Das digitale Abbild und seine Grenzen

WOLFGANG COY
Institut für Informatik
der Humboldt-Universität zu Berlin

Die Geburt der Informatik aus der Statikrechnung

›Eine ausgesprochene Abneigung hatte ich gegen die statischen Rechnungen, mit denen man die Bauingenieurstudenten quälte. Die Professoren, die diese Rechnerei beherrschten, bewunderte ich wie Halbgötter aus einer anderen Welt. Würde ich das jemals begreife? Später sollte ich über das Problem des statischen Rechnens auf die Idee des programmgesteuerten Rechenmaschine kommen.‹ Der Bauingenieur Konrad Zuse erinnert sich an seine *alma mater*, die TH Charlottenburg, in der wir hier tagen, und die Wirkungen seines soliden Ingenieurstudiums.

Computer sind also als Rechenautomaten konzipiert worden. Daran muß man heutzutage, in Zeiten des Internets und der Multimediaspielkonsolen schon gelegentlich erinnern. Digitale Technik heißt in unserem Nachbarland *techniques numériques*, eine präzise und akademische Erinnerung an ihre Geburtsumstände. Die ersten Computer, vor 1950, speichern Zahlen und sie rechnen mit ihnen.

Zahlen und Rechnen

Zahlen und Zählen sind eine der ältesten medialen Techniken, entstanden wohl aus dem Wunsch, wirtschaftlichen Besitz zu überprüfen. Zählen ist die Basisoperation des Rechnens. Ohne Zählen gibt es nur einige Zahlen: ›Eins, zwei, drei, viele‹, aber durch das Zählen eröffnet sich eine symbolische Unendlichkeit: eins, zwei, drei, … und einundneunzig, zweiundneunzig, dreiundneunzig, … und einemillionzweitausendeins, einemillionzweitausendzwei, einemillionzweitausenddrei … und immer so weiter. *›Nenn' mir eine Zahl, ich nenn Dir die nächst größere.!‹* Das Zählverfahren ist ein Algorithmus, eine Vorschrift, die zu jeder gegeben natürlichen Zahl die nächst größere angibt. So wird ein abstrakter, symbolischer Zahlenraum aufgespannt, später ergänzt um Brüche, dann Irrationalzahlen wie die berüchtigte Quadratwurzel $\sqrt{2}$, die das pythagoräische Weltbild beschädigte oder transzendent-irrationale Zahlen wie die Kreiszahl π bis hin zu komplexen und imaginären Zahlen. Die Geschichte der Mathematik ist eine Folge von solchen Erweiterungen des symbolischen Raumes. Auch die Rechenverfahren sind immer komplizierter geworden.

Die digitalen Techniken erben aus dieser symbolischen Herkunft eine Reihe nützlicher Eigenschaften.

• Zahlen lassen sich in verschiedene Formate kopieren, darunter auch in die technisch gut beherrschbaren Binärformate.

• Binärzahlen lassen sich als gut unterscheidbare elektrische oder magnetische Signale darstellen. Diese lassen sich über Leitungen oder per Funksignal übertragen.
• Binärzahlen lassen sich technisch gut speichern. Es gibt Halbleiterspeicher, magnetische oder optische Speicher unterschiedlichster Speicherkapazität und Zugriffsgeschwindigkeit.
• Derart gespeicherte Zahlen lassen sich exakt wiederholen und von anderen Zahlen gut unterscheiden. Damit lassen sich digitalisierte Werte sehr zuverlässig kopieren.
• Für Binärzahlen lassen sich einfache programmierbare Schaltungen angeben, so daß Maschinen mit Binärzahlen rechnen können.

Die digitale Technik erlaubt derart eine schnelle und effektive Speicher- und Übertragungstechnik mit guter Kopierbarkeit und dem Vorteil maschineller Berechenbarkeit.

Zahlen-Messen

Zahlen bilden die Basis des Messens und damit eines wesentlichen Aspektes moderner Wissenschaften. Numerisches Rechnen hat sich deshalb als Paradeaufgabe für die ersten Rechenautomaten erwiesen. Wissenschaftliches und technisches Rechnen bildete den erste großen Aufgabenbereich des Computereinsatzes. Konrad Zuse, der Ingenieur, der die statischen Bauingenieurrechnungen automatisieren wollte, hat dies schon 1936 erkannt. Und er hat eine wesentliche Entscheidung getroffen, um diese Art Rechnungen zu vereinfachen. Er führte die halblogarithmische Zahldarstellung in die Rechentechnik ein, wo sie heute noch als Fließkommarechnung oder *Floating Point Arithmetic* weiterlebt, längst durch einen technischen Standard IEEE-754 fixiert ist. Fließkommazahlen oder *real*-Zahlen sind Zahldarstellungen, die in einem großen Zahlbereich alle Zahlen durch eine feste Anzahl von Stellen (typischerweise 8 Dezimalstellen) und ihre Größenordnung als Exponent repräsentieren..

