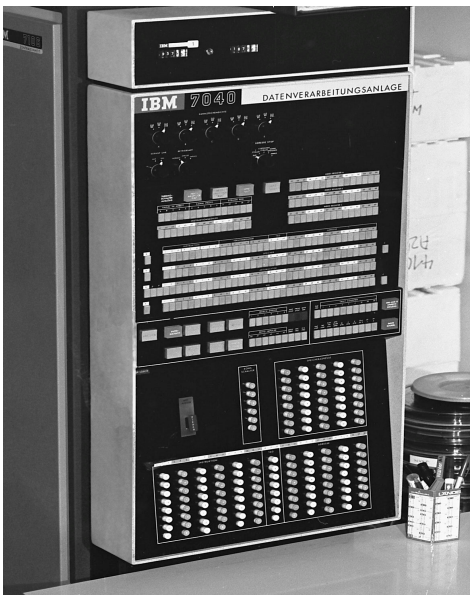
Back To The Future

Helmut Schauer
Universität Zürich
schauer@ifi.uzh.ch

40 Jahre Informatik-Studium und 25 Jahre Schulinformatik in Österreich sind ein willkommener Anlass für eine Zeitreise zurück zu den Anfängen. Ich bitte um Verständnis dafür, diesen Rückblick aus meiner persönlichen Sicht vornehmen zu dürfen. Neben einigen Seitenhieben auf die "heroic failures" der Pionierzeit möchte ich das Hauptaugenmerk auf die Beobachtung lenken, dass einige der damals vermittelten Informatikkompetenzen heute obsolet sind. Die daraus resultierende Frage, welche Informatikkonzepte auch in der Zukunft relevant sein werden, bilden den Schwerpunkt meiner Überlegungen.



Mein persönlicher Rückblick

Die Informatik zog mich erstmals während meines Studiums der Elektrotechnik an der damaligen Technischen Hochschule in Wien Mitte der 60er Jahre in ihren Bann und ließ mich seitdem bis heute nicht mehr los. Alles begann in den Vorlesungen über Schaltalgebra von Prof. Heinz Zemanek[1], dem österreichischen Computerpionier, die dieser reichlich mit Anekdoten über die Entwicklung des "Mailüfterls", des ersten volltransistorisierten Computers Europas, zu garnieren verstand. Gleichzeitig verbrachte ich jede freie Minute mit der Programmierung der damals neu installierten Grossrechenanlage IBM 7040 am Institut für Numerische Mathematik. Sie war gross im wahrsten Sinn des Wortes! Zu ihrer Aufstellung musste ein ganzer Hörsaal umgebaut, klimatisiert und mit einem doppelten Boden für die Kabelstränge versehen werden. Die nächste Rechenanlage gleicher Bauart befand sich damals in Darmstadt!

Der Arbeitsspeicher der 7040 hatte die gigantische Kapazität von 32 K Worten! Zum Vergleich: die Kapazität des in meinem Geschirrspüler heute eingebauten Speicherchips ist mehr als 5 mal so groß! Sicherlich übertreffen allein die in meinem Auto standardmäßig eingebauten Prozessoren die Rechnerleistung aller im Europa der 60er-Jahre installierten Computer -

[1] Heinz Zemanek (*1920) gründete das Wiener IBM-Labor und war für die Formale Definition der Programmiersprache PL/I mitverantwortlich.

von Handy, Navigationssystem und Kamera ganz zu schweigen!
Die 7040 arbeitete im sogenannten Batch-Betrieb: Programme wurden händisch auf Lochkarten gestanzt, und einmal eingegeben, dauerte es im Normalfall (nur!) 24 Stunden, bis die Ergebnisse ausgedruckt und abholbereit waren. Naturgemäß war das Benutzerverhalten von großer Umsicht geprägt, verlor man doch durch den kleinsten Tippfehler gleich einen ganzen Tag. Eine "Trial and Error"-Vorgehensweise war völlig undenkbar! Die "Kunst der Computernutzung" bestand einfach darin, die beschränkten Hardwareressourcen optimal zu nutzen. Unnötig ist zu erwähnen, dass die wahren Cracks in Assembler programmierten, denn nur so konnte man das Letzte aus der Anlage herausholen. Assemblerkenntnisse waren unumgänglich, um die Funktionsweise eines Computers zu verstehen. Davon waren zumindest wir alle überzeugt.

Die Hippies in San Francisco trugen Blumen im Haar, in Europa brannten die Unis und Bob Dylan sang "**The Times They Are A-Changin'** ".

Ich erwähne das alles, um den Zeitgeist widerzuspiegeln, der 1969 in Österreich vorherrschte als das Informatikstudium eingeführt worden ist, und ich mir kurz danach im Unterrichtsministerium kalte Füße holte, als ich im Auftrag der Studienkommission vorstellig wurde, um die Einführung eines Lehramtsstudiums Informatik zu beantragen. Der zuständige Ministerialbeamte lehnte das Ansuchen mit dem Verweis auf bereits genügend arbeitslose Lehrer schlichtweg ab. Um die Arbeitsmarktchancen der Schüler machte er sich keine Gedanken ... Tatsächlich wurde das Lehramtsstudium Informatik erst ein Viertel Jahrhundert später im Jahr 2000 eingeführt. Vermutlich hat dieses Versäumnis auch seine guten Seiten: Die damalige Kompetenz und der informatische Ereignishorizont der Lehrer wäre sicherlich auf Programmieren in Assembler, BASIC und ein wenig Hardwarekunde beschränkt gewesen. Er hätte für Schule und Schüler mehr Schaden als Nutzen angerichtet.

