Gabriele Gramelsberger

Digitale Wissenschaft

1. Einleitung – »Wunder« der Digitalisierung

Es ist die Wissenschaft, die in ihrer (Neu-)Gier nach Neuem die Digitalisierung in die Welt gebracht hat. Bereits 1685 erfand Gottfried Wilhelm Leibniz in seiner Studierstube das binäre Rechnen und kritzelte sein Notizbuch mit Einsen und Nullen voll (Siemens 1966). Leibniz, besessen von der Automatisierung des Rechnens, entwarf nicht nur waghalsige Kalküle, darunter 1675 den Infinitesimalkalkül und 1693 den Kalkül der Lage, sondern lieferte auch gleich die Ideen für mechanische Analog- und Digitalrechner dazu mit. Seine Erfindung der Staffelwalze von 1627, die alle vier Grundrechenoperationen umsetzte, war bis weit in das 20. Jahrhundert in mechanischen Tischrechnern im Einsatz. Was Leibniz als ersten Forscher einer digitalen Wissenschaft antrieb, war Folgendes: Mit Kalkülen »[...] all die Fragen, für die das Vermögen der Anschauung nicht mehr zureicht, weiter [zu] verfolgen, so daß der hier geschilderte Kalkül [...] die Ergänzung der sinnlichen Anschauung und gleichsam ihre Vollendung darstellt. Ferner wird er, außer in der Geometrie, auch in der Erfindung von Maschinen und in den Beschreibungen der Mechanismen der Natur bisher unbekannte Anwendungen verstatten« (Leibniz 1693, 76). Diese Motivation treibt die digitale Wissenschaft und die künstliche Intelligenzforschung bis heute an, auch wenn nach Leibniz' Tod sein umfassendes Projekt der *Mathesis universalis* in Vergessenheit geriet.

Erst zweihundert Jahre später, gegen Mitte des 19. Jahrhunderts, hat die Erweiterung der sinnlichen Anschauung durch automatisierbare Kalküle wieder an Fahrt aufgenommen; und zwar mit der wirkmächtigsten Kalkülisierung: Der Kalkülisierung der Logik selbst. 1847 rekonstruierte George Boole in seiner *Mathematical Analysis of Logic* die Syllogistik, die sich seit Aristoteles nicht verändert hatte, in Form logischer Gleichungen neu. Dies gelang ihm durch die Darstellung der Kombinationsgesetze logischer Aussagen durch Auswahloperationen der Klassenbildung. So wird aus der syllogistischen Formel ›alle X sind Y‹ die logische Gleichung: $xy = y$; aus ›kein X ist Y‹ wird $xy = 0$; aus ›einige X sind Y‹ wird $v = xy$ und aus ›einige X sind keine Y‹ wird $v = x(1-y)$. Der Preis für die Logik war jedoch ihre Loslösung von der aristotelischen Metaphysik und deren Bezug auf reale Entitäten. Was blieb, war die Verortung der Logik als rein kalkülisierte Zeichensprache in der Mathematik.

Das Praktische an Booles' Kalkül war, dass sich dieser sowohl aussagen- wie klassenlogisch interpretieren ließ, indem die Wahrheitswerte

(wahr/falsch) als Nullklasse/Allklasse und diese mit den numerischen Werte 0 und 1 darstellbar sind. Booles »Algebra of Symbols« wurde in der Weiterentwicklung durch Logiker wie John Venn, William S. Jevons, Charles S. Peirce, Ernst Schröder und Giuseppe Peano zur Booleschen Algebra vereinfacht. Deren Operatoren AND, OR, NOT spielten eine wichtige Rolle für den Bau digitaler Computer. Bereits 1886 schlug Peirce vor, die sechzehn möglichen Verknüpfungen der binär interpretierten Booleschen Algebra durch eine einzige Operation, die NAND Operation (NOT, AND), darzustellen und diese auf elektrische Schaltungen zu transferieren (Pierce 1886). Heute realisiert jeder Digitalcomputer auf Schaltungsebene unzählige NAND-Operationen, aus welchen die gesamte Leistungsfähigkeit der Computer und damit der digitalen Wissenschaft resultiert.

Das wirklich Erstaunliche an der Digitalisierung ist, dass sie aus der Zuspitzung auf diese elementaren Operationen und deren binäre Zustände die gesamte Fülle ihrer Erscheinungen generiert. Bereits Leibniz wurde dafür getadelt, dass er der Trivialität von nur zwei Ziffern solche Bedeutung zumaß. »Das Ganze schien unerklärlich, und daß überdies der ›große Leibniz‹ es der Mühe wert erachtet hatte, sich mit nur zwei Ziffern zu beschäftigen, machte alles noch geheimnisvoller« (Greve 1966, 22). Wie lässt sich dieses »Wunder« der Digitalisierung verstehen, aus nur zwei Zuständen ganze Welten zu erschaffen? Oder anders gefragt: Welche Ordnungsstrukturen lassen sich auf solch elementare Operationen rückführen? Mit dieser Frage der Rekonstruktion komplexer Ordnungsstrukturen durch zahlentheoretisch elementare, maschinell umsetzbare Operationen beschäftigt sich die Theorie der Rekursivität. Ausgehend von der Nullfunktion, der Nachfolgerfunktion (n+1) sowie der Identitätsfunktion berechnet die Rekursion die Funktionswerte für gegebene Argumente durch Rückgriff auf die vorhergehenden Funktionswerte. Dabei entspricht die Anzahl der Argumente der Anzahl der Elementaroperationen, die von einer Registriermaschine durchlaufen werden, bevor sie stoppt und im n+1-ten Register der Funktionswert $f(x_1, \ldots, x_n)$ steht. Die Registriermaschine wiederholt dieselbe Operation, beispielsweise Verminderung des Wertes um 1, so oft, bis im Register der zu berechnende Wert steht. Als »rekurrierende Denkweise« stellt die Rekursion eine neue Erkenntnisform dar, die den Vorteil hat, dass sie, »in einer einzigen Formel zusammengedrängt, eine unendliche Anzahl von Syllogismen enthält« (Poincarè 1904, 10; Skolem 1923). Computer sind in diesem Sinne simple Registriermaschinen und können die rekurrierende Denkweise – oder im Fachterminus: primitiv-rekursive Funktionen – hervorragend darstellen.

