PROGRAMMIEREN

HARALD SACK

ANEKDOTE Als berüchtigt, ausschweifend oder gar verrückt galt er seinen Zeitgenossen, der Vater von Augusta Ada King, da er als Literat gerne den wilden Mann spielte, doch blieb er der Nachwelt besser bekannt als Lord Byron, der romantische Poet und Freiheitskämpfer. Von ihrem berühmten Vater erbte sie sicherlich die Kunst, mit Wörtern umzugehen, aber auch ihre Neugier. 1843 veröffentlichte Augusta Ada King, Countess of Lovelace, eine Reihe von Notizen, in denen sie die *Analytical Engine* des britischen Mathematikers Charles Babbage beschrieb, einer mechanischen, frei programmierbaren Rechenmaschine, die heute als Meilenstein auf dem Weg der Entwicklung zum modernen Computer gilt, die aber lediglich als Entwurf existierte und niemals gebaut wurde.[1] Babbage hatte die erste Beschreibung seiner bahnbrechenden Maschine bereits 1837 publiziert. Fünf Jahre später demonstrierte er das Konzept der Analytical Engine vor ausgewählten Wissenschaftlern. Unter ihnen war der junge italienische Mathematiker Federico Luigi Menebrae, der später zum italienischen Ministerpräsidenten aufsteigen sollte, der eifrig Notizen anfertigte und aus diesen zusammen mit einigen Kommentaren Babbages 1842 seinen Artikel „*Notions sur la machine analytique de Charles Babbage*" veröffentlichte.[2]

Babbage hatte der damals 17-jährigen Ada kurz nach ihrer ersten Begegnung den halbfertigen Prototypen seiner *Difference Engine,* einem Vorläufer der Analytical Engine, vorgeführt, was großen Eindruck bei Ada hinterließ. Damit

1 Dies lag in der Hauptsache daran, dass es Babbage nicht gelang, die notwendige Finanzierung aufzutreiben. Das britische Parlament weigerte sich, Babbages utopisch anmutendes Forschungsprogramm zu finanzieren. Die Analytical Engine sollte von einer Dampfmaschine angetrieben werden, aus 55.000 feinmechanisch hochpräzise gearbeiteten Einzelteilen bestehen bei einer geplanten Dimension von 19 Metern mal 3 Metern. Hatte es doch die von Babbage projektierte Vorgängermaschine, die Difference Engine, bereits mit der damals sagenhaften Summe von 17.000 Pfund Sterling gefördert und dafür lediglich einen halbfertigen Demonstrator erhalten. (Vgl. Gleick: The Information, S. 104).

2 Vgl. Menabrea/Lovelace: Sketch of the Analytical Engine invented by Charles Babbage with notes by the translator, S. 666–731.

begann ihre Zusammenarbeit und er ließ sie teilhaben an seinem neuesten Projekt, der universell programmierbaren Analytical Engine, einer Maschine, der Babbage die restlichen 38 Jahre seines Lebens widmen sollte. Natürlich verstieß Ada mit ihrem Interesse an Mathematik und ihrem Umgang mit einem der führenden Mathematiker Englands gegen alle damals geltenden Regeln gesellschaftlicher Konvention. Im Alter von 19 Jahren heiratete sie William King, den 8. Baron King und Earl of Lovelace, der, so ungewöhnlich es erscheinen mag, ebenfalls über eine mathematische Ausbildung verfügte. Da Frauen der Zutritt zu Bibliotheken und Universitäten untersagt war, ließ er sich ihr zuliebe sogar in die Londoner Royal Society aufnehmen, wo er für sie wissenschaftliche Artikel und Abhandlungen kopierte.[3]

Ada hatte Manebraes Abhandlung über die *Analytical Engine* aus dem Französischen ins Englische übersetzt und Babbage ermutigte sie, ihre eigenen Gedanken mit in den Artikel einfließen zu lassen. Daraufhin erweiterte Ada ihre Übersetzung um eine Diskussion der Möglichkeiten, wie dieser „Computer" programmiert werden könnte.[4] Die Analytical Engine könne algebraische Muster weben, ebenso wie Jacquards Webstühle Blumen und Blätter weben. Neben der Möglichkeit von Programmverzweigungen, Verschachtelungen, Sprungbefehlen und Symbolmanipulationen diskutierte sie, was die Maschine theoretisch leisten könne und was nicht. Sie verstand, dass die Analytical Engine nicht nur eine einfache Rechenmaschine war, sondern tatsächlich die Möglichkeiten eines modernen Computers vorwegnahm. Während Babbage den Einsatz seiner Maschine lediglich in der Berechnung von Zahlentabellen sah, die für die Schiffsnavigation benötigt wurden und manuell erstellt oder kopiert oftmals zahlreiche Fehler enthielten[5], erkannte Ada deren wahres Potential. Neben Zahlen könnte die Maschine auch Buchstaben manipulieren und sogar Musik komponieren. Allerdings nennt sie uns auch deren Grenzen: „Die Maschine kann [nur] das tun, was wir ihr zu befehlen vermögen, sie kann [unserer] Analyse folgen. Sie hat jedoch keine Fähigkeit

3 Vgl. Kim/Toole: Ada and the First Computer, S. 78.

4 Vgl. Menabrea/Lovelace: Sketch of the Analytical Engine invented by Charles Babbage with notes by the translator, S. 666–731.