In den 60er-Jahren wurden vorwiegend Anwendungen aus bereits formalisierten naturwissenschaftlichen und administrativen Aufgaben mittels damaliger Großcomputer im Batch-Betrieb gelöst. Gleichzeitig begann die Regelungstechnik Computer zur Steuerung von Maschinen, Robotern und Raketen einzusetzen.[2] Die damaligen - im physischen Sinn- "großen" Computer wurden zuerst in spezialisierten Programmiersprachen wie FORTRAN für technische, und COBOL für kommerzielle Anwendungen programmiert. An den Universitäten wurden "algorithmische" Sprachen wie ALGOL 60 oder Lisp (für die ersten Gehversuche der Künstlichen Intelligenz) verwendet. Mitte der 60er-Jahre versuchte IBM mit der universellen Programmiersprache PL/I dem Babylonischen Sprachgewirr Einhalt zu gebieten und alle Aufgaben mit einer einzigen Sprache zu lösen. Ende der 60er-Jahre verfolgten die Universitäten mit ALGOL 68 ein ähnlich hochgestecktes Ziel.

In den 70er-Jahren standen Timesharing Systeme im Vordergrund, bei denen viele Benutzer parallel über Terminals an einem Rechner angeschlossen waren. Unix begann sich durchzusetzen, "small is beautiful" war die Devise. Pascal startete seinen Siegeszug und am Palo Alto Research Center von Xerox wurde mit der Programmiersprache Smalltalk der Grundstein der objektorientierten Programmierung gelegt.

[2] Dies führte 1962 auch zu einem der ersten grossen Fehlschläge des Computereinsatzes, dem Absturz der Mariner 1 Venus Sonde aufgrund eines Schreibfehlers (Punkt statt Komma) im FORTRAN Programm der dazu führte, dass die Rakete nicht mehr steuerbar war und von der Bodenstation aus zerstört werden musste. Der Schaden belief sich auf 18.5 Millionen U.S. Dollar.

Die 80er-Jahre standen im Zeichen des Personal Computers und der damit Hand in Hand gehenden Demokratisierung der Informatik (jedem Benutzer sein eigener Computer!). Im Jahr 1981 lancierte IBM den ersten PC, 1983 folgte Apple's Lisa (mit Maus) und 1984 Apple's klassischer Macintosh. Gleichzeitig eroberte Commodore's 8-bit Heimcomputer C64 mit seiner Spielkonsole die Herzen der Kinder.

Die 90er Jahre waren das Jahrzehnt der Vernetzung - Internet und World Wide Web erlaubten es, vom Schreibtisch aus die Welt zu erobern. Es entstanden Virtuelle Welten und mikroprozessor-gesteuerte "embedded sytems" bemächtigten sich vieler Gebrauchsgegenstände. Dank "ubiquitous computing" kamen auf einen Benutzer mehrere Prozessoren und "ICT" wurde (neben Rechnen. Lesen und Schreiben) zur unverzichtbaren 4. Kulturtechnik. Die objektorientierte Programmiersprache Java war zum richtigen Zeitpunkt verfügbar und verbreitete sich dank ihrer Eignung für portable und architekturneutrale Internetanwendungen rasant.

Das erste Jahrzehnt des 3. Jahrtausends - die sogenannten "Nullerjahre" - wird von so genannten „sozialen Netzwerken" dominiert. Wer sein persönliches Profil nicht in Facebook publiziert und seine Meinung nicht auf Twitter online stellt, läuft Gefahr von der Web-Community nicht wahrgenommen zu werden.

Heute entfallen auf jeden Menschen etwa 1 Milliarde Transistoren[3]. Jährlich werden so viele produziert, wie es Ameisen auf der Erde gibt und Gartner prognostiziert für 2010 einen Absatz von nahezu 400 Millionen Computern weltweit.

Fundamente und langlebige Aspekte der Informatik

Konzepte der Informatik

Vergleichen wir die Entwicklung der Informatik der letzten Jahrzehnte mit dem, was heute relevant ist, so zeigt sich, dass produktspezifisches Wissen (wie etwa über den IBM 7040 Assemblercode) eine geringe Halbwertszeit von wenigen Jahren aufweist. Selbst einst gesuchte Spezialisten für COBOL oder MS-DOS haben heute Schwierigkeiten bei der Jobsuche. Obwohl aktuelle ICT- Kenntnisse für viele Informatikanwendungen unumgänglich sind, sind diese so kurzlebig, dass ihr allgemeinbildender Stellenwert mehr als fraglich ist. Konzeptionelles Wissen hingegen hat die Jahrzehnte überdauert! Welche sind nun diese langlebigen Konzepte der Informatik? Meines Erachtens gehören dazu

- **Modellierung** und die damit Hand in Hand gehende **Abstraktion**
- **Notationsformen** (textuelle und graphische) mit der damit verbundenen Unterscheidung zwischen **syntaktischer** äußerer Form und **semantischer** Bedeutung einschließlich **Rekursion**
- **Strukturen** und **Relationen** wie sie bei **statischen** Zusammenhängen aber auch bei **dynamischen** Abläufen auftreten
- **Formalisierte Systeme** und ihre **Spezifikation**

Darüber hinaus möchte ich, rückblickend auf meine eigene Unterrichtserfahrung, exemplarisch einige Informatikaspekte illustrieren, die ich für „Dauerbrenner" halte und die vermut-

[3] Quelle: http://www.zehn.de/ eine-milliarde-transistoren-pro-kopf-178603-3 (im Jahr 2005 waren es noch knapp 100 Millionen). Im Vergleich dazu nehmen sich die 200 Millionen Insekten, die auf einen Erdenbewohner kommen, bereits mager aus …

lich auch noch in den kommenden Jahrzehnten für den Informatikunterricht relevant sein werden.