Meßgenauigkeit Realzahlen

Diese Fließkommadarstellungen sind Abstraktionen von Messungen, denn Meßgeräte können ja auch nur eine Genauigkeit mit einer gewissen Anzahl von Stellen repräsentieren. Leider akkumulieren sich diese Rundungen, die bei Messungen durchaus vertretbar sind, bei wiederholten Rechenschritten unter Umständen sehr schnell, so daß die Zahl der verwertbaren Stellen der Zahldarstellung rasch abnehmen kann. Banken verzichten deshalb auf diese Zahldarstellungen bei der Kontenführung; Techniker und Naturwissenschaftler setzen sie dagegen eher unbekümmert ein.
Mit der wachsenden Rechnerleistung steigt die Zahl der in einer gegebenen zeit ausführbaren Rechenschritte gewaltig an – etwa um das Doppelte in 18 Monaten. Die Rechengenauigkeit wird entsprechend unsicherer. Man kann zeigen, daß schon bei Rechnungen mit wenigen hundert oder tausend Iterationen Fehler beliebiger Größenordnung auftreten können.
Versuche, Fließkommadarstellungen präziser zu fassen, sei es durch bessere Compiler oder zusätzliche Hardwareeinrichtungen, sind am Markt und an der Ignoranz der programmierenden Kundschaft gescheitert, so daß heute die Fließkommaarithmetik zu einem erheblichen Unsicherheitsfaktor beim Einsatz digitaler Technik geworden

ist. Die Komplexität der Rechnungen macht es zudem extrem schwer, auftretende Fehler überhaupt zu erkennen – es sei denn, daß sie in gröbster Weise gegen Plausibiltätserwägungen sprechen.

Alan Turing

Während Konrad Zuse in Charlottenburg die Bauelemente seines praktischen Rechenautomates in fleißiger Laubsägearbeit erstellte, erdachte im gleichen Jahr 1936 in Cambridge der Mathematiker und Logiker Alan M. Turing eine *Paper Machine*, die viel mehr können sollte – nämlich alles, was sich überhaupt berechnen läßt. Diese *Paper Machine* war als Modell der Berechenbarkeit gedacht, die menschliche Rechenfähigkeiten in allgemeinster Weise nachbilden sollte – eine theoretische Herausforderung, die den Test der Zeit bislang bestanden hat. Turings *Diskrete Maschine* gilt auch 50 Jahre später noch als gültige Formulierung des mathematischen Berechenbarkeitsbegriffs gilt – und von vielen Mathematikern und Informatikern als ewig gültige Fassung des Begriffs *Algorithmus* angesehen wird.
Turings Leistung, die auf Vorarbeiten Hilberts und Gödels zurückgreift, liegt darin, daß sein Kalkül sehr wohl eine Paper Machine wie die Blaupause eines Rechenautomaten war. Begriffe wie Instruktion, Programm oder Speicher, aber auch die umfassende Kodierbarkeit aller Zahl- und Schriftdarstellung sowie die interne Speicherbarkeit von Programmen und Daten sind in Turings Arbeit bereits formuliert.
Darüber hinaus zeigt Turing, daß Programmierung durch einfache Regeln, logische Abfragen und Schleifen in umfassender Weise stattfinden kann – und er macht sich Gedanken über die Grenzen der programmierbaren Aufgaben. Unter anderem zeigt Turing, daß keineswegs alle mathematisch formulierbaren Funktionen auch maschinelle berechnet werden können. Seitdem wissen wir, daß es harte Grenzen der Berechenbarkeit und der Entscheidbarkeit wohldefinierter logischer Fragen gibt

Im zweiten Weltkrieg baut Turing an Computern zur Dechiffrierung verschlüsselter Wehrmachtsnachrichten – eine nicht-numerische Anwendung von Computern. Diese nicht-numerischen Anwendungen öffnen nach dem Krieg den Blick auf die Möglichkeiten des Computereinsatzes erheblich - soweit, daß heute die Bezeichnung Computer für die weltweit vernetzen digitalen Multimediamaschinen zum klaren *Misnomer* geworden ist.