5 Vgl. Budge et al. (Hrsg.): The Dictionary of Nineteenth-Century British Philosophers, S. 35.

zur Erkenntnis analytischer Verhältnisse oder Wahrheiten.“[6] Die Vision einer „künstlichen Intelligenz“ lässt die Romantikerin jedoch nicht zu. Erst 1950 sollte der britische Mathematiker und Kryptologe Alan Turing ihren Einwand in seiner Frage „*Can a Machine think?*“ ernsthaft diskutieren.[7]

Adas Notizen enthalten auch die Beschreibung eines Programms, mit dem die Analytical Engine eine Serie von Bernoulli-Zahlen berechnen und in Form eines Diagramms ausgeben kann. Zwar wird in der Fachwelt davon ausgegangen, dass Babbage selbst erste Programme für die Analytical Engine entworfen haben muss, aber Adas Programm gilt als das erste tatsächlich publizierte Computerprogramm der Welt, auch wenn sich unter Historikern lange ein Disput über ihre Rolle als „welterste Programmiererin“ hinzog, der bis heute nicht verstummt ist. Babbage selbst bezeichnete Ada als „Interpretin“. Sein epochales Werk legte die Grundlage für Adas Notizen, doch waren es ihre leicht nachvollziehbaren „Interpretationen“, die eine einzigartige Einsicht in die Bedeutung und die zukünftigen Potentiale dieser Maschine ermöglichten.

Adas Notizen begründeten ihren Ruhm als Computerpionier, aber ihr faszinierendes Leben, ihre Herkunft und insbesondere ihre Rolle als weiblicher Pionier auf einem Gebiet, in dem Frauen bis heute hoffnungslos unterrepräsentiert sind, machten sie zu einer Ikone, deren Leben in Literatur, Film und sogar in einem eigenen Comic aufgegriffen wurde.[8] Obwohl Ada nicht die einzige bedeutende Frau in der Entwicklung der Informatikwissenschaft war, wurde nur nach ihr eine eigene Programmiersprache benannt – ADA.[9]

6 Eigene Übersetzung. Vgl. (Art.) „From Note G, p. 722“. In: o.A.: Ada Lovelaces's Notes and The Ladies Diary. Unter: http://www.cs.yale.edu/homes/tap/Files/ada-lovelace-notes.html [aufgerufen am 13.09.2017].

7 Vgl. Turing: Computing Machinery and Intelligence, S. 433–460.

8 Vgl. Padua: The Thrilling Adventures of Lovelace and Babbage.

9 Ihre Arbeit blieb lange Zeit unbemerkt, bis Bertram V. Bowden 1953, ein Jahrhundert nach Adas Tod, in *Faster than Thought* eine Geschichte des Computers zusammenstellte und sie darin als „Prophetin“ bezeichnete. Einige der modernen Computerpioniere kannten die Arbeiten Babbages oder auch Adas Notizen. Dennoch erzielten sie ihre konzeptionellen Durchbrüche unabhängig davon. (Vgl. dazu Kim/Toole: Ada and the First Computer, S. 81; Vgl. Bowden [Hrsg.]: Faster than Thought).

ETYMOLOGIE Das Wort *programmieren* stammt vom altgriech. Substantiv *πρόγραμμα* (prógramma), das so viel bedeutet wie „öffentlich und schriftlich bekannt gemachte Nachricht, Befehl“[10]. *Programmieren* bezeichnet den Vorgang oder schöpferischen Akt, der zu einem Programm führt. Das Programm als manifestiertes Ergebnis des Programmierens ist als Wort bereits um 1700 als Entlehnung vom lat. *programma* sowie dem griech. *πρόγραμμα* (prógramma) nachweisbar. Anfangs wird es direkt in der griech.-lat. Form *Programma* mit der Bedeutung „öffentliche Bekanntmachung, öffentliche Einladungsschrift“ verwendet und schließlich um 1800 in *Programm* eingedeutscht. Zugrunde liegt das griech. Verb *προγράφειν* (prográphein) mit der Bedeutung „vor-, voranschreiben, durch öffentlichen Anschlag im Voraus anzeigen, bekanntmachen“[11]. Das *Computerprogramm* sowie der zu dessen Erzeugung notwendige Vorgang des Programmierens ist erst ab Mitte des 20. Jhs. sowohl im Dt. als auch im Engl. mit der Entwicklung des Computers nachweisbar.[12] Der in diesem Kontext neugeprägte Begriff bezieht sich darauf, eine konkrete Problemstellung als „Programm für einen Computer so zu formulieren, dass dieser in der Lage ist, die ihm gestellte Aufgabe zu lösen.“[13] Der *Google Book* Corpus[14] weist für die Verwendung des Verbs *programmieren* eine erste signifikante Relevanz in der zweiten Hälfte der 1940er Jahre nach, der ab 1960 einen deutlichen Anstieg erfährt und ab Mitte der 1990er Jahre erneut sprunghaft anwächst.[15]

