MAGISTERARBEIT

Das Problem der universellen Realisierbarkeit bei Putnam und Searle

zur Erlangung des Grades Magister Artium im Fach Philosophie

am Institut für Philosophie
der Philosophischen Fakultät I

vorgelegt von
Thomas Goldstraß

Dekan: Prof. Dr. Oswald Schwemmer

Gutachter: 1. PD Dr. Uwe Scheffler
2. Prof. Dr. John Michael Krois

eingereicht: 08. November 2002

Zusammenfassung

In dieser Arbeit wird gezeigt, dass das Problem der universellen Realisierbarkeit – ein Argument von John R. Searle und Hilary Putnam gegen die Kognitionswissenschaft – auf einer irreführenden Vorstellung des Endlichen Automaten basiert. Nach einer Richtigstellung dieser Vorstellung bleibt ein allgemeines Beschreibungs- oder Zuschreibungsproblem zurück, das jeder empirische Ansatz in der Wissenschaft hat und deshalb nicht gut dazu geeignet ist, die Kognitionswissenschaft im Besonderen zu kritisieren.

Abstract

In this paper I claim that the problem of universal realizability – an argument by John R. Searle and Hilary Putnam against cognitive science – is based on a misleading notion of the finite state automaton. After a correction of this notion, there remains a general problem of description or ascription, which is a problem common to every empirical approach in science and hence not very useful for a critique specific to cognitive science.

Schlagwörter:

Philosophie des Geistes, universelle Realisierbarkeit, Kognitionswissenschaft, Funktionalismus, Künstliche Intelligenz, Kognitivismus, Endlicher Automat, digitaler Computer, Turingmaschine, Computerprogramm, Symbol, Geist, Gehirn.

Keywords:

philosophy of the mind, universal realizability, cognitive science, functionalism, artificial intelligence, cognitivism, finite state automaton, Turing machine, computer program, digital computer, symbol, mind, brain.

Inhaltsverzeichnis

1 Einleitung ... 4
2 Grundlagen ... 8
2.1 Kognitionswissenschaft ... 8
2.1.1 Searles Kognitivismus ... 11
2.1.2 Putnams Funktionalismus ... 17
2.2 Zum Konzept des Endlichen Automaten (EA) ... 23
2.2.1 Beispiel: Ein EA Für das Mann/Wolf/Ziege/Kohl-Problem ... 23
2.2.2 Die Rolle des EA in der Theoretischen Informatik ... 29
2.2.3 Turingmaschinen und die Church/Turing-Hypothese ... 34
3 Das Problem der universellen Realisierbarkeit ... 40
3.1 Searle: Symbolmanipulationen sind universell realisierbar ... 40
3.2 Putnam: Jedes beliebige physikalische System ist eine Realisierung jedes beliebigen Endlichen Automaten (EA) ... 49
3.3 Searle: Die Syntax ist der Physik nicht intrinsisch ... 54
3.4 Fazit: Das Problem der universellen Realisierbarkeit ist kein spezielles Problem der Kognitionswissenschaft ... 63
4 Anmerkungen ... 68
5 Literaturverzeichnis ... 73

1 Einleitung

In dieser Arbeit dreht es sich um ein negatives Argument, das die beiden amerikanischen Philosophen Hilary Putnam und John R. Searle zeitgleich und unabhängig voneinander um 1990 in die Debatte der Philosophie des Geistes gebracht haben. Bezeichnenderweise befindet sich dieses negative Argument in zwei insgesamt relativ negativen Büchern, nämlich *Repräsentation und Realität* von Putnam (Putnam 1999)[1] und *Die Wiederentdeckung des Geistes* von Searle (Searle 1993), worin die beiden Philosophen mit vielen verschiedenen Argumenten unterschiedliche Positionen der Philosophie des Geistes kritisieren. Das Problem der universellen Realisierbarkeit, um das es hier geht, ist eines davon und richtet sich explizit gegen computerorientierte Auffassungen innerhalb des interdisziplinären Forschungsprojektes Kognitionswissenschaft.

In der Kognitionswissenschaft wird der menschliche Geist mitunter mit einem Computerprogramm verglichen und das Gehirn mit einem Computer. Mit dem Problem der universellen Realisierbarkeit werfen Putnam und Searle der Kognitionswissenschaft vor, dass man, wenn man das Gehirn mit einem Computer vergleicht, mit dem gleichen Recht alles andere, Wände, Badewannen, Tauben und jedes weitere beliebige physikalische System, auch mit einem Computer vergleichen könne; – damit würde dieser Vergleich sinnlos. Dies ist für die beiden eine Folge aus den Definitionen des Computers bzw. des dem Computer als definierendes abstraktes Modell zugrundeliegenden Endlichen Automaten (EA).

Putnam und Searle hatten leicht verschiedene Positionen im Sinn, als sie das Problem der universellen Realisierbarkeit formulierten. Putnam kritisierte damit den so genannten Funktionalismus und Searle eine Ansicht, die er Kognitivismus nennt und von der er sagt, dass sie die zentrale Position der computerorientierten Kognitionswissenschaft sei. Weil das Problem damit sehr voraussetzungsreich ist, wird in dieser Arbeit besonders viel Wert auf eine ausführliche Darstellung seiner Grundlagen gelegt: Auf die beiden kritisierten Ansätze, Kognitivismus und Funktionalismus (Abschnitt 2.1) und auf das Konzept des Endlichen Automaten (Abschnitt 2.2), das bei diesen beiden Ansätzen die Grundlage für die computerorientierte Vorgehensweise bildet.

In Abschnitt 2, *Grundlagen*, wird vor allem darauf geachtet, eine genaue Vorstellung davon zu vermitteln, wie es in der Kognitionswissenschaft gemeint ist, dass der Endliche

Automat als Grundlage für die Erforschung des menschlichen Geistes benutzt werden kann. Dabei wird für die Darstellung des Endlichen Automaten bewusst Fachliteratur aus der Theoretischen Informatik, aus der das Konzept das Endlichen Automaten ursprünglich kommt, benutzt und keine aus der philosophischen Debatte selbst, in die es durch die Kognitionswissenschaften importiert wurde. Nur auf diese Art kann man den begrifflichen Unschärfen, die bei diesem Import und seiner Kritik vorkommen, auf die Spur kommen. Schon bei der Darstellung der Grundlagen wird immer wieder parenthetisch auf das Problem der universellen Realisierbarkeit Bezug genommen, um es schrittweise genauer zu erfassen.

In Abschnitt 3, *Das Problem der universellen Realisierbarkeit*, werden die Argumentationen von Putnam und Searle getrennt voneinander, aber aufeinander aufbauend dargestellt, interpretiert und gleichzeitig kritisiert. Der Schwerpunkt liegt auf Searle:

Abschnitt 3.1, *Searle: Symbolmanipulationen sind universell realisierbar*, stellt Searles Herleitung des Problems der universellen Realisierbarkeit vor. Um die Grundannahme des Kognitivismus: „Das Gehirn ist ein digitaler Computer“ ad absurdum zu führen, führt er eine gedanklich-empiristische Untersuchung durch mit der Frage: „Was würde ich tun, um herauszufinden, dass etwas ein digitaler Computer ist?“ Seine Untersuchung führt ihn zu dem Ergebnis, dass man auf die von ihm vorgeschlagene Weise für jeden beliebigen Gegenstand herausfinden könne, dass er ein digitaler Computer ist; auf dem obendrein jedes beliebige Programm implementiert ist. Es wird gezeigt, dass Searle in seiner Argumentation den Computer unzulässig mit der Turingmaschine – einer bestimmten Form des Endlichen Automaten – gleichsetzt, und dass dies das einzig Absurde an seiner Untersuchung ist.

In Abschnitt 3.2, *Putnam: Jedes beliebige physikalische System ist eine Realisierung jedes beliebigen EA*, wird referiert, wie Putnam das Problem der universellen Realisierbarkeit als Theorem beweist. Er stellt dafür einen Beispiel-EA mit zwei Zuständen ohne In- und Output vor und beweist mit Hilfe zweier Axiome aus den Grundsätzen der Physik, dass dieser EA jedes beliebige physikalische System realisiert. Danach überträgt Putnam das Ergebnis seines Beweises auf einen einfachen EA mit In- und Output. Es wird bestätigt, dass dieser Beweis Putnams formal in Ordnung ist und die Vermutung geäußert, dass er nicht den Funktionalismus in seiner ersten, von Putnam selbst vorgeschlagenen Version betrifft. Ursprünglich formulierte er mit dem Funktionalismus eine empirische

Hypothese, über die dieser Beweis keinerlei Aussagen macht.

In Abschnitt 3.3, *Searle: Die Syntax ist der Physik nicht intrinsisch*, wird Searles Analyse des tiefen Grundes für die universelle Realisierbarkeit vorgestellt. Der tiefe Grund ist nach Searle der, dass Computer rein syntaktisch definiert seien und also für die Identifikation eines Gegenstandes als Computer eine Zuordnung von Zeichen vorgenommen werden müsse. Solcherlei Zuordnungen seien aber beobachterrelativ. Man könne damit prinzipiell nichts identifizieren, das der Physik wirklich intrinsisch ist. Hier wird Searles Gleichsetzung von Computer und Turingmaschine, auf die seine Analyse beruht, korrigiert und gezeigt, dass der Computer nicht rein syntaktisch definiert ist. Searles spezielle Version der universellen Realisierbarkeit ist nach dieser Korrektur nicht mehr formulierbar. Seine Feststellung, dass die Syntax der Physik nicht intrinsisch ist, wird als wahr anerkannt aber zugleich als relativ unproblematisch eingestuft. Sie ist für die Kognitionswissenschaften nicht problematischer als für andere empirische Wissenschaften auch.

Im Fazit, Abschnitt 3.4, wird zum Abschluss festgestellt, dass das Problem der universellen Realisierbarkeit in seiner allgemein formulierbaren Form kein spezielles Problem der Kognitionswissenschaft ist. Es handelt sich dabei um ein altbekanntes, allgemeines Problem, das die Wissenschaft insgesamt immer hat, wenn sie die Dinge in der Welt mit Hilfe von Zeichen – oder auch mit Hilfe von Zeichnungen – beschreiben will. Dabei ist es gleichgültig, ob es um computationale Beschreibungen im Sinne der Kognitionswissenschaft geht oder um andere Beschreibungsarten. Dieses Problem eignet sich entgegen der Intentionen von Putnam und Searle nicht dazu, besonders die Kognitionswissenschaft zu kritisieren. Um dieses Ergebnis im Allgemeinen zu belegen, wird ein hundert Jahre altes Beispiel (1895) gebracht, worin ein seriöser Astronom dem Planeten Mars versehentlich Zeichen zugeordnete, die nichts von dem beschreiben, was dem Mars intrinsisch ist. Als Beleg für die konkrete Debatte um das Problem der universellen Realisierbarkeit in den Kognitionswissenschaften folgt zum Schuss ein Auszug aus einem aktuellen Dialog (2000) in dem Internetdiskussionsforum PSYCHE-B. Darin stellt ein Philosoph einem computerorientierten Neurobiologen direkt die Frage nach der universellen Realisierbarkeit, und der Biologe reagiert sehr gelassen darauf: Das Problem sei derart allgemein, dass es seine spezifischen Fragestellungen nicht im Besonderen betreffe; er tendiere dazu, es im Alltag seiner wissenschaftlichen Praxis zu ignorieren. Zu Recht, wenn das Ergebnis dieser Arbeit stimmt.

Es fragt sich vielleicht, wozu diese kleine negative Arbeit über zwei große negative Bücher von philosophischem Nutzen sein könnte. Darauf kann der Autor zusammen mit Putnam (vgl. Putnam 1999: 9) antworten, dass er wegen der insgesamt negativen Ausrichtung seiner Arbeit nicht niedergeschlagen ist. Denn eigentlich ist eine negative philosophische Arbeit oft auch ein positive Arbeit. Die vorliegende hat neben ihrem negativen Ergebnis, dass das Problem der universellen Realisierbarkeit nicht notwendig die gesamte Kognitionswissenschaft in unlösbare Schwierigkeiten bringt, durch ihren argumentativen Weg dahin auch zwei nützliche positive Ergebnisse. Erstens liefert sie während der Diskussion der Argumentation Searles einen Vorschlag, wie eine in der computerorientierten Philosophie des Geistes nach Searle aktuelle und ungelöste Frage zu beantworten ist; – die Frage: „Wie ist die Elektrotechnik mit der Mathematik zu verknüpfen?“ (vgl. Searle 1993: 226). Der Vorschlag, der in Abschnitt 3.3 dazu gemacht wird, ist ein mehr oder weniger Wittgensteinscher: Wenn die Begriffe, die es in der Theoretischen Informatik seit Alan Turing dazu gibt, so gelassen werden wie sie sind, ergibt sich die gewünschte Verknüpfung daraus ganz von selbst; – in diesem Falle ist die Verknüpfung eine klare Trennung. Zweitens kann man diese Arbeit, wenn nicht direkt, dann doch zumindest in ihrem Subtext, als kleines, spezielles, positives philosophisches Plädoyer für den ruhigen allgemeinen praktischen Umgang mit großen negativen philosophischen Büchern lesen.

2 Grundlagen

Das Problem der universellen Realisierbarkeit steckt tief in einer voraussetzungsreichen philosophischen Debatte. Besonders zwei theoretische Stränge laufen hier zusammen. Zum einen ein interdisziplinärer Forschungsansatz namens Kognitionswissenschaft und zum anderen ein abstraktes Konzept aus der Theorie der formalen Sprachen und Berechenbarkeit, der Endliche Automat (EA). Man kann die philosophische Problematik der Debatte in erster Annäherung so umreißen: Die Kognitionswissenschaft hat das Konzept des Endlichen Automaten als zentrales Axiom ihrer Grundannahmen importiert, und ihre Kritiker, Putnam und Searle, behaupten nun, dass sie sich damit das Problem der universellen Realisierbarkeit eingehandelt hat. Es wird hier also zuerst zu untersuchen sein, was es bedeutet, das Konzept des EA in die Kognitionswissenschaft zu importieren. In den folgenden beiden Abschnitten, 2.1 und 2.2, werden die Kognitionswissenschaft und das Konzept des Endlichen Automaten – mit Blick auf die spätere Analyse der Argumentationen bei Putnam und Searle – saparat besprochen

2.1 Kognitionswissenschaft

Searle richtet sich mit seinem Argument der universellen Realisierbarkeit ausdrücklich gegen das Forschungsprogramm der Kognitionswissenschaft – cognitive science. Er schreibt in einem Kapitel in *Die Wiederentdeckung des Geistes*, das er programmatisch *Die Kritik der kognitiven Vernunft* (Searle 1993: 218-249)[2] nennt:

> Über mehr als ein Jahrzehnt hinweg [...] war ich ein praktizierender »Kognitionswissenschaftler«. Während dieser Zeit habe ich viele Fortschritte und wertvolle Arbeit auf diesem Gebiet gesehen. Als Disziplin leidet die Kognitionswissenschaft jedoch daran, daß einige ihrer hochgeschätzten Grundannahmen falsch sind. (Searle 1993: 218)

Auch bei Putnam schwingt die Vermutung etwas zurückhaltender mit, dass seine Argumentation die gesamte Kognitionswissenschaft betreffen könnte. Er schreibt im Vorwort zu *Repräsentation und Realität* (Putnam 1999):

> In diesem Buch werde ich zu zeigen versuchen, daß die Computeranalogie – man mag sie die »Computeranalogie des Geistes«, die »kalkülmäßige Auffassung des Geistes«, »Funktionalismus« oder sonst wie nennen – eben doch nicht die Frage beantwortet, die wir Philosophen (ebenso wie viele Kognitionswissenschaftler) eigentlich beantworten

> wollen, nämlich die Frage: »Was ist das Wesen der mentalen Zustände?« (Putnam 1999: 11)

Es geht beiden Autoren um die Grundannahmen der Kognitionswissenschaften. Wenn man allerdings aktuelle Einführungen in die Kognitionswissenschaft liest, zum Beispiel Max Urchs' *Maschine, Körper, Geist* (Urchs 2002), erfährt man, dass dieser junge Forschungsansatz noch einigermaßen weit davon entfernt ist, überhaupt eine allgemein akzeptierte Beschreibung seiner Grundannahmen oder Ziele zu besitzen. Man stößt auf absichtlich vorsichtig formulierte, allgemein gehaltene Explikationsversuche wie:

> Ziel der Kognitionswissenschaft ist es, die intelligentem Handeln zugrunde liegenden Mechanismen zu verstehen. (Urchs 2002: 9)

> Kognitionswissenschaft ist die interdisziplinäre Wissenschaftsdisziplin, die den Geist als ein informationsverarbeitendes System untersucht. (Urchs 2002: 14)

Allein eine Liste der an der Kognitionswissenschaft beteiligten wissenschaftlichen Disziplinen lässt vermuten, dass eine einfache, klare und widerspruchsfreie begriffliche Bestimmung des Faches überhaupt schwierig werden könnte. Urchs zählt die untenstehenden zehn Disziplinen auf, wobei er die ersten fünf zu Kernbereichen und die anderen zu Randgebieten erklärt (vgl. Urchs 2002: 10f.):

- kognitive Psychologie (ältester Zweig der heutigen Kognitionswissenschaft, Wilhelm Wundt gründete sein Institut für experimentelle Psychologie bereits 1879)
- kognitive Sprachwissenschaft (Untersuchung der Sprache als einem wichtigen Teil menschlicher Kognition)
- künstliche Intelligenz (Konstruktion zumindest teilweise nichtbiologischer intelligenter Systeme)
- Neurowissenschaft (Untersuchung der Gehirnvorgänge mit dem Ziel, den Geist zu verstehen)
- kognitive Logik (Untersuchung der Struktur menschlicher Problemlösungsroutinen)
- Algorithmentheorie (Grundlagen der maschinellen Simulation intelligenten Verhaltens)
- Klinische Psychologie (Analyse kognitiver Aspekte von Hirntraumata)
- kognitive Anthropologie (phylogenetische Entwicklung der menschlichen Kognition)
- Pädagogik
- Kulturwissenschaft

Und noch immer, sagt Urchs, sei kein Ende dieser Liste in Sicht. Ein Forschungsprojekt, das derartig viele Disziplinen in sich vereinen will, dürfte gut daran tun, bei seiner Begriffsbestimmung einiges in der Schwebe zu lassen, um das gesamte Projekt nicht von vornherein in einem einzigen Metadiskurs um eindeutig explizierte gemeinsame Grundannahmen aufzureiben. Die beteiligten Betrachtungsebenen, Methoden und Begrifflichkeiten sind zu unterschiedlich. Es hat sich für die Kognitionswissenschaft schließlich als fruchtbar herausgestellt, mit einem Minimum an gemeinsamer Zielsetzung und vielleicht höchstens einigen Familienähnlichkeiten in den Grundannahmen zu arbeiten.