Terminologie

Nicht nur Programmiersprachen, sondern vor allem die natürliche Sprache, mit der wir Informatik unterrichten sowie die dabei verwendeten Fachbegriffe, verdienen unsere Beachtung. Umgangssprachlich werden zum Beispiel gerne die Begriffe “Ziffer” und “Zahl” missbräuchlich verwendet. Informatikern sollte völlig klar sein, dass sich Ziffern zu Zahlen ebenso verhalten wie Buchstaben zu Wörtern. Zahlen sind ebenso aus Ziffern zusammengesetzt wie Wörter aus Buchstaben. Selbstverständlich gibt es auch einstellige Zahlen die nur aus einer einzigen Ziffer bestehen. Wörter, die nur aus einem einzigen Buchstaben bestehen, sind hingegen selten. Ziffern sind Symbole, während mit Zahlen Berechnungen durchgeführt werden können. Eine Telefonnummer besteht somit aus Ziffern, das Kennzeichen eines Autos kann aus Ziffern und Buchstaben zusammengesetzt sein, Schuhnummern hingegen sind Zahlen ebenso wie die Tages-, Monats- und Jahresangaben eines Kalenderdatums oder die Nummern der Stunden am Ziffern(!)blatt einer Uhr. Es kann verlangt werden, dass Passwörter neben Buchstaben auch Ziffern (aber keinesfalls Zahlen) enthalten müssen. Aber wenn im Nationalrat über Budgetziffern diskutiert wird sind natürlich Budgetzahlen gemeint.

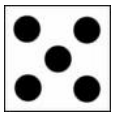
Vergebens haben wir in den 70er Jahren die abendländische Kultur gegen Amerikanismen zu verteidigen versucht. Wo sind die Zeiten, in denen Siemens noch Rechner (nicht “Computer”) baute und Zahlen “sedezimal” und nicht “hexadezimal” (welch grausige Vermischung von Griechisch und Latein!) verschlüsselte. Ich erinnere auch an die penible Unterscheidung zwischen “binären Codes” und dem “dualen Zahlensystem”, die zwar humanistische Bildung widerspiegelt, Amerikanern aber völlig unverständlich ist. Amüsant sind auch die Versuche, Fachbegriffe wie “Hardware”, “Software” oder “Browser” ins Deutsche zu übersetzen. Vielen ist vielleicht noch die Bezeichnung “Kellerspeicher” für einen “Stack” erinnerlich. In der Bundesrepublik kursierte sogar eine Pascal-Variante mit deutschsprachigen Schlüsselwörtern. Erst kürzlich überraschte mich die Bezeichnung “Heimseite” als deutsches Pendent zu “homepage”.

Inzwischen haben wir das Handtuch geworfen und akzeptieren Anglizismen wie “online”, “E-Mail” oder “Notebook” und das ist vermutlich gut so, unterstützt es doch internationalen Austausch und verhilft dem Englischen auf dem Weg zur “lingua franca” der Informatik. Zumeist sind die englischen Fachbegriffe auch “griffiger” als die deutschen. Kurioserweise gegenläufig setzt sich der im deutschsprachigen Raum verwendete Begriff “Handy” gegen das amerikanische “cell phone” und das britische “mobile phone” durch. Vermutlich, weil es Englisch klingt.

Sorgsam umgehen sollten wir jedoch mit der mit den Fachbegriffen verknüpften Semantik. So werden die Begriffe Daten und Information im alltäglichen Sprachgebrauch häufig synonym verwendet. In der Informatik jedoch haben sie jedoch unterschiedliche Bedeutungen.

Daten und Information

Daten können in unterschiedlichsten Formen (durch Bits und Bytes binär kodiert) repräsentiert und in diesen Repräsentationen gespeichert und übertragen werden.

5 V 0101   **fünf**

Bewusst wahrgenommen hingegen werden Daten zu Information! Daten können von Maschinen, Informationen von Menschen verarbeitet werden. Ob und in welchem Ausmaß wir aus Daten Informationen entnehmen, hängt allein von uns ab! Nachrichten in Zeitungen, Radio oder Fernsehen können die gleichen Nachrichten an Millionen Menschen verbreiten und dennoch kann jeder Einzelne aus den übermittelten Daten individuell unterschiedliche Informationen entnehmen! Der in den Naturwissenschaften des 20. Jahrhunderts vielstrapazierte "Beobachter" findet sich somit auch in der Informatik wieder! Die oft zitierte über uns hereinbrechende "Informationsflut" ist somit eher eine "Datenflut" und braucht uns nicht zu verängstigen, zumal es an uns liegt zu selektieren und zu reflektieren!

Wissen und Weisheit

Sobald wir unterschiedliche Informationen in einem Erkenntnisprozess miteinander verknüpfen, entsteht Wissen. Während Daten elektronisch im Mikrosekundenbereich übertragen werden können, hat der Mensch eine biologisch beschränkte Aufnahmekapazität von Information. Wir schaffen maximal 50 bit pro Sekunde bewusst wahrzunehmen. Das wussten bereits die Ingenieure des 19. Jahrhunderts, die die Übertragungsgeschwindigkeit der Telegraphie mit 50 Baud (1 Baud entspricht bei binärer Verschlüsselung 1 bit/sek) standardisiert haben. Um Informationen zuzuordnen und daraus Wissen zu erzeugen, benötigen wir manchmal Stunden. Oft hilft es sogar darüber zu schlafen... Jahre, wenn nicht Jahrzehnte, dagegen benötigen wir, um Wissen zu Weisheit werden zu lassen! Manche schaffen es nie - doch das ist eine andere Geschichte.

Plausibilität

Wenn eine computergenerierte Lösung einer kubischen Gleichung aus drei komplexen Werten besteht, so ist das ebensowenig glaubhaft wie die von 40 Radiostationen ausgesandte Nachricht, dass die Post in Hinkunft ein Porto auf E-Mails einhebt, um das Defizit der Briefpost zu reduzieren, insbesondere wenn man weiß, dass diese Meldung vom 1.4.2010 stammt. Wir sollten unsere Kinder dringend dazu anhalten, alle Recherchen – vor allem jene im Web - nicht nur am 1. April, sondern regelmässig einem Plausibilitätscheck zu unterziehen!

