Emulation

Alle Denkmaschinen sind Turingmaschinen

Dr. Hartwig Thomas

Fragt der Computerneuling seinen befreundeten Experten: 'Welchen Rechner soll ich mir kaufen?', worauf dieser ihm antwortet: 'Das kommt darauf an, was Du damit tun willst.' Der Anfänger ist enttäuscht, sein Freund, der Experte, hat versagt; glaubt er selbst doch, daß Computer alles können, und wollte nur wissen, welches die beste Maschine sei. – 'Schuld' an diesem Mißverständnis trägt nicht zuletzt Alan Turing und der von ihm geprägte Begriff der Universellen Turingmaschine.

Der britische Mathematiker Alan M. Turing hat den Computer erfunden. Außerhalb eines engen Kreises von Spezialisten sind sein Name und Werk kaum bekannt. Seit 1987 ist eine deutsche Sammlung vorwiegend nicht-technischer Artikel Turings für 'Laien' als Geheimtip unter Kennern im Umlauf. Alle abgedruckten Beiträge befassen sich mit der Frage, was Denkmaschinen sind und was sie sein könnten. Dabei spielt der Kernbegriff Emulation eine ausschlaggebende Rolle. Die Sammlung ist eine wichtige Grundlage für eine intelligente Auseinandersetzung mit intelligenten Maschinen auch außerhalb einer technisch orientierten Insidergruppe.

In seinem Artikel 'Über berechenbare Zahlen mit einer Anwendung auf das Entscheidungsproblem' [1] führt Alan Turing den Beweis, daß das berühmte 'Entscheidungsproblem' nicht lösbar ist. Das 'Entscheidungproblem' stammt von David Hilbert. Dieser Vater des mathematischen Formalismus zählte im Jahre 1900 auf dem Internationalen Mathematikerkongreß in Paris eine Reihe von wichtigen, im neuen Jahrhundert zu lösenden Problemen auf. Das zehnte Problem war das später so genannte Entscheidungsproblem:

Eine diophantische Gleichung[1] mit irgendwelchen Unbekannten und mit ganzen rationalen Zahlkoeffizienten sei vorgelegt: man soll ein Verfahren angeben, nach welchem sich mittels einer endlichen Anzahl von Operationen entscheiden läßt, ob die Gleichung in ganzen rationalen Zahlen lösbar ist.

Der berühmte Beweis von Kurt Gödel [2] zeigte, daß mit keinem Axiomensystem der Zahlentheorie alle Sätze entscheidbar sind [3].

Alan Turing führt in seinem Beitrag zum Entscheidungsproblem in sehr klarer und unprätentiöser Weise die – seither so genannte – Turingmaschine als theoretische Konstruktion ein. Er identifiziert die Tätigkeit eines Mathematikers bei der Berechnung einer Zahl in einem Kalkül mit dem Funktionieren einer hypothetischen Maschine. Diese wird mit dem notwendigen Minimum ausgestattet, um dem Mathematiker geistig ebenbürtig zu sein: einem Speicher (vorgestellt als potentiell unendliches Band mit Feldern, auf denen Zeichen aus einem endlichen Alphabet stehen können, und das von der Maschine über einen Lesekopf Feld für Feld examiniert werden kann) und einer Steuereinheit, die endlich viele Zustände annehmen kann und vom Band lesen und auf das Band schreiben kann.

Symbolmanipulation

Eine solche Turingmaschine ist in ihrem Funktionieren eindeutig festgelegt, wenn man für jeden Zustand und jedes gelesene Zeichen festhält, in welchen neuen Zustand die Maschine beim nächsten Schritt übergeht. Da das Zeichenalphabet und die Zustandsmenge der Steuereinheit endlich sind, kann eine solche Maschine mit einer endlichen Liste von Zustandsänderungen beschrieben werden.

Jede Turingmaschine verarbeitet einen Input auf dem Band zu einem Output und kommt beim Endzustand zum Halten, sofern sie nicht in einer endlosen Ramifikation [2] steckenbleibt. Nachdem Turing gezeigt hat, daß es für eine Reihe von elementaren Problemen angebbare Maschinenbeschreibungen gibt (Kopiervorgänge, arithmetische Operationen ...), holt er zum großen Schlag aus: er stellt fest, daß seine Maschinen nicht nur reichlich immaterielles Material verarbeiten, nämlich Symbole auf einem Band, sondern daß sie selbst auch aus ähnlich immateriellem Material bestehen. Sie können durch eine Folge von Symbolen vollständig beschrieben werden (in seinem Artikel hat er ja auch nur mit solchen Beschreibungen gearbeitet, da 1937 noch keine realen Computer zur Verfügung standen). Nichts naheliegender also, einer Maschine den Bauplan einer anderen Maschine als Input zu 'füttern'.

Universalmaschine

Dank dieser überraschenden Wendung stieß Turing zum Wesen der Symbolmanipulation vor. Er erfand die Universelle Turingmaschine. Es stellt sich als einfach heraus, eine Turingmaschine zu konstruieren, die, nach Lesen des Bauplans einer beliebigen anderen Turingmaschine, diese 'nachmacht', das heißt sich in jedem Punkt so verhält, als ob sie diese spezielle Turingmaschine wäre. Dieser Vorgang und diese Idee sind uns im Computerzeitalter derart vertraut, daß wir das Erstaunen darüber vergessen haben. Die Bezeichnung Emulation hat sich für dieses Verfahren eingebürgert.

Mit dieser Universalmaschine hat Turing den modernen Computer erfunden. Alle modernen Computer sind ihrem Konzept gemäß Turingmaschinen (in der Praxis setzen mechanische Abnutzung und endliche Menge von zur Verfügung stehendem Speicher – etwa Magnetband oder Disketten – diesem Konzept Grenzen). In dem halben Jahrhundert seit dieser Erfindung wurden verschiedenste Versuche unternommen, den Bereich der im Prinzip von Menschen berechenbaren Formeln, Zahlen, Sätze anders und allgemeiner als Turing zu formulieren. Bis heute haben sich alle diese Versuche als der Turingmaschine äquivalent erwiesen.

Obwohl man eine Definition eines Begriffs wie Berechenbarkeit natürlich nicht beweisen kann, da es sich ja um eine Definition handelt, wird unter

[1] Eine diophantische Gleichung ist eine Gleichung mit ganzzahligen Koeffizienten, in der nur die vier Grundrechenarten und ganzzahlige Exponentiation vorkommen.

[2] Ramifikation = Verästelung

Mathematikern angenommen, daß alle 'vernünftigen' Definitionen dieses Begriffs auf dasselbe herauskommen. (Dieser Sachverhalt wird unter Mathematikern auch Church-Hypothese genannt, was zur weiteren Verwirrung beiträgt, da sich eine Definition natürlich auch nicht als 'Hypothese' formulieren läßt.)

Und der zentrale 'Trick' des Computerzeitalters ist schon in Turings erstem Artikel klar erkannt: der Computer kann als Maschine gebaut werden, die alles kann, als universelle Metamaschine also, die sich je nach Bauplan – heute nennt man das 'Programm' –, mit dem man sie füttert, in eine konkrete Maschine verwandelt. Auch diese 'konkreten' Maschinen können natürlich wieder mehr oder weniger universell sein.

Ein Software-Hersteller, der es mit den Spezialvorstellungen seiner Kunden über irgendein Thema nicht verderben will, offeriert im allgemeinen eine 'Makrosprache' (Textverarbeitung) oder 'Befehlssprache' (dBase), in der man alles das selber programmieren kann, was vom Hersteller nicht vorgesehen war, und behauptet dann stinkfrech, sein Programm könne alles, was dasjenige der Konkurrenz auch könne (man müßte es nur mit dem 'Bauplan' des Programms der Konkurrenz füttern, damit die Emulation läuft ...).

Füttern unumgänglich

Auch der PC-Benutzer ist sich heute in den seltensten Fällen darüber im klaren, daß er für seine 3000 Mark mitnichten eine Maschine gekauft hat, die Buchhaltung 'kann' und CAD 'kann' und Typographie 'kann' und übers Telefon kommunizieren 'kann', sondern daß seine neue Errungenschaft mit Bauplänen konkreter Maschinen gefüttert werden muß, bevor sie brauchbar ist. Von einem Neugeborenen sagen wir auch nicht, es 'könne' lesen, obwohl wir die meisten von ihnen bei richtiger 'Programmierung' dazu veranlassen können, diese Disposition zu verwirklichen.

Um also die universelle Metamaschine zur benutzbaren Maschine zu machen, müssen wir noch einmal teures Geld auf den Tisch legen, um sie mit Programmen zu füttern. Beim Auto hat man sich daran gewöhnt, daß es nicht ohne Benzin fährt. Beim PC tun sich viele Benutzer noch schwer mit dem Softwarekauf und kommen sich betrogen vor, weil man ihnen ja eine Maschine verkauft hat, die alles kann.

Dieses Konzept einer kannibalistischen Maschine, welche die allgemeinst vorstellbaren Denkvorgänge übernehmen kann und die nach Verzehren des Bauplans ihrer Artgenossen deren Intelligenz erbt, ist genügend mit scheinbaren Paradoxien geladen, so daß man mit ihrer Hilfe auch die Unlösbarkeit des hilbertschen Entscheidungsproblems herleiten kann.

Dies überrascht uns aus heutiger Perspektive kaum mehr. Auch Alan Turing war an seinen denkenden Maschinen nicht nur als geistreiche theoretische Hilfskonstruktion im Beweis eines berühmten Theorems interessiert. In den weiteren Beiträgen im besprochenen Band schreibt er unter Titeln wie 'Lösbare und unlösbare Probleme', 'Intelligente Maschinen', 'Spielprogramme', 'Rechenmaschinen und Intelligenz' immer wieder über Universelle Turingmaschinen und die Konsequenzen ihrer Existenz.

Uns ist das zentrale Konzept der Emulation dermaßen alltäglich geworden, daß wir kaum mehr merken, wovon wir abstrahieren, wenn wir sagen, 'Mit dem Amiga kann man einen PC emulieren', 'Der Laserdrucker kann sowohl einen Epson als auch einen HP emulieren', 'Dieses Programm emuliert die Difference Engine von Charles Babbage' oder 'Mit Sprint kann man die MS-Word-Oberfläche emulieren'. Wir meinen, daß der symbolische Intelligenzaspekt nachgeahmt wird, und sehen von äußeren Unterschieden als völlig unerheblich ab.

Eine heute weit verbreitete feinsinnige Bemerkung über das Wesen der Computer lautet: digitale Maschinen könnten ja nur – in einem manichäischen Maschinen-Universum – zwischen 0 und 1, zwischen Ja und Nein, zwischen Schwarz und Weiß unterscheiden. Die von solchen 'Meinungen' implizierte Überlegenheit des psychisch reifen Menschen, der auch Zwischentöne kennt, wird von Turing dadurch widerlegt, daß er zeigt, daß jegliche diskrete Maschine (mit endlichem 'Grauton-Alphabet') mit einer Turingmaschine emuliert werden kann und daß sogar Analogmaschinen von Turingmaschinen beliebig genau nachgeahmt werden können.

Wie den Mathematikern zu Beginn dieses Jahrhunderts ist Turing und einigen Spezialisten klar, daß der Computer – trotz seines Namens –, ebenso wie die Mathematik, keine Rechenmaschine ist, sondern eine Maschine zur freien Manipulation von Symbolen.

Konkurrenzangst?

In den weiteren Artikeln der Sammlung [1] verfolgt Turing lange vor der allgemeinen Verbreitung solcher Maschinen die häretische Frage, warum seine Turingmaschine nicht auch das menschliche Denken emulieren können sollte. In 'Rechenmaschinen und Intelligenz' beschäftigt ihn die Frage, inwiefern man solchen Maschinen das Denken absprechen könne. Er zählt Einwände auf, die er argumentativ zu wiederlegen versucht, und unterstellt den entschiedensten Gegnern der Denkmaschinen, nämlich gewissen Intellektuellen, eine aus ökonomischer Konkurrenzangst verzerrte, unredliche Argumentationsweise.

Einwände gegen die Intelligenz der Computer gemäß Turings Klassifikation:

1. Der theologische Einwand: nur die unsterbliche Seele kann denken und Gott gab diese nur den Menschen.
2. Der 'Vogel-Strauß'-Einwand: die Folgen wären zu schrecklich.
3. Der mathematische Einwand: Turingmaschinen können die Lösbarkeit gewisser Gleichungen nicht entscheiden.
4. Der Bewußtseinseinwand: Maschinen können nicht fühlen und nicht kreativ tätig sein.
5. Verschiedene Unfähigkeiten: Maschinen können (noch) nicht Initiative besitzen, Sinn für Humor haben, Recht von Unrecht unterscheiden, Fehler machen, sich verlieben, gerne Erdbeeren mit Schlagsahne essen ...
6. Der Einwand von Lady Lovelace: Computer können immer nur das tun, was ihnen befohlen wird.
7. Der Mensch ist nicht digital, sondern etwa ein Hybridrechner.
8. Das Argument aus dem informellen Charakter des menschlichen Verhaltens.

Turings Bemerkung, daß es nicht ausgeschlossen sei, daß Denkmaschinen einmal die Macht übernehmen könnten, erinnert uns daran, daß 'Aemulatio' auf lateinisch nicht nur 'Nachahmung', sondern auch 'Wettstreit' heißt. (bb)

Literatur

[1] Alan Turing, On Computable Numbers with an Application to the Entscheidungsproblem, Proceedings of the London Mathematical Society (2)42, 1937.

In einem sorgfältig edierten Band liegen diese mathematischen Artikel und andere Schriften Turings, die sich an ein nicht-mathematisches Publikum richten, auch auf deutsch vor:

Dotzler, Kittler (Hrsg.), Alan Turing, Intelligence Service, Brinkmann & Bose, Berlin, 1987 (siehe auch Buchkritik in c't 10/88, S. 261)

[2] Kurt Gödel, Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme I, Monatshefte Math. Phys., 38 (1931), 173–198

[3] *Eine bemerkenswerte Variation dieses Resultats wurde 1987 von Gregory J. Chaitin vorgestellt zusammen mit einem expliziten Beispiel einer diophantischen Gleichung (5 Seiten), von der man zeigen kann, daß es unmöglich ist, zu entscheiden, ob sie endlich oder unendlich viele ganzzahlige Lösungen besitzt, in:*

G. J. Chaitin, Algorithmic Information Theory, Cambridge Tracts in Theoretical Computer Science 1, Cambridge University Press, 1987