Searle ist offensichtlich selbst von den vielen Erfolgen angetan, die er bisher auf diesem Gebiet beobachten konnte. Aber leider scheinen ihm einige ihrer impliziten Grundannahmen falsch zu sein – für das weitere Vorwärtskommen der Sache hinderlich. Er beschwert sich in einigermaßen polemischem Ton darüber, dass in kognitionswissenschaftlichen Publikationen kein Versuch gemacht würde, die gemeinsamen Grundannahmen endlich, nachträglich, auf den Tisch zu legen und eine fächerübergreifende Diskussion darüber zu beginnen.

> Die übliche Vorgehensweise in diesen Büchern und Artikeln besteht darin, ein paar Bemerkungen über Einsen und Nullen zu machen, die Church-Turing-These populärwissenschaftlich zusammenzufassen und sich dann faszinierenderen Dingen (wie den Erfolgen und Fehlschlägen mit Computern) zuzuwenden. (Searle 1993: 226)

Also benennt Searle die Grundannahmen der, wie er sie nennt, Standard-Kognitionswissenschaft – mainstream cognitive science – kurzerhand selbst, um sie hinterher zu kritisieren. Mit dem Argument der universellen Realisierbarkeit hat er es dabei speziell auf einen Standpunkt abgesehen, den er Kognitivismus – cognitivism – nennt. In Abschnitt 2.1.1 werden Searles Herleitung des Kognitivismus und die von ihm explizierten Grundannahmen dieser Disziplin besprochen.

Bei Putnam zielt das Argument der universellen Realisierbarkeit in eine ähnliche Richtung, aber er hat damit, präziser, einen bestimmten Ansatz innerhalb der Kognitionswissenschaften im Auge, den er einige Jahre vorher selbst mit entwickelt hatte: den Funktionalismus. Dieser Ansatz hat verschiedene real existierende Vertreter, die seine Grundannahmen explizit ausgearbeitet haben, sodass man die Darstellung des von Putnam kritisierten Standpunkts nicht ausschließlich Putnam selbst überlassen muss. Der grundlegende Überblick über Putnams, und nicht nur Putnams, Funktionalismus in Ab-

schnitt 2.1.2 folgt neben Putnam (Putnam 1975a-e und 1982) im Wesentlichen Ned Block (Block 1980a, 1980b, 1990, 1995 und 2001) sowie den Arbeiten von David Armstrong, Jaegwon Kim und David Lewis (Armstrong 1980, Kim 1980, Lewis 1980).

2.1.1 Searles Kognitivismus

Das Herz der Kognitionswissenschaften ist nach Searle die so genannte Künstliche Intelligenz (KI) – Artificial Intelligence (AI). Und das Kritisieren von Ansätzen der KI hat bei Searle eine längere Tradition. Um bei seinen KI-skeptischen Argumentationen den begrifflichen Überblick zu behalten, teilt er die Sichtweisen der KI in drei Bereiche ein:

> I call the view that all there is to having a mind is having a program, Strong AI, the view that brain processes (and mental processes) can be simulated computationally, Weak AI, and the view that the brain is a digital computer, cognitivism. (Searle 1992: 201f.)

Diese Unterscheidung in a) Schwache KI, b) Starke KI und c) Kognitivismus lässt sich am besten im Hinblick auf die Unterschiede in Searles Argumentation für bzw. gegen diese Standpunkte verstehen:

a) Schwache KI

Den Standpunkt der Schwachen KI, die mentalen Prozesse, und zwar tatsächlich sämtliche mentalen Prozesse, könne man auf einem digitalen Computer simulieren, findet Searle vertretbar. Die Church/Turing-Hypothese, die er hier Churchs These nennt, bestätigt Searle, dass man alles auf einem digitalen Computer simulieren kann, was sich als präzise Abfolge von Schritten charakterisieren lässt:

> [G]iven Church's thesis that anything that can be given a precise enough characterization as a set of steps can be simulated on a digital computer, it follows trivially that the question ["Can the operations of the brain be simulated on a digital computer?"] has an affirmative answer. (vgl. Searle 1992: 200)

Mentale Prozesse ließen sich in diesem Sinne genau in der gleichen Weise auf einem digitalen Computer simulieren, wie sich darauf Wetterlagen, Erdbeben, Börsenbewegungen, Photosynthese, die Prozesse am synaptischen Spalt oder ähnliche präzise Abfolgen von Schritten simulieren lassen. Diese Art von Simulation empfindet Searle als bewusstseinsphilosophisch harmlos. Niemand, schreibt er, käme auf die Idee zu sagen,

allein schon weil man die Photosynthese auf einem digitalen Computer simulieren kann, ist die Photosynthese ein Computerprogramm und ein Blatt ein Computer, oder allein schon weil man Erdbeben auf einem Computer simulieren kann, sind Erdbeben ein Computerprogramm und die Erde ein Computer usw. Die Photosynthese eines Blattes ließe sich leicht von einer Computer-Photosynthese unterscheiden. Zum Beispiel so: Die Photosynthese des Blattes produziert Sauerstoff als Output, die des Computers produziert die Formel für Sauerstoff, die Zeichenkette „O_2“, als Output. (Dies ist nur die nicht sehr subtile Andeutung einer Möglichkeit von vielen, Photosynthese auf einem digitalen Computer zu simulieren.) Der Stoff der Photosynthese im Blatt ist grün und heißt Chlorophyll, der Stoff der Photosynthese im Computer hat keine besondere Farbe und heißt Eingabealphabet und Überführungsfunktion. Es gibt Kriterien, mit denen man die Photosynthese im Blatt von dieser Photosynthesesimulation im Computer unterscheiden kann. Simulation (Computer-Photosynthese) und Simuliertes (Blatt-Photosynthese) sind verschieden. Das ist für Searle auch bei der Simulation von mentalen Prozessen auf einem digitalen Computer der Fall.

Man könnte hier u. a. einwenden, dass es durchaus Fälle von Simulation gibt, bei denen es schwer fällt, ein sinnvolles Kriterium anzugeben, das sie vom Simulierten unterscheidbar macht. Wie simuliert man beispielsweise eine Pantomime, ohne Pantomime zu machen? (vgl. Urchs 2002: 82) Wie simuliert man Träumen auf einem digitalen Computer, ohne dass der digitale Computer träumt? Sind die Sätze: „Der Computer macht keine Photosynthese, er simuliert sie nur“ und „Der Computer träumt nicht, er simuliert es nur“ gleichermaßen sinnvoll? Um diesen Einwand zu entkräften und die letzte Frage mit Ja zu beantworten, braucht Searle ein Argument, das zeigt, weshalb die Simulation von mentalen Prozessen nicht eine Simulation der Pantomimesorte ist. Ein reductio ad absurdum Argument, das man aus seinem Text herauslesen kann und das für diese Arbeit relevant ist, lautet: Es muss einen Unterschied zwischen dem mentalen Prozess und seiner Simulation auf einem digitalen Computer geben, denn die Ansicht, dass es keinen gibt, führt zur Starken KI, wenn nicht gar zum Kognitivismus; – und diese beiden Standpunkte hält Searle für falsch bzw. im Fall des Kognitivismus tatsächlich für absurd.

b) Starke KI

Bei seiner Argumentation gegen das, was er Starke KI nennt, hat Searle das Verhältnis zwischen Semantik und Syntax im Auge. Starke KI sei die Ansicht: „Einen Geist haben heißt ein Programm zu haben, und mehr ist am Geist nicht dran.“ (Searle 1993: 223) Um

Geist zu haben, reicht es für Searle aber nicht aus, ein Programm zu haben. Programme, sagt er, sind rein syntaktischer Natur, und das Semantische ist dem Syntaktischen nicht intrinsisch. Um Geist zu haben, braucht es aber ein Verständnis der Semantik, d. h. der Bedeutung dessen, was mit den Zeichen transportiert wird.

Das Argument, mit dem er zu beweisen versucht, dass die Semantik nicht der Syntax intrinsisch sei, ist das inzwischen sehr berühmte Chinese-Room-Argument. Wenn jemand, der kein Chinesisch versteht, so das Argument, auf rein syntaktischer Ebene chinesische Zeichen so verarbeitet, dass man diesen Prozess nicht vom normalen Sprachgebrauch eines chinesischen Muttersprachlers unterscheiden kann – was anscheinend funktioniert, und sei es nur ‚for the sake of the argument' –, wird er dennoch nie zu einem bewussten Verständnis dieser Zeichen gelangen. Zur Illustration dieses Arguments stellt Searle ein Gedankenexperiment an: Ein Mann, sagen wir ein Deutscher, der kein einziges chinesisches Zeichen versteht, es auch nicht von den Zeichen anderer ostasiatischer Sprachen unterscheiden kann, befindet sich allein in einem Zimmer. Durch eine Klappe erhält er als Input einen Zettel mit für ihn unverständlichen, chinesisch anmutenden, ostasiatischen Zeichen darauf. Er hat in seinem Zimmer nur ein syntaktisches Wörterbuch, worin in seiner Muttersprache Regeln stehen, die ihm vorschreiben, wie er die Zeichen auf dem Zettel mit anderen Zeichen in Beziehung setzen soll, etwa so[3]:

Wenn Sie

お元気ですか。

lesen, dann schreiben Sie bitte

はい, 元気です。

Das Ergebnis gibt er auf einem anderen Zettel als Output wieder durch die Klappe aus dem Zimmer heraus. Wenn man annimmt, so Searle, dass auf dem ersten Zettel, dem Input, „Fragen" und auf dem zweiten Zettel, dem Output, „Antworten" stehen, dann wäre leicht einzusehen, dass jeder, der diese Zeichen lesen und verstehen kann, die Antworten, die aus diesem Mann-im-Zimmer-System herauskommen, als korrekte Antworten einer existierenden natürlichen Sprache erkennen würde. Gleichzeitig sei aber völlig klar, dass der Deutsche in dem Zimmer auf diese Weise nie ein Wort dieser Sprache zu verstehen lernt, wie viele solcher formalen Operationen er auch durchführen mag. (vgl. hierzu Searle 1980. Einen recht guten, im Internet zugänglichen Überblick über die Debatte rund um das Chinese-Room-Argument liefert Hauser 1993.)

Das Chinese-Room-Argument ist nicht das Thema dieser Arbeit und kann hier nicht umfassend besprochen werden, es sei aber darauf hingewiesen, dass einige Gegenargumente, die in der Fachliteratur diskutiert werden, sehr plausibel zu sein scheinen. Besonders sei der so genannte System-Einwand genannt, der besagt, dass Searle die Aufmerksamkeit des Lesers zu stark auf den Mann im Zimmer lenkt, sodass dabei leicht übersehen wird, dass sich das Mann-im-Zimmer-System als Ganzes doch frappant so verhält, wie etwas, dem man ein normales Sprachverstehen und Sprachbenutzen zuschreiben sollte. Und diese Übertragungsebene wäre einer Computeranalogie zudem die angemessenere. Man könnte nun, dem Bild schlicht folgend, in einer Art Chinese-Head-Argument das gesamte Searle'sche Chinesisch-Zimmer in den Kopf eines Menschen verlegen – im Rahmen solcher leicht verstiegenen Gedankenexperimente dürfte so etwas gestattet sein – und nach diesem Schritt, wenn man ihn mitgeht, hätte man es sehr schwer abzustreiten, dass der Mensch mit dem Chinesisch-Zimmer im Kopf in ganz normalem Sinn Chinesisch kann. (vgl. hierzu auch Block 1995: Abschnitt 4, und Urchs 2002: 125f.) Hier läuft wie in a) die Schwierigkeit darauf hinaus, ein Kriterium anzugeben, das die Simulation vom Simulierten unterscheidet.

c) Kognitivismus

Bei seiner Argumentation gegen das, was er Kognitivismus nennt, geht es Searle um das Verhältnis zwischen Syntax und Physik. Die Argumentationsebene ist gegenüber b) gewissermaßen um eine Stufe der Hierarchie Semantisches → Syntaktisches → Physisches tiefer gelegt worden. Kognitivismus ist nach Searle „die Auffassung, das Gehirn sei ein digitaler Computer" (Searle 1993: 223), und sein Vorwurf gegen diese Auffassung ist formal ganz ähnlich, wie der, den er der Starken KI macht. Das Syntaktische, argumentiert Searle, ist dem Physischen nicht intrinsisch. Computer seien nun aber per definitionem syntaktische, symbolverarbeitende Maschinen und Gehirne demgegenüber rein physische Objekte in der Welt. Unternimmt man den Versuch, Gehirne ebenfalls als syntaktische, symbolverarbeitende Maschinen aufzufassen, und zwar im absolut wörtlichen Sinne, dann führt das für Searle zu absurden Konsequenzen: Man könne dann mit genau dem gleichen Recht jedes beliebige andere Objekt in der Welt als syntaktische, symbolverarbeitende Maschine auffassen. Wände, Tische, Steine, Blumentöpfe und immer so weiter, jedes beliebige materielle Ding. Die Information, dass das Gehirn ein digitaler Computer ist, würde damit nutzlos, ohne interessante Aussage, da auf einmal schlicht und einfach alles ein digitaler Computer wäre. Die Ansicht, dass das Gehirn syntaktische Symbolmanipulation realisiert – um es noch einmal prozessual zu fassen –, führt für Searle dazu, zu behaupten, dass jedes beliebige Objekt in der Welt jede beliebige syntaktische

Symbolverarbeitung realisiert. Symbolverarbeitung ist nichts intrinsisch Physisches, sondern immer eine – beliebige – Zuordnung von außerhalb; – universell realisierbar im dem Sinne, dass sie jedem beliebigen Gegenstand beliebig zuzuordnen sei.

Das Argument, mit dem Searle begründet, warum Computer per definitionem syntaktische, symbolverarbeitende Maschinen und syntaktische Symbolmanipulation universell realisierbar seien, ist Gegenstand der vorliegenden Arbeit. Es wird in Abschnitt 3 detailliert behandelt. Zuvor sollte allerdings noch geklärt werden, was genau – nach Searle – die Annahmen des Kognitivismus sind, die zum Problem der universellen Realisierbarkeit führen. Er fasst sie in einem zentralen Abschnitt von *The Rediscovery of the Mind* (Searle 1992) wie folgt zusammen, wobei zu beachten ist, dass er hier aus dem Mund der Kognitivisten spricht, um ihren Standpunkt zunächst so stark wie möglich zu begründen. Im Originaltext ist diese Passage ein wenig eingerückt gesetzt, damit auch grafisch deutlich wird, dass hier nicht Searle, sondern sein hypothetischer Gegner im Plural spricht. Er nennt sie *The Primal Story of Cognitivism, Die Urgeschichte des Kognitivismus*:

> We begin with two results in mathematical logic, the Church-Turing thesis and Turing's theorem. For our purposes, the Church-Turing thesis states that for any algorithm there is some Turing machine that can implement that algorithm. Turing's thesis says that there is a universal Turing machine that can simulate any Turing machine. Now if we put these two together, we have the result that a universal Turing machine can implement any algorithm whatever. […] It is clear that at least some human mental abilities are algorithmic. For example, I can consciously do long division by going through the steps of an algorithm for solving long-division problems. It is furthermore a consequence of the Church-Turing thesis and Turing's theorem that anything a human can do algorithmically can be done on a universal Turing machine. I can implement, for example, the very same algorithm that I use for long division on a digital computer. In such a case, as described by Turing (1950), both I, the human computer, and the mechanical computer are implementing the same algorithm. I am doing it consciously, the mechanical computer nonconsciously. Now it seems reasonable to suppose that there might be a whole lot of other mental processes going on in my brain nonconsciously that are also computational. And if so, we could find out how the brain works by simulating these very processes on a digital computer. Just as we got a computer simulation of the processes for doing long division, so we could get a computer simulation of the processes for understanding language, visual perception, categorization, etc. (Searle 1992: 202f.)

Diese *Urgeschichte* führt laut Searle unmittelbar zum Problem der universellen Realisierbarkeit. Was, zumindest hier in diesem Text, in dem die beiden Darstellungen von

Schwacher KI und Kognitivismus so dicht beieinander stehen, eine einigermaßen verwirrende Feststellung sein dürfte. Um es noch einmal zu rekapitulieren: Bei seiner Argumentation für die Schwache KI sagt Searle, die Church/Turing-Hypothese, die er dort Churchs These nennt, bestätige harmloserweise nur, dass man jeden beliebigen mentalen Prozess auf einem digitalen Computer simulieren kann; – und nichts weiter. Bei seiner Argumentation gegen den Kognitivismus führt die Church/Turing-Hypothese, die er dann Church-Turing These nennt, im Gegenteil fatalerweise zu der Annahme, dass das Gehirn selbst, wie jeder andere Gegenstand, im wörtlichen Sinne ein digitaler Computer sei.

Man könnte annehmen, dass Searle glaubt, Churchs These, die er bei der Schwachen KI anführt und die Church-Turing These, die er dem Kognitivismus zuschreibt, seien zwei verschiedene Thesen. Es handelt sich dabei aber nur um zwei Namen für dieselbe Sache, was man zum Beispiel sehr unterhaltsam in Douglas Hofstadters' *Gödel, Escher, Bach* (vgl. Hofstadter 1992: 598) nachlesen kann. Hofstadter unterscheidet darin elf verschiedene Lesarten dieser Hypothese; und man sollte daher fairerweise vermuten, dass Searle den Unterschied zwischen Schwacher KI und Kognitivismus in verschiedenen Interpretation der Church/Turing-Hypothese sieht: Kognitivisten intepretierten demnach die Hypothese so, dass daraus folgt, das Gehirn sei ein digitaler Computer; für Vertreter der Schwachen KI folgt diese Annahme nicht. Worin besteht aber – laut Searle – der Unterschied zwischen den beiden Lesarten, die zu diesen sehr unterschiedlichen Sichtweisen führen?