Signifikanz

Häufig beobachten wir den Unfug, dass nach der Division zweier nur grob geschätzter Zahlen das Ergebnis auf 8 Dezimalziffern angegeben wird, nur weil es vom Computer so genau berechnet wurde. So berechnet sich zum Beispiel das mittlere Alter der Kinder einer Familie mit zwei 11-jährigen
Zwillingen und einer 13-jährigen Tochter zu 11.666667. Vernünftiger wäre es hingegen, dieses Ergebnis auf 11.7 zu runden.

Beim Drucken eines online-Tickets der ÖBB fiel mir kürzlich der Hinweis "Mit dieser Bahnfahrt entlasten Sie unser Klima um 74,2512 kg CO2" auf.
Trotz aller Sympathie sowohl für die ÖBB wie auch für alle Bemühungen zur Entlastung unserer Umwelt halte ich diesen Hinweis aus folgenden Gründen für - gelinde gesagt - schwachsinnig. es macht meines Erachtens absolut keinen Sinn, einen angeblich ersparten CO_2-Gehalt auf Zehntel Gramm genau anzugeben - vergleichsweise wird die Entfernung ja auch nicht auf Zentimeter genau angegeben. Darüber hinaus haben die ÖBB keine Ahnung, welche Route und welche Strecke ich gefahren wäre, wie umweltfreundlich mein Auto ist oder ob ich nicht alternativ zu Hause geblieben wäre... Freundlicherweise haben mir die ÖBB auf mein Ansuchen die Erlaubnis gegeben, diesen Ausdruck (selbstverständlich unter Quellenangabe) als illustratives Beispiel für numerische (nicht)-Signifikanz zu verwenden.

Größenordnungen

Der grosse Informatiker und Turing-Award Träger Edsgar W. Dijkstra[4] hat in einem anschaulichen Vergleich illustriert, dass ein Faktor 1000 an Geschwindigkeit Quantität zu Qualität werden lässt: Angenommen ein einjähriges Baby krabbelt in seiner Gehschule mit einer Geschwindigkeit von 1 km/h - die Geschwindigkeit von 1000 km/h eines Passagier-Jets eröffnet völlig andere Perspektiven der Fortbewegung! Und das bei einem Faktor von “bloss” 10^3 (Kilo). In der Informatik sind jedoch Größenordnungen von 10^6 (Mega), 10^9 (Giga) und 10^{12} (Tera) üblich, Werte die sonst nur in der Physik oder Astronomie (und neuerdings auch in der Finanzwirtschaft) verwendet werden. Die Exponenten entsprechen den Logarithmen dieser Zahlen (und spiegelt die Anzahl der Nullen wider, die Politikern oft Schwierigkeiten bereiten). Schade, dass logarithmische Funktionen im Mathematikunterricht eher stiefmütterlich behandelt werden, denn in der Informatik sind Logarithmen unverzichtbar.[5]

Ein anschauliches Beispiel für Größenordnungen, mit denen die Natur problemlos fertig wird, bietet die Informationsverarbeitung beim Menschen.

Während zum Beispiel die Auflösung unser Augen mit etwa 6 Megapixel etwa der von handelsüblichen Digitalkameras entspricht, werden davon - wie schon erwähnt - nur etwa 50 bit pro Sekunde bewusst wahrgenommen. Insgesamt schafft es somit ein Mensch in einem erfüllten Leben etwa $3x\ 10^{10}$ bit (das entspricht etwa 4 Gigabyte) aufzunehmen. Dank der enormen Speicherkapazität unseres Gehirns von etwa 10^{12} bit können wir diese Information auch problemlos speichern. Die Befürchtung, dass ein allzu intensiver Schulunterricht einen “memory overflow” bewirkt ist somit erwiesenermaßen unbegründet!

[4] Edsgar Wybe Dijkstra (1930-2002) erhielt 1972 den Turing Award verliehen.
[5] So ist zum Beispiel der Shannon’sche Informationsgehalt einer Nachricht als Logarithmus vom Reziprokwert der Wahrscheinlichkeit mit der diese Nachricht erwartet wird definiert.

Interessanterweise sind in unseren Genen Daten im Ausmaß von etwa 10^{10} bit verschlüsselt. Unsere Erbinformation hat somit in etwa die gleiche Größenordnung wie das, was wir im Laufe unseres Lebens dazulernen können.

Dauerbrenner Programmiersprachen

In den 70er-Jahren wurde die Wahl der für den Schulunterricht "richtigen" Programmiersprache heftig und kontroversell diskutiert. Die "Praktiker", die die Eignung einer Sprache an ihrer unmittelbar umsetzbaren praktischen Einsatzmöglichkeit festmachten, favorisierten BASIC und FORTRAN. Als Leiter eines von IBM gesponserten Projektteams zur Einführung der Datenverarbeitung an allgemeinbildenden höheren Schulen wurde ich damals mittels "Kabel" vom IBM Europa Headquarter in Paris instruiert, dass IBM nur bereit ist, die Einführung von FORTRAN an Schulen zu unterstützen. Dagegen argumentierten "Theoretiker" (mich eingeschlossen), dass Sprachen das Denkvermögen beeinflussen und dies nicht nur für die Muttersprache sondern auch für die erste Programmiersprache gelte und befürworteten Pascal. Lange Zeit erfreute sich auch die am MIT entwickelte Programmiersprache Logo als ernstzunehmende Alternative hoher Beliebtheit. Die Tatsache, dass auch Kinder erfolgreich in Logo programmieren lernen, hat allerdings zu dem Fehlschluss geführt, dass Logo nur für Kinder geeignet ist. Schade, erlaubt doch die aktuelle Version von NetLogo einen hervorragenden Einstieg in die Informatik bis hin zur Entwicklung von kollaborativen Computerspielen.