Neben der technischen Interpretation kann *programmieren* auch ganz allgemein als „nach einem Programm ansetzen“ bzw. „etwas (im Ablauf) festlegen“[16] verstanden werden. Dabei bezieht sich das Programm etwa auf den Ablauf von

10 (Art.) πρόγραμμα. In: Handwörterbuch der griechischen Sprache, S. 1099.

11 (Art.) programmieren. In: Pfeifer (DWDS online). Unter: https://www.dwds.de/wb/programmieren [aufgerufen am 29.05.2017].

12 Vgl. ebd.

13 Vgl. ebd.

14 Der Google Ngram Corpus umfasst mehr als 5 Millionen digitalisierte Druckwerke aus der Zeit zwischen 1500 und 2008. (Vgl. dazu: Michel et al.: Quantitative Analysis of Culture Using Millions of Digitized Books. In: Science, S. 176–182).

15 Vgl. (Art.) programmieren. In: Google Ngram Viewer. Unter: https://goo.gl/hxG5Z5 [aufgerufen am 05.09.2017].

16 (Art.) programmieren. In: Duden GWDS (DWDS online). Unter: https://www.dwds.de/wb/programmieren [aufgerufen am 29.05.2017].

Veranstaltungen oder Darbietungen wie z.B. ein Theaterprogramm, ein Festaktprogramm oder ein musikalisches Programm, auf ein Verzeichnis der Darbietungen und Darsteller oder auch auf die Darlegung der Absichten, Grundsätze und Vorhaben politischer Parteien und Organisationen.[17] Der Vorgang des Programmierens besteht hier ebenfalls in der stets vorab stattfindenden Festlegung des jeweiligen Programms, in der Absicht, diesem möglichst ohne jegliche Abweichung in dessen Ausführung zu folgen. Der letztere Aspekt tritt v.a. in den Vordergrund, wenn die Programmierung das menschliche Verhalten im Sinne einer Konditionierung betrifft. Dies wird in Redewendungen wie z.B. „die Mannschaft ist auf Erfolg programmiert“[18] deutlich.

KONTEXTE Vor der Erfindung des Computers als frei programmierbare Rechenmaschine Mitte des 20. Jhs. bezog sich das *Programmieren* entweder auf die Erstellung eines nach einer festen Reihenfolge vorgegebenen Ablaufs von Ereignissen, wie z.B. bei einem Theaterprogramm, oder auf die Erstellung von Maximen und Direktiven, von denen bei ihrer Umsetzung, wie z.B. bei einem Parteiprogramm, nicht abzuweichen war. Beide Aspekte kommen insbesondere beim *Programmieren* von Computern zum Tragen. Das *Programmieren* eines Computers bezeichnet den Vorgang, der – ausgehend von einer vorgegebenen Problemformulierung – zu einem durch den Computer ausführbaren Programm führt.[19] Das *Programmieren* beginnt meist bereits mit der Analyse eines zu lösenden Problems, bei dem zuerst ein Verständnis des Problems entwickelt werden muss, damit ein Lösungsweg erarbeitet und schließlich ein diesen Lösungsweg verarbeitender Algorithmus konstruiert werden kann.[20] Trennt man den konzeptionellen Algorithmus vom Medium Computer, kann dieser prinzipiell auch von einem Menschen mit entsprechenden Hilfsmitteln, wie z.B. Stift und Papier, zielführend ausgeführt werden, wobei sich lediglich

17 Vgl. ebd.

18 (Art.) programmieren. In: Duden (online). Unter http://www.duden.de/rechtschreibung/programmieren [aufgerufen am 05.09.2017].

19 Vgl. (Art.) programmieren. In: Pfeifer (DWDS online). Unter: https://www.dwds.de/wb/programmieren [aufgerufen am 29.05.2017].