Leider ist Searles Text an dieser Stelle sehr dunkel. Er führt zwei verschiedene Namen für die selbe Hypothese ein, ohne je darauf hinzuweisen, dass es sich um dieselbe handelt, und den Grund für diesen sprachlichen Griff verschweigt er. Zudem stellt sich Searle, der sich darüber beschwert, dass die Church/Turing-Hypothese in der Philosophie immer nur populärwissenschaftlich abgehandelt würde, nicht besser, wenn er selbst einfach keine Abhandlung der Church/Turing-Hypothese bringt und nur zwei verschiedene Interpretationen vorträgt. Hier noch einmal die beiden Interpretationsvarianten Searles gegenübergestellt:

Schwache-KI-Variante:

> Church's thesis [states] that anything that can be given a precise enough characterization as a set of steps can be simulated on a digital computer. (vgl. Searle 1992: 200)

Kognitivismus-Variante:

> The Church-Turing thesis states that for any algorithm there is some Turing machine that can implement that algorithm. (Searle 1992: 202)

Der große Unterschied dieser beiden Varianten dürfte darin bestehen, dass Searle die Kognitivismus-Variante mit den (Fach-)Begriffen „Algorithmus“ und „Turingmaschine“ formuliert und die Schwache-KI-Variante dagegen eher allgemein verständlich in Begriffe des normalen Wortschatzes fasst: „Abfolge von Schritten“ und „digitaler Computer“. Die Kognitivismus-Formulierung führt nach Searle zum Problem der universellen Realisierbarkeit, die der Schwachen KI hingegen nicht. Und dies liegt, wie später in den Abschnitten 3.1 und 3.3 gezeigt wird, genau an dem Begriff der Turingmaschine, den Searle den Kognitivisten zuschreibt, und der für Searle bei der Schwachen KI, so scheint es, entweder keine oder eine andere, harmlose Rolle spielt.

Um Searles Kognitivismus-Verwendung des Begriffs der Turingmaschine, und was dabei schief läuft, genauer fassen zu können, muss ein von Searles Darstellung unabhängiger Blick auf den Endlichen Automaten und die Turingmaschine, die eine Spezialform des Endlichen Automaten ist, geworfen werden. Abschnitt 2.2 liefert deshalb eine detaillierte Darstellung des Endlichen Automaten, sowie eine hoffentlich nicht populärwissenschaftliche Zusammenfassung der Church/Turing-Hypothese.

2.1.2 Putnams Funktionalismus

Der Funktionalismus ist von Putnam als Gegenvorschlag zu der so genannten klassischen Identitätstheorie des Geistes entworfen worden. Wobei der Funktionalismus im Bereich der Identitätstheorien bleibt; es wird nur etwas anderes als in den klassischen identifiziert. In der klassischen Identitätstheorie, die hauptsächlich von H. Feigl, U. T. Place und J. J. C. Smart um 1960 in die philosophische Debatte gebracht wurde (vgl. hierzu Borst 1970), ging es darum, Zustände des Geistes mit Zuständen des Gehirns zu identifizieren. Diese Theorie wird in der Literatur auch unter dem Namen Gehirnzustandstheorie – brain-state theory – diskutiert. Thomas Nagel unterscheidet in seinem Aufsatz *Physikalismus* (Nagel 1993) vier Arten, wie in verschiedenen Gehirnzustandstheorien eine Identifikation von Zuständen des Geistes (zum Beispiel Schmerzen) und Zuständen des Gehirns (zum Beispiel Feuern der C-Fasern[4]) versucht wird (vgl. Nagel 1993: 56). Sämtliche Versionen der Gehirnzustandstheorie haben dabei – wie es der Titel *Physikalismus*

von Nagels Aufsatz andeutet – gemeinsam, dass sie versuchen, physikalische Zustände des Gehirns mit mentalen Zuständen zu identifizieren.

An diesem Punkt setzt Putnams Kritik an. Er sagt, dass er mit dieser Fixierung auf das Physische des Gehirns in der Gehirnzustandstheorie nicht zufrieden sei. Sie würde zu einem extrem exklusiven Chauvinismus führen. Es könne immerhin sein, dass es irgendwo im Weltall Bewohner eines fremden Planeten gebe, deren Gehirne, oder besser deren Gehirnentsprechungen, aus einem ganz anderen Material bestünden als unsere, die aber genau die gleichen mentalen Zustände haben wie wir. Mentale Zustände, so Putnams Argument, sind multipel realisierbar. Sie sind nicht notwendigerweise an irgendeinen physischen Stoff gebunden, deshalb darf man die Untersuchung des Wesens des Geistes nicht von vornherein an die zufällige materielle Beschaffenheit unserer Gehirne binden. Man müsse viel liberaler an die Sache herangehen.

Das Problem der multiplen Realisierbarkeit, das die Gehirnzustandstheorie in arge Bedrängnis gebracht hat[5] (vgl. z. B. Kim 1980), ist nicht zu verwechseln mit dem Problem der universellen Realisierbarkeit, das in dieser Arbeit hauptsächlich behandelt wird. Das Problem der universellen Realisierbarkeit hat sich Putnam erst eingehandelt, als er versuchte, das Problem der multiplen Realisierbarkeit los zu werden. Bei dem Versuch, die Philosophie des Geistes vom extremen Chauvinismus zu befreien, führte er sie in einen extremen Liberalismus hinein. So würden die Kritiker – unter ihnen ein späterer Putnam – sagen, die ihm das Problem der universellen Realisierbarkeit vorwerfen. Zuerst einmal aber noch etwas mehr zum Versuch des jüngeren Putnam, die Philosophie des Geistes aus dem Problem der multiplen Realisierbarkeit herauszuführen:

Im Ansatz, sagt Putnam, sei es schon richtig, Zustände des Geistes mit Zuständen des Materials zu identifizieren, die mit ihnen korrespondieren. Nur die Art der Zustände sei falsch gewählt. Es ginge vielmehr um nichtphysikalische Zustände des Gehirns, die das Wesentliche des Geistes ausmachen. Und eben weil die Philosophie sich prinzipiell mit den wesentlichen und nicht mit den zufälligen Eigenschaften eines Phänomens beschäftigt, sollten von Philosophen auch für eine Identifikation der Zustände des Geistes nichtphysikalische Zustände des Gehirns genommen werden. Wie man sich solche nichtphysikalischen Zustände vorzustellen hat, darüber schreibt Putnam:

> Nun, was meine ich mit der Aussage, daß das Gehirn nichtphysikalische Eigenschaften hat? Ich meine Eigenschaften, die in Begriffen, die den physikalischen oder chemischen Aufbau des Gehirns unerwähnt lassen, definiert werden können. Falls es seltsam scheint, daß ein physikalisches System nichtphysikalische Eigenschaften hat, denke man an eine Rechenmaschine. Eine Rechenmaschine hat viele physikalische Eigenschaften. Sie hat z. B. ein bestimmtes Gewicht, und sie hat eine bestimmte Anzahl von Schaltelementen o. dgl. Sie hat ökonomische Eigenschaften, wie z. B. einen bestimmten Preis, und sie besitzt auch funktionale Eigenschaften, wie z. B. ein bestimmtes Programm. Diese letztere Art von Eigenschaften ist nichtphysikalisch in dem Sinne, daß ein System sie unabhängig von seiner sozusagen metaphysischen oder ontologischen Zusammensetzung zeitigen kann. Es könnte sein, daß ein körperloser Geist, ein Gehirn und eine Maschine je ein bestimmtes Programm aufweisen und daß der funktionale Aufbau dieser drei Programme – des körperlosen Geistes, des Gehirns und der Maschine – genau gleich wären, obwohl ihre Materie – ihr Stoff – völlig verschieden ist. (Putnam 1982: 111)

Es sind, wie Putnam hier darlegt, funktionale Zustände des Gehirns, die ihm besonders für die Identifikation mit Zuständen des Geistes geeignet zu sein scheinen. Er vergleicht das Gehirn, einen körperlosen Geist und eine Rechenmaschine und stellt fest, dass der funktionale Aufbau der drei in gewissem Sinne gleich sein könne. Um diesen funktionalen Aufbau näher zu explizieren, benutzt er die Begrifflichkeiten des Konzepts der Endlichen Automaten bzw. der Turingmaschine; in seinen Texten nennt er sie auch Probabilistische Automaten – Probabilistic Automata:

> Natürlich ist eine Turingmaschine einfach eine besondere Art eines Probabilistischen Automaten, eine mit Übergangswahrscheinlichkeiten 1,0. (Putnam 1993: 127)

Dass es bei dieser Übertragung – dem Import, wie oben einleitend gesagt – der Begriffe des Endlichen Automaten aus der Theorie der formalen Sprachen und Berechenbarkeit in die Philosophie des Geistes zu einem Vorwurf in Richtung universelle Realisierbarkeit kommen könnte, schien Putnam auch 1968 schon zu ahnen, als er seinen ersten Aufsatz zu diesem Thema schrieb[6]. Und wie um sich präventiv davor zu schützen, führte er den Begriff der Beschreibung eines Systems ein:

> Since an empirically given system can simultaneously be “a physical realisation“ of many different Probabilistic Automata, I introduce the notion of a *Description* of a system. A Description of *S* where *S* is a system, is any true statement to the effect that *S* possesses distinct states $S_1, S_2 ..., S_n$ which are related to one another and to the motor outputs and sensory inputs by the transition probabilities given in such-and-such a Machine Table.

> The Machine Table mentioned in the Description will then be called the Functional Organization of *S* relative to that Description. (Putnam 1975e: 434)

1968 brauchte Putnam den Begriff der Beschreibung, um seine Hypothese „Schmerz zu haben ist ein funktionaler Zustand des Organismus“ (Putnam 1993: 128) präzise zu formulieren und das Konzept des Probabilistischen Automaten theoretisch mit Organismus und mentalem Zustand in einen Zusammenhang bringen zu können:

1. All organisms capable of feeling pain are Probabilistic Automata.
2. Every organism capable of feeling pain possesses at least one Description of a certain kind (i.e. being capable of feeling pain is possessing an appropriate kind of Functional Organisation).
3. No organism capable of feeling pain possesses a decomposition into parts which separately possesses Descriptions of the kind referred in (2).
4. For every Description of the kind referred in (2), there exists a subset of sensory inputs such that an organism with that Description is in pain when and only when some of its sensory inputs are in that subset. (Putnam 1975e: 434)

Diese Hypothese und ihre Ausformulierung in den Begriffen des Endlichen Automaten dürfte hier für die allermeisten philosophischen Leserinnen und Leser trotz aller Bemühung Putnams um analytische Präzision durchaus noch immer schwer zu greifen sein. Das liegt an dem außerphilosophischen Voraussetzungsreichtum, den diese Begriffe mit sich bringen und dem hohen Grad an Abstraktion, den Putnam hier vorlegt, und von dem er an keiner Stelle seiner Texte bereit ist hinabzusteigen. Zwar führt er in verschiedenen Texten den Endlichen Automaten und die Turingmaschine ein (vgl. Putnam 1975c und 1975d), doch tut er das in einer derartigen Kürze und Dichte, dass man, wie in einem hermeneutischen Zirkel, die Turingmaschine immer schon verstanden haben muss, um sie in Putnams Texten zu verstehen.

Der einzige philosophische Autor, bei dem sich etwas Konkreteres dazu finden lässt, wie man sich einen Endlichen Automaten als Modell für Schmerzzustände vorstellen kann, ist Ned Block. In seinen verschiedenen, immer wieder überarbeiteten Einführungsaufsätzen zum Funktionalismus (erstmals Block 1980a, aktuell Block 2001)[7] führt Block zunächst einen Endlichen Automaten (EA) für irgendein einfaches mathematisches Problem ein und führt danach im Groben vor, wie man dieses Modell für die funktionalistische Beschreibung des Zustandes „Schmerzen haben“ benutzen könnte. In seinem aktuellsten Aufsatz *Functionalism* (Block 2001) nimmt er einen EA für den Ungerade-Gerade-Test

und stellt ihn für den Anfang in Form einer Transitionstabelle – auch, wie bei Putnam oben, Machine Table genannt – vor:

	S_1	S_2
1	**"Odd"** S_2	**"Even"** S_1

Abb. 2.1: Transitionstabelle für den Ungerade-Gerade-EA

Dieser EA hat zwei Zustände, S_1 und S_2, ein Zeichen als Input, „1", das beliebig oft hintereinander eingegeben werden kann, und zwei Outputs, „Odd" – „Ungerade" und „Even" – „Gerade". Der EA ermittelt, ob eine gerade oder ungerade Anzahl von Einsen eingegeben wurde. Die Tabelle ist wie folgt zu lesen: In der ersten Spalte steht der Input, der einen Zustandswechsel auslöst. Spalte zwei besagt, dass der EA, wenn er sich in Zustand S_1 befindet, „Odd" ausgibt und in Zustand S_2 überwechselt, Spalte drei besagt, dass der EA, wenn er sich in Zustand S_2 befindet, „Even" ausgibt und in Zustand S_1 überwechselt. Der EA wird in Zustand S_1 anhalten, wenn eine ungerade Anzahl von Einsen eingegeben wurde, andernfalls in Zustand S_2. (Genaueres zum Konzept des EA in Abschnitt 2.2.) In ein formelähnliches Format gebracht und vom Standpunkt „in Zustand S_1 sein, heißt ..." aus betrachtet, könnte der Sachverhalt so ausgedrückt werden:

> Being in S_1 = being in the first of two states that are related to one another and to inputs and outputs as follows: being in one of the states and getting a '1' input results in going into the second state and emitting "Odd"; and being in the second of the two states and getting a '1' input results in going into the first and emitting "Even". (Block 2001)

Um die Quantifikation über Zustände besser herauszustellen, gibt Block noch eine zweite Formulierung mit den entsprechenden Existenzquantoren an:

> Being in S_1 = Being an x such that ∃P∃Q [If x is in P and gets a '1' input, then it goes into Q and emits "Odd"; if x is in Q and gets a '1' input it goes into P and emits "Even" & x is in P] (Note: read '∃P' as There is a property *P*.) (Block 2001)

Damit zeigt sich deutlicher, wie es gemeint ist, dass über die Zustände eines Systems geredet werden kann, ohne dessen physikalische Beschaffenheit zu erwähnen. Obiger

Ungerade-Gerade-EA könnte das Verhalten eines gängigen elektronischen PC aus Silizium beschreiben, man könnte sich aber durchaus genauso gut eine mechanische Konstruktion aus Holz oder eine mit fließendem Wasser aus kleinen Schläuchen und Ventilen vorstellen, auf die diese Beschreibung passt, man könnte sogar ein ausgetüfteltes System als Labyrinth aus Gängen und Türchen bauen, mit dressierten Mäusen darin, das sich gemäß des EA verhält (vgl. dazu auch Block 1995, insbesondere das UND-Gatter aus Katzen und Mäusen in Abb. 5). Blocks Transitionstabelle und Formeln lassen das Material außer Acht und sprechen nur darüber, was all diesen multiplen physikalischen „Realisierungen“ des Ungerade-Gerade-EA wesentlich gemeinsam zu sein hat: die funktionalen Zustände S_1 und S_2.

Angenommen, schreibt Block weiter und setzt damit zur philosophischen Pointe an, wir hätten eine Theorie, die mentale Zustände ganz genau so spezifiziert wie die obigen Formeln die Zustände des Ungerade-Gerade-EA: als kausale Relationen über Zustände, sensorische Inputs und behaviorale Outputs. Eine solche Theorie würde dann die Relation zwischen dem mentalen Zustand „Schmerzen haben“, dem sensorischen Input „Auf einer Reißzwecke sitzen“ und dem behavioralen Output „’Autsch’ rufen“ wie folgt auf eine Formel bringen:

> Being in pain = Being an x such that $\exists P \exists Q$[sitting on a tack causes P & P causes both Q and emitting ‘ouch’ & x is in P]
>
> More generally, if T is a psychological theory with n mental terms of which the 17th is ‘pain’, we can define ‘pain’ relative to T as follows (the ‘F_1’... ‘F_n’ are variables that replace the n mental terms, and i_1, etc. and o_1, etc. indicates):
>
> Being in pain = Being an x such that $\exists F_1 ... \exists F_n$ [T(F_1...F_n, i_1, etc, o_1, etc) & x is in F_{17}] (Block 2001)

Über diese funktionalistische psychologische Theorie T, die er hier hypothetisch einführt, schreibt Block ausdrücklich, dass es sich dabei um eine empirisch erforschte Theorie oder um eine common sense Volkspsychologie handeln könne, die natürlich ebenfalls einer empirischen Überprüfung standhalten müsste. Man befindet sich demnach beim Projekt des Funktionalismus voll und ganz auf der Ebene der Formulierung empirischer Hypothesen. Putnam sah das 1965 genau so. Er schrieb:

> Ich werde mich nicht dafür entschuldigen, daß ich eine empirische Hypothese vorbringe. [...] Die detaillierte Entwicklung und Verifikation meiner Hypothese wäre eine genau so utopische Aufgabe wie die detaillierte Entwicklung und Verifikation der Gehirnzustandshypothese. Aber das Aufstellen nicht von detaillierten und wissenschaftlich „fertigen“ Hypothesen, sondern von Schemata für Hypothesen, ist schon seit langem eine Funktion der Philosophie. (Putnam 1993: 127)

Man muss dem nicht zustimmen, dass es tatsächlich eine Aufgabe der Philosophie ist, einer anderen Disziplin utopische Projekte vorzuschlagen, aber es ist nicht von der Hand zu weisen, dass so etwas ab und zu passiert.