Turing war diese Weite des möglichen Computereinsatzes sehr wohl bewußt. So schreibt er an W. Ross Ashby unverblümt: »Daß ich am ACE-Computer arbeite, liegt daran, daß ich mehr an der Möglichkeit interessiert bin, Modelle der Handlungen des Gehirns zu verstehen als an praktischen Anwendungen des Rechnens.«[i]

Kodes

Der entscheidende Schritt über das numerische Rechnen hinaus ist die Kodierung von Zahlen als Schrift und Schrift als Zahlen, Vielleicht hat die *Académie Française* doch ein wenig voreilig Digitaltechnik *mit technique numérique* übersetzt. Digitaltechnik umfaßt Zahlen ebenso wie Buchstaben, längst aber auch Bilder, Töne, Filme und sogar digitalisierte Gerüche – eben alle Signale, die sich in meßbare Werte umwandeln lassen.

Mit der Öffnung des maschinellen Rechnens zum Abarbeiten von logischen Ausdrücken und Regeln, dem Speichern und Umformen von Texten wird das automatische Rechnen in eine andere Tradition gestellt, die auf die kartesische Methode und auf Gottfried Wilhelm Leibniz *characteristica universalis* und andere formalisierten Wissenschaftssprachen verweist.

René Descartes fixierte die methodischen Anweisungen künftiger Wissenschaft (hier in der Formulierung Polyas):
›Erstens: Man reduziere jede Art von Problem auf ein mathematisches Problem.

Zweitens: Man reduziere jede Art von mathematischem Problem auf ein algebraisches Problem.

Drittens: Man reduziere jedes algebraische Problem auf die Lösung einer einzigen Gleichung.‹
Descartes Ziel war es, eine höhere Gewißheit der Erkenntnis zu erreichen. Er reduziert die sinnliche Erfahrung deshalb auf die Algebra, die er als fundiert annimmt. Ähnliches unternahm Isaac Newton mit der Physik, die er auf eine mathematischer Form reduzierte, wobei Physik mit (Punkt-) Mechanik gleichgesetzt ist. Seit dem Ende des 18. Jahrhunderts wurde dies zur dominierenden wissenschaftlichen Ideologie, als deren entschiedenster Sprecher Laplace gelten kann. Laplace schreibt 1812 in seinem *Essai philosophique sur les probabilités*:
›Wir müssen also den gegenwärtigen Zustand des Weltalls als die Wirkung seines früheren und als die Ursache des folgenden Zustands betrachten. Eine Intelligenz, der in einem gegebenen Zeitpunkt alle in der Natur wirkenden Kräfte sowie die gegenseitige Lage aller Dinge, aus denen die Welt besteht kennte und überdies umfassend genug wäre, alle diese Daten der Analyse zu unterwerfen, könnte in einer und derselben Formel die Bewegungen der größten Körper des Weltalls und die der leichtesten Atome zusammenfassen; nichts wäre ihr ungewiß, und Zukunft wie Vergangenheit wäre ihren Augen gegenwärtig.
Der menschliche Geist liefert in der Vollkommenheit, die er der Astronomie zu geben wußte, eine schwache Skizze dieser Intelligenz.‹
Für diesen allwissenden *Laplaceschen Dämon* sind die Vorstellungen eindeutig definierter Zustände und eindeutig definierter wirkender Kräfte von wesentlicher Bedeutung. Beides wird im folgenden die physikalisch-mathematische Denkwelt beherrschen. Unter »allen in der Natur wirkenden Kräften« versteht Laplace der Zeit entsprechend alle in der Natur wirkenden *mechanischen* Kräfte.
Der Physiker Philipp Franck zerlegt die Arbeitsweise des Laplaceschen Dämons in vier Schritte:

1. Alle Gesetze für die zwischen Massen geltenden Gesetze zu kennen.
2. Die Lagen und Geschwindigkeiten aller Massen im gegenwärtigen Weltzustand numerisch genau festzustellen.
3. Die komplizierten Differentialgleichungen bis zu numerisch angebbaren Lösungen für jeden Zeitpunkt aufzulösen.
4. Aus der Kenntnis der Lage und Geschwindigkeit der Massenpunkte die Erlebnisse anzugeben, die damit für die Menschen verbunden sind.