Aus heutiger Sicht ist es verwunderlich, dass Diskussionen über die Wahl der Programmiersprache im Schulunterricht gelegentlich immer noch geführt werden und dabei das Argument der Praxisrelevanz ebenso ins Treffen geführt wird wie das "Verstehen" der Funktionsweise eines Computers. Dabei sollte inzwischen allen klar geworden sein, dass jede in der Schule vermittelte Programmiersprache spätestens beim Berufseintritt der Schüler obsolet ist und dass man auch gut Autofahren kann, ohne die technischen Details des Verbrennungsmotors zu verstehen, geschweige denn einen solchen konstruieren zu können. Auch wenn es mir schwer fällt es einzugestehen: Der Stellenwert des Programmierens im Informatikunterricht nimmt zunehmend ab.

Algorithmische Komplexität

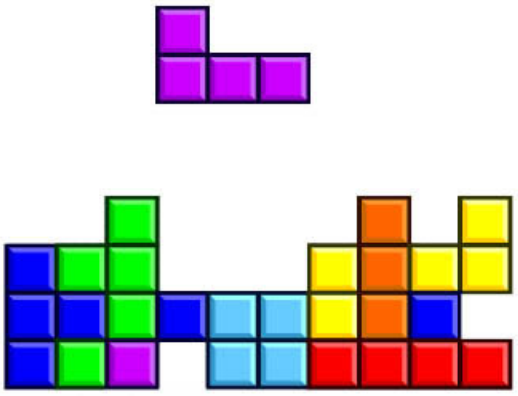
Größenordnungen spielen auch bei der Abschätzung der Komplexität von Algorithmen eine wichtige Rolle. Ein anschauliches Beispiel ist das "Verflixte Schildkrötenpuzzle" (das ge-

zeigte 3x3 Puzzle wird im Handel 6- bis 99-Jährigen empfohlen). Weil es mir vor meinen beiden damals noch schulpflichtigen Töchtern peinlich war, dass ich das Puzzle nicht lösen konnte, habe ich ein kleines Java Applet programmiert um mein Image zu rehabilitieren.

Während ein 2x2 Puzzle nach wenigen Versuchen lösbar ist, empfiehlt es sich zur Lösung des 3x3 Puzzles eine Strategie (einen Algorithmus) anzuwenden.
Die naheliegende Strategie, die Teile in zufälliger Reihenfolge herzunehmen und wenn eines nicht passt zu drehen und allenfalls wieder wegzulegen und ein anderes zu versuchen, benötigt etwa 2500 Versuche.

Aufgrund der kombinatorischen Vielfalt gibt es für ein Puzzle mit n Teilen 4^n n! unterschiedliche Anordnungen, das sind 4 triviale Lösungen des 1x1 Puzzles (die aber wenig Spass machen) und 6144 Anordnungen beim 2x2 Puzzle (dazu braucht man nun wirklich keinen Computer). Das verflixte 3x3 Puzzle mit 9 Teilen bietet hingegen bereits etwa 10^{11} Möglichkeiten, was in meinen Augen die Verwendung eines Computers rechtfertigt! Zur Lösung eines 4x4 Puzzles mit seinen mehr als 10^{22} Anordnungen ist allerdings selbst ein schneller Computer wenig hilfreich. Selbst wenn für einen Versuch nur eine Mikrosekunde benötigt wird benötigt ein Rechner etwa 3 Milliarden Jahre um alle Anordnungen zu überprüfen. Nur zum anschaulichen Vergleich: Seit dem Urknall sind etwa 15 Milliarden Jahre vergangen. Die Verwendung eines Computers ist hier offensichtlich völlig sinnlos!. Auch dieses Beispiel illustriert wie Quantität zu Qualität werden kann!

Die Lösung eines Puzzles gehört zwar nicht gerade zu den brennendsten Problemen der Menschheit, leider gibt es jedoch zahlreiche praxisrelevante Aufgaben vergleichbarer Komplexität, die auch mit den schnellsten verfügbaren Supercomputern nicht sinnvoll gelöst werden können, obwohl korrekte Lösungsalgorithmen existieren! Dazu gehören zum Beispiel das Problem des Handlungsreisenden (gesucht ist die kürzeste Rundreise die n vorgegebene Orte verbindet), das BinPacking Problem (wie können n vorgegebene Kisten in eine möglichst geringe Anzahl von Containern geschlichtet werden), das Rucksackproblem (welche Auswahl von n vorgegebenen Gegenständen ergeben - in einen Rucksack gegebener Größe gepackt - zusammen den größten Wert) oder Tetris (staple herab fallende Bausteine so in einem Rechteck, dass dabei ganze Zeilen gefüllt werden) und Sudoku (ergänze die Matrix so, dass die Ziffern[6] von 1 bis 9 nur je einmal in jeder Reihe, in jeder Spalte und in jedem umrahmten Kästchen vorkommen).



4		2		9		6	7	
1				5	2			9
	3				4		8	
	4	5				2	1	
			5	4				3
	6	8		7	1		4	
8					3			4
	7	3		8	5		9	
5			1				2	8

[6] manche Anleitungen sprechen von Zahlen...

Alleebäume und Zwischenräume

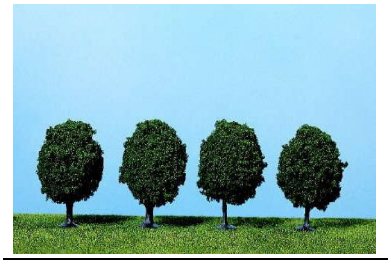
Die folgenden Überlegungen basieren auf der simplen Beobachtung, dass es zwischen n Alleebäumen genau n-1 Zwischenräume gibt[7]. Eng verwandt damit ist auch die Beobachtung, dass eine Salami mit n Messerschnitten in n+1 Stücke geteilt werden kann. Diese Diskrepanz führt gelegentlich zu Fehlern um genau „Eins zuviel“ oder „Eins zuwenig“, so genannten „Plus-Minus Eins Fehlern“.