2.2 Zum Konzept des Endlichen Automaten (EA)

Der Endliche Automat (EA) – finite/discrete state automaton – ist ein abstraktes mathematisches Modell für ein System mit einer endlichen Anzahl interner Zustände und diskreten Ein- und Ausgaben. Ein Zustand dieses Systems wird als die Menge der Aktionen beschrieben, die das System in diesem Zustand ausführt sowie als die Information, die sich aus ihm ablesen lässt. Als Eingabe für den EA wird eine Zeichenkette aus einem vorgegebenen Eingabealphabet angesehen. Die Aktionen, die der Endliche Automat in einem Zustand maximal ausführen kann, sind: *Ein Symbol aus der Menge der Eingabesymbole lesen, ein Symbol aus der Menge der Eingabesymbole (über)schreiben, den Zustand wechseln oder nichts tun* (= *anhalten*). Nicht jede dieser Aktionen muss notwendig in jedem Zustand vorgeschrieben sein. Der Endliche Automat besitzt genau einen Startzustand; eine Teilmenge aus der Menge seiner Zustände wird als Menge der Endzustände bezeichnet. Ein Endlicher Automat beschreibt (erkennt, akzeptiert) eine Zeichenkette aus dem Eingabealphabet genau dann, wenn die Eingabe dieser Zeichenkette dazu führt, dass der EA einen Endzustand erreicht und dort anhält.

2.2.1 Beispiel: Ein EA Für das Mann/Wolf/Ziege/Kohl-Problem

Das Mann/Wolf/Ziege/Kohl-Problem besteht aus der folgenden Situation: Ein Mann steht mit einem Wolf, einer Ziege und einem Kohlkopf an einem Flussufer und will den Fluss gern zusammen mit seinen drei Besitztümern überqueren. Leider hat er dazu bloß ein sehr kleines Ruderboot zur Verfügung, in dem er nur jeweils eines der drei mitnehmen kann. Er muss daher mehrmals fahren, um seine Tiere und seinen Kohlkopf einzeln über den Fluss zu bringen. Dabei kommt komplizierend hinzu, dass er weder die Ziege mit dem

Kohlkopf zusammen an einem Ort allein lassen kann, weil die Ziege den Kohlkopf dann auffressen würde, noch den Wolf mit der Ziege, denn in diesem Fall wäre die Ziege, wenn er zurückkommt, nicht mehr da.

Um einen EA zu konstruieren, mit dessen Hilfe der Mann eine Lösung für sein Dilemma finden kann, muss man seine Situation und das, was er in dieser Situation tun kann, zuerst irgendwie als System mit einer endlichen Menge von Zuständen beschreiben. Eine Möglichkeit wäre, die denkbaren Kombinationen von Mann (M), Wolf (W), Ziege (Z) und Kohlkopf (K) an den beiden Ufern des Flusses als Zustandsmenge eines Systems anzusehen. Diese Menge ist klarerweise endlich. Einer der so beschriebenen Zustände wäre z. B. *MZ – WK*; zu lesen als: *Der Mann steht jetzt mit der Ziege am linken Ufer, der Wolf mit dem Kohlkopf am rechten*. Ein Zustand wie *ZK – MW* wäre fatal und dürfte von einem EA, der zu einer Lösung des Problems kommen soll, nicht angenommen werden. Die Zustandswechsel können bei diesem Ansatz mit der Bootsfahrt assoziiert werden. Wenn der Mann etwa bei Zustand *MWK – Z* zusammen mit dem Kohlkopf ans rechte Flussufer fährt, dann ergibt sich daraus der Zustand *W – MZK*. Zustandswechsel kommen bei Endlichen Automaten normalerweise in Reaktion auf eine Eingabe zustande, deshalb ist es in diesem Mann/Wolf/Ziege/Kohl-Fall hier sinnvoll, die Aktionen des Mannes als Eingaben aufzufassen, verkürzt geschrieben als: *Fluss allein überqueren*: Eingabe *m*, *Fluss zusammen mit dem Wolf überqueren*: Eingabe *w*, *Fluss zusammen mit der Ziege überqueren*: Eingabe *z* und *Fluss zusammen mit dem Kohlkopf überqueren*: Eingabe *k*. Das gültige Eingabealphabet dieses EA besteht damit aus der Menge {*m*, *w*, *z*, *k*}.

Als anschauliche formale Darstellung der Zustände und Zustandswechsel eines EA bietet sich u. a. der so genannte Transitionsgraph an. Dabei werden die Zustände als Kreise, in der Fachsprache Knoten genannt, dargestellt und die Zustandswechsel als Pfeile, wobei das Eingabesymbol jeweils an den Pfeil herangeschrieben wird. Endzustände werden doppelt eingekreist. Im hier besprochenen Beispiel gibt es nur einen Endzustand.

Ein Transitionsgraph zur Lösung des Problems, einen Wolf, eine Ziege und einen Kohlkopf unter beengten logistischen Bedingungen über einen Fluss zu bringen, sähe dann so aus (Hopcroft/Ullman 1990: 15):

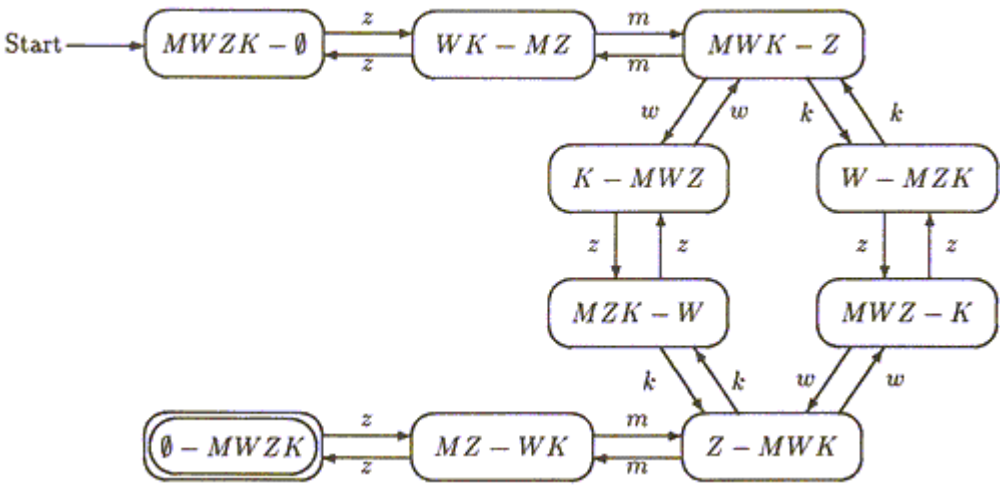


Abb. 2.2: Transitionsgraph für den Mann/Wolf/Ziege/Kohl-EA

Etwas in dieser Art muss zum Beispiel Ned Block gemeint haben, als er sagte, man könne auch ein ausgefuchstes System aus Katzen, Mäusen und Käse als Endlichen Automaten auffassen (vgl. Block 1990 und Block 1995). Hier wird das Ganze zufällig an einem ausgefuchsten System aus einem Mann, einem Wolf, einer Ziege und einem Kohlkopf durchgespielt.

Abstrakt gesprochen, sozusagen in der Sprache der Endlichen Automaten, akzeptiert ein EA, der gemäß dem obigen Transitionsgraph funktioniert, die beiden Zeichenketten *zmwzkmz* und *zmkzwmz* des Eingabealphabets. Denn nur unter Eingabe einer dieser Zeichenketten erreicht der EA, vom Startzustand aus gesehen, seinen Endzustand. Dass dieser EA auch rückwärts läuft, liegt in der symmetrischen Natur des Problems; das muss nicht immer der Fall sein, es ist keine notwendige Eigenschaft eines EA. Versuchte man, eine andere Zeichenkette aus dem Eingabealphabet an diesem EA auszuprobieren, dann würde er nicht bis zu seinem Endzustand kommen. Gäbe man zum Beispiel *zmwm...* ein, dann käme der EA nur bis zu Knoten *K – MWZ* und müsste dort anhalten, weil es für die Eingabe *m* an dieser Stelle keine Aktion gibt, die er auszuführen hat.

Im konkreten Fall der Mann/Wolf/Ziege/Kohl-Situation, für die der EA entworfen wurde, bedeutet sein Ergebnis, dass der Mann zwei Möglichkeiten hat, das ihm vorliegende logistische Problem zu lösen. Nämlich entsprechend einer der Anweisungen, die sich hinter einer der beiden Zeichenketten *zmwzkmz* und *zmkzwmz* verbergen, zu handeln. Das heißt z. B., gemäß *zmwzkmz,* zuerst die Ziege hinüberbringen, allein zurückfahren, den

Wolf mitnehmen, danach zusammen mit der Ziege zurückfahren, um den Kohl mitzunehmen und danach noch ein Mal alleine zurück, um die Ziege abzuholen. Handelt er nicht gemäß einer der beiden Ergebnisse, dann hat das zur Folge, dass ihm entweder sein Kohlkopf oder seine Ziege abhanden kommt. Bei einer Entscheidung, entsprechend *zmwm* zu handeln, würde ihm hinterher die Ziege fehlen.

Zum Vergleich mit der in 2.2 genannten allgemeinen Beschreibung des EA: Der hier vorgestellte EA hat eine endliche Menge von möglichen Zuständen, nämlich genau zehn; sie entspricht der Anzahl der Knoten im Transitionsgraphen aus Abb. 2.2. Er hat, da er so konstruiert wurde, einen Input, der durch das Eingabealphabet vorgegeben ist, und er liefert Ergebnisse: Zwei Zeichenketten aus dem Eingabealphabet, die dadurch abzulesen sind, dass der EA bei ihrer Eingabe und nur bei ihrer Eingabe den Endzustand erreicht. Dieser EA hat keinen Output, denn er verändert die Zeichen seiner Eingabe während des Durchlaufs seiner Prozedur nicht.

Oben wurde zuerst das zu lösende Problem als System mit endlich vielen Zuständen beschrieben. Man entwarf also ein System, das noch gar nicht arbeitete, man beschrieb eine Menge möglicher zukünftiger Prozesse. Danach wurde ein EA konstruiert, der die gewünschte prozessuale Lösung herausbrachte, und zuletzt konnte einer der beiden (Handlungs-)Abläufe, die zum Ziel führen, umgesetzt werden. Man kann sich aber auch andersherum vorstellen, dass ein Wissenschaftler, eine Art Quinescher Ethnologe beispielshalber, einen Mann beobachtet, wie er im Augenblick dabei ist, in beschriebener Manier mit Hilfe eines kleinen Ruderbootes einen Wolf, eine Ziege und einen Kohlkopf über einen Fluss zu transportieren. Und dieser Ethnologe könnte auf die Idee kommen, die Szene, die er im Moment beobachtet, mit Hilfe eines EA zu beschreiben. Es ist denkbar, dass er die Szene genau auf die gleiche Weise wie oben als ein System mit endlichen Zuständen beschreibt und in seinem Notizbuch Sätze wie: „Jetzt ist die Situation *KW – MZ* eingetreten“, „Jetzt geschieht *z*“ usw. notiert. Möglicherweise zeichnet er auch gleich Pfeile und Kreise wie in Abb. 2.2 hinein. Und es ist denkbar, dass der Wissenschaftler hinterher einen erfolgreichen Pfad, das heißt einen der möglichen Verbindungen zwischen Start- und Endknoten des obigen Graphen, in seinem Notizbuch stehen hat.

Dies ist, zumindest wenn man Putnam und Searle folgt, eine erste Annäherung an die Idee, die dem Forschungsprogramm kognitionswissenschaftlicher Ansätze zugrunde liegt. Es wird etwas in Echtzeit mit Hilfe eines EA beschrieben. Einem System, das läuft, wird

gleichzeitig aus einer passiven Beobachterposition heraus sein funktionaler Schaltplan abgeschaut. Oder, wie die Kritiker der Kognitionswissenschaft Putnam und Searle sagen, er wird ihm angedichtet; allein schon die Zustände werden, den Kritikern zufolge, dem System willkürlich von außen als reine Projektion übergestülpt.

Eine Schwierigkeit beim Konzept des EA, die für das Problem der universellen Realisierbarkeit, vor allem bei Searle, eine zentrale Rolle spielt, lässt sich an dieser Stelle besonders gut veranschaulichen. Es scheint, dass die Modellierung eines Systems durch Endliche Automaten zu der Annahme verführt, das System selbst würde, und zwar in der gleichen Weise wie der EA, Symbole verarbeiten. Hier hat es den Anschein, als sei der Mann so etwas wie das symbolverarbeitende Modul des Mann/Wolf/Ziege/Kohl-Systems. Bei genauer Betrachtung des theoretischen Status des EA dürfte aber deutlich werden, dass dieses Konzept so nicht gemeint sein kann und dass nur die naheliegenden Beispiele, wie das obige hier, und die Illustrationen zum EA (siehe Abb. 2.4 weiter unten) zu dieser falschen Vermutung führen. Wahrscheinlich trägt auch das Wort „Automat“ in Namen dieses Konzepts und die mechanistische Redeweise von „Zuständen“, die „durchlaufen“ werden, einiges zu dem Missverständnis bei, ein EA könne tatsächlich gebaut werden. Das ist falsch und irreführend. Der EA ist ein abstraktes Modell, und wenn man im Zusammenhang mit Endlichen Automaten von „konstruieren“ spricht, dann meint man damit eine Tätigkeit, die normalerweise mittels Strichen auf einem Blatt Papier stattfindet. Bauen, simulieren oder nachstellen kann man nur den Prozess innerhalb eines Systems, den der EA beschreibt oder vorschreibt. Das Verhältnis ist ähnlich, wie das des Wortes „Badewanne“ zu dem Gegenstand, den es bezeichnet. Das Wort „Badewanne“ kann man nicht bauen, man kann nur eine Badewanne bauen.

Wahrscheinlich trägt darüber hinaus die ein erkennendes Bewusstsein implizierende Redeweise, der EA „erkennt“ eine Zeichenkette, er „liest“ ein Zeichen etc., zu dem weiteren Missverständnis bei, das vom EA beschriebene System würde Symbole verarbeiten. Bei Schöning (zum Beispiel Schöning 1992: 28), einem theoretischen Informatiker, kann man deutlich ein leichtes Unwohlsein beim Gebrauch dieser Begriffe spüren, er setzt sie ab und zu in Anführungsstriche; denn diesem theoretischen Modell „lesen“ zuzugestehen, kann nicht 100%ig im gleichen Sinne gemeint sein, wie einem wirklichen Automaten – wie einem Zigarettenautomaten oder einem Barcodescanner – oder einem Menschen lesen zuzugestehen. Der Sinn „lesen“ wurde in der theoretischen Schwebe gelassen.

Im ersten Fall, in dem der Mann den EA entwirft, um eine Lösung für sein Problem zu finden, ist das Verhältnis zwischen Papier und Wölfen/Ziegen etc. allein durch den zeitlichen Abstand leicht zu verdeutlichen. Der EA, der vor der eigentlichen Szene da ist, erhält in Zustand *MWZK* - ∅ die Eingabe *z* und verarbeitet dieses Symbol, indem er daraufhin in den Zustand *WK* – *MZ* überwechselt. Dies alles wird auf einem Blatt Papier dargestellt und vollzieht sich in der Vorstellung des Mannes. Falls er ein versierter EA-Anwender ist, benötigt er unter Umständen noch nicht einmal ein Blatt Papier zu seiner Hilfe. Wenn er aber hinterher die Lösung umsetzt, die der EA ergeben hat, dann rudert er einfach zuerst zusammen mit seiner Ziege über den Fluss und steht danach mit der Ziege am anderen Flussufer. Dem Mann wird dabei kein Symbol *z* eingegeben und er befindet sich dabei niemals in einem Zustand *MWZK* - ∅ . Das Verhältnis zwischen EA und der Szene, die er modelliert, ist das Verhältnis zwischen Beschreibung und Beschriebenem oder zwischen einem Plan und seiner Umsetzung. Mit Hilfe des EA modelliert man ein System auf der Ebene der Zeichen. Das System verarbeitet keine Symbole, sondern es durchläuft einen Prozess. In einem Fall wie hier, in dem jemand den Prozess Schritt für Schritt bewusst durchführt, könnte man sagen, dass der EA eine Art normative Wirkung hat. Der EA schreibt mit Hilfe von Symbolen vor, wie der Prozess auf der Ebene von Menschen, Wölfen, Ziegen und Kohlköpfen abzulaufen hat. Der EA normiert damit gewissermaßen einen Handlungsablauf. (Als erstes Beispiel für Algorithmen werden Informatikstudenten im ersten Semester häufig Kochrezepte genannt. Kochrezepte geben auch einen normierten Handlungsablauf vor, den man – schon sehr umständlich – mit Hilfe eines EA beschreiben könnte.)

Auch aus dem zweiten Fall, in dem der Ethnologe die vor ihm ablaufende Szene mit Hilfe des EA beschreibt, dürfte deutlich hervorgehen, dass der Beschreibende mit „Jetzt geschieht *z*“ nicht meinen kann, dass dem Mann jetzt ein *z* eingegeben wird und er dieses *z* jetzt verarbeitet. Auch hier geschieht die Verarbeitung des Zeichens *z* auf der Beschreibungsebene. Der Mann bringt eine Ziege über den Fluss und der Ethnologe bezeichnet dies als Eingabe *z* für den Endlichen Automaten, den er soeben in sein Notizbuch zeichnet.

Im Hinblick auf das Problem der universellen Realisierbarkeit sind an dieser Stelle zwei Zwischenbeobachtungen angebracht. Zum einen lässt sich feststellen, dass es durchaus alternative Endliche Automaten zur Beschreibung des Mann/Wolf/Ziege/Kohl - Ablaufs gibt. Man könnte alternativ beispielsweise die Besatzung des Bootes als Zustand des

Systems annehmen. Zustand 1: *Jetzt ist das Boot leer*. Zustand 2: *Jetzt ist der Mann zusammen mit der Ziege im Boot* usw. Wenn man diese Version als Transitionsgraphen notieren wollte, müsste man nur die Bezeichnungen der Knoten mit denen der Pfeile vertauschen und in den Start und den Endknoten jeweils eine *0* für *Jetzt ist das Boot leer* hineinschreiben. So hätte man einen alternativen EA zur Beschreibung des Ablaufs.