Die Turingsche Maschine ist, nebenbei bemerkt, eine diskrete *hommage* an den Laplaceschen Dämon. So schreibt er: »Es will scheinen, als ob es bei gegebenem Anfangszustand der Maschine und gegebenen Eingabesignale immer möglich sei, alle zukünftigen Zustände vorherzusagen. Das erinnert an die Laplacesche Ansicht, daß es möglich sein müßte, aus dem vollständigen Zustand des Universums zu einem bestimmten Zeitpunkt, beschrieben durch Lage und Geschwindigkeiten sämtlicher Partikel, alle zukünftigen Zustände vorherzusagen. Die von uns hier betrachtete Vorhersage ist jedoch praktikabler als die von Laplace erwogene. Das System des ›Universums als Ganzem‹ ist so beschaffen, daß minimale Fehler in den Anfangsbedingungen zu einem späteren Zeitpunkt einen überwältigenden Einfluß haben können. Die Verschiebung eines einzigen Elektrons um einen billionstel Zentimeter in einem Augenblick könnte ein Jahr später darüber entscheiden, ob ein Mensch von einer Lawine getötet wird oder ihr entkommt. Es ist eine wesentliche Eigenschaft der mechanischen Systeme, die wir ›diskrete Maschinen‹ genannt haben, daß dieses Phänomen nicht auftritt. Selbst wenn wir die tatsächlichen physikalischen Maschinen anstelle der idealisierten Maschinen betrachten, ergibt sich aus einer verhältnismäßig genauen Kenntnis des jeweiligen Zustandes eine verhältnismäßig genaue Kenntnis aller späteren Schritte.«

Doch auch dieses diskrete Revival des Laplaceschen Dämons war nur von kurzer Dauer. Ironischerweise bricht diese Reduktion gerade auf der Basis der Turingschen Rekursionstheorie in den Ergebnissen von Untersuchungen zum deterministischen Chaos zusammen, denn auch deterministische Berechnungen zeigen unter bestimmten Bedingungen ein unvorhersehbares ›chaotisches Grenzwertverhalten, bei dem die Lawine auf Grund ihrer infinitesimalen Vorgeschichte eben abreißt – oder nicht.

Wir sollten uns im klaren sein, daß diese spezifischen naturwissenschaftlich-mathematischen Ideologien die Basis der digitalen Techniken bilden und daß diese Techniken nicht ohne ihre Basis in andere Wissenschaftsformen übertragen werden.

Modellierung & Simulation

Programme werden erfolgreich eingesetzt, um Algorithmen und exakt beschriebene Verfahren maschinell zu steuern. In diesem Sinne haben die Rechenautomaten zu einer Maschinisierung und Automatisierung des Rechnens geführt. Leider ist dies nur ein kleiner und immer kleiner werdender Teil des Computereinsatzes. Mehr und mehr werden Verfahren programmiert, die keineswegs auf gesicherten Algorithmen oder exakten Beschreibungen des erwarteten Verhaltens zurückgreifen.

Regeln und Heuristiken

Eine große Gruppe von Programmen, beginnend mit Fakturier- und Buchhaltungsverfahren, Abrechnungsverfahren, Lagerhaltungsprogrammen, Arbeitsablaufsteuerungen und ähnlichem, die schon in der Frühzeit der Informatik einen wesentlichen Anteil am praktischen Einsatz hatten, greift nicht auf feste Algorithmen zurück, sondern auf Regelsammlungen. Bei anspruchsvolleren Programmen zur Transportwegeoptimierung, Maschinenbelegung oder

Musterkennung werden heuristische Verfahren eingesetzt. Als zwingende Programmiertechnik werden Regelsteuerungen oder Heuristiken bei Programmen zur Analyse, Diagnose, Wartung oder Planung komplexer Systeme eingesetzt.

Unsicherheit

Solche Programme sind mit hohen oder doch zumindest schwer abschätzbaren Unsicherheiten, Vagheiten und Unschärfen behaftet – und zwar nicht wegen Mängeln der Programmiertechnik sondern wegen der inhärenten Unsicherheit der zugrunde liegenden Modelle.
Während mit algorithmisch fundierter Programmierung in der Tat eine neue Phase stabiler Nutzung erreicht werden kann, kann die Digitalisierung vagen und unscharfen Wissens nur bedingt die inhärenten Unsicherheiten solcher Programmierung überwinden. Viele Projekte der Künstlichen Intelligenz, wie z.B. Expertensysteme, maschinelle Sprachübersetzung oder gar die maschinelle Textanalyse belegen dies.