Die vieldiskutierte Frage, ob das dritte Jahrtausend am 1.1.2000 oder erst am 1.1.2001 beginnt illustriert diese Problematik. Wer Ersteres favorisiert nimmt in Kauf, dass erste Jahrtausend nur aus 999 Jahren besteht (Ursache dieses Dilemmas ist übrigens das fehlende Jahr Null zwischen dem Jahr 1 vor und nach Beginn der Zeitrechnung). Die zweite Lösung hat den unschönen Nachteil, dass die Tausenderziffer der Jahre innerhalb eines Jahrtausends wechselt und sich ein analoges Problem konsequenterweise auch für die einzelnen Jahrzehnte stellt. Im ersten Fall haben die Jahre eines Jahrzehnts die fortlaufenden Endziffern 0 bis 9, im zweiten Fall hingegen 1 bis 9 gefolgt von 0 (diese Problematik manifestiert sich auch auf den Tastaturen unserer Computer und Telefone bei denen die Ziffer 0 nach der Ziffer 9 angeordnet ist).

Obwohl wir am Ziffernblatt einer Uhr die vollen Stunden mit 1 bis 12 beschriften, laufen die dazwischenliegenden Stundenintervalle sinnvollerweise zyklisch von 0 bis 11. Beim ersten Schlag der Kirchturmuhr ist es somit “Null Uhr Fünfzehn”. Die “Stunde Null” bezeichnet den Anfang aller Dinge, ist es dagegen “Fünf vor Zwölf”, so ist das Ende nicht mehr weit...

Variable und Konstante

Typisch für Variable in der Informatik ist, dass sie nicht (wie in der Mathematik) variieren, sondern solange konstante Werte haben bis diese (per Wertzuweisung) verändert werden!
Alan Perlis[8] verdanken wir den erkenntnisreichen Ausspruch „One man's constant is another man's variable“. Schöner kann man nicht zum Ausdruck bringen, dass unterschiedliche Blickwinkel unterschiedliche Sichten nach sich ziehen. Genau genommen kennt die Informatik nur drei fundamentale Konstante: 0, 1 und 2 - alle anderen Konstanten sind schlechte Approximationen von Unendlich! Allenfalls sind noch einige wenige kulturell bedingte Konstante berechtigt wie zum Beispiel 10 (für die Anzahl unserer Finger und somit die Basis des Dezimalsystems), 12 (für die Anzahl der Monate eines Jahre) oder etwa e (als Basis der natürlichen Logarithmen) und π (der Umfang des halben Einheitskreises).



Eine psychologische Konstante ist für die Informatik ebenfalls von Bedeutung: „Sieben plus/minus zwei“, da wir maximal fünf bis neun Objekte (chunks) in unserem Kurzzeitgedächtnis halten können.[9] Nicht von ungefähr gibt es sieben Tage pro Woche, sieben Zwerge und sieben Todsünden. Wegen dieser Beschränkung unserer Wahrnehmungsfähigkeit sollte zum

[7] Man beachte jedoch, dass diese Behauptung nur für lineare Alleen gilt - bei zyklischer Anordnung gibt es ebensoviele Bäume wie Zwischenräume!
[8] Alan Perlis (1922-1980), einer der Väter von ALGOL60 erhielt 1966 den Turing Award
[9] G. A. Miller (1956): "The magical number seven, plus or minus two: Some limits on our capacity for processing information"

Beispiel die Auswahl bei Menüs 7±2 Möglichkeiten nicht überschreiten. Sind mehr Unterscheidungen notwendig, so empfiehlt sich eine hierarchische Struktur. Die Tiefe dieser Hierarchie wächst naturgemäß logarithmisch mit der Anzahl dieser Unterscheidungen.

Umkehrbarkeit

Manche Abläufe sind umkehrbar (reversibel), andere nicht. Ein irrtümlich geöffnetes Fenster kann zum Beispiel sowohl im Klassenzimmer wie auch am Computer mittels eines inversen Prozesses wieder geschlossen werden. Benutzerfreundliche Computersysteme stellen dazu eigens inverse "undo"-Funktionen zur Verfügung. Das Schälen eines Apfels hingegen kann ebensowenig rückgängig gemacht werden wie das Formatieren einer Festplatte. Sicherheitshalber fragen Computersysteme vor solchen irreversiblen Aktionen zumeist nach ob man das auch wirklich will.

Verblüffenderweise gibt es auch Aktivitäten, die bei wiederholter Ausführung den ursprünglichen Zustand wiederherstellen, die also zu sich selbst invers sind! So entsteht zum Beispiel durch spiegeln des Spiegelbildes wieder das Original und durch zweimaliges Vertauschen der Werte zweier Variabler wieder der Ausgangszustand.

Quantität versus Qualität

So wie die Natur oft mit einer großen Anzahl von Individuen versucht deren mangelhafte Qualität auszugleichen. So produziert ein Krötenweibchen jährlich bis zu 6000 Nachkommen, von denen nur einige wenige das Kaulquappenstadium überleben, während zum Beispiel Elefantenkühe ihre Einzelkinder mehrere Jahre hindurch mit großem Aufwand groß ziehen.

1996 gelang es dem IBM Schachcomputer Deep Blue erstmals den amtierenden Weltmeister Garri Kasparow zu schlagen. Dieser Erfolg war allerdings nicht einer ausgeklügelte Software, sondern der Schnelligkeit der Hardware zu verdanken (Deep Blue konnte 200 Millionen Stellungen pro Sekunde analysieren).

Doch nicht immer kann Qualität durch Quantität kompensiert werden. Bei der Software Erstellung zum Beispiel leisten kleine Teams hochqualifizierter Spezialisten viel bessere Arbeit als viele schlechtausgebildete Programmierer (vgl. den oft zitierten „Million Monkeys Approach").