Andererseits wird hier schon deutlich, dass es, wenn man die vorgeschlagene Einteilung der Situation in Zustände und die Menge des Eingabealphabets als angemessen akzeptiert, ziemlich viele Endliche Automaten gibt, die als Beschreibung des Lösungsalgorithmus des Mann/ Wolf/Ziege/Kohl-Problems nicht gültig sind. Ein EA etwa, der einen Zustand *WZ – KM* als Endzustand annehmen kann, „realisiert" die Lösung nicht. Das Problem der universellen Realisierbarkeit muss also vor einer solchen Vereinbarung auf Zustände und Eingabealphabete liegen.

2.2.2 Die Rolle des EA in der Theoretischen Informatik

Ursprünglich waren Endliche Automaten dafür gedacht, Zeichenketten auf ihre Gültigkeit innerhalb eines formalen Systems zu testen. Das Ergebnis eines solchen Testvorgangs wird daraus abgelesen, ob der EA, dem die zu testende Zeichenkette eingegeben wird, bei ihrer Bearbeitung einen Endzustand erreicht (positives Ergebnis), an einem Nicht-Endzustand anhält oder ob er in einen zirkulären Zustandswechsel gerät (negative Ergebnisse).

In der Theoretischen Informatik, der Wissenschaft, aus der das Konzept des EA als Ansatz für den Computerfunktionalismus übernommen wurde, wird der EA formal wie folgt definiert (Definition und Beispiel vgl. Schöning 1992: 27ff.):

Definition 2.2.2

Ein Endlicher Automat EA wird spezifiziert durch ein 5-Tupel

$$EA = (Z, \Sigma, \delta, z0, E).$$

Hierbei bezeichnet Z die Menge der Zustände und Σ ist das Eingabealphabet, $Z \cap \Sigma = \emptyset$. Z und Σ müssen endliche Mengen sein. $z0 \in Z$ ist der Startzustand, $E \subseteq Z$ ist die Menge der Endzustände und $\delta : Z \times \Sigma \rightarrow Z$ heißt die Überführungsfunktion.

Beispiel 2.2.2

EA = (Z, Σ, δ, $z0$, E), mit

$Z = \{z0, z1, z2, z3\}$

$\Sigma = \{a, b\}$

$E = \{z3\}$

$\delta(z0, a) = z1$

$\delta(z0, b) = z3$

$\delta(z1, a) = z2$

$\delta(z1, b) = z0$

$\delta(z2, a) = z3$

$\delta(z2, b) = z1$

$\delta(z3, a) = z0$

$\delta(z3, b) = z2$

und dem entsprechenden Transitionsgraphen (aus Schöning 1992: 28):

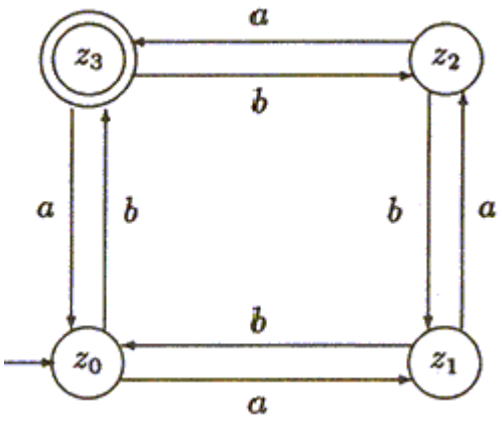


Abb. 2.3: Transitionsgraph für den EA aus Beispiel 2.2.2

Dieser Beispielautomat beschreibt (erkennt, akzeptiert) u. a. alle Zeichenketten (Zk) der Art:

Zk = { x $\in \Sigma^*$ | ((Anzahl a's in x) – (Anzahl b's in x)) MOD 4 = 3}

Das Summenzeichen mit dem Sternchen bedeutet eine beliebige Zeichenkette zusammengesetzt aus dem Eingabealphabet $\sum = \{a, b\}$. MOD ist die Bezeichnung für die Rechenart modulo.

In der Theoretischen Informatik hat der Endliche Automat einen sehr hohen Stellenwert. Er ist letztendlich das geeignete Konzept, mit dem bestimmte Aussagen über die Möglichkeiten und Grenzen eines formalen Systems gemacht werden können. Über das obige Eingabealphabet kann man zum Beispiel jetzt behaupten, dass man aus seinen Zeichen nur Reihenfolgen herstellen kann, zu denen sich auch ein EA ähnlich dem aus Beispiel 2.2.2 konstruieren lässt, der sie akzeptiert. Für durchaus irgendwie formulierbare Reihenfolgen, die aber unmöglich konkret hinzuschreiben sind, lässt sich auch kein solcher EA konstruieren. Zum Beispiel wird es niemandem gelingen, für Zeichenketten der Art:

$$Zk = \{ x \in \sum^* \mid ((\text{Anzahl a's in } x) + (\text{Anzahl b's in } x)) = -1\}$$

einen EA zu konstruieren.

Es kursieren diverse bildliche Darstellungen des Endlichen Automaten in der informatischen Literatur. Eine typische wäre die folgende (aus Schöning 1992: 29):

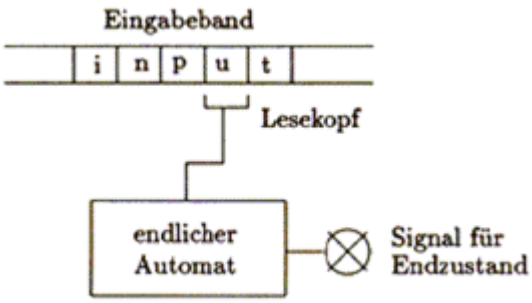


Abb. 2.4: Typische Illustration zum EA

Solche Illustrationen sind im philosophischen Zusammenhang aber viel eher hinderlich als hilfreich, denn sie legen die Vermutung nahe, ein Endlicher Automat sei eine reale Maschine in der wirklichen Welt der konkreten Dinge, an der man herumschalten kann. So ist diese Illustration aber nicht gemeint. Sie veranschaulicht nur die Bedeutung dessen, was Mathematiker mechanische Lösungen von Problemen nennen: einen Ablauf von Symbolveränderungen, der vollkommen planmäßig, Schritt für Schritt, algorithmisch abläuft. Es

gibt zwei wesentliche Gründe, weshalb man einen Endlichen Automaten nicht in der Welt der Dinge antreffen kann. Erstens wird das Eingabeband in einigen Versionen des EA als unendlich gedacht, und Unendliches kann man nicht bauen. Zweitens ist in dieser Illustration überhaupt nichts darüber ausgesagt, wie dieser kleine Kasten, auf dem vielversprechend die Wörter „Endlicher Automat“ und „Lesekopf“ stehen, das Programm, das durch so etwas wie den Transitionsgraphen gegeben ist, überhaupt umsetzt. Es müssten irgendwelche kleinen Mechaniken oder Organe angenommen werden, die mit der notwendigen, minimalen Beschreibung des einzelnen Zustands nichts zu tun haben. Um dieses Problem zu umgehen, setzen viele Autoren einfach ein kleines, gerne schwachsinniges Männchen in diesen Kasten hinein, das die einzelnen Befehle, *lies, schreibe, gehe links, gehe rechts,* ausführt, der EA und der Lesekopf werden also der Anschaulichkeit halber einfach personifiziert:

> We are not concerned with the mechanics or the electronics of the matter. Perhaps the simplest way to picture the thing is quite crudely: Inside the box there is a man, who does all the reading and writing and erasing and moving. (Boolos/Jeffrey 1989: 21)

Es wird ein Homunkulus in die Maschine gesetzt. Bei einem theoretischen, abstrakten Modell für mechanische Algorithmen ist das kein Problem. Man darf nur nicht den Fehler machen, das Modell eins zu eins ins Praktische, Konkrete zu übertragen.

Autoren zur Theoretischen Informatik wie John Hopcroft und Jeffrey Ullman sind sich weitgehend darüber einig, dass der Computer kein Endlicher Automat ist, sondern auf einer anderen ontologischen Ebene liegt. Aussagen der folgenden Art lassen sich recht häufig finden:

> Obwohl die Spannung an den Gattern jeden aus einer unendlichen Menge an Werten annehmen kann, ist der elektronische Schaltkreis so konzipiert, dass nur die Spannungen, die 1 und 0 entsprechen, stabil sind und sich alle anderen Spannungen fast augenblicklich auf diese Spannungen einstellen. Schaltkreise werden absichtlich so entwickelt, dass sie als Systeme mit endlichen Zustandsmengen angesehen werden können. Dadurch wird der logische Entwurf eines Computers von der elektronischen Implementation getrennt. (Hopcroft/Ullman 1990: 13)

Hier wird die Trennung mit den Worten „logischer Entwurf“ versus „elektronische Implementation“ deutlich gemacht. Den logischen Entwurf kann man zum Beispiel mit Hilfe eines Endlichen Automaten anfertigen, er befindet sich auf der Ebene der Zeichen und

Symbole. Bei der elektronischen Implementation hat man es dagegen mit fließendem Strom zu tun, dessen Spannungen erst als „1“ oder „0“ aufgefasst werden müssen, und die mit Absicht so konstruiert wurden, dass diese externe Lesart auf der Entwurfs- und Beschreibungsebene leicht ist. Diese Spannungen selbst sind keine Einsen und Nullen, und es gibt in den Schaltkreisen kein Bewusstsein oder kleines Männchen, das diese Spannungen als solche auffasst. Auf der Ebene der elektronischen Implementation befindet sich pure Mechanik/Elektronik. Insofern muss man im philosophischen Kontext sehr vorsichtig damit sein, wenn man den Computer eine symbolverarbeitende Maschine nennt. Genaugenommen ist nur das Modell des Computers, der Endliche Automat, eine symbolverarbeitende Maschine, und damit verarbeitet letztlich derjenige Symbole, der sich den EA vorstellt oder ihn auf dem Papier nachvollzieht. Der reale Computer verarbeitet auf der Hardwareebene keine Symbole, sondern dort knistert, klackert und surrt er allenfalls. Er befindet sich in einem Prozess, der mit Hilfe eines EA beschrieben, entworfen oder nachvollzogen werden kann. Genau so wie der Mann in Beispiel 2.2.1 sich in einem Prozess, in diesem Fall in einem Handlungsablauf, befindet, der als EA entworfen, beschrieben oder nachvollzogen werden kann. Symbolverarbeitung findet beim Computer erst auf der Ebene der Interpretation statt. Der Benutzer instrumentalisiert den Computer zur Symbolverarbeitung. Inzwischen instrumentalisiert er ihn aber auch schon für viele andere Dinge; zum Musik hören oder zum Bilder betrachten beispielsweise.

Für die abstrakte theoretische Analyse der Eigenschaften von formalen Systemen hat sich der EA sehr bewährt. Für die Praxis der Computerwissenschaften dagegen, wenn es um konkrete Computer und deren Programmierung geht, erwies sich der EA eher als unergiebig. Zwar findet er dort Einsatz bei sehr begrenzten Problemen wie dem Entwurf von lexikalischen Analysern (das sind kleine Programm-Module, die Wörter erkennen müssen) oder bei der Beschreibung kleinster Schaltkreise, üblicherweise für die Grundrechenarten. Für den Entwurf oder die Betrachtung großer Computer oder Computerprogramme wird er aber nicht herangezogen. Auch über die Brauchbarkeit des Konzepts für Untersuchungen über existierende Systeme in der Welt – Pflanzen, Wetterlagen oder Gehirne – äußern sich Hopcroft und Ullman skeptisch:

> Es ist verführerisch, das menschliche Gehirn als System mit einer endlichen Zustandsmenge anzusehen. Die Zahl der Gehirnzellen oder Neuronen ist begrenzt – wahrscheinlich höchstens 2^{35}. Es ist vorstellbar – obwohl es Anhaltspunkte dafür gibt, dass dies nicht der Fall ist – dass der Zustand jedes Neurons mit einer kleinen Anzahl Bits beschrieben werden kann. Wäre dies der Fall, so ließe sich die Theorie der endlichen Zustandsmen-

gen auf das Gehirn anwenden. Die Zahl der Zustände wäre jedoch so groß, dass es unwahrscheinlich ist, dass dieser Ansatz zu nützlichen Beobachtungen über das menschliche Gehirn führen könnte, ebenso wenig, wie Annahmen über endliche Zustandsmengen uns helfen, große, aber endliche Computersysteme zu verstehen.
(Hopcroft/Ullman 1990: 14)

2.2.3 Turingmaschinen und die Church/Turing-Hypothese

Um diese Zurückhaltung und Skepsis besser nachvollziehen zu können, ist es am besten, sich die elaborierteste Version des Endlichen Automaten, die so genannte Turingmaschine vor Augen zu führen. Bisher wurden nur begrenzte Endliche Automaten (nur Input) mit einem relativ wenig aussagekräftigen Eingabealphabet betrachtet. Bei der Turingmaschine sieht man den Bruch zwischen theoretischer Beweiskraft und praktischer Anwendbarkeit am deutlichsten. Das Konzept Turingmaschine macht Aussagen über die Berechenbarkeit von Funktionen in den formal mächtigsten Systemen überhaupt, z. B. der klassischen Arithmetik, möglich. Aber schon bei der konkreten algorithmischen Darstellung sehr einfacher arithmetischer Funktionen wird der Einsatz der Turingmaschine sehr schnell sehr mühsam, wenn nicht lebenszeitverbrauchend aussichtslos.

Die Turingmaschine wurde in den Jahren 1935 bis 1936 von dem englischen Mathematiker Alan Mathison Turing entwickelt, um das so genannte Entscheidungsproblem zu lösen: die damals in der theoretischen Mathematik virulente Frage, ob es unberechenbare Funktionen gibt, und wenn ja, wie man entscheiden kann, ob man eine solche vor sich hat.

Üblicherweise (nicht notwendigerweise) besteht das Eingabealphabet basaler Beispiel-Turingmaschinen in der Literatur nur aus der Eins. Zahlen werden durch Reihen von Einsen dargestellt, die sich auf einem unendlichen, in kleine Kästchen eingeteilten Band befinden, pro Kästchen ein Zeichen, und die durch so genannte Blanks, Leerstellen, voneinander getrennt sind. Die Zahlendarstellung erfolgt dabei nach der so genannten monadischen Notation: 1 = 1, 11 = 2, 111 = 3, 1111 = 4 usw. Die Blanks, die man natürlich auch als Zeichen betrachten kann, werden nicht zum Eingabealphabet gezählt. Der Schreibekopf – alternativ „Lese/Schreibe-Kopf“, „Scanner“ oder „kleines Männchen in der Kiste“ – kann pro Zustand und pro Kästchen ein Zeichen lesen, schreiben oder löschen, ein Kästchen nach rechts oder links gehen oder anhalten sowie den Zustand wechseln oder beibehalten.

Als einfaches Beispiel hier, auch um noch eine zum Transitionsgraphen alternative Darstellung für Endliche Automaten vorzuführen, eine Transitionstabelle für eine Turingmaschine, die Additionen durchführt. Oder, in der funktionalen Redeweise gesagt, die eine Funktion der Form f(x,y) = x+y berechnet, wobei x und y aus der Menge der Natürlichen Zahlen stammen (vgl. Hodges 1994: 116):

	Gelesenes Zeichen: Blank	Gelesenes Zeichen: ‚1'
Zustand 1	Gehe nach rechts. Bleibe in Zustand 1.	Gehe nach rechts. Wechsle zu Zustand 2.
Zustand 2	Schreibe ‚1'. Gehe nach rechts. Wechsle zu Zustand 3.	Gehe nach rechts. Wechsle zu Zustand 2.
Zustand 3	Gehe nach links. Wechsle zu Zustand 4.	Gehe nach rechts. Bleibe in Zustand 3.
Zustand 4 Endzustand	Halte an. Bleibe in Zustand 4.	Lösche die ‚1'. Halte an. Bleibe in Zustand 4.

Abb. 2.5: Transitionstabelle für eine Additions-Turingmaschine

Die Aussage der Tabelle 2.5 kann man auch mit einem Transitionsgraphen machen. Die beiden Darstellungsweisen sind äquivalent.

Das für Turings Intention interessante Ergebnis dieser Turingmaschine ist eigentlich das Faktum, dass sie, auf eine Funktion der Form f(x,y) = x +y angesetzt, irgendwann, und zwar korrekt beim Endzustand, anhält. Damit wurde gezeigt, dass diese Funktion berechenbar ist. Dass sie auch das Ergebnis der Addition als Zeichenkette auf ihrem Band als Output liefert, ist für Turings Zwecke nur ein Nebenprodukt gewesen.

Unten ist ein Beispiel des Outputs der obigen Additions-Turingmaschine zu sehen, für die Funktion f(4,5) = 4+5. Der erste Ausriss zeigt einen interessanten Teil des unendlichen Bandes und die Position des Scanners am Anfang der Prozedur, die Turingmaschine befindet sich dabei in Zustand 1. Der zweite zeigt den gleichen Teil des Bandes und die Position des Scanners nach Ablauf der Prozedur, die Turingmaschine befindet sich dabei in Zustand 4, dem Endzustand (vgl. Hodges 1994: 117).