20 Vgl. Brookshear: Computer Science, an Overview, S. 2 ff.

die zur Ausführung des Programms benötigte Zeit verlängert.[21] So betrachtet lässt sich *programmieren* als das Festlegen eines Algorithmus historisch sogar bereits im 2. Jt. v. Chr. nachweisen.[22] Der Algorithmus wird zur Ausführung auf einem Computer mit Hilfe einer oder auch verschiedener Programmiersprachen formuliert und implementiert – ein Vorgang, der auch heute noch als *Kodierung* bezeichnet wird.[23] Das entstandene Computerprogramm in Form einer Abfolge von Befehlen einer oder mehrerer Programmiersprachen wird als *Quellcode* bezeichnet.[24] Der Zweck des Programmierens besteht in der automatisierten Lösung eines speziellen Problems oder in der Automatisierung von sich wiederholenden Abläufen. Die Kunst des Programmierens ist ein kreativer Akt und erfordert die gleichzeitige Expertise in mehreren Domänen, einschließlich Wissen aus dem jeweiligen Anwendungsgebiet, zu grundlegenden Algorithmen und aus der formalen Logik.[25]

Bereits heute haben sich Computer so tief bis in unser alltägliches Leben vorgearbeitet, dass wir uns ihrer Anwesenheit oft überhaupt nicht mehr bewusst sind. Von der intelligenten Waschmaschine, den als „smart" bezeichneten Geräten der Unterhaltungselektronik über das Auto, in dem sich heute ein ganzer Zoo von Rechensystemen verbirgt, bis hin zum Smart Home, das in unserem Zuhause Temperatur, Licht und damit auch unser Wohlbefinden regelt. Im Gegensatz zum freien *Programmieren* eines Computers, das mit Hilfe einer speziellen Programmiersprache erfolgt, gestatten diese in unserem Alltag angesiedelten Systeme zahlreiche Varianten der Interaktion über speziell an unsere Bequemlichkeit angepasste Benutzerschnittstellen. Doch bleiben wir beim *freien* Programmieren. Ein Computer ist dann *universell programmierbar,* d. h. ein richtiger Computer, wenn er die Berechnung sämtlicher Funktionen zulässt, die von einer sogenannten *universellen Turingmaschine* ausgeführt

21 So bezeichnete der Begriff *Computer* ursprünglich auch Menschen, die mathematische Berechnungsfolgen von Hand oder mit Hilfe mechanischer Hilfsmittel in großer Menge und ohne dabei davon abzuweichen ausführten (vgl. Turing: Computing machinery and intelligence, S. 433–460).

22 Vgl. Gericke: Mathematik in Antike und Orient, S. 16 ff.

23 Vgl. Brookshear: Computer Science, an Overview, S. 208 ff.

24 Vgl. ebd.

25 Vgl. Chun: On Software, or the Persistence of Visual Knowledge In: Grey Room, S. 26–51.

werden können.[26] Turingmaschinen machen die Begriffe des Algorithmus und der Berechenbarkeit mathematisch greifbar, sie formalisieren diese Begriffe. Damit wird der Computer zu einem mathematischen Objekt und kann mit mathematischen Methoden untersucht und beschrieben werden. Erlaubt ein Computer alle Berechnungen, die mit einer universellen Turingmaschine durchgeführt werden können, dann wird er als *turing-vollständig* bezeichnet.[27] Die von Charles Babbage konzipierte Analytical Engine gilt bereits als turing-vollständiger Computer.[28] Mit Hilfe seines Gedankenmodells der Turingmaschine war Turing in der Lage, bereits in den 1930er Jahren die Grenzen der Berechenbarkeit und damit die Grenzen unserer heutigen und zukünftigen Superrechner auszuloten.

KONJUNKTUREN Unabhängig vom jeweiligen Anwendungskontext bezieht sich das *Programmieren* auf die Festlegung einer fixierten Reihenfolge von Ereignissen und deren fester Einhaltung im Zuge ihrer Ausführung. Die ersten praktisch einsetzbaren Computer, die als turing-vollständig galten, wurden in den 1940er Jahren entwickelt. In diesem Zusammenhang gerät das Programmieren erstmals häufiger in unseren Sprachgebrauch.[29] Weitere Konjunkturen korrelieren mit der Weiterentwicklung des Computers und seiner Programmiersprachen. Die Programmierung der ersten Computergeneration erfolgte direkt über sogenannte *Maschinenbefehle,* d.h. Zahlencodes, die für die Ausführung eines bestimmten Verarbeitungsbefehls standen. Diese Art der Programmierung macht das Erlernen dieser zahlreichen Codes notwendig. Zudem erforderte die Begutachtung, Anpassung oder Veränderung eines Programms zusätzlich einen aufwändigen Dekodierungsschritt. Die damit verbundene hohe Komplexität erforderte sehr spezielles Expertenwissen und

26 Die Turingmaschine ist ein einfaches theoretisches Berechnungsmodell, das vom britischen Mathematiker und Computerpionier Alan Mathison Turing 1936 eingeführt wurde, also bereits zu einer Zeit, in der an einen praktischen Einsatz dieser neuen Rechenmaschinen noch gar nicht zu denken war (vgl. Ernst: Grundkurs Informatik, S. 397 ff.).