Doch Computer können noch mehr. Sie können über primitiv-rekursive Funktionen hinaus auch μ-rekursive Funktionen berechnen; dies wurde in einem einzigen Jahr, 1936, mehrfach bewiesen (Kleene 1936;

Church 1936; Post 1936; Turing 1936/37). Bereits 1926 untersuchte Wilhelm Ackermann eine Funktion, die wesentlich schneller wächst als primitiv-rekursive Funktionen, denn Ackermann hatte eine weitere Konstruktionsregel in die Funktion eingeführt, die heute als μ-Operator bezeichnet wird (Ackermann 1928). Fragt man nach der Funktion des μ-Operators, so stellt sich diese als Suchoperation heraus. Wie Stephen Kleene 1936 beschrieb, besteht der Unterschied zwischen primitiv-rekursiven und μ-rekursiven Funktionen in der »operation of seeking indefinitely through the series of natural numbers for one satisfying a primitive recursive relation [is ad-ded]« (Kleene 1936, 736). μ-rekursive Funktionen sind mit Registriermaschinen berechenbar, solange für den μ-Operator der »Normalfall« vorliegt, das heißt solange es einen kleinsten Wert gibt auf den hin die Funktion zuläuft. Damit ist klar, dass eine Rechenmaschine, eine rekursive Funktion, deren μ-Operator nicht definiert ist, nicht berechnen kann, beziehungsweise dass die Maschine zu keinem Ende kommt.

Diese abstrakte mathematisch-logische Darstellung der Digitalisierung zeigt sich konkret in zwei Klassen von Programmtypen. LOOP-Programme, die nur Additionen, Wertzuweisungen und endlich oft durchlaufene Schleifen erlauben. Sie terminieren immer, weswegen sich mit ihnen keine Endlosschleifen realisieren lassen und damit nur primitiv-rekursive Funktionen berechnet werden können. WHILE-Programme hingegen zählen darüber hinaus auch nach oben und können somit μ-rekursive Funktionen berechnen, solange der Normalfall vorliegt. Allgemein gesprochen bedeutet dies, dass heutige Computer (Turing-Maschinen) LOOP- und WHILE-Programme ausführen, insofern entsprechend der Church-Turing-These jedes intuitiv berechenbare Programm durch eine Turing-Maschine gelöst werden kann (Mahr 2007). Algorithmen und Programmiersprachen (formale Sprache zur Formulierung von Datenstrukturen und Algorithmen), die diesen Anforderungen genügen, sind Turing-vollständig. Darunter fallen prozedurale Sprachen wie C oder Fortran, objektorientierte Sprachen wie C++, Fortran90 oder Java, funktionale Programmiersprachen wie LISP und Haskell als auch Logiksprachen wie Prolog.

2. Logik der Automatisierung

Ist der Computer eine Maschine, die einfache logische Operationen automatisiert ausführen und damit Zahlen rekursiv berechnen kann, so sind Computerprogramme Automatisierungen der Nutzung dieser rekursiven Berechnungen. Als erste Programmiersprache gilt Fortran (Formula Translator), die 1954 von John Backus für IBM entwickelt wurde und sich schnell in der Wissenschaft durchsetzte. Zum einen, weil

sie mit den damals marktbeherrschenden IBM-Computern an die Forschungsinstitute ausgeliefert wurde, zum anderen, weil die Idee, »a concise, fairly natural mathematical language« für die Programmierung von LOOP- und WHILE-Programmen zu verwenden, den Ansprüchen der Wissenschaftler und Ingenieure entgegenkam (Backus und Herrick 1954, 112; Metropolis et al. 1980; Gramelsberger 2010). Zuvor mussten alle Programme per Hand in Form maschinentauglicher Befehle in den Rechner eingegeben werden. Ein solcher Befehl war beispielsweise: »S(x) → Ac: Clear accumulator and add number located at position x in the Sceletrons into it« (Goldstine und von Neumann 1947, 85). Ein Programm setzte sich bereits damals aus hunderten oder tausenden von Einzelbefehlen zusammen und obwohl es nicht einfach war, eine solche Anzahl maschinentauglicher Befehle mit Papier und Bleistift zu koordinieren, lehnten die ersten Computerprogrammierer kurioser Weise Computersprachen ab – wie Backus berichtete: »At that time [1954], most programmers wrote symbolic machine instructions exclusively [...] they firmly believed that any mechanical coding method would fail to apply the versatile ingenuity which each programmer felt he possessed and constantly needed in his work« (Backus und Heising 1964, 382).

Interessanter Weise handelt es sich bei der »mechanical coding method« um den ersten Fall von Automatisierung durch Software. Viele weitere werden folgen und die Motivation ist immer noch dieselbe wie damals: »What could be done now to ease the programmer's job?« (Backus 1980, 131). Im Falle der Automatisierung der Programmierung selbst lag die Antwort in der Mathematisierung: »Once asked, the answer to this question had to be: Let him use mathematical notations. But behind that answer [...] there was the really new and hard question: Can a machine translate a sufficiently rich mathematical language into a sufficiently economical machine program to make the whole affair feasible?« (ebd.). Diese Frage ist alles andere als trivial, denn es muss gewährleistet sein, dass die programmierte Berechnung tatsächlich vom Computer ausgeführt wird. Solange ein menschlicher Programmierer seine Berechnungen direkt in Form maschinentauglicher Befehle in den Computer eingibt, kann er dies selbst überprüfen. Wird dies jedoch an eine Programmiersprache delegiert und damit automatisiert, muss sich der Programmierer auf diese verlassen können.

Zur selben Zeit ereignete sich eine weitere, folgenschwere Automatisierung menschlicher Tätigkeiten, nämlich »to control a machine or process directly from numbers« (Johnson 1956, 80). Zu diesem Zweck wurde Anfang der 1950er Jahre die Numerically Controlled Milling Machine vom Servomechanism Laboratory des Massachusetts Institute of Technology (M.I.T.) in Boston gebaut. Das M.I.T verfügte über eine der ersten Forschungsumgebung einer digitalen Wissenschaft mit Digitalrechner (Whirlwind I), Programmieren und eben der Numerically

Controlled Milling Machine. Ziel war es, numerischen Code direkt in maschinelle Bewegung zu übersetzen, um damit die industrielle Produktion der Fabrikation von Bauteilen zu automatisieren. Und wie schon zuvor wird die Automatisierung der Steuerung von maschinell-ausführbaren Prozessen selbst zum Thema der frühen Softwareforschung, wie dies die Konstrukteure James McDonough und Alfred Susskind treffend beschrieben: »The computational and coding steps can frequently be performed by machines. The use of digital computers in programming becomes particularly advantageous in those cases where the work surface can be conveniently described in terms of equations. Several parts have been machined on the numerically controlled milling machine which were programmed by Whirlwind I, the digital computer developed at Massachusetts Institute of Technology. Whirlwind I is capable of supplying at its output tapes which can be used directly to machine parts« (McDonough und Susskind 1953, 136). Dabei ist es nicht einfach, die Bewegung der Maschine – beispielsweise um ein kreisrundes Bauteil zu fertigen – mit numerischem Code darzustellen. Douglas T. Ross und Kollegen gingen daher noch einen Schritt weiter, indem sie eine erste Programmiersprache dafür entwickelten: Automatically Programmed Tools (APT), die Geometrie-Spezifikationen und Bewegungskommandos automatisierte (Ross und Pople 1956). Numerischer Code wurde durch leichter zugängliche Sprachbefehle ersetzt.