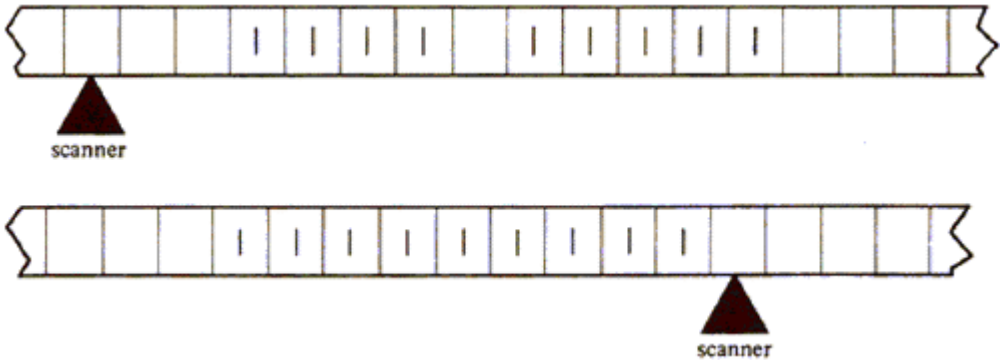


Abb. 2.6: Erster und letzter Output der obigen Addtions-Turingmaschine

Eine Turingmaschine für die Addition ist noch überschaubar, und auf diesem Niveau ist es auch sinnvoll, wissenschaftlich und technisch hilfreich, mit Tabellen oder Graphen für Endliche Automaten zu arbeiten. Selten – und wenn, dann auch nur zum Zweck der höheren Fingerübung – findet man in der Literatur Beispiele, die komplexere Funktionen als die Addition behandeln. Eines dieser Fundstücke ist das Buch *Computabiltity and Logic* von George S. Boolos und Richard C. Jeffrey (Boolos/Jeffrey 1989), in dem eine Turingmaschine für die Multiplikation positiver ganzer Zahlen, also Funktionen der Form f(x,y) = x ∗ y, wiederum mit x und y aus der Menge der Natürlichen Zahlen, vorgestellt wird

Das Beispiel soll hier vor allem zeigen, wie schnell es geht, dass die Transitionsgraphen von Endlichen Automaten als Darstellungswerkzeug für Algorithmen unübersichtlich werden. Der Graph für die Multiplikation ist ein hoher Genuss der theoretischen Spielerei im Abstrakten und dürfte schon kurz hinter der oberen Grenze des praktisch Sinnvollen liegen. Die Multiplikation beim gängigen PC funktioniert völlig anders, und es würde wahrscheinlich niemand unter kommerziellen Bedingungen auf die Idee kommen, eine Rechenmaschine zu bauen, die nach dem untigen Turingmaschinen-Schema arbeitet.

In der Darstellung in Abb. 2.7 unten wird wie bei der Additions-Turingmaschine (Abb. 2.6) die monadische Notation verwendet. Knoten 1 bezeichnet den Startzustand, Knoten 18 den Endzustand. An den Pfeilen steht auf der linken Seite des Doppelpunktes jeweils das gelesene Zeichen und auf der rechten die auszuführende Aktion. Demnach ist *1:R* zu lesen als der Befehl für den Scanner *wenn du eine Eins liest, gehe ein Kästchen nach rechts, B:1 als wenn du ein Blank liest, überschreibe es mit ‚1'* usw. Die Zustandswechsel sind durch die Pfeile angegeben (Boolos/Jeffrey 1989: 31):

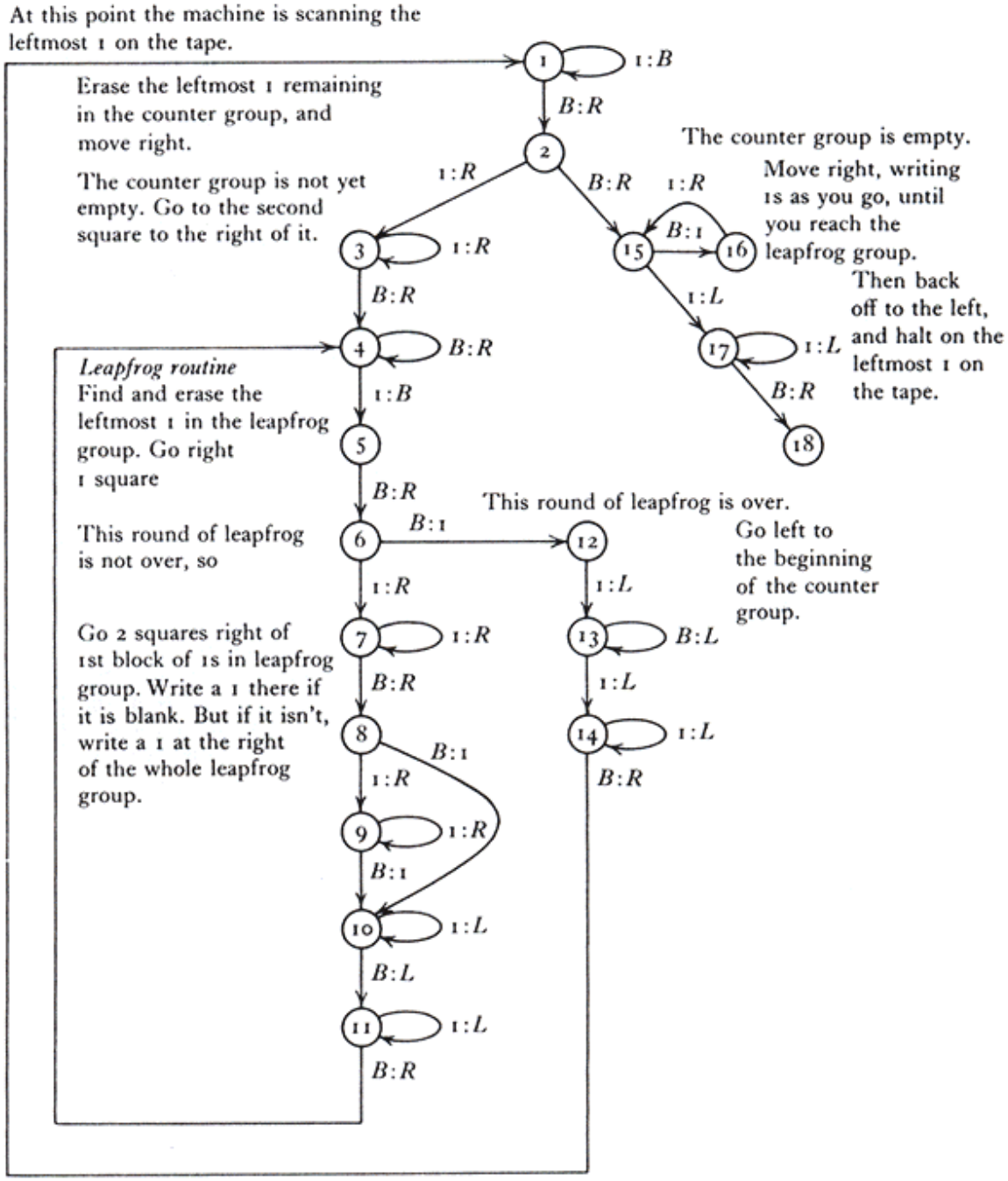


Abb. 2.7: Transitionsgraph für die Multiplikations-Turingmaschine

Im Vergleich zu den Funktionen, die man bräuchte, um die Vorgänge in so komplexen Systemen wie einem handelsüblichen PC oder sogar dem menschlichen Gehirn angemessen zu beschreiben, ist die Multiplikation vergleichsweise simpel, sie mutet in diesem Kontext geradezu atomar an. Es ist nur noch schwer vorstellbar, dass jemand ernsthaft den Versuch unternehmen würde, einen Transitionsgraphen für eine Turingmaschine aufzuzeichnen, die alle relevanten Prozesse detailliert beschreibt, die sich auf der Hard-

wareebene eines PC abspielen, wenn das Wordstar-Programm läuft. Die notwendige Menge der anzunehmenden Zustände nur zu schätzen, dürfte ausgesprochen schwierig sein. Wenn man sich das vor Augen führt, kann man sehr gut nachvollziehen, weshalb Putnam seinen Vorschlag, die Zustände des Geistes mit Turingmaschinen zu beschreiben, schon 1968 utopisch fand.

Für die Theorie der Berechenbarkeit, die Turing im Sinne hatte, war es nicht nötig, den Lösungsalgorithmus für komplexere Funktionen tatsächlich als Transitionsgraph für Turingmaschinen hinzuschreiben. Es genügten auch für diese Zwecke einfache Beispiele völlig. Der Rest der Theorie der Berechenbarkeit wurde in Form von formalen Betrachtungen über die Eigenschaften der Turingmaschine formuliert. Eines der wichtigsten Ergebnisse dieser Betrachtungen war, dass jede beliebige Art, eine Funktion zu berechnen, etwa mit Hilfe von WHILE-Schleifen oder GOTO-Anweisungen, genauso viel leisten kann, wie die Art, Funktionen zu berechnen, die das Konzept der Turingmaschine anbietet. Diese Erkenntnis ist in der berühmten Church/Turing-Hypothese folgendermaßen formuliert (vgl. Schöning, 1992: 88):

> **Church/Turing-Hypothese**
>
> Die durch die formale Definition der Turing-Berechenbarkeit erfasste Klasse von Funktionen stimmt genau mit der Klasse der intuitiv berechenbaren Funktionen überein.

Die Church/Turing-Hypothese lässt sich nicht beweisen, aber sie ist „allgemein akzeptiert“ (Schöning 1992: 88). Aus ihr folgt, dass jede Funktion, die nicht turingberechenbar ist, überhaupt nicht berechenbar ist sowie selbstverständlich umgekehrt, dass jede Funktion, die irgendwie berechenbar ist, auch turingberechenbar ist.

Der oben angedeutete große Bruch zwischen theoretischer Mächtigkeit und praktischer Anwendbarkeit ist der, dass in Praxis schnell effektivere Verfahren zur Berechnung von Funktionen benutzt werden als das minimalistische, aber auch aufwendige der Turingmaschine. Man könnte jedes Mal, das behauptet die Church/Turing-Hypothese, einen Transitionsgraphen für eine Turingmaschine entwerfen, die den gleichen Output liefert. Andere Arten, Funktionen zu berechnen, WHILE-Programme, rekursive Programme und viele andere mehr, beinhalten nicht unbedingt Annahmen über endliche Zustandsmengen. Wenn sie es doch tun, so sehen diese Zustände völlig anders aus als die der äquivalenten Turingmaschine. Die in Abb. 2.7 vorgestellte Multiplikations-Turingmaschine ist in

keiner Weise eine angemessene Beschreibung dessen, was derzeit auf der Hardwareebene eines handelsüblichen Computers geschieht, wenn er einen Multiplikationsalgorithmus durchläuft.

3 Das Problem der universellen Realisierbarkeit

Searle und Putnam argumentieren unterschiedlich für das Problem der universellen Realisierbarkeit. Die beiden Darstellungen ergänzen sich eigentlich sehr gut. Searles Form ist etwas breiter angelegt, bleibt aber an einigen für die Anschaulichkeit wichtigen Stellen etwas in Andeutungen stecken. Putnams Beweis ist kurz, von formaler Strenge und sehr konkret. Über die Analyse der Ursache des Problems schweigt Putnam beinahe vollständig; Searle dagegen bietet eine Erklärung an. Die Reihenfolge des Referates hier, Searles Version des Problems, 3.1, Putnams Version des Problems, 3.2, Searles Analyse des Problems, 3.3, ergibt sich daraus.

3.1 Searle: Symbolmanipulationen sind universell realisierbar

Wie in Abschnitt 2.1.1 dargestellt wurde, wendet sich Searle mit dem Argument der universellen Realisierbarkeit gegen eine Forschungsrichtung innerhalb der Kognitionswissenschaft, der er den Namen Kognitivismus gibt. Hinter dem Kognitivismus stecke die Annahme, „dass das Hirn ein Computer ist und geistige Prozesse computational sind.“ (Searle 1993: 219) Um die Grundannahme „Das Gehirn ist ein Computer“ zu testen, fasst Searle sie als empirische Hypothese auf und stellt sich, im für ihn charakteristischen Gestus des philosophischen Gedankenexperiments, vor, was man tun würde, um herauszufinden, ob etwas ein Computer ist. Das ist für ihn gleichbedeutend mit der Frage: Was würde man tun, um herauszufinden, ob etwas eine Computation[8] durchführt? Für diese empiristische Untersuchung vom Schreibtisch aus sucht Searle zuallererst, so wie ein Lehrbuch des Empirismus es empfehlen würde, nach einer brauchbaren Definition des Computers. Doch wiederum, wie schon bei den Grundannahmen, kann er in der kognitionswissenschaftlichen Literatur keine Übereinstimmung in den fundamentalen Fragen finden und entschließt sich, zu der ursprünglichen Definition zurückzukehren, zu der von Alan Turing:

> According to Turing, a Turing machine can carry out certain elementary operations: It can rewrite a 0 on its tape as a 1, it can rewrite a 1 on its tape as a 0, it can shift the tape 1 square to the left, or it can shift the tape 1 square to the right. It is controlled by a program of instructions and each instruction specifies a condition and an action to be carried out if the condition is satisfied. (Searle 1992: 205)

Das ist alles, was Searle zur Turingmaschine sagt. Das Zitat ist vollständig, und es gibt in seinem Text keine weitere Passage dazu. Bei dem argumentativen Gewicht, das er im

Folgenden auf diese Definition stützt, dürfte ihm eine bloße ungenaue Beschreibung dessen, was die Turingmaschine tut, nicht genügen. Wenn man das mit dem vergleicht, was in Abschnitt 2.2 zur Turingmaschine gesagt wurde, ist es nicht schwierig zu sehen, dass damit eine grob verkürzte, wenn nicht gar eine falsche Version der Turingmaschine und dessen, was sie tut, vorgestellt wurde. Searle unterschlägt oder verkürzt darin mindestens vier essenziell wichtige Punkte. Erstens erwähnt er mit keinem Wort, dass die Turingmaschine ein unendliches Band haben muss. Zweitens verschweigt er, dass es sich bei der Turingmaschine um einen Endlichen Automaten handelt, d. h. dass es bei der computationalen Beschreibung mit Hilfe einer Turingmaschine um eine Beschreibung mit Hilfe einer endlichen Menge von Zuständen geht. Drittens legt er den In- und Output der Turingmaschine sofort auf Einsen und Nullen fest, obwohl nach Turing jedes beliebige Zeichen auf dem Band stehen kann, sofern es bei der Definition des Eingabealphabets berücksichtigt wurde. Und viertens tut Searle so, als würde die Turingmaschine schon einen Mechanismus beinhalten, der das Programm selbsteigen umsetzt; als wäre das Symbollesen und Symbolschreiben des Turing-Scanners eine durch die Definition geklärte Angelegenheit. Dabei sind gerade dies die wesentlichen Punkte, die deutlich machen, dass die Turingmaschine eine abstrakte Maschine ist mit Komponenten, die sich nicht bauen lassen und deren Mechanismus für das, was das Modell leisten soll, keine Rolle spielt.

All diese Verkürzungen suggerieren, dass mit dem Wort „Turingmaschine“ dasselbe gemeint sei wie mit dem Wort „Computer“[9]. Mit dem, was man heute im Allgemeinen über derzeitig handelsübliche Computer[10] weiß, und mit ein bisschen Phantasie lässt sich der Computer mit Searles Version der Turingmaschine problemlos eins zu eins in Deckungsgleichheit bringen. Mit dem Band könnte vielleicht die Festplatte gemeint sein, man weiß ja, dass sich irgendwie Einsen und Nullen darauf befinden, und man weiß auch, dass diese Einsen und Nullen irgendwie von einem Lese/Schreibekopf gelesen und überschrieben werden. Bei dem Programm, das Searles Turingmaschine steuert, denkt man vielleicht an Pascal- oder C++ oder Java-Programme, von denen man auch weiß, dass sie irgendwie in Einsen und Nullen übersetzt, in dieser Form auf der Festplatte gespeichert und von dort aus bei Bedarf irgendwie aufgerufen werden usw. Leider ist diese Übertragung aus vielen Gründen absolut unzulässig und führt in große begriffliche Schwierigkeiten. Die Gründe dafür wurden in Abschnitt 2.2 bereits erwähnt. Hier sollen sie, um Searles Argumentation zu folgen, noch einmal in Kürze dargestellt und direkt auf diese Übertragung bezogen werden:

Das Band der Turingmaschine trägt den In- und Output, und der Mensch, der „Benutzer“ der Turingmaschine, muss den Input auf das Band schreiben und den Output vom Band ablesen; die Turingmaschine hat, bildlich gesprochen, eine flache Architektur, alles Notwendige zwischen In- und Output liegt offen; der verborgene Mechanismus ist irrelevant. Die Turingmaschine besitzt kein Analogon zur Festplatte. Die Festplatte ist ein internes Modul des Computers, worauf der Mensch, der Benutzer des Computers, nichts lesen und schreiben muss und normalerweise auch nichts lesen und schreiben kann. Der Computer hat eine hierarchische Architektur, zwischen In- und Output spielt sich einiges Relevantes im Verborgenen ab: Speicher – Datenbus – Prozessor. Der Lese/Schreibekopf des Computers liest und schreibt Einsen und Nullen nur in der Weise und auf der Beschreibungsebene, in der Tonköpfe von Kassettenrekordern Noten lesen und schreiben, Zigarettenautomaten einen Euro erkennen, Strom den geringsten Weg des Widerstands sucht und Sehzellen die Information des Lichtes interpretieren (vgl. Czihak/Langer/Ziegler 1992: 112, „Licht als Informationsträger“). Es handelt sich um etwas, das Searle selbst Als-ob-Intentionalität (vgl. Searle 1993: 176) nennen würde; man könnte das Lesen des Computers von der Festplatte auch eine Als-ob-Interpretation nennen, der ein vollständig homunculusfrei beschreibbarer und gar nicht geheimnisvoller elektro-mechanischer Prozess zugrunde liegt. Die Als-ob-Interpretation ist eine Redeweise, die sich für manche Prozessbeschreibungen in Naturwissenschaft und Technik – wie auch in der Alltagssprache – eingebürgert hat und dort im Normalfall keine Schwierigkeiten macht. Es ist nicht leicht einzusehen, weshalb sie auf einmal bei Computern oder Gehirnen problematisch werden sollte. Der abstrakte Lese- und Schreibevorgang des abstrakten Lese/Schreibekopfes einer Turingmaschine ist nicht näher bestimmt. Der Einfachheit halber wird bei ihm häufig etwas angenommen, das Searle eine bewusste, intrinsische Intentionalität (vgl. Searle 1993: 177) nennen würde. Es würde dem Bild nicht widersprechen, zu sagen, die Turingmaschine könne Einsen und Nullen so lesen, wie ein Mensch Einsen und Nullen liest. Man könnte es intrinsische Interpretation nennen. Da es sich um eine Papiermaschine handelt, die sowieso als ganze auf diese Weise vom Menschen gelesen wird, ist das auch nicht weiter überraschend. In vielen Illustrationen der Turingmaschine sitzt ein Homunculus. In handelsüblichen Computern dagegen sitzt garantiert kein Homunculus. Wenn, dann sitzt ein ausgewachsener Homunculus davor, schaut auf den Bildschirm und tippt auf die Tasten. Aber er interpretiert normalerweise keine Vorgänge im Computer computational; das ist nicht nötig, der innere Mechanismus des Computers funktioniert von ganz allein, ohne interne Interpretation.