27 Vgl. ebd.

28 Vgl. Johnson: Programming of Life, S. 13.

29 Vgl. (Art.) programmieren. In: Google Ngram Viewer. Unter: https://goo.gl/hxG5Z5 [aufgerufen am 05.09.2017].

verhinderte eine rasche Verbreitung und Adaption der neuen Technologie.[30] Die Computer der ersten Generation wurden schnell von Systemen abgelöst, die eine Programmierung in sogenanntem *Assemblercode* gestatteten. Die Zahlencodes wichen einer einfachen intuitiveren Befehlsnotation, die aber keinerlei Abstraktion erlaubte und die Programmierung direkt an die jeweiligen Hardwarekomponenten des verwendeten Computers band. Jeder Rechner besaß einen unterschiedlichen Befehlssatz, der auf unterschiedliche Weise binär kodiert wurde und verantwortlich dafür war, dass die Programmierung zu einer aufwändigen und langwierigen Angelegenheit geriet.

Der 1947 entwickelte Transistor revolutionierte das Design und die Entwicklung der Computer, die jetzt zuverlässiger und v.a. auch kleiner als ihre Röhrenvorgänger wurden. Mit dieser zweiten Generation der Computer hielten in der zweiten Hälfte der 1950er Jahre sogenannte höhere Programmiersprachen wie COBOL oder FORTRAN ihren Einzug, mit denen die Programmierung von der jeweilig verwendeten Computerhardware abstrahiert werden konnte. Das *Programmieren* gestaltete sich wesentlich einfacher im Vergleich zum kryptischen Maschinencode, die Verbreitung der Computertechnologie schritt voran, v.a. bedingt durch die Einführung des integrierten Schaltkreises als nächsten Schritt der Miniaturisierung, der energieeffizientere, kleinere und auch preiswertere Computer ermöglichte.[31] Die Einführung der Mikroprozessoren in den 1970er Jahren, verbunden mit hochintegrierten Schaltkreisen, ermöglichte es schließlich, Millionen von Transistoren auf einem einzelnen Mikrochip unterzubringen. Der damit einhergehende Preisverfall ebnete dem Computer als *Personal Computer* und schließlich sogar als *Homecomputer* den Weg bis hinein in unsere Wohn- und Kinderzimmer. Einfach erlernbare Programmiersprachen, wie BASIC oder Logo, erlauben selbst Kindern das *Programmieren*.[32] So wurde die aufwändige Kodierung von einst, eine Kunst, die nur von wenigen Experten beherrscht wurde, zu einer allgemeinen Kulturtechnik für jedermann, die Eingang in unsere Schulen gefunden hat und in nicht ferner Zukunft zu unseren Basisfertigkeiten zählen wird, wie das Lesen

30 Vgl. Meinel/Sack: Digitale Kommunikation, S. 73 f.

31 Vgl. ebd.

32 Vgl. Papert: An Evaluative Study of Modern Technology in Education. Unter http://www.papert.org/articles/AnEvaluativeStudyofModernTechnology.html [aufgerufen am 05.09.2017].

und Schreiben. Mit der exponentiell wachsenden Verbreitung des *Internets* und des *World Wide Webs* beginnend in den 1990er Jahren, weiter angefacht durch fortschreitende Entwicklung mobiler Endgeräte wie Smartphones und Tablets, ging eine weitere bis heute ungebrochene Popularisierung des Programmierens einher. Um heute eine erfolgreiche *App* zu *programmieren,* benötigt man kein Hochschulstudium. Zahlreiche frei verfügbare Programmbibliotheken und Baukastensysteme, sogenannte *Frameworks,* unterstützen den Programmierer mit wiederzuverwendenden Komponenten, so dass dieser auf einen reichen Schatz an bereits vorhandenen Lösungen zurückgreifen und diese auf einfache Weise zu neuen Anwendungen kombinieren kann.[33]

GEGENBEGRIFFE Ein direktes Antonym zu *programmieren* als schöpferischen Akt zur Erstellung eines ausführbaren Computerprogramms scheint es nicht zu geben. Spontane Selbstorganisation und selbstlernende Systeme könnten als Gegenpol zu den durch menschliche Intelligenz entworfenen Computerprogrammen herangezogen werden. Auch ihnen liegen aber oft basale, vom Menschen programmierte Systeme zu Grunde. Diese bilden in diesem Zusammenhang aber lediglich ein Grundgerüst aus selbstlernenden Verfahren, sogenanntes *maschinelles Lernen,* das mit Hilfe statistischer Verfahren aus vorgegebenen Beispielen neue Programme lernen und trainieren kann (überwachtes Lernen) oder sich über ein abgegebenes Feedback automatisch an neue Situationen anpassen kann (bestärkendes Lernen).[34] Semantisch stellt das Verb *hacken* eine Spezialisierung des Programmierens dar. Als Hacker bezeichnet man umgangssprachlich eine Person, die in ein Computertsystem unberechtigt eindringt oder ein Mitglied der als Hacker bezeichneten computeraffinen Szene. Im Gegensatz zum bloßen *Programmieren* beinhaltet Hacken oft eine Form der Exzellenz oder Virtuosität, um etwa die Grenzen des Machbaren zu erkunden und dabei gleichzeitig einen sinnvollen Beitrag zu leisten.[35] Bezogen auf *programmieren* als die Vorgabe einer fixierten Reihenfolge von Ausführungsanweisungen, stellt *improvisieren* das Verlassen dieses

33 Vgl. Pree: Komponentenbasierte Softwareentwicklung mit Frameworks.

34 Vgl. Russel/Norvig: Künstliche Intelligenz, ein moderner Ansatz, S. 793 f.

35 Vgl. Stallmann: The Hacker Community and Ethics. Unter: https://www.gnu.org/philosophy/rms-hack.html [aufgerufen am 29.05.2017].

fest vorgegebenen Pfades dar, wobei auch die Improvisation, wie z.B. in der Musik, bestimmten Regeln folgt, jedoch einen weitaus größeren Freiheitsgrad in deren Ausführung und Ausgestaltung gestattet. Allerdings bezieht sich *Programmieren* auf den Vorgang des Erstellens, d.h. dem Vorbereiten eines Programms, während sich die Improvisation meist direkt auf die Ausführung, ohne dezidierte Vorbereitung und aus dem Stehgreif beschränkt und sich bei der Problemlösung des spontanen Gebrauchs von Kreativität bedient.[36]

PERSPEKTIVEN Die grundsätzliche, ursprünglich schon von Ada Augusta King aufgeworfene Frage nach der von ihr angezweifelten Fähigkeit der Maschine zum schöpferischen Denken, geriet seit den 1960er Jahren mit dem Aufkommen der *künstlichen Intelligenz* immer wieder in den Fokus der öffentlichen Aufmerksamkeit.[37] Bereits vor 40 Jahren waren sich Wissenschaftler einig in der Frage, ob der Computer den Menschen eines Tages intellektuell überflügeln würde. Ein Meilenstein dabei war 1997 der erste Sieg eines Schachcomputers über einen menschlichen Schachweltmeister.[38] Wurden diese ersten Siege der Maschine noch rein analytisch durch schiere Rechenkraft und Speicherfähigkeit (Brute Force) errungen, befindet sich seit jüngster Zeit das *maschinelle Lernen,* insbesondere das der biologischen Informationsverarbeitung nachempfundene Lernen in neuronalen Netzwerken (Deep Convolutional Neural Networks) auf der Überholspur.[39] Maschinelles Lernen setzt Algorithmen voraus, die in der Lage sind, aus vorgegebenen Beispielen einen Vorgang, wie z.B. die Klassifikation eines Informationsinhaltes, zu „erlernen". Dazu wird ein internes mathematisches Modell „trainiert", bis es in der Lage ist, die zum Training verwendeten Vorlagen möglichst gut selbst zu klassifizieren, bevor man das System mit noch unbekannten „Testdaten" konfrontiert. In diesem Falle „programmiert" sich das System quasi selbst nach dem Prinzip

36 Vgl. (Art.) improvisieren. In: DWDS online. Unter: https://www.dwds.de/wb/improvisieren [aufgerufen am 29.05.2017].

37 Vgl. Russel/Norvig: Künstliche Intelligenz, ein moderner Ansatz, S. 36 f.

38 Vgl. Heßler: Der Erfolg der „Dummheit". In: NTM Zeitschrift für Geschichte der Wissenschaften, S. 1–33.

39 Vgl. Goodfellow/Bengio/Courville: Deep Learning, S. 1–8.

mathematischer Optimierungsverfahren.[40] Einen eindrucksvollen Beweis für die Leistungsfähigkeit dieser neuen Technologien konnte z.B. mit *IBM Watson*[41] erbracht werden, der die Champions im sprachlich anspruchsvollen Jeopardy TV-Quiz schlagen konnte, oder Googles künstlicher Intelligenz *AlphaGo,*[42] die bislang noch von keinem Menschen im japanischen Go-Spiel geschlagen werden konnte.

FORSCHUNG Denkt man diesen Ansatz weiter, stellt sich die Frage, ob die Fähigkeit des Programmierens in Zukunft überhaupt noch eine Rolle für den Menschen spielen wird, sei es, dass Maschinen ihm diese Aufgabe komplett abnehmen werden oder dass die von den Maschinen selbst geschriebenen Programme so komplex werden, dass sie vom Menschen nicht mehr verstanden werden können. Lange Zeit wurde davon ausgegangen, dass, wenn ein Computer eine Entscheidung nach vorgegebenen oder erlernten, also vermeintlich objektiven Entscheidungsregeln trifft, menschliche Vorurteile im Entscheidungsprozess vermieden werden könnten. Allerdings sind die von einer maschinellen Intelligenz erlernten Entscheidungsregeln stets nur so gut, wie die ihnen zugrunde liegenden Daten, mit denen diese trainiert wurden. Da die gewaltigen Mengen an Trainingsdaten, die für eine qualitativ gute Entscheidungsfindung benötigt werden, meist aus dem Web oder aus digitalisierten Textcorpora stammen, die ursprünglich von Menschen verfasst wurden, konnten in den darauf basierenden maschinell gelernten Entscheidungen ebenso soziale oder rassistische Vorurteile nachgewiesen werden.[43] Darüber hinaus lässt sich spekulieren, ob das menschliche Denken selbst in ein Computerprogramm übertragbar wäre. Dies würde bedeuten, dass mentale Prozesse in Form von Algorithmen darstellbar wären. Dann allerdings wären diese Prozesse deterministisch, d.h. ausgehend vom selben mentalen Zustand – falls dieser in seiner Gesamtheit erfasst und repräsentiert werden kann – würde in

40 Vgl. Russel/Norvig: Künstliche Intelligenz, S. 793 f.

41 Vgl. Ferrucci et. al: The AI Behind Watson – The Technical Article. In: The AI Magazin. Unter: http://www.aaai.org/Magazine/Watson/watson.php [aufgerufen am 29.05.2017].

42 Vgl. Silver et al.: Mastering the Game of Go with Deep Neural Networks and Tree Search. In: Nature, 529, S. 484–489.

43 Vgl. Barocas/Selbst: Big Data's Disparate Impact.

Folge einer einfachen Kausalkette stets derselbe mentale Effekt bzw. dieselbe Verhaltensreaktion hervorgerufen.[44] Dem widerspricht die Auffassung, dass das menschliche Denken als nicht-algorithmischer Prozess einzuordnen ist, dem nach Ansicht des Mathematikers und Physikers Roger Penrose (zufällige) Quantenphänomene zugrunde liegen könnten.[45]

LITERATUREMPFEHLUNGEN

Kürzel, Werner/Beste, Peter: Maschinendenken/Denkmaschinen – An den Schaltstellen zweier Kulturen, Frankfurt/M. (1996).

Hofstadter, Douglas: Gödel, Escher, Bach – ein Endloses Geflochtenes Band, 18. Aufl., Stuttgart (2008).

Minsky, Marvin: The Society of Mind. New York (1988).

VERWEISE

einloggen |II 133|, operieren |II 316|, serialisieren |I 498|, übertragen |II 458|, vernetzen |II 482|

BIBLIOGRAFIE

(Art.) Babbage, Charles. In: The Dictionary of Nineteenth-Century British Philosophers, London (2002).

Barocas, Solon/Selbst, Andrew D.: Big Data's Disparate Impact. In California Law Review, Vol. 104–3, Juni 2016, S. 671–732.

Bowden, Bertram V. (Hrsg.): Faster than Thought: A Symposium of Digital Computing Machines, London (1953).

Brookshear, J. Glenn: Computer Science, an Overview, Heft 7, Boston (2003).

Chun, Wendy Hui Kyon: On Software, or the Persistence of Visual Knowledge. In: Grey Room, Winter 2005, S. 26–51.

Ernst, Hartmut: Grundkurs Informatik, 3. Aufl., Wiesbaden (2003).

Gericke, Helmuth: Mathematik in Antike und Orient, Berlin (1984).

Goldenberg, E. Paul: Logo – A Cultural Glossary. In: BYTE, August 1982, S. 210–228.

Goodfellow, Ian/Bengio, Yoshua/Courville, Aaron: Deep Learning, Cambridge (2016).

Heßler, Martina: Der Erfolg der „Dummheit". Deep Blues Sieg über den Schachweltmeister Garri Kasparov und der Streit über seine Bedeutung für die Künstliche Intelligenz-Forschung. In: NTM Zeitschrift für Geschichte der Wissenschaften, Technik und Medizin Nr. 25 (1), 2017, S. 1–33.

Hofstadter, Douglas: Gödel, Escher, Bach – ein Endloses Geflochtenes Band, 18. Aufl., Stuttgart (2008).

Hyman, Anthony: Charles Babbage, Pioneer of the Computer, Oxford (1982).

44 Vgl. Pinker: So How Does the Mind Work?, S. 1–24.

45 Vgl. Penrose: The Emperor's New Minds and the Laws of Physics.

Johnson, Donald E.: Programming of Life, Big Mac Publishers (2010).
Kim, Eugene Eric/Toole, Betty Alexandra: Ada and the First Computer, In: American Scientific, Mai 1999.
Kürzel, Werner/Beste, Peter: Maschinendenken/Denkmaschinen – An den Schaltstellen zweier Kulturen, Frankfurt/M. (1996).
Meinel, Christoph/Sack, Harald: Digitale Kommunikation – Vernetzen, Multimedia, Sicherheit, Heidelberg (2009).
Menabrea, Luigi Federico/Lovelace, Ada: Sketch of the Analytical Engine invented by Charles Babbage... with notes by the translator. Translated by Ada Lovelace. In: Taylor, Richard: Scientific Memoirs. 3, London, S. 666–731.
Michel, Jean-Baptiste et al.: Quantitative Analysis of Culture Using Millions of Digitized Books. In: Science 2011, 331 (6014), S. 176–182.
Minsky, Marvin: The Society of Mind. New York (1988).
O'Reagan, Gerard: A Brief History of Computing, Heidelberg (2008).
Padua, Sidney: The Thrilling Adventures of Lovelace and Babbage: The (Mostly) True Story of the First Computer, London (2015).
Pape, Wilhelm/Sengebusch, Max (Bearb.): Handwörterbuch der griechischen Sprache, 3. Aufl., 6. Abdruck, Braunschweig (1914).
Passow, Franz: Handwörterbuch der griechischen Sprache, neu bearbeitet und zeitgemäß umgestaltet von Dr. Val. Chr. Fr. Rost und Dr. Friedrich Palm, 5. Aufl., Bd. 2,1: Lambda–Pi, Leipzig (1852).
Penrose, Roger: The Emperor's New Minds and the Laws of Physics, New York (1989).
Pinker, Steven: So how Does the Mind Work? In: Mind & Language, Vol. 20, No. 1 February 2005, S. 1–24.
Russel, Stuart/Norvig, Peter: Künstliche Intelligenz, ein moderner Ansatz, 2. Aufl., München (2004).
Silver, David et al.: Mastering the Game of Go with Deep Neural Networks and Tree Search. In: Nature, 529 (2016), S. 484–489.
Turing, A. M.: Computing Machinery and Intelligence. In: Mind, 59 (1950), S. 433–460.

Internet

(Art.) „From Note G, p. 722". In: o.A.: Ada Lovelaces's Notes and The Ladies Diary. Unter: http://www.cs.yale.edu/homes/tap/Files/ada-lovelace-notes.html [aufgerufen am 13.09.2017].
(Art.) improvisieren. In: DWDS (online). Unter: https://www.dwds.de/wb/improvisieren [aufgerufen am 29.05.2017].
(Art.) programmieren. In: Duden (online). Unter http://www.duden.de/rechtschreibung/programmieren [aufgerufen am 05.09.2017].
(Art.) programmieren. In: Duden GWDS (DWDS online). Unter: https://www.dwds.de/wb/programmieren [aufgerufen am 29.05.2017].
(Art.) programmieren. In: Google Ngram Viewer. Unter: https://goo.gl/hxG5Z5 [aufgerufen am 05.09.2017].
(Art.) programmieren. In: Pfeifer. Etymologisches Wörterbuch des Deutschen (DWDS online). Unter: https://www.dwds.de/wb/programmieren [aufgerufen am 29.05.2017].
Ferrucci, David et.al.: The AI Behind Watson – The Technical Article. In: The AI Magazin, 2010. Unter: http://www.aaai.org/Magazine/Watson/watson.php [aufgerufen am 29.05.2017]
Papert, Seymour: An Evaluative Study of Modern Technology in Education (1976), Unter http://www.papert.org/articles/AnEvaluativeStudyofModernTechnology.html [aufgerufen am 05.09.2017]
Stallmann, Richard: The Hacker Community and Ethics: An Interview with Richard M. Stallman, 2002, Unter: https://www.gnu.org/philosophy/rms-hack.html [aufgerufen am 29.05.2017]