Ross nannte diesen neuen Umgang mit Computern 1956 *Gestalt programming* und markierte damit den Auftakt der Mensch-Maschine-Interaktion: »The purpose of a Gestalt system is to facilitate the transmission of general ideas as in a conversation, between a human and a computer, so that the maximum use of their respective capabilities can be made« (Ross 1958, 5). Diese frühe Mensch-Maschine-Interaktion bestand nicht mehr in der direkten Instruktion von Computern, sondern in der Programmierung von Software durch die Sprachbefehle der höheren Programmiersprachen, die dann die Computer- oder Maschinenschaltungen steuerten. Im Falle der Numerically Controlled Milling Machine ließ sich dies sogar noch weitertreiben, indem nicht die Sprache in Form getippter Sprachbefehle, sondern das Zeichnen von Bauteilen selbst automatisiert wurde. Ivan Sutherland nutzte die Bildschirme der Radartechnologie des M.I.T., um ein Computerprogramm zu entwickeln, Sketchpad, das mit Hilfe eines Lichtstifts die direkte Eingabe auf den Bildschirm ermöglichte. »The Sketchpad program by eliminating typed statements (except for legends) in favor of line drawings, opens up a new area of man-machine communication« (Sutherland 1963, 8). Eine Linie mit Sketchpad zu zeichnen, bedeutete, Anfangs- und Endpunkt auf dem Bildschirm zu markieren und den Button »draw« zu drücken, um eine lineare Interpolationsgleichung zu starten, die beide Punkte in kürzestem Abstand verband. »Drawing with the Sketchpad system

is different from drawing with an ordinary pencil and paper. Most important of all, the Sketchpad drawing itself is entirely different from the trail of carbon left on a piece of paper. Information about how the drawing is tied together is stored in the computer as well as the information which gives the drawing its particular appearance. Since the drawing is tied together, it will keep a useful appearance even when parts of it are moved« (Sutherland 1963, 21) Sketchpad markierte damit den Ursprung computerbasierter Design (CAD)-Programme, die heute Standard in der Industrie wie den Ingenieurswissenschaften sind.

Bereits 1957 orderte das U.S. Militär für über 30 Millionen Dollar Computer Numeric Control (CNC) gefertigte Bauteile (Ross 1978, 63). 1994 ist die Boeing 777 das erste technische Artefakt »to be 100 percent digitally designed using three-dimensional solids technology« (Boeing 2018). Die »digitale Kette« von den Sprach- und Zeichenbefehlen der Programmiersprachen und CAD-Programme über Computer als Übersetzer in einfache logische Operationen zu CNC-Maschinen stellt die wirkmächtige Kulturtechnik der Automatisierung des 20. und 21. Jahrhunderts dar. Erweitert wird diese seit einigen Jahren durch das 3D-Drucken und durch Machine-Learning-Methoden. Die digitale Kette ist nicht nur das Ergebnis wissenschaftlicher Forschung, sondern auch die Grundlage der digitalen Wissenschaft selbst, sowohl im Bereich der Natur- als auch der Ingenieurswissenschaften. Sie begründet die Vorhersage- wie Artefaktkultur der digitalen Wissenschaft.

3. Digitale Wissenschaft – Vorhersagekultur (Programm-Computer-Visualisierung)

Die wohl augenfälligste Folge des ersten Teils dieser digitalen Kette – also von Programmiersprachen und Programmen über computerbasierte Berechnungen zu deren Visualisierungen auf Bildschirmen – ist die Transformation der Wissenschaft in die Vorhersagekultur der digitalen Wissenschaft. Der automatisierte Zugriff auf mögliche Zukünfte ist heute selbstverständlich geworden (Gramelsberger 2010a; Seefried 2015). Grundlage dafür ist die mathematische Extrapolation, die Poincaré bereits Anfang des 20. Jahrhunderts beschrieben hat: »Wir [Mathematiker] sind daran gewöhnt zu extrapolieren; das ist ein Mittel, die Zukunft aus der Vergangenheit und aus der Gegenwart abzuleiten« (Poincaré 1914, 17). Um dieses Mittel zu nutzen, sind unzählige Berechnungen vonnöten, die vor der Entwicklung elektronischer Digitalrechner per Hand ausgeführt wurden. So rechnete beispielsweise Johannes Kepler vier Jahre lang per Hand, um aus den Beobachtungsdaten von Tycho Brahe die Formen und Gesetze der Planetenbahnen abzuleiten (Kepler

1619). Auf Basis dieser Formen und Gesetze wiederum konnten nun astronomische Vorhersagen mit Hilfe von Leibniz' Infinitesimalkalkül berechnet werden. Wie David A. Grier in seinem Buch *When Computer were Human* treffend beschrieb, wurde auf diese Weise das Eintreffen des Halley'schen Kometen jedes Jahrhundert genauer und aufwendiger berechnet. »[Andrew] Crommlin, with the assistance of an observatory colleague, Phil Crowell (1879–1949), identified the basic differential equations [Infinitesimalkalkül] that described the path of the comet and created a computing plan for the Greenwich Observatory computing staff. The computing plan had certain similarities to the plan that Clairaut had used in 1757. It located all the key objects in space and described the forces acting between them. At each step of the calculation, the computers advanced the comet, Saturn, Jupiter, and the other planets forward by a small distance. They did not worry about elliptical orbits but instead followed the direction of the forces. Once they had moved the objects, they had to recalculate all the forces. It was a slow and methodological process, one that required much grinding of Brunsvigas and other calculating machines« (Grier 2005, 121).

Das Verlangen, die Zukunft vorherzusagen, führte zur Einrichtung immer größer werdender Computergruppen, die bis Mitte des 20. Jahrhunderts per Hand mit mechanischen Tischrechnern wie der Brunsviga, die zumeist Leibniz' Staffelwalze nutzten, immer aufwendigere Vorhersagen berechneten: für die Seefahrt, die Astronomie, die Versicherungsbranche, das Militär oder die Wissenschaft. Die Entwicklung elektronischer Digitalrechner resultierte aus diesem wachsenden Bedarf an Berechnungen, wie dies die Computerpioniere Herman H. Goldstine und John von Neumann für komplexere Probleme vorrechneten. Für eine Trajektorie (Bewegungsbahn), so schätzten sie, werden 750 Multiplikationen benötigt. Dafür musste eine Person sieben Stunden permanent rechnen. Für komplexere Probleme wie die Ausbreitung einer Welle in zwei, drei oder sogar vier Dimensionen (Raum und Zeit) genügte jedoch eine Trajektorie nicht mehr. »For such problems the number of multiplications rises enormously due to the number of lattice points. It is not unreasonable to consider between 10^6 and 5×10^6 multiplications for a 3-variable problem and between 2.5×10^7 and 2.5×10^8 multiplications for a 4-dimensional situation. Hence these are roughly equivalent to 1,300 to 6,700 trajectories and to 33,000 to 330,000 trajectories« (Goldstine und von Neumann 1946, 12). Ein menschlicher Computer müsste für letztere Situation über 140 Jahre ununterbrochen rechnen; ENIAC (Electronic Numerical Integrator and Computer), der erste elektronische Universalrechner von 1946, benötigte dafür nur noch vier Tage; heutige Supercomputer erledigen dies in Mikrosekunden.