Es gibt keinen Zufall

Die Informatik beschäftigt sich ausschließlich mit deterministischen Prozessen. Diese liefern unter gleichen Voraussetzungen auch immer die gleichen Ergebnisse und sind somit prognostizierbar. In der Informatik ist nichts zufällig. Selbst sogenannte Zufallszahlengeneratoren liefern bei gleichen Anfangsbedingungen immer dieselben Zahlenfolgen, und diese sind daher alles andere als zufällig.[10] Selbst Monte Carlo - Methoden und genetische Algorithmen sind reprodzierbar. Um Zufälligkeit im herkömmlichen Sinn zu erzeugen, müsste der Zu-

[10] Donald Knuth widmet den 1981 erschienenen Band 2 seines "opus magnum" The Art of Computer Programming" grösstenteils den Zufallszahlen.

fallsgenerator durch Hardware implementiert sein wie zum Beispiel beim Würfeln, am Roulettetisch oder beim Atomzerfall.

Es scheint allerdings, dass wir etwas genau dann als zufällig bezeichnen wenn wir den Wirkungszusammenhang nicht durchschauen. Genau genommen gibt es keinen Zufall! Schon Albert Einstein hat erkannt, dass "der Alte nicht würfelt"![11]

Drücken oder Ziehen – Push or Pull

Immer wenn ich das gewaschene Geschirr vom Geschirrspüler in die Küchenkästchen räume, quält mich die gleiche brennende Frage: Soll ich das Besteck nehmen wie es kommt und danach auf die einzelnen Fächer der Bestecklade aufteilen oder lieber selektiv zuerst alle Messer, dann alle Gabeln und zuletzt die Löffel sammeln.

Zwei triviale Sortierverfahren bringen dieses Prinzip algorithmisch auf den Punkt. Beim Sortieren durch Einfügen (Insertion Sort) werden die einzelnen Elemente der Reihe nach dort eingefügt wo sie hingehören, während beim Sortieren durch Auswählen (Selection Sort) das jeweils nächste Element entsprechend der Sortierreihenfolge gesucht und plaziert wird. Sortieren durch Einfügen eignet sich daher hervorragend bei unvollständiger Information wie zum Beispiel für die Aktualisierung der Anzeigetafel bei Schirennen. Sortieren durch Auswählen wiederum hat den Vorteil, dass ein Element, sobald es plaziert ist, endgültig auf diesem Platz bleibt (was zum Beispiel für schwer bewegliche Elemente wie etwa Marmorstatuen empfehlenswert sein kann).

Dieses Push-and-Pull Prinzip begegnet uns vielerorts, nicht nur in der Fertigungsplanung, wo zwischen Produktion auf Lager (Bring-Prinzip) oder nachfrageorientierter Fertigung (Hol-Prinzip) entschieden werden muss, sondern auch bei der Wissensvermittlung. So wird etwa durch Läuten der Kirchturmglocken die Uhrzeit mittels Push-Technologie verbreitet, während ein Blick auf die Kirchturmuhr dem Hol-Prinzip entspricht. Gleiches gilt für Nachrichten, die über Medien verbreitet werden, ganz im Gegensatz zum World Wide Web, das es dem Benutzer erlaubt, selbst nach bestimmten Informationen zu suchen.

Vergleichen wir die Verschlüsselung von Daten durch einen Code mit der alternativen Möglichkeit, die Nachricht zu erfragen, so begegnen wir auch hier dem Bring- und Holprinzip. Der Binärcode 0110 der Zahl 6 sagt uns zum Beispiel, dass die Zahl kleiner als 8 aber nicht kleiner als 4, nicht kleiner als 6 und gerade ist. Anstatt die durch 0110 verschlüsselte Zahl zu übermitteln, könnte aber auch ihr Wert durch folgende vier geschickt formulierte Fragen erraten werden: „Ist die Zahl kleiner als 8?" Falls ja lautet die nächste Frage "Ist die Zahl kleiner als 4?", falls nein folgt die Frage "Ist die Zahl kleiner als 6?" und schliesslich die Frage "Ist die Zahl ungerade?". Verschlüsseln wir jede positive Antwort durch eine Null und jedes Nein durch eine Eins, so erhalten wir den ursprünglichen Code 0110. Sind alle Zahlen zwischen 0 und 15 gleichwahrscheinlich, so ist interessanterweise der Informationsgehalt von 4 bit derselbe, unabhängig davon, ob wir den Code nach dem Bring-Prinzip geliefert bekommen oder nach dem Hol-Prinzip erfragen![12]

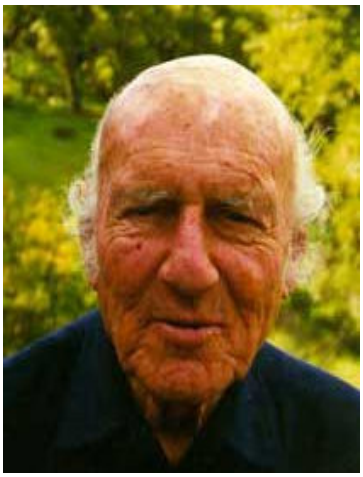
[11] Albert Einstein schrieb 1926 in einem Brief an Max Born: „Die Theorie liefert viel, aber dem Geheimnis des Alten bringt sie uns doch nicht näher. Jedenfalls bin ich überzeugt davon, dass der nicht würfelt."

[12] Die Fragen müssen immer so gestellt werden, dass die binären Antworten gleichwahrsheinlich sind, dann ist der Informationsgehalt jeder Antwort maximal!

Nicht zuletzt begegnet uns das Bring- oder Holprinzip auch in der Schule: Alternativ zum Nürnberger Trichter, der versucht Wissen auf Vorrat in die Köpfe der Schüler zu drücken, erlauben es konstruktivistische Lernmethoden den Lernenden ihren Lernprozess selbst zu steuern.

Ein kurzer interdiszplinärer Überblick

Konstruktivismus



Informatik ist gelebter Konstruktivismus! Nicht nur Computerspiele schaffen virtuelle Realitäten, nahezu jede Informatikanwendung spiegelt die konstruktivistische Sicht ihrer Entwickler wieder! Nicht zuletzt aber konstruieren manche Informatiksysteme tatsächlich Wirklichkeiten wie uns Börsencrashes und Finanzkrisen deutlich vor Augen führen. Der an der Ideengeschichte der Informatik nicht unwesentlich Beteiligte und vor allem als Mitbegründer des Konstruktivismus bekannte gebürtige Österreicher Heinz von Förster[13] ("Wahrheit ist die Erfindung eines Lügners") formulierte darüber hinaus seinen ethischen Imperativ „Handle stets so, dass die Anzahl der Wahlmöglichkeiten größer wird!“, der interessante Bezüge zu Informationsgehalt und Entropie nahelegt.

Kreativität

Für Alan Key[14] ist Kreativität die Fähigkeit, bewährte Konzepte in neue Einsatzgebiete zu übertragen. Die Objektorientierung war bereits von der Mitte der 60er-Jahre entwickelten Programmiersprache Simula her bekannt und Mausklicks gab es bereits bei der Digitalisierung von Landkarten, als Alan Key diesen Ideen ein Jahrzehnt später bei Xerox Research mit der objektorientierten Programmiersprache Smalltalk und den Mouse/Windows Benutzeroberflächen zum Durchbruch verhalf. Alan Kay betonte übrigens, dass die Transformation geläufiger Konzepte auf neue Gebiete sowohl in der Kunst wie auch in der Wissenschaft gang und gäbe ist und nicht zuletzt häufig die Grundlage für einen guten Witz liefert.

Interdisziplinarität

Den Einwurf, dass viele der hier diskutierten Themen nicht allein die Informatik betreffen, sondern in anderen Fächern beheimatet sind, lasse ich gerne gelten. Informatik ist aber in hohem Maße auch interdisziplinär und lebt von ihren Anwendungen. Diese Anwendungen sind es auch, die zur Motivation als Einstieg in die Informatik herangezogen werden können (und realiter auch werden).

[13] Heinz von Förster (1911 - 2002) wurde in Wien geboren und war langjähriger Direktor des legendären Biological Computer Laboratory der University of Illinois.

[14] Alan Curtis Kay (*1940) begründete in den 70er-Jahren im Xerox Palo Alto Research Center die Objektorientierte Programmierung und entwickelte die ersten Mouse/Windows Systeme. Er ist auch Träger des „Nobelpreises der Informatik“, dem Turing-Award.

Nicht nur Mathematik und naturwissenschaftliche Fächer, sondern auch Musik, Sprachen oder Geographie bergen eine Fülle von Informatikaspekten, die auch nicht an Mathematik interessierten Schüler ansprechen sollten.

Selbstverständlich ist es unverzichtbar, die Grundlagen der Informatik von fachkompetenten Lehrkräften in einem eigenen Gegenstand zu vermitteln. Darüber hinaus bietet die Informatik jedoch willkommene Gelegenheiten, den klassischen Frontalunterricht durch interdisziplinäre Projekte (gegebenenfalls mittels Teamteaching) zu ergänzen. Erfahrungsgemäß stellt ein solcher projektorientierter Unterricht mannigfaltige Herausforderungen an die Lehrer dar. Insbesonders engagierte Schüler, die als "digital natives" Erfahrungen im technischen Umgang mit den neuesten technologischen Geräten gewonnen haben (und dafür auch enorm viel Zeit investieren), werden ihren Lehrern immer eine Nasenlänge voraus sein. Umso wichtiger ist es für die Lehrkräfte, ihre Autorität nicht allein durch technologisches Know-how sondern vielmehr (als "primi inter pares") durch fundierte soziale und fachliche Kompetenz zu untermauern.

Die Zeiten ändern sich …

Digital Natives

Noch sind die meisten Lehrer "digital immigrants", während ihre Schüler als "digital natives" das Leben in der digitalen Welt von Geburt an kennengelernt haben. Es ist zu erwarten, dass eine neue Lehrergeneration, die selbst mit Computern aufgewachsen ist, einen (völlig) anderen Zugang zur Informatik vermittelen wird. Trotzdem bin ich überzeugt, dass es eine (häufig anzutreffende) Illusion ist zu glauben, dass der Informatikunterricht jemals den aktuellen Stand der Technologie vermitteln wird - zu rasch nämlich erfolgt der technologische Wandel und zu träge ist im Vergleich dazu das Schulsystem. Und das ist auch gut so, **geht es doch im Informatikunterricht vor allem um die Vermittlung langfristig gültiger Konzepte** und nicht um technologische Modeerscheinungen und Zufälligkeiten!

The Times They Are A-Changin'

Während sich die Welt im Wandel befindet, scheint die Zeit an der Schule (eher) stillzustehen, obwohl die derzeitige Rhetorik einiger (geplanter) Reformen vermeintlich das Gegenteil suggeriert. Wie in meiner Kindheit und der Kindheit meiner Eltern und Grosseltern laufen Tafelklassler immer noch mit Schultüten und hochgesteckten Erwartungen zur Schule. Viele Pädagogen haben die Entwicklung verschlafen.

Auf der interdisziplinären Partnersuche nach kooperationswilligen Pädagogen wurde ich mit der Aussage überrascht, dass für Pädagogen Computer nichts Neues sind - sie wüssten seit Erfindung des Buchdruckes mit Medien umzugehen. Auf meinen Einwand, dass Computer im Gegensatz zu Büchern Interaktivität erlauben, wurde ich belehrt, dass Interaktivität auch bei physikalischen und chemischen Experimenten möglich sei und dort pädagogisch schon hinlänglich untersucht wäre...

Aber auch wenn sich Schule und Lehrer verändern: **Die "digital natives" von heute werden die technologischen Immigranten von morgen sein - for the times they are a-changin'**[15]

[15] © Bob Dylan (1963)