bildung lässt sich die Projekte erklären. Quelle: ITCC Afghanistan*


*Abb. 2 Nazir Peroz von der Fachgruppe Informatik und soziale Entwicklung übergibt die Preise. Quelle: ITCC Afghanistan*

Taxi-App für Afghanistan und ein intelligenter Helm für Minenarbeiter, der neben zahlreichen anderen Funktionen auch gefährliche Gaskonzentrationen erkennen kann. Studierende der Universität Nangarhar überzeugten die Jury mit einem auf Arduino basierten smarten Blindenstock. Als Preis wurden von der Fachgruppe drei Notebooks gestiftet, die am ersten Tag der IT-Konferenz am 17. Dezember den Preisträgern überreicht wurden.

## *LNI-Neuerscheinungen*

Sämtliche LNI-Bände finden Sie in der Digitalen Bibliothek der GI unter https://dl.gi.de/handle/20.500.12116/21.

LNI P-286 Projektmanagement und Vorgehensmodelle 2018 (PVM2018), Mikuzs, Martin; Volland, Alexander; Engstler, Martin; Fazal-Baqaie. Masud; Hanser, Eckhart; Linssen, Oliver (Hg.)

LNI P-285 Workshops der INFORMATIK 2018 – Architekturen, Prozesse, Sicherheit und Nachhaltigkeit, Czarnecki, Christian; Brockmann, Carsten; Sultanow, Eldar; Koschmider, Agnes; Selzer, Annika (Hg.)

LNI P-284 DeLFI 2018 – Die 16. E-Learning Fachtagung Informatik der Gesellschaft für Informatik e. V., Krömker, Detlef; Schroeder, Ulrik (Hg.)

LNI P-283 11. DFN-Forum Informationstechnologien, Müller, Paul; Neumair, Bernhard; Reiser, Helmut; Dreo Rodosek, Gabi (Hg.)

LNI P-282 BIOSIG 2018, Brömme, Arslan; Busch, Christoph; Dantcheva, Antitza; Rathgeb, Christian; Uhl, Andreas (Hg.)

# GI-Veranstaltungskalender

**16.05.2019 – Neu-Ulm**
14. Neu-Ulmer Test-Engineering-Day
**NU-TED**
https://www.nu-ted.de

**20.05.–23.05.2019 – Copenhagen/ Denmark**
15th Workshop on Dependability and Fault Tolerance (VERFE'19)
**VERFE 2019**
http://arcs2019.itec.kit.edu/downloads/VERFE.pdf

**23.05.–24.05.2019 – Potsdam**
Potsdamer Konferenz für Nationale CyberSicherheit
https://www.potsdamer-sicherheits-konferenz.de/konferenz.html

**18.06.–21.06.2019 – Rennes/France**
CARS 2019 Computer Assisted Radiology and Surgery – 33rd International Congress and Exhibition
**CARS 2019**
https://www.cars-int.org

**24.06.–26.06.2019 – Wolfsburg**
Innovations for Community Services
**I4CS**
http://www.i4cs-conference.org/

**26.06.–28.06.2019 – Seville/Spain**
22nd International Conference on Business Information Systems
**BIS 2019**
http://www.bis.ue.poznan.pl

**30.06.–03.07.2019 – Boston/USA**
11th International ACM Web Science Conference 2019
**WebSci19**
https://websci19.webscience.org/

**26.08.–30.08.2019 – Leipzig**
Digital Operating Room Summer School
**DORS 2019**
https://www.iccas.de/dors

**08.09.–11.09.2019 – Hamburg**
Tagung Mensch und Computer 2019
**MuC 2019**
http://muc2019.mensch-und-computer.de/

**16.09.–18.09.2019 – Dortmund**
Informatik und Schule
**INFOS 2019**
https://infos2019.cs.tu-dortmund.de/

**16.09.–19.09.2019 – Berlin**
Die 17. e-Learning Fachtagung Informatik
**DeLFI 2019**
http://delfi2019.de

**18.09.–20.09.2019 – Chemnitz**
Simulation in Produktion und Logistik
**ASIM 2019**
http://www.asim-fachtagung-spl.de/

**23.09.–26.09.2019 – Kassel**
INFORMATIK 2019: Jahrestagung der Gesellschaft für Informatik
**INFORMATIK 2019**
http://www.informatik2019.de

**26.09.–27.09.2019 – Darmstadt**
Science · Peace · Security '19
**SPS'19**
https://sps.peasec.de

**30.09.–02.10.2019 – Berlin**
Lernen. Wissen. Daten. Analysen.
**LWDA**
https://hu.berlin/lwda2019

**21.11.–22.11.2019 – Boppard**
40. Fachtagung Echtzeit „Autonome Systeme – 50 Jahre PEARL"
**Echtzeit 2019**
https://www.real-time.de/echtzeit.html

**09.12.–12.12.2019 – Frankfurt am Main**
IT-Tage 2019
http://www.it-tage.org

**06.09.–09.09.2020 – Magdeburg**
Tagung Mensch und Computer 2020
**MuC2020**
https://muc2020.mensch-und-computer.de

**29.09.–01.10.2020 – Karlsruhe**
INFORMATIK 2020: Jahrestagung der Gesellschaft für Informatik
**INFORMATIK 2020**
gs@gi.de