Insbesondere von Neumann verschrieb sich der Entwicklung und Etablierung elektronischer Digitalrechnern für die Wissenschaft. Durch die

schiere Rechengewalt erhoffte er sich wissenschaftliche Durchbrüche für den Bereich komplexer Systeme, die erst durch schnelle Digitalrechner zugänglich wurden (von Neumann 1954). Denn für komplexe Systeme, die Rückkopplungseffekte berücksichtigen, gibt es schlichtweg keine algebraisch formulierten Lösungen. Ein möglicher Ausweg aus diesem Dilemma ist die numerische Berechnung eines solchen komplexen Systems und eben dazu braucht es leistungsfähige Computer. Allerdings sind numerische Berechnung komplexer Systeme – auch Computersimulation genannt – nur näherungsweise (approximativ) möglich. Das bedeutet, dass die Ergebnisse nie exakt sind, sondern nur Wahrscheinlichkeitsaussagen ermöglichen.

Das klassische Beispiel komplexer Systeme ist das Wettersystem. Wetter ist ein komplexes System, das sich mathematisch als rückgekoppeltes, hydro- und thermodynamisches Gleichungssystem für die sieben meteorologischen Variablen formulieren lässt. Diese Variablen sind die Temperatur, die Luftfeuchte und -dichte, der Luftdruck sowie die Windgeschwindigkeit in drei Richtungen. Die meteorologischen Variablen sind komplex miteinander verbunden, was dazu führt, dass ihre Wirkung aufeinander nicht vorhersagbar ist. Da es keine exakte, algebraische Lösung für das Wettervorhersageproblem gibt, lässt sich dieses nur numerisch, näherungsweise simulieren. Zum Problem der Approximation kommt noch ein weiteres, denn leicht variierende Anfangsbedingungen können in komplexen Systemen zu stark variierenden Resultaten führen. Dies hatte bereits Poincaré erkannt. »Es kann der Fall eintreten, dass kleine Unterschiede in den Anfangsbedingungen große Unterschiede in den späteren Erscheinungen bedingen; ein kleiner Irrtum in den ersteren kann einen außerordentlich großen Irrtum für die letzteren nach sich ziehen. Die Vorhersage wird unmöglich und wir haben eine ›zufällige Erscheinung‹« (Poincaré 1904, 56–57).

Obwohl Wettervorhersagen mathematisch eigentlich unmöglich sind, haben wir uns an sie gewöhnt. Wettervorhersagen sind das Paradebeispiel der digitalen Wissenschaft und der durch sie etablierten Vorhersagekultur (Heymann et al. 2017). Klimavorhersagen wären ein weiteres Paradebeispiel. Sie basieren, wie die Wettervorhersagen, auf der numerischen Simulation des rückgekoppelten, hydro- und thermodynamischen Gleichungssystems der sieben meteorologischen Variablen. Allerdings geben Klimamodelle statistisch gemittelte Projektionen möglicher Klimazukünfte wieder (Gramelsberger und Feichter 2011). Durch die statistische Mittelung sind Klimaprojektionen – im Unterschied zu Wettervorhersagen – über einen langen Zeitraum möglich. Dabei führen sie uns ein Spiegelbild unseres kollektiven Handelns vor Augen, das Vorhersagewissen zum dominanten Orientierungswissen in einer Vorhersagekultur werden lässt.

4. Digitale Wissenschaft – Artefaktkultur (CAD-Computer-CNC/3D)

Erweitert man die digitale Kette durch CNC-Maschinen und 3D-Drucker, dann gelangt man von der Vorhersagekultur zur Artefaktkultur der digitalen Wissenschaft. Die Artefaktkultur knüpft direkt an die Automatisierung menschlicher Tätigkeiten an; und zwar der bereits beschriebenen Idee folgend, »to control a machine or process directly from numbers« (Johnson 1956, 80). Die industriell gefertigten Produkte von heute – Autos, Flugzeuge, Möbel, Gebrauchsgegenstände etc. – sind direkte Folgen dieser ersten Welle der Automatisierung, die menschliche Bewegung in maschinelle transformierte. Industrieroboter in nahezu menschenleeren Werkhallen sind *das* Paradebeispiel der Artefaktkultur der digitalen Ingenieurswissenschaft. Doch die Digitalisierung und mit ihr die Automatisierung entwickeln sich weiter. Zum einen, indem die Automatisierung der Logik zunehmend die Boolesche Algebra überschreitet; zum anderen, indem immer größere Bereiche der Lebenswelt unter die Digitalisierung und Automatisierung subsummiert werden.

Booles Projekt der Kalkülisierung der Logik und Pierces Mechanisierung der Booleschen Algebra durch elektrische Schaltungen stellten den Auftakt dar, das großangelegte, neuzeitliche Projekt der Operationalisierung des (menschlichen) Geistes maschinell umzusetzen (Gramelsberger 2014). Das Produkt dieser Umsetzung ist der, dem Paradigma der Turing-Machine verschriebene Digitalcomputer (Turing 1936). Doch schon Alan Turing dachte über sein eigenes Paradigma hinaus, als er in dem Essay *Computing Machinery and Intelligence* von 1950 Maschinen beschrieb, »whose manner of operation cannot be satisfactorily described by its constructors because they have applied a method which is largely experimental« (Turing 1950, 460). Experimentell trainierte Maschinen kennen wir heute als Deep Neural Networks (DNNs), die eine der wichtigsten Grundlagen des Maschinenlernens darstellen (Nilsson 2009). Insbesondere die aktuell in der Bild- und Mustererkennung so erfolgreichen Convolutional Neural Networks (CNNs) kalkülisieren und mechanisieren Wahrnehmungsfunktionen des visuellen Kortex (Fukushimas 1988). Kombiniert mit Maschinen und Robotern, die über entsprechende Sensoren mit der Umwelt verbunden sind, kündet sich hier nach der Automatisierung der Bewegung die Automatisierung der Wahrnehmung an. Welche Folgen dies für unsere Lebenswelt haben wird (Industrie 4.0), ist noch nicht absehbar, denn diese neuen Maschinen sind – im Unterschied zu Menschen – permanent miteinander vernetzt und selten offline. Sie beginnen, sich aus den Maschinenhallen zu befreien und in die Alltagswelt zu migrieren. Dabei sind sie uns in punkto Geschwindigkeit, Stärke und der Verarbeitung großer Datenmengen durch Mustererkennung weit überlegen.

Doch die Digitalisierung und Automatisierung machen auch vor antagonistischen Bereichen wie Gefühlen oder der Biologie nicht Halt, die bislang als das Nicht-Mechanisierbare, Nicht-Kalkülisierbare oder Nicht-Maschinelle galten. Affective Computing (Picard 1997, 2015) und Synthetische Biologie (Ellowitz und Leibler 2000) heißen die neuen Operationalisierungsprojekte der digitalen Wissenschaft. »Gefühle« digitalisieren bedeutet folgender Logik zu folgen: »objective data related to emotion is more believable than verbal reports about feelings. Shared affective data can improve communication between people and lead to better understanding and sometimes to beneficial changes in behavior« (Picard 2015, 12). Solche objektiven Daten sind nicht-invasive, physiologische Sensormessungen des Hautwiderstandes, der Atmung, der Herzfrequenz oder der Stimmfrequenz sowie automatisierte Gesichts-, Gesten- und Emotionserkennungsdaten durch Bildanalysen. Die automatisierte Affekterkennung – Affective Computing genannt – eliminiert dabei den menschlichen Beobachter durch Algorithmen. Ob wir den »objektiven Daten« und der automatisierten Affekterkennung in Zukunft mehr Glauben schenken als unserer eigenen Wahrnehmungen, wird sich zeigen, doch schon jetzt spiegeln uns Computerprogramme immer häufiger maschinenlogisch interpretierte Gefühle wider.

Das mit Abstand jedoch avancierteste Operationalisierungsprojekt ist das der Digitalisierung und Automatisierung der Biologie respektive des Lebens selbst. »Programing Life«, wie der Slogan der Biological Computing Group von Microsoft lautet (Microsoft Research 2016), dokumentiert nicht nur die Zielrichtung dieser neuen Form der digitalen Wissenschaft, sondern macht auch deutlich, dass Forschung im 21. Jahrhundert und damit digitales Wissen zunehmend von einigen IT-Großkonzernen dominiert wird. Dabei geht es um Folgendes: »Trimming microorganisms genetically to optimize their productivity is therefore a key technology of immense industrial importance« (Tomita 2001, 1091). Trimmen bedeutet, dass Forscher einen Organismus respektive dessen genetischen Code mit Hilfe von CAD-Programmen wie TinkerCell oder F*rna*kenstein designen (Schwille 2011). Danach wird der designte Code mit DNA-Synthesizern »ausgedruckt« und in genetisch modifizierte Bakterien (*E. coli*) oder Hefezellen implementiert. Mit etwas Glück und Geschick reproduziert sich der synthetisierte Organismus und produziert Proteine, Lipide und andere Biomoleküle, die zur Herstellung von Medikamenten, Biokraftstoffen oder Lebensmitteln benötigt werden (Friedrich und Gramelsberger 2011).

Diese Idee des »Engineering of Biology« stellt die neueste Ausprägung der Artefaktkultur der digitalen Wissenschaft dar (Endy 2005). Ihre Entwicklung ist industriell getrieben. Doch auch der alte Traum der Menschheit, Leben künstlich herzustellen, findet durch diese Art von Biologie neuen Aufwind. Bereits Ende des 19. Jahrhunderts gestand der

Biologe Jacques Loeb dem Physiker Ernst Mach: »The idea is now hovering before me that man himself can act as a creator, even in the living Nature, forming it eventually according to his will. Man can at least succeed in a technology of living substances« (Loeb zitiert in Pauly 1987, 5). Neben Loeb waren es vor allem Alfonso L. Herrera und Stéphane Leduc, die zu Beginn des 20. Jahrhunderts mit künstlichem Leben experimentierten. Insbesondere Leduc formulierte in seinem Buch *La Biologie synthétique* von 1912 die moderne Vision dieses Traums: »Just as synthetic chemistry began with the artificial formation of the simplest organic products, so biological synthesis must content itself at first with the fabrication of forms resembling those of the lowest organisms« (Leduc 1912, 113). Fast hundert Jahre später verkünden Craig Venter und Kollegen die *Creation of a Bacterial Cell Controlled by a Chemically Synthesized Genome* und nennen den artifiziellen Organismus JCVI-syn1.0 (Gibson et al. 2010). JCVI-syn1.0 ist zwar nicht komplett artifiziell reproduziert, da hierfür immer noch die Hilfe der Natur benötigt wird, doch JCVI-syn1.0 wird als »proof of principle for producing cells based on computer-designed genome sequences« angesehen (Gibson et al. 2010, 55).

5. Digitale Wissenschaft und Lebenswelt – Gewöhnungs- und Kolonialisierungseffekte

Digitale Wissenschaft hat eine umfangreiche Vorhersage- und Artefaktkultur hervorgebracht, die sich aktuell in ihrem Handlungsspielraum durch künstliche Intelligenz (KI) und künstliches Leben (KL) enorm erweitert. Künstliche, präzedenzlose Moleküle, durch Molecular-Modelling-Verfahren errechnet, erweitern die Ontologie künstlicher Entitäten (Jansen und Schön 2006). Diese Trias wird die Lebenswelt wie nie zuvor umgestalten. Da Digitalisierung in der Regel mit Automatisierung einhergeht, haben und werden die Folgen der Automatisierung unsere Vorstellung von Selbstbestimmung beeinflussen. Denn die Automatisierung delegiert menschliche Fähigkeiten an Maschinen, die diese dann noch dazu besser, präziser, schneller, vernetzter und günstiger ausführen können. Dass dies Folgen für die Arbeitswelt hatte und haben wird, versteht sich von selbst.

Vergegenwärtigt man sich die Dynamik der Digitalisierung anhand der dargestellten Zeitleiste (Abb. 1), dann wird die explosionsartige Entwicklung deutlich. Damit ist nicht nur die Durchdringung der Lebenswelt mit Milliarden von Endgeräten gemeint, sondern die Transformation der Digitalisierung selbst: aus IT mit Stand-alone-Geräten wurde zu Beginn der 1990er Jahre mit der Entwicklung des World Wide Webs

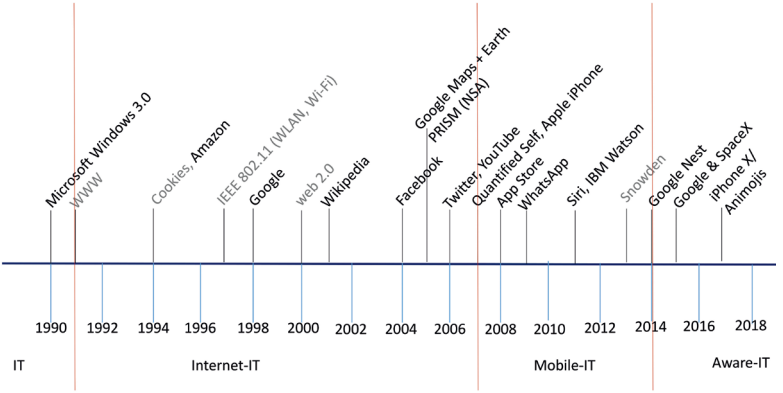

Abbildung 1: Timeline Digitalisierung (Quelle: Gabriele Gramelsberger)

Internet-IT; aus Internet-IT wurde Mitte der 2000er Jahre durch Smart Phones Mobile-IT; und aus Mobile-IT wird heute durch die zunehmende Sensorialisierung der Endgeräte Aware-IT. Smart Phones sind längst keine Telefone mehr, sondern High-Tech-Messinstrumente, die über mehr Sensorik verfügen als mancher Satellit. Im Unterschied zu Satelliten tragen wir sie jedoch permanent und allzeit online bei uns. Ein Smartphone wie das iPhone X beispielsweise verfügt über einen Beschleunigungssensor, einen Magnetometer, einen Barometer, ein Sonar. Es gibt einen Annäherungssensor, der die Bildschirmbeleuchtung und die Touch-Funktion abschaltet, wenn man es an den Kopf hält, und einen Lichtsensor, der die Umgebungshelligkeit erfasst. Doch alleine die beiden letzteren Sensoren können wesentlich mehr: Sie messen Körperfunktionen. »Nach Apples Vorstellung können so schon Puls, Blutdruck, Körperfettanteil, Sauerstoffsättigung und Durchblutung ermittelt werden« (Becker 2017). Wie klein diese Sensoren sind, sieht man an dem verbauten Barometer. Er misst gerade mal 2,5 mal 2 Millimeter und ist weniger als einen Millimeter hoch. Die Sensorik in Smart Phones ist mittlerweile so ausgereift, dass damit Forschung betrieben werden kann (RWTH Aachen 2018).
Die Entwicklung der Digitalisierung ist seit Beginn auch durch IT-Großkonzerne getrieben, doch mit Aware-IT versteht sich Kapitalismus neu als Datenkapitalismus, der Mark Weisers Vision des Ubiquitous Computing realisiert (Weiser 1991). Dieser Datenkapitalismus der Generierung und Analyse großer Datenmengen ist mithilfe der digitalen Wissenschaft entstanden, er treibt diese aber auch – in zunehmender Konkurrenz mit den IT-Großkonzernen – an. Was dies für gesellschaftliche Strukturen und Institutionen bedeutet, ist noch wenig einschätzbar. Denn digitale (öffentlich finanzierte) Wissenschaft und digitale Forschung der IT-Großkonzerne teilen sich das Herrschaftswissen der Digitalisierung

und Automatisierung. Dieses Herrschaftswissen wird zunehmend zum Orientierungswissen der technologisierten Lebenswelt, wie es sich exemplarisch an der Vorhersagekultur zeigt. Dass die rationale Prognostik der Vorhersagekultur nur digital verfügbar gemacht werden kann, liegt auf der Hand. Schnellere Computer und bessere Analyse- und Vorhersagealgorithmen bestimmen die Reflektionen auf mögliche Zukünfte. Dazu gehören heute weit mehr als Wettervorhersagen und Klimaprojektionen, die Milliardeninvestitionen in Klimaschutz, aber auch in die Energiewirtschaft motivieren. Angefangen von den Verhaltensvorhersagen des Predictive Policing bis hin zum Predictive Shipping von Amazon, das schon einmal vorsorglich das verschickt, was wir morgen mit verblüffend hoher Wahrscheinlichkeit kaufen werden, bestimmen Vorhersagealgorithmen immer größere Bereiche der Lebenswelt.

Doch wie steht es um die Artefaktkultur? Hier stehen wir vor dem Paradox, dass das artefaktische Orientierungswissen der digitalen Wissenschaft und Forschung in erster Linie den Maschinen zugutekommt. Die Rolle des Menschen als Homo Faber und die Rolle der Maschinen als das Gemachte beginnen sich zu verkehren. Natürlich konstruieren und programmieren sich die Maschinen bislang (noch) nicht selbst, aber das heißt nicht – mit Hans Jonas gesprochen –, dass der Mensch nicht zunehmend selbst »unter die Objekte der Technik« fällt (Jonas 1979, 47). Norbert Wiener hatte dies zu Beginn der Entwicklung der Digitalisierung als *The Human Use of Human Beings: Cybernetics and Society* bereits vorhergesagt (Wiener 1950). Doch erst heute beginnen wir die Folgen dieser Kolonialisierung des Menschen durch seine eigenen Technologien zu ahnen. Digital umgeschriebener genetischer Code von Neugeborenen durch Gene Editing gehört hier ebenso dazu wie die für Menschen unerreichbare Präzession von Operationsrobotern. Doch dieses »unter die Objekte der Technik fallen« hat noch eine weitere Komponente, die sich in den gigantischen Projekten der Maschinenlesbarkeit der gesamten Welt zeigen; heißen diese Wikipedia, Google, oder anders. Denn artefaktisches Orientierungswissen ist algorithmen- und datengetrieben und die Maschinenlesbarkeit der menschlichen Wissenswelten ist die Basis der automatisierten Ökonomie – ergänzend zum Datenkapitalismus. Diese Entwicklung ist jedoch einseitig, denn uns bleibt die Lesbarkeit der maschinischen Wissenswelten vorenthalten. Selbst Programmierer haben den Zugang zu ihren eigenen (selbstlernenden) Algorithmen verloren und können nicht mehr erklären, wie und warum ihre Systeme so und nicht anders entscheiden. Auch wenn die Europäische Kommission das Recht auf Erklärbarkeit algorithmischer Entscheidungen in ihrer Datenschutzverordnung ab 2019 verpflichtend festgeschrieben hat (Goodman und Flaxman 2016), so liegt es doch in der Natur der Automatisierung, uns Handlungsspielräume systematisch zu entziehen. Neben der Bewegung und Produktion sowie der Wahrnehmung

gerät nun das Wissen selbst unter das Automatisierungskapital. Eben darin liegt der Clou der Maschinenlesbarkeit der Welt, die sich KI-basierte Expertensysteme wie IBM Watson zunehmend zu Nutze machen (Ferrucci et al. 2010). Maschinenlesbarkeit ist dabei nur ein anderer Begriff für automatisierbare Zugänglichkeit von Wissen. Das Internet 2.0, Common-Sense-Datenontologien wie Concept Net der M.I.T. Media Lab Ausgründung Luminoso oder die Standardisierung der Maschine-Maschine-Kommunikation (M2M) des Internets der Dinge und der Industrie 4.0 werden zunehmend maschinenaffiner gestaltet, um die automatisierte Ökonomie von uns abzukoppeln. Es ist nur eine Frage der Zeit, bis die KI-basierte Expertensysteme die Maschinen und Roboter trainieren.

Es ist in der Tat ein Wunder, dass sich aus nur zwei Zuständen ganze Welten erschaffen lassen. Der Mensch wie auch seine Lebenswelt passen allerdings nur bedingt in diese maschinenlogischen, digitalen Welten. Was oder wer sich nicht rekonfigurieren lässt, bleibt außen vor. Noch tragen wir die Portale zu den maschinischen Wissenswelten in Form von Smart Phones und Tablets mit uns herum, auch wenn die Maschinen-Maschinen-Kommunikation die des Menschen mit seinen Maschinen bei weitem übersteigt. An direkten Schnittstellen wie dem Internet of Brain oder Neurochips wird bereits geforscht. Das alles mag sich wie Science Fiction anhören, doch eines ist gewiss: Es ist der Mensch, der sich an seine Technologie anpasst; nicht umgekehrt. Das Zauberwort heißt »Gewöhnung.« Wir werden uns daran gewöhnen, dass Algorithmen für uns Entscheidungen treffen; was sie bereits tun. Wir werden uns daran gewöhnen, dass Roboter uns operieren, pflegen und trainieren, was sie bereits tun. Und wir werden uns daran gewöhnen, dass Maschinen ein Teil unseres Körpers werden, wie Neuroimplantate wie künstliche Retinas hinreichend belegen. Doch, und das ist die Frage, werden Maschinen ein Teil von uns sein, oder werden wir Teil eines global umspannenden, vernetzten und miteinander kommunizierenden Maschinennetzes sein? Und sind wir das nicht bereits?

Literatur

Ackermann, Wilhelm (1928): »Zum Hilbertschen Aufbau der reellen Zahlen«, in: *Mathematische Annalen*, 99, 118–133.

Backus, John und Harlan Herrick (1954): »IBM 701 Speedcomputing and other automatic-programmingsystems«, in: *Proceedings of the Symposium Automatic Programming Digital Computers*, Washington: Office of Naval Research, Department of the Navy, 106–113.

Backus, John und William P. Heising (1964): »FORTRAN«, in: *IEEE Trans. Electron. Comp.* 13, 382–385.

Backus, John (1980): »Programming in America in the 1950s«, in: Nicholas Metropolis, Jack Howlett und Gian-Carlo Rota (Hg.): *A History of Computing in the Twentieth Century*, New York: Academic Press, 125–135.

Becker, Leo (2017): »Apple-Patent: iPhone-Sensoren erfassen Körperdaten«, in: *Heise-online*, 08.08.2017. Online unter: https://www.heise.de/mac-and-i/meldung/Apple-Patent-iPhone-Sensoren-erfassen-Koerperdaten-3795605.html [Zugriff 22.05.2019].

Boeing (2018): *Amazing Technology Facts from Boeing Commercial Airplanes*. Online unter: http://boeing.mediaroom.com/2002-09-20-Amazing-Technology-Facts-from-Boeing-Commercial-Airplanes [Zugriff 10.12.2018].

Boole, George (1847): *The Mathematical Analysis of Logic: Being an Essay towards a Calculus of Deductive Reasoning*, Cambridge: Macmillan.

Church, Alonzo (1936): »A note on the Entscheidungsproblem«, in: *The Journal of Symbolic Logic*, 1(1), 40–41.

Elowitz, Michael B. and Stansilas Leibler (2000): »A synthetic oscillatory network of transcriptional regulators«, in: Nature, 403(6767), 335–338.

Endy, Drew (2005): »Foundations for engineering biology«, in: *Nature*, 438, 449–453.

Ferrucci, David, Eric Brown, Jennifer Chu-Carroll et al. (2010): »Building Watson: An overview of the DeepQA project«, in: *AI Magazine*, 313, 59–79.

Friedrich, Kathrin und Gabriele Gramelsberger (2011): »Techniken der Überschreitung. Fertigungsmechanismen ›verlässlich lebensfähiger biologischer Entitäten‹«, in: *Zeitschrift für Medienwissenschaften*, 4, 15–21.

Fukushima, Kunihiko (1988): »Neocognitron: A hierarchical neural network capable of visual pattern recognition«, in: *Neural Networks*, 1(2), 119–130.

Gibson, Dan et al. (2010): »Creation of a Bacterial Cell Controlled by a Chemically Synthesized Genome«, in: *Science*, 329(5987), 52–56.

Goodman, Bryce und Seth Flaxman (2016): »European Union regulations on algorithmic decision-making and a ›right to explanation‹«, in: *arXiv:1606.08813v3* (31.8.2016).

Gramelsberger, Gabriele (2010): »Story Telling with Code«, in: Andrea Gleininger, Georg Vrachliotis (Hg.): *Code. Zwischen Operation und Narration*, Basel: Birkhäuser, 29–40.

Gramelsberger, Gabriele (2010a): »Die kausale Mechanistik der Prognosen aus dem Computer«, in: Heinrich Hartmann, Jakob Vogel (Hg.): *Zukunftswissen. Prognosen in Wirtschaft, Politik und Gesellschaft seit 1900*, Frankfurt/New York: Campus, 213–230.

Gramelsberger, Gabriele (2014): *Operative Epistemologie. Begriffstheoretische Studie zur Erkenntnis- und Formkraft der Mathematik*, (Habilitationsschrift am Fachbereich Gesellschafts- und Geschichtswissenschaften), Darmstadt: Technischen Universität Darmstadt.

Gramelsberger, Gabriele und Johann Feichter (Hg.) (2011): *Climate Change and Policy. The Calculability of Climate Change and the Challenge of Uncertainty*, Heidelberg und Berlin.

Greve, Hermann J. (1966): »Entdeckung der binären Welt«, in: Siemens Aktiengesellschaft (Hg.): *Herrn von Leibniz' Rechnung mit Null und Eins*, Siemens AG: Berlin, München 21–31.

Grier, David A. (2005): *When Computers were Human*, Princeton University Press: Princeton.

Goldstine, Herman H. und John von Neumann (1946): »On the Principles of Large Scale Computing Machines«, in: John von Neumann: *Collected Works*, 5. Bd.: *Design of Computers, Theory of Automata and Numerical Analysis* (hrsg. von A. H. Taub 1963), Oxford: Pergamon Press, 1–32.

Goldstine, Hermann H. und John von Neumann (1947): »Planing and Coding Problems for an Electronic Computing Instrument, Part II, Vol 1«, in: John von Neumann: *Collected Works*, 5. Bd.: *Design of Computers, Theory of Automata and Numerical Analysis* (hrsg. von A. H. Taub 1963), Oxford: Pergamon Press, 80–151.

Heymann, Matthias, Gabriele Gramelsberger und Martin Mahony (Hg.) (2017): *Cultures of Prediction in Atmospheric and Climate Science: Epistemic and Cultural Shifts in Computer-based Modelling and Simulation*, (Routledge Environmental Humanities), London: Routledge.

Jansen, Martin J. und Christian Schön (2006): »›Design‹ in der chemischen Synthese – eine Fiktion? «, in: *Angewandte Chemie*, 118(21), 3484–3490.

Johnson, E. C. (1956): »A Numerically Controlled CAM-Milling Machine«, in: *IRE Transactions on Industrial Electronics*, 80–86.

Jonas, Hans (1979): *Das Prinzip Verantwortung. Versuch einer Ethik für die technologische Gesellschaft*, Frankfurt/M.: Suhrkamp.

Kepler, Johannes (1619): *Harmonice mundi*, (Gesammelte Werke, 2. Bd. 1940), München: Beck.

Kleene, Stephen C. (1936): »General recursive functions of natural numbers«, in: *Mathematische Annalen*, 112, 727–742.

Leibniz, Gottfried W. (1693): »De analysi situs«, in: ders.: *Philosophische Werke*, (hrsg. von Artur Buchenau, Ernst Cassirer 1996), 1. Bd., Meiner: Hamburg, 69–76.

Leduc, Stéphane (2012): *La Biologie synthétique*, Paris: Poinat.

Mahr, Bernd (2007): »Womit können wir rechnen?«, in: *Spektrum der Wissenschaft*, (Spezial: Ist das Universum ein Computer?), 27–35.

McDonough, James C. und Alfred W. Susskind (1953): »A Numerically Controlled Milling Machine«, in: *International Workshop on Managing Requirements Knowledge*, (New York City: American Institute of Electrical Engineers), 133–137.

Metropolis, Nicholas, Jack Howlett und Gian-Carlo Rota (Hg.) (1980): *A History of Computing in the Twentieth Century*, New York: Academic Press.

Microsoft Research: *Biological Computing*, in: https://www.microsoft.com/en-us/research/ group/biological-computation/ [Zugriff: 10.8.2016].

Neumann, John von (1954): »Entwicklung und Ausnutzung neuerer mathematischer Maschinen«, in: John von Neumann: *Collected Works*, 5. Bd.:

Design of Computers, Theory of Automata and Numerical Analysis (hrsg. von A. H. Taub 1963), Oxford: Pergamon Press, 248–268.
Nilsson, Nils J. (2009): *The Quest for Artificial Intelligence. A History of Ideas and Achievements*, New York: Cambridge University Press.
Pauly, Philip J. (1987): *Controlling Life: Jacques Loeb and the Engineering Ideal in Biology*, New York: Oxford University Press.
Peirce, Charles Sanders (1886): »Letter, Peirce to A. Marquand«, in: ders.: *Writings of Charles S. Peirce*, 5. Bd., Indianapolis: Indiana University Press 1993, 421–423.
Picard, Rosalind (1997): *Affective Computing*, Cambridge: The MIT Press.
Picard, Rosalind (2015): »The Promise of Affective Computing«, in: Rafael Calvo et al. (Hg.): *The Oxford Handbook of Affective Computing*, Oxford: Oxford University Press, 12–19.
Poincaré, Henri (1904): *Wissenschaft und Hypothese*, Leipzig: Teubner.
Poincaré, Henri (1914): *Wissenschaft und Methode*, Teubner: Leipzig.
Post, Emil L. (1936): »Finite combinatory processes, formulation I«, in: *The Journal of Symbolic Logic*, 1(1), 103–105.
Ross, Douglas T. und H.E. Jr. Pople (1956): *Automatic Programming of Numerically Controlled Machine Tools*, (Report Nos.6873-IR-1 und 6873-IR-2), Cambridge: MIT Servo Lab.
Ross, Douglas T. (1958): »Gestalt Programming: A New Concept in Automatic Programming«, in: *AFIPS Proceedings*, (WJCC Western Point Computer Conference from 7-9.2.1956 at San Francisco), 5–9.
Ross, Douglas T. (1978): »Origins of the APT Language for Automatically Programmed Tools«, in: *ACM SIGPLAN Notices*, 13(8), 61–99.
RWTH Aachen (2018): *Phyphox*, (Physikalisches Institut), Aachen: Rheinisch-Westfälische Technische Hochschule. Online unter: https://phyphox.org/de/home-de/.
Seefried, Elke (2015): *Zukünfte. Aufstieg und Krise der Zukunftsforschung 1945–1980*, Berlin: de Gruyter Oldenbourg.
Schwille, Petra (2011): »Bottom-Up Synthetic Biology: Engineering in a Tinkerer's World«, in: *Science*, 333(1252), 1252–1254.
Siemens Aktiengesellschaft (Hg.) (1966): *Herrn von Leibniz' Rechnung mit Null und Eins*, Berlin, München: Siemens AG.
Skolem, Thoralf (1923): »Begründung der elementaren Arithmetik durch die rekurrierende Denkweise ohne Anwendung scheinbarer Veränderlicher mit unendlichem Ausdehnungsbereich«, in: *Skrifter utgit. av Videnskapsselskapet i Kristiania*, 1. Mat.-nat. kl, 6, 1–38.
Sutherland, Ivan (1963): *A man-machine graphical communications system*. (M.I.T. Technical Report 296), Boston: M.I.T. Lincoln Laboratories.
Tomita, Masaru (2001): »Towards Computer-aided Design (CAD) of useful microorganisms«, in: *Bioinformatics*, 17, 1091–1092.
Turing, Alan (1936/37): »On Computable Numbers, with an Application to the Entscheidungsproblem«, in: *Proceedings of the London Mathematical Society*, 42(2), 230–265 und 544–546.

Turing, Alan (1959): »Computing Machinery and Intelligence«, in: *Mind*, 49, 433–460.
Weiser, Mark (1991): »The computer for the 21st century«, in: *Scientific American*, 265(3), 94–104.
Wiener, Norbert (1950): *The Human Use of Human Beings: Cybernetics and Society*, Boston, MA: Houghton Mifflin.