Pascal-, C++ oder Java-Programme beinhalten Konzepte, die für Programme der Turing-

maschine nicht vorgesehen sind: WHILE-Schleifen, bedingte Sprungbefehle, Zwischenspeicherung in einem Stack, Rekursivität und vieles mehr. Anders als Programme für Turingmaschinen machen sie keine Aussagen über Zustände. Zwar sind die Konzepte gängiger Programmiersprachen ins Konzept der Turingmaschine übersetzbar – das sagt die Church/Turing-Hypothese – aber diese Übersetzbarkeit liegt auf einer Ebene, die man nicht direkt mit der Festplatte eines Computers in Verbindung bringen sollte.

Das wäre ungefähr so, wie das Wissen, dass man jede natürliche Sprache in jede andere natürliche Sprache übersetzen kann, direkt mit Büchern in Verbindung zu bringen. Wenn man ein chinesisches Buch vor sich hat, sollte man nicht versuchen, einen deutschen Text darin zu finden, indem man auf den Seiten, auf denen chinesische Schriftzeichen stehen, die Struktur lateinischer Schriftzeichen sucht, – und das mit der Begründung, dass dieses Buch ins Deutsche übersetzbar ist. Um ein chinesisches Buch lesen zu können, muss man zunächst einmal chinesische Zeichen identifizieren können, und auf dieser Ebene helfen einem Deutschkenntnisse wenig.

Computerprogramme, die z. B. in C++ geschrieben worden sind – vielleicht ist z. B. das Wordstar-Programm in C++ geschrieben worden –, werden tatsächlich in eine Struktur übersetzt[11] und auf der Festplatte gespeichert. Die Festplatte ist Träger der Steuerungsinformation; – und zwar in der gleichen Art und Weise, wie ein gelochtes Band oder eine Audiokassette Träger von Musik oder Gene die Träger der Erbinformation sind (vgl. Czihak/Langer/Ziegler 1992: 179, „Nucleinsäuren als Träger der Erbinformation“). Kein Mensch muss diese Informationen lesen, damit sie wirksam sind. Es ist, mit Searles Begrifflichkeiten ausgedrückt, eine Als-ob-Information für die Als-ob-Interpretation. Das Programm der Turingmaschine hat keinen definierten Speicherort. Wenn man die Illustration in Abb. 2.4 zum Beispiel nimmt und einen Homunkulus in den Kasten, auf dem „Endlicher Automat“ steht, hineinsetzt, so wäre es mit dieser Illustration vereinbar, dass der Homunkulus einen Zettel hat, auf dem das Programm in Form einer Tabelle wie in Abb. 2.5 oder eines Transitionsgraphen wie in Abb. 2.7 geschrieben steht. Das Programm der Turingmaschine ist so gespeichert wie ein Text, vielleicht ein Kochrezept oder ein Gesetzestext, in einem Buch gespeichert ist. Ein Mensch muss es lesen und verstehen und anwenden, damit es wirksam wird. Es ist eine Information für die intrinsische Interpretation.

Mit diesen Unterscheidungen im Hintergrund lassen sich Searles ontologische Vermischung von Turingmaschine und Computer und die Folgerungen, die er daraus zieht, gut

nachvollziehen: Searle sucht nach einer Definition für Computer und liefert stattdessen eine (verkürzte bis falsche) Definition der Turingmaschine. Allem Anschein nach macht er für die Verbindung von Turingmaschine und Computer die Hilfsannahme, dass alles, was man irgendwie computational, d. h. für ihn mit Hilfe einer Turingmaschine, beschreiben kann, ein Computer ist. Unglücklicherweise verschwimmen in Searles Text an dieser Stelle die Markierungen von Kognitivismus-Referat und Darstellung der eigenen Meinung. Man weiß nicht genau, ob Searle sagen will, dass der Kognitivismus diese falschen Annahmen über die Turingmaschine vertritt und auch notwendig vertreten muss, oder ob Searle selbst das tut. Man sollte vermutlich Ersteres annehmen.

Unmittelbar nach seiner oben zitierten Darstellung der Turingmaschine fährt Searle folgendermaßen fort:

> That is the standard definition of computation, but, taken literally, it is at least a bit misleading. If you open up your home computer, you are most unlikely to find any 0's and 1's or even a tape. But this does not really matter for the definition. To find out if an object is really a digital computer, it turns out that we do not actually have to look for 0's and 1's, etc.; rather we just have to look for something that we could *treat as* or *count as* or that *could be used to* function as a 0's and 1's. (Searle 1992: 206)

Hier macht Searle also seiner Suggestion gemäß den Vorschlag, den Computer, den wir zu Hause haben, aufzuschrauben und nachzusehen, ob er eine Turingmaschine ist. Die Überraschung, dass man keine Einsen und Nullen auf der Festplatte – oder sonst wo darin – findet, irritiert ihn nicht, denn er weiß, dass wir nach etwas suchen müssen, das wir wie eine Eins oder eine Null behandeln können. Searle meint an dieser Stelle aber nicht, dass wir nach etwas suchen müssen, das der Lese/Schreibekopf auf der Festplatte in der Art einer Als-ob-Interpretation als Einsen und Nullen auffasst, sondern er meint, dass wir nach Einsen und Nullen suchen müssen, wie sie auf dem Band einer Turingmaschine stehen könnten und die von uns in der Art einer intrinsischen Interpretation gelesen werden. Er legt also recht deutlich die eins zu eins Übertragung Festplatte - Turingband nahe. Diese Übertragung legt er auch einigen Kognitionswissenschaftlern in den Mund, indem er ihre Argumentation für die multiple Realisierbarkeit zitiert und sie in diese Richtung deutet:

> Furthermore, to make the matter more puzzling, it turns out that this machine could be made out of just about anything. As Johnson-Laird says, "It could be made out of cogs and levers like an old fashioned mechanical calculator; it could be made out of a hydraulic system through which water flows; it could be made out of transistors etched into a silicon

> chip through which electric current flows; it could even be carried out by the brain. Each of these machines uses a different medium to represent binary symbols. The positions of cogs, the presence or absence of water, the level of the voltage and perhaps nerve impulses" (Johnson-Laird 1988, p. 39). Similar remarks are made by most of the people who write on this topic. For example, Ned Block (1990) shows how we can have electrical gates where the 1's and 0's are assigned to voltage levels of 4 volts and 7 volts respectively. So we might think that we should go and look for voltage levels. But Block tells us that 1 is only "conventionally" assigned to a certain voltage level. The situation grows more puzzling when he informs us further that we need not use electricity at all, but we can use an elaborate system of cats and mice and cheese and make our gates in such as way that the cat will strain at the leash and pull open a gate that we can also treat as if it were a 0 or a 1. The point, as Block is anxious to insist, is "the irrelevance of hardware realization to computational description. These gates work in different ways but they are nonetheless computationally equivalent" (p. 260). In the same vein, Pylyshyn says that a computational sequence could be realized by "a group of pigeons trained to peck as a Turing machine!" (1984, p. 57) (Searle 1992: 206)

Bei genauer Analyse dieser Zitate zeigt sich, dass keiner der obigen Autoren die Einsen und Nullen einer Turingmaschine unbedingt auf eine verborgene Ebene verlegt. Zwar könnte man Pylyshyn so lesen, dass er sagt, man könne eine Turingmaschine aus Tauben bauen, aber diese philosophisch irreführende rhetorische Verkürzung passiert schnell, – die Fachbücher für Theoretische Informatik sind voll davon. Sie ist an sich harmlos, es wird erst gefährlich, wenn man die falschen Schlüsse daraus zieht. Pylyshyns Aussage kann genau so gut und viel richtiger wiedergegeben werden als: „Man kann Tauben so dressieren, dass ihr Pickverhalten einem Algorithmus entspricht, der zuvor mit Hilfe einer Turingmaschine entworfen wurde.“ Nichts von dem, was er behaupten will, geht dabei verloren.

Um die Taubenkonstruktion Pylyshyns etwas besser zu verstehen, könnte man noch einmal zu Abb. 2.6 zurückkehren und sich vorstellen, dass diese Kästchen in den Sand gemalt sind und überall dort, wo ein Strich ist, ein Korn liegt, und überall dort, wo das Kästchen frei ist, kein Korn liegt. Angenommen, eine Taube verschiebe immer, wenn sie solch einer Situation ausgesetzt ist, pickend die Körner auf genau die Art, wie es in Tabelle 2.5 vorgeschrieben ist. Da es zugegebenermaßen unrealistisch ist, dass eine Taube solche Leistungen vollbringen kann, sei weiterhin angenommen, dass die Taube den Algorithmus nur unvollkommen umsetzt und manchmal, sagen wir bei circa jedem fünften Durchlauf, einen Fehler macht. Es dürfte keine gute Idee sein, eine Repräsentation der Tabelle 2.5

im Kopf der Taube zu suchen, denn die Taube interpretiert die Körner nicht als Einsen und Nullen, sie pickt einfach. Für die computationale Beschreibung des Verhaltens der Taube ist, so wie das Beispiel gemeint ist, das Gehirn der Taube nicht relevant. Die Situation eines Forschers, der das Verhalten der Taube beschreiben möchte, wäre ganz ähnlich wie die des Ethnologen mit dem Mann/Wolf/Ziege/Kohl-Problem aus dem Beispiel 2.2.1.

Searle findet die Tatsache, dass man unter gewissen begrenzten – zudem meist hochgradig konstruierten – Umständen das Verhalten von Tauben, Männern, Wölfen, Ziegen, Katzen, Hunden, Mäusen, Wasser, Holz und vielem anderen mehr mit Hilfe einer Turingmaschine angemessen beschreiben kann, hoch problematisch:

> Computationally speaking, on this view, you can make a "brain" that functions just like yours and mine out of cats and mice and cheese or levers or water pipes or pigeons or anything else provided the two systems are, in Block's sense, "computationally equivalent." You would just need an awful lot of cats, or pigeons or water pipes, or whatever it might be. The proponents of cognitivism report this result with sheer and unconcealed delight. (Searle 1992: 207)

In der Hinsicht der multiplen Realisierbarkeit, schreibt Searle weiter, seien Computer scheinbar mit Thermostaten oder Vergasern vergleichbar. Vergaser könne man auch aus verschiedenen Materialien herstellen. Es gibt welche aus Kupfer, es gibt welche aus Stahl.

> But there is a difference: The classes of carburetors and thermostats are defined in terms of the production of certain *physical* effects. That is why, for example, nobody says you can make carburetors out of pigeons. But the class of computers is defined syntactically in terms of the *assignment* of 0's and 1's. The multiple realizability is a consequence not of the fact that the same physical effect can be achieved in different physical substances, but that the relevant properties are purely syntactical. The physics is irrelevant except in so far as it admits of the assignments of 0's and 1's and of state transitions between them[12]. (Searle 1992: 207)

Anders als Thermostate oder Vergaser sind Computer für Searle rein syntaktisch definiert. Und aus dem Umstand, dass wir keine Einsen und Nullen als Ziffern im Inneren von Computern finden können, sondern diese erst – z. B. der Festplatte – zuschreiben müssen, folgert Searle weiterhin, dass Computation durch die Zuschreibung von Einsen und Nullen

definiert sei. Die physische Ebene müsse eine solche syntaktische Zuschreibung von Einsen und Nullen bloß zulassen, ansonsten sei sie irrelevant. Diese Folgerungen scheinen ihm recht absurde Konsequenzen zu haben: die universelle Realisierbarkeit, die Searle, gleichfalls mit nicht wenig „sheer and unconcealed delight“, in zwei verschiedenen, sich ergänzenden Versionen vorträgt:

1. The same principle that implies multiple realizability would seem to imply universal realizability. If computation is defined in terms of the assignment of syntax, then everything would be a digital computer, because any object whatever could have syntactical ascriptions made to it. You could describe anything in terms of 0's and 1's.
2. Worse yet, syntax is not intrinsic to physics. The ascription of syntactical properties is always relative to an agent or observer who treats certain physical phenomena as syntactical. (Searle 1992: 207f.)

1. For any object there is some description of that object such that under that description the object is a digital computer.
2. For any program and for any sufficiently complex object, there is some description of the object under which it is implementing the program. Thus for example the wall behind my back is right now implementing the Wordstar program, because there is some pattern of molecule movements that is isomorphic with the formal structure of Wordstar. But if the wall is implementing Wordstar, then if it is a big enough wall it is implementing any program, including any program implemented in the brain. (Searle 1992: 208f.)

An dieser Stelle beendet Searle seine empiristische Untersuchung „Was tut man, um herauszufinden, dass etwas ein digitaler Computer ist?“ und stellt fest, dass diese Frage sinnlos ist, weil man bei jedem beliebigen Gegenstand feststellen könne, dass er ein digitaler Computer ist. Mehr noch: Jeder beliebige Gegenstand ist jeder beliebige Computer. Denn jedes denkbare Programm läuft auf jedem beliebigen Gegenstand, der für eine Zuschreibung hinreichend groß ist. Das hat die seltsame Konsequenz, dass jedes beliebige Programm in jedem beliebigen handelsüblichen PC implementiert ist. Jedes ohne Unterschied, auch alle, die im Moment nicht vom Benutzer oder vom Hersteller installiert wurden. Wenn jemand, ein schreibfauler Grafiker vielleicht, nur das Bildbearbeitungsprogramm Photoshop benutzt, und nur dieses Programm als einziges bewusst auf seinem Computer installiert hat, läuft auf seinem Computer dennoch gleichzeitig genau im gleichen Sinne das Textverarbeitungsprogramm Wordstar – und alle anderen denkbaren Pro-

gramme, für die seine Festplatte groß genug ist, um ihr ihre formale Struktur zuzuordnen. Er wird es nicht verhindern können. Er wird noch nicht einmal ein Kriterium angeben können, nach dem er entscheiden könnte, welches Programm auf seinem Computer installiert ist und welches nicht. Alle sind sie installiert. Irgendetwas scheint daran seltsam zu sein. Wie kann man herausfinden, welche Konfiguration ein Computer wirklich im Augenblick besitzt? Searles Analyse gibt keinen Hinweis darauf, wie man diese Seltsamkeit aufklären könnte. Zwar schreibt er:

> I do not think that the problem of universal realizability is a serious one. I think it is possible to block the result of universal realizability by tightening up our definition of computation. Certainly we ought to respect the fact that programmers and engineers regard it as a quirk of Turing's original definitions and not as a real feature of computation. Unpublished works by Brian Smith, Vinod Goel, and John Batali all suggest that a more realistic definition of computation will emphasize such features as the causal relations among program states, programmability and controllability of the mechanism, and situatedness in the real world. (Searle 1992: 209)

Doch scheint ihm dieser Ansatz, den Begriff der Computation so zu verschärfen, wie Smith, Goel und Batali es vorschlagen, keine Lösung für sein Problem und damit auch eigentlich keine Lösung für die universelle Realisierbarkeit und für die oben angesprochene Seltsamkeit zu sein, denn:

> [T]hese further restrictions on the definition of computation are no help in the present *discussion because the really deep problem is that syntax is essentially an observer-relative notion. The multiple realizability of computationally equivalent processes in different physical media is not just a sign that the processes are abstract, but that they are not intrinsic to the system at all. They depend on an interpretation from outside*. (Searle: 1992: 209)

Ein Versuch, das aufzuklären, ohne Turings Definitionen einzuschränken oder anderswie zu verändern, wird hier in Abschnitt 3.3 angeboten, worin Searles Vermutung, dass das tiefe Problem der universellen Realisierbarkeit daher komme, dass die Syntax der Physik nicht intrinsisch ist, genauer betrachtet wird.

Zuvor sei das Problem der universellen Realisierbarkeit aber noch mit Hilfe eines Beweises von Putnam etwas genauer präzisiert. Searles Argumentation bleibt an einigen Stellen etwas abstrakt, an anderen ist sie unklar oder verwirrend. Er spricht oft von der Zuord-

nung von Zeichen. Syntax sei die Zuordnung von Zeichen. Zustände und Zeichen würden der physischen Beschaffenheit nur zugeordnet, sie seien ihr nicht intrinsisch usw. Aber wenn er schreibt, dass man ein Muster von Molekülbewegungen in der Wand hinter ihm finden könne, das der formalen Struktur des Wordstar-Programms isomorph ist, wie meint er das genau? Wie soll man diese Isomorphie in der Wand suchen, ohne dabei wieder zum Physikalisten zu werden? Wie sieht die formale Struktur des Wordstar-Programms überhaupt aus? Welchen Formalismus und welches Suchverfahren schlägt er vor? Searle gibt auf diese Fragen keine Antwort. Er führt der Leserin und dem Leser keine beliebige Suche nach einer beliebigen Computation in einem beliebigen physischen Ding in dieser Welt anschaulich vor. Putnam tut das wohl.