Skalierung

Zu den besonderen Möglichkeiten des programmierten Umgangs mit Datenbeständen gehören die Möglichkeiten der Skalierung. Programme werden mit kleinen Datenbeständen getestet, aber auf beliebig große Datenbestände angewendet. Algorithmen beschreiben Verfahren, die allgemeingültig sein sollen – für spezielle Anwendungsfälle ebenso wie für allgemeine. Die Grenzen des Einsatzes werden selten exakt bestimmt, sie sind auch selten exakt bestimmbar.

Ein klassisches Skalierungsproblem entsteht schon bei der Langzeitspeicherung von Daten. Digitale Dateien lassen sich zwar sehr viel besser kopieren und speichern als analoge Signale; fehlerfrei ist die digitale Speicherung allerdings auch nicht, so daß eine Vielzahl von Korrekturverfahren auf allen Ebenen eingesetzt werden. Trotzdem muß man z.B. mit einem Bitfehler bei jeder zweitausendsten CD-Rom rechnen. Andere Speicher sind viel anfälliger.

Skalierung wird zum großen Problem durch die rasante Entwicklung der Hardware. Bill Gates fragte bei der Vorstellung des berüchtigten DOS-Betriebssystems: ›Wer braucht mehr als 640KB Speicher?‹ Hinter der harmlosen Frage versteckt sich eines der ärgerlichsten Probleme für eine ganz Generation von Wintel-Programmierern, die genau diese mißliche und unnötige Schranke mit allerlei Tricks überwinden mußten. Es gibt eine Fülle weiterer Probleme, die auf Ignoranz gegenüber der technischen Weiterentwicklung zurückzuführen sind. Selbst das Jahr2000-Problem zeigte seine Hardwarevarianten durch schlecht konstruierte Real-Time-Clocks und BIOS-Roms.

Man kann das Skalierungsproblem etwas theoretischer fassen. Im Kern greift jede digitale Modellierung auf Okhams *Razor* zurück: Die wichtigen Charakteristika werden modelliert, unwichtige fallen weg. ›Entia non sunt multiplicanda sine necessitate!‹ Dies ist einer der wichtigsten Leitsätze der modernen Wissenschaften. Leider ist das Verhältnis von wichtigen und unwichtigen nicht zwingend invariant gegenüber wachsender Größe der Daten. Was für einen Größenbereich unwichtig zu sein scheint, mag für andere Bereiche wichtig sein. Es ist extrem schwierig, die

genauen Geltungsbereiche von Programmen abzuschätzen. Man kann eben nicht die Stabilitätsverhältnisse eines Modellautos auf das richtige Auto übertragen, aber bei Programmen tauchen viel schwierigere Skalierungsprobleme auf – deren Fehler typischerweise erst entdeckt werden, wenn etwas schief geht – und gelegentlich nicht einmal dann.

Ein eigentümliches Skalierungsproblem ergibt sich bei der Fehlerkorrektur selber. Große Programme sind niemals fehlerfrei; das ist eine Binsenweisheit. Als groß kann dabei jedes Programm gelten, das mehr als ein paar hundert Programmzeilen enthält. Programme wie etwa Betriebssysteme enthalten mehrere Millionen Programmzeilen. Es gibt also eine Vielzahl von Fehlern in solchen Systemen, die von Version zu Version bearbeitet werden.
Die Vorstellung ist nun weit verbreitet, man könne allmählich alle Fehler eines solch komplexen Systems ausmerzen – und nur die Einführung neuer Programm-Features verhindere dies. Es ist schon richtig, daß das Einfügen neuer Programmzeilen mit einer gewissen Wahrscheinlichkeit auch neue Fehler verursacht. Im Prinzip kann man die Wahrscheinlichkeit abschätzen, mit der eine neue oder geänderte Programmzeile einen Fehler trotz sorgfältiger Tests einführt; sie ist sehr viel kleiner als 1, aber leider größer als 0. Man kann im Prinzip auch für jeden erkannten Fehler die Wahrscheinlichkeit seines Auftretens abschätzen.
Nun tritt die kuriose Situation auf, daß bei manchen sehr selten auftretenden Fehlern, die Gesamtfehlerwahrscheinlichkeit durch die Korrektur ansteigt statt abzunehmen. Solche Fehler werden nicht mehr korrigiert, sie sind *known bugs*. Natürlich wächst diese Zahl der nicht-korrigierwerten Fehler mit dem Umfang des Programms - ein Skalierungsproblem.

Kontext

Zahlen und Schrift sind abstrakte und explizite Abbilder dessen, was sie beschreiben oder berechnen sollen. Diese Abstraktheit erlaubt die programmierte oder maschinelle Bearbeitung, weil sie von jedem Kontext abstrahiert - eine Stärke, die es erlaubt, die gleichen Verfahren auf unterschiedliche Daten oder Modelle anzuwenden. Diese Stärke der Abstraktion ist freilich auch eine substantielle Schwäche in dem Moment, in dem Kontext von Bedeutung ist. Der Einsatz digitaler Technik wird nun immer häufiger in solchen Situationen gewünscht, wo der Kontext eine Rolle spielt. Die einfachste Antwort besteht darin, eine programmiertes Verfahren mit einem menschlichen Nutzer oder Entscheider zu kombinieren. Es gibt aber keine Garantie, daß Kontextfragen nicht plötzlich und unerwartet auftauchen.
Sprachübersetzungssysteme oder Bilderkennungssysteme sind notorisch mit diesem Problem konfrontiert.

Erfahrung

Die explizite Formulierung durch Schrift und Zahl wird ein weiteres Problem auf, wenn es um den Wissenserwerb geht. Seine banalste Form ist Lernen durch Erfahrung. Erfahrung ist nun keineswegs nur die Ansammlung von Falldaten, was Computer angesichts der heutigen Speichermöglichkeiten ja sehr gut können, sondern

die Herausbildung von Erkenntnis- oder Handlungsmustern in impliziter Form. Diese körperlich oder in Kollektiven gebundenen Muster werden nur gelegentlich (und oft unter großer Anstrengung) zu explizierten Mustern – in Form von Texten, Büchern, Regeln oder manchmal auch als Algorithmen oder Computerprogramme. Zwar gibt es einige Versuche, in Datenbeständen regelhafte Muster zu erkennen, die aktuellen Schlagworte heißen z.B. *data mining, Expertensysteme* oder *case based reasoning*, aber dies sind noch immer Marginalien.
Die übliche Art, ein Programm zu schreiben, ist es, sich auf Beschreibungen, Regeln und Verfahren zu stützen, gelegentlich ergänzt durch Befragungen von Fachleuten, durch Introspektion bei denjenigen im Programmierteam, die Anwendungskenntnisse erworben haben und allzu oft durch phantasievolle Vorstellungen, was der Kunde wohl haben wolle. Implizite Erfahrung ist ein kostbares Gut, das leider nur sehr schwer zu explizieren und dadurch digitalisierbar wird. Explizierte Erfahrung ist die Basis des Programmierens – und selbst dieser Transformationsprozeß ist hinreichend dornig und fehlerträchtig. Überhaupt sind Fehler ja ein Grundstoff der Erfahrung, die an ihren –erkannten- Fehlern wächst. Programme freilich lernen von ihren Fehler nicht. Deshalb sind Fehler in Programmen unbeliebt – auch wenn sie unausrottbar scheinen. Erfahrung hat also diesen zweiten Aspekt: Menschen lernen durch Erfahrung, Programme nicht – jedenfalls nicht von selbst. Erfahrung wird in Versionsverwaltung umgewandelt. Die Erfahrungen der Nutzer und Nutzerinnen werden in der nächsten Version berücksichtigt – die Ware reift beim Kunden; auch bekannt als *Bananenprinzip*.

Urteilskraft

Nun werden alle Bedenken die Digitalisierung des Denkens und darüber vermittelt des Handelns nicht wirklich aufhalten. Zu mächtig sind die wirkenden Kräfte, aber auch die damit verbundenen Hoffnungen. Bei allem sollten wir jedoch nicht übersehen, daß wir mehr über das Denken und Handeln wissen als uns vielleicht alltäglich bewußt ist. Ich will deshalb zum Schluß auf den großen philosophischen Analytiker und Systematiker Immanuel Kant zurückgreifen. Er hat uns drei Fähigkeiten genannt, mittels derer wir denken:
Verstand, Urteilskraft und Vernunft, soll heißen die praktische Fähigkeiten des logischen, ableitenden Denkens, die Fähigkeit, uns über Zwecke oder ästhetische Qualitäten ein Urteil zu bilden und die Fähigkeit, im Kontext abwägend und bestimmend zu planen.
Computer und die ihnen unterliegende Strategie der Digitalisierung verstärken unseren Verstand in diesem Kantischen Sinne ungemein, doch sie bleiben arme Gehilfen bei der Ausbildung der Urteilskraft oder der Vernunft.