3.2 Putnam: Jedes beliebige physikalische System ist eine Realisierung jedes beliebigen Endlichen Automaten (EA)

Putnams Beweis zur universellen Realisierbarkeit, in dem er in drei Schritten beliebige Endliche Automaten auf beliebige physikalische Systeme bezieht, befindet sich im Appendix seines Buches *Repräsentation und Realität* (Putnam 1999: 213-218) bzw. *Representation and Reality* (Putnam 1998: 121-125). Er ist sehr kurz, aber äußerst vielschichtig, sodass aufgrund des Rahmens und der Richtung dieser Arbeit nicht alle Aspekte des Beweises zitiert und diskutiert werden können. Im ersten Schritt erläutert Putnam die für seinen Beweis relevanten Grundannahmen über die Eigenschaften von physikalischen Systemen, von denen er sagt, dass es anerkannte Grundannahmen der Physik seien, und beweist ein Lemma, einen Hilfssatz. Das Lemma und sein Beweis können hier ausgeklammert werden[13]. Im zweiten Schritt beweist Putnam, dass man einen Endlichen Automaten mit zwei Zuständen und ohne In- und Output mit einem beliebigen physikalischen System so identifizieren kann, dass man darüber sagen kann, das System realisiert diesen EA. Im dritten Schritt, Diskussion, überträgt Putnam das Ergebnis seines Beweises auf beliebige Endliche Automaten mit In- und Output. Das Problem der universellen Realisierbarkeit lautet in Putnams Formulierung:

> *Theorem.* Every ordinary open system is a realization of every abstract finite automaton. (Putnam 1998: 121)

Die Grundannahmen, die Putnam im ersten Schritt zum Beweis seines Theorems einführt, sind das Kontinuitätsprinzip – Principle of Continuity:

> The electromagnetic and gravitational fields are continuous, except possibly at a finite or denumerably infinite set of points. (Putnam 1998: 121)

Und das Prinzip des nichtzyklischen Verhaltens – Principle of Noncyclical Behavior:

> The system *S* is in different maximal states at different times. This principle will hold true of all systems that can "see" (are not shielded from electromagnetic and gravitational signals from) a clock. Since there are natural clocks from which no ordinary open system is shielded, all such systems satisfy this principle. (Putnam 1998: 121)

Es sei darauf hingewiesen, beim Lesen des Folgenden bitte genau darauf zu achten, wie Putnam seinen in- und outputlosen EA immer stärker an die von ihm durch sein Prinzip des nichtzyklischen Verhaltens in die physikalischen Systeme selbst verlegte Uhr bindet, denn um diesen Trick Putnams kümmert sich die weitere Argumentation in diesem Text am meisten. Putnams Beweis des Theorems:

> (I have stated the theorem in terms of finite automata, but the technique is easily adapted to other formalisms.) A finite automaton is characterized by a table which specifies the states and the required state-transitions. Without loss of generality, let us suppose the table calls for the automaton to go through the following sequence of states in the interval (in terms of "machine time") that we wish to simulate in real time: *ABABABA.* Let us suppose we are given a physical system *S* whose spatial boundary we have exactly defined, at least during the real-time interval we are interested in (say, a given 7-minute interval, e.g., from 12:00 to 12:07). We wish to find physical states *A* and *B* such that during the time interval we are interested in the system *S* "obeys" this table by going through the sequence of states *ABABABA,* and such that given just the laws of physics (including the Principle of Continuity) and the boundary conditions of *S,* a Laplacian supermind could predict the next state of the system (e.g., that *S* will be in state *B* from 12:03 to 12:04) given the previous state (given that *S* was in state *A* from 12:02 to 12:03). This will show that *S* "realizes" the given table during the interval specified. Since the technique of proof applies to *any* such table, we will have proved that *S* can be ascribed any machine table at all, and the description will be a "correct" one, in the sense that there really are physical states with respect to which *S* is a realization of the table ascribed. (Putnam 1998: 122) [14]

Man könnte Putnam an dieser Stelle schon vorwerfen, dass Endliche Automaten ohne In- und Output in der Theoretischen Informatik eigentlich nicht definiert sind. Der einfachste EA, der definiert ist, hat wenigstens Input (vgl. Definition 2.2.2). Sein EA nur mit den Zuständen *A* und *B* ist also streng genommen sinnlos (Chalmers erhebt diesen Vorwurf in

etwas schwächerer Form auch. Vgl. Chalmers 1996). Aber vielleicht ist es trotzdem interessant, was Putnam aus seinem eigentlich sinnlosen Beispiel-EA macht:

> I shall use the symbolic expression *St(S, t)* to denote the maximal state of *S* at *t* (in classical physics this would be the value of all the field parameters at all the points inside the boundary of *S* at *t*). Let the beginnings of the intervals during which *S* is to be in one of its stages *A* or *B* be $t_1, t_2, \ldots, t_n$ (in the example given, $n = 7$, and the times in question are t_1 = 12:00, t_2 = 12:01, t_3 = 12:02, t_4 = 12:03, t_5 = 12:04, t_6 = 12:05, t_7 = 12:06). The end of the real-time interval during which we wish *S* to "obey" this table we call t_{n+1} (= t_8 = 12:07, in our example). For each of the intervals t_i to t_{i+1}, $i = 1,2, \ldots, n$, define a (nonmaximal) *interval state* s_i which is the "region" in phase space consisting of all the maximal states *St(S, t)* with $t_i \leq t < t_{i+1}$. (I.e., *S* is in s_i just in case *S* is in one of the maximal states in this "region.") Note that the system *S* is in s_1 from t_n to t_{n+1}, in s_2 from t_2 to $t_3, \ldots,$ in s_n from t_n to t_{n+1}. (Left endpoint included in all cases but not the right – this is a convention to ensure the "machine" is in exactly one of the s_i at a given time.) The disjointness of the states s_i is guaranteed by the Principle of Noncyclical Behavior.
> Define $A = s_1 \vee s_3 \vee s_5 \vee s_7$; $B = s_2 \vee s_4 \vee s_6$.
> Then, as is easily checked, *S* is in state *A* from t_1 to t_2, from t_3 to t_4, from t_5 to t_6, and from t_7 to t_8, and in state *B* at all other times between t_1 and t_8. So *S* "has" the table we specified, with the states *A, B* we just defined as the "realizations" of the states *A, B* described by the table. (Putnam 1998: 122f.)

Das ist, die Betrachtung über kausale Verursachung ausgelassen, Putnams Beweis der universellen Realisierbarkeit. Seine spätere Übertragung auf Endliche Automaten mit In- und Output hat nicht mehr den Charakter eines Beweises, sie gehört zum dritten Schritt – Diskussion. Zunächst zum Beweis, zu dem vorweg zu sagen ist, dass er formal durchaus korrekt ist. Es ist nur die Frage, ob er das beweist, was er beweisen soll:

Putnams Kritiker David Chalmers (Chalmers 1996) und Ronald L. Chrisley (Chrisley 1994) werfen ihm hauptsächlich vor, erstens zu wenige Voraussetzungen von den betrachteten physikalischen Systemen zu verlangen und zweitens zu einfache Endliche Automaten zu betrachten. Chalmers führt, um das Problem der universellen Realisierbarkeit los zu werden, noch eine weitere Bedingung ein, die ein physikalisches System erfüllen muss, damit es für eine bewusstseinsphilosophisch relevante Beschreibung mit Hilfe eines Endlichen Automaten in Frage kommt. Das System solle neben einer Uhr – clock – auch ein Zifferblatt – dial – enthalten, auf dem sich die Zustandswechsel abbilden lassen. Der Endliche Automat, den er zur angemessenen Beschreibung des Bewusstseins für angemessen

hält, nennt er Kombinatorischen Endlichen Automaten – Combinatorial State Automata (CSA). Der CSA ist hinsichtlich seiner formalen Mächtigkeit äquivalent mit dem EA, eine Folge der Church/Turing-Hypothese, aber die Bedingungen für seine Implementation sind weitaus eingeschränkter. Nach einer Reihe äußerst komplizierter formaler Betrachtungen über Systeme mit Uhr und Zifferblatt, die mit Hilfe von CSAs beschrieben werden, kommt er zu dem Schluss, dass er bestimmt immer noch zu viele Systeme in seine Betrachtung eingeschlossen hat, gleichzeitig aber bestimmt auch schon zu viele ausgeschlossen. Und mit diesem unbefriedigenden, zwischen Chauvinismus und Liberalismus schwankenden Ergebnis bricht er seine Untersuchung ab (vgl. Chalmers 1996). Chalmers' Analyse ist insofern auf dem richtigen Weg, als er auf der Suche nach angemesseneren computationalen Beschreibungsarten als denen des klassischen EA ist. Aber mit seinen funktionalistischen Abbildungen mit Hilfe eines CSA, die abhängig sind von einem inneren Takt des Systems, wird er sein für funktionalistische Theorien charakteristisches Schwanken zwischen Chauvinismus und Liberalismus nicht los. Der Endliche Automat scheint auch in seiner kombinatorischen Variante nicht die richtigen Beschreibungsebenen zu treffen, auf denen sinnvolle empirische Arbeit möglich ist. Möglicherweise krankt der Funktionalismus als empirische Wissenschaft an der von Putnam schon 1968 festgestellten Utopie, die aus dem EA-Formalismus resultiert (vgl. Abschnitt 2.1.2).

Hier, für diese Argumentation, interessieren vor allem die Beschreibungsmöglichkeiten, die Putnam in seinem Beweis bereits durch die Bindung an das Prinzip des nichtzyklischen Verhaltens ausschließt. Es scheint, dass Putnam mit der Annahme einer Uhr im System und der Bindung seines EA an den Takt dieser Uhr schon alle Beschreibungsebenen, die seinen Funktionalismus ursprünglich interessierten, von vornherein ausgeschlossen hat. Putnams Funktionalismus – in der in Abschnitt 2.1.2 gegebenen Darstellung von Block – interessierte sich für kausale Relationen zwischen sensorischen Inputs und behavioralen Outputs. Solche In- und Outputs sind nicht unbedingt an den Takt einer internen Uhr gebunden. Man betrachte noch einmal das Mann/Wolf/Ziege/Kohl-Szenario: Es ist für die Beschreibung mit Hilfe des EA gleichgültig, ob der Mann irgendwann einmal eine Pause macht. Die Beschreibung ist nicht an einen internen Taktgeber gebunden. Genau so scheint es, dass ein Schmerzzustand nicht unbedingt von einer inneren Uhr abhängt, man kann eine Tablette nehmen, und er ist früher zu Ende als ohne. Das abstrakte Konzept des EA verlangt keinen Takt, es spielt keine Rolle, ob das Männchen in der Kiste der Turingmaschine ab und zu einmal eine Pause macht. Es scheint, dass Putnam hier eine Art physikalistischen EA-Funktionalismus kritisieren will und nicht mehr seinen ursprünglichen, der dezidiert auf das Sprechen über physische Eigenschaften verzichtet.

Unterstellt man bei seiner Betrachtung eine im physischen System enthaltene Uhr, die z. B. durch seine Molekülbewegungen gegeben ist, beginnt man wieder über physische Eigenschaften zu reden. Gerade das wollte Putnams Funktionalismus ursprünglich vermeiden. Auch bei Searles nicht ausformulierter Suche nach der Isomorphie der Wand mit der formalen Struktur des Wordstar-Programms könnte er wieder versehentlich eine klassische physikalistische Identifikation im Sinne gehabt haben. Aber die klassischen Identitätstheoretiker sind bekanntlich Chauvinisten und interessieren sich deshalb prinzipiell nicht für die Struktur von Wänden.

Zum Schluss noch etwas zu Putnams Diskussion, in der er seinen Beweis auf einen einfachen EA mit In- und Output überträgt:

> Imagine, however, that an object *S* which takes strings of "1"s as inputs and prints such strings as outputs behaves from 12:00 to 12:07 exactly as *if* it had a certain description *D*. That is, *S* receives a certain string, say "111111", at 12:00 and prints a certain string, say "11", at 12:07, and there "exists" (mathematically speaking) a machine with description *D* which does this (by being in the appropriate state at each of the specified intervals, say 12:00 to 12:01, 12:01 to 12:02 …, and printing or erasing what it is supposed to print or erase when it is in a given state and scanning a given symbol). In this case, *S* too can be *interpreted* as being in these same logical states *A,B,C,* ... at the very same times and following the very same transition rules; that is to say, we can find *physical* states *A,B,C, . . .* which *S* possesses at the appropriate times and which stand in the appropriate causal relations to one another and to the inputs and the outputs. The method of proof is exactly the same as in the theorem just proved (the unconstrained case). Thus we obtain that *the assumption that something is a "realization" of a given automaton description (possesses a specified "functional organization") is equivalent to the statement that it behaves as if it had that description.* (Putnam 1998: 124)

Hier begegnet der Leserin und dem Leser wiederum eine Argumentation mit einem Als-ob. Diesmal ist das Als-ob auf der Seite des Beobachters. Der Beobachter tut so, als ob das System eine funktionalistische computationale Beschreibung hätte, und weil es kein Kriterium gibt, das diese Als-ob-Beschreibung von einer objektiven Beschreibung – bzw. einer Beschreibung von intrinsischen Prozessen – unterscheidbar macht, sind diese beiden Beschreibungen äquivalent. An diesem Punkt kann man nicht mehr umhin, Putnam vorzuwerfen, dass er es sich etwas zu einfach macht. Sein EA mit In- und Output ist denkbar simpel, und wie die Beschreibung *D* aussehen soll, das sagt er nicht. Er betont aber ausführlich, dass diese Übertragung für alle Systeme mit funktioniere, auch für solche mit

Mund, Nase, Augen und Ohren als In- und Output-System, die ab und zu Fehler machen.

Leider gibt Putnam, genau so wie Searle, nicht die empiristischen Methoden an, die man benutzen soll, um die Als-ob-Beschreibung von einer objektiven Beschreibung zu unterscheiden. Da beide jedoch dezidiert empirische Theorien kritisieren wollen, wäre dieser Punkt ein interessanter. Es ist zum Beispiel ein Grundsatz der empirischen Forschung, dass man bei der Formulierung einer Hypothese immer die Bedingungen mit angeben muss, unter denen sie falsifiziert ist. Weder Putnam noch Searle machen eine Aussage dazu, wie das bei ihren Argumentationen berücksichtigt werden könnte. Auch gibt es keine Aussagen zu den üblichen Minimalforderungen wie Gebundenheit an eine akzeptierte Theorie und Wiederholbarkeit der Untersuchung. Putnam und Searle gehen beide von einem passiven Beobachter aus, der das zu betrachtende System bei seiner Untersuchung nicht beeinflussen darf. Auch das ist eine relativ überholte Vorstellung von der empirischen Wissenschaft. Die heutige empirische Wissenschaft hat vielmehr einen Benutzer der Dinge zum Ideal, einen Labormenschen, der systematisch testet oder simuliert, wie die Dinge funktionieren.

3.3 Searle: Die Syntax ist der Physik nicht intrinsisch

So wie Searle das Problem der universellen Realisierbarkeit analysiert, liegt der tiefe Grund dafür in falschen Annahmen über das Verhältnis zwischen Syntax und Physik. Die Syntax sei kein intrinsisches[15] Merkmal der Physik, aber Kognitivisten würden sie für ein solches halten.

> The aim of natural science is to discover and characterize features that are intrinsic to the natural world. By its own definitions of computation and cognition, there is no way that computational cognitive science could ever be a natural science, because computation is not an intrinsic feature of the world. It is assigned relative to observers. (Searle 1992: 212)

Intrinsische Merkmale der Physik sind für Searle alle Merkmale, die noch da sind, wenn der Beobachter verschwunden ist. Eine Pflanze, sagt er, macht intrinsischermaßen Photosynthese, ein Herz pumpt intrinsischermaßen Blut, „Masse“, „Schwerkraft“ und „Molekül“ bezeichnen intrinsische Eigenschaften der Welt. Wenn alle menschlichen Beobachter auf einmal verschwinden, gibt es immer noch Masse, Schwerkraft, Moleküle, Photosynthese und Herzen, die Blut pumpen. „Stuhl“, „Badewanne“ und „ein hübscher Tag für ein Pick-

nick“ sind Searles Beispiele für Bezeichnungen nichtintrinsischer Merkmale. Eine Badewanne ist nur relativ zu ihrem Benutzer eine Badewanne; wenn dieser Bezugspunkt verschwindet, jemand, der einen so und so geformten Gegenstand als Badewanne auffasst und als Badewanne benutzt, verschwinden mit ihm auch die Badewannen. Wenn alle Menschen auf einmal aus der Welt verschwinden, dann verschwinden damit auch die Badewannen aus der Welt.

Searles empiristische Frage „Wie findet man heraus, dass etwas ein Computer ist?“ – vollständig zu lesen als „Wie findet man heraus, dass etwas intrinsischermaßen ein Computer ist?“ – führte für ihn deshalb ins Absurde, zur universellen Realisierbarkeit, weil „Computer“, neben „Badewanne“ und „Stuhl“, in die Reihe der Wörter gehört, die keine intrinsischen Merkmale bezeichnen, sondern eine beobachterrelative Zuschreibung vornehmen.

Nun ist diese Frage aber in einer bestimmten Hinsicht den Kognitivisten gegenüber unfair gestellt. Kognitivisten haben, Searles eigener Definition zufolge, die Auffassung „Das Gehirn ist ein Computer“ als eine ihrer Grundannahmen gewählt, und es ist eine gängige Praxis der empirischen Wissenschaften, dass Grundannahmen selbst nicht zum Gegenstand der Forschung werden. Es ist selbstverständlich legitim, die Grundannahmen zu hinterfragen, aber Searle tut in seiner reductio ad absurdum Argumentation so, als sei es das Forschungsprojekt des Kognitivismus selbst, herauszufinden, ob das Hirn ein Computer ist:

> Wir wollten wissen, ob das Hirn nicht in irgendeinem Sinn an sich, intrinsischermaßen, ein digitaler Computer ist – so, wie ein grünes Blatt intrinsischermaßen Photosynthese vollzieht und ein Herz intrinsischermaßen Blut pumpt. (Searle 1993: 230)

Wenn Searle aber das Forschungsprojekt des Kognitivismus ad absurdum führen will, täte er vielleicht besser daran, auch eine Frage ad absurdum zu führen, die sich der Kognitivismus wirklich stellt. Der Kognitivismus stellt sich eher, wenn man Searles *Urgeschichte* (vgl. Abschnitt 2.1.1) folgt, die Fragen: „Vollzieht das Hirn intrinsischermaßen einen Sprachalgorithmus?“ oder „Vollzieht das Hirn unter den und den Bedingungen intrinsischermaßen einen Schmerzalgorithmus?“ Und diese Fragen sind, wiederum nach Searles eigener Darstellung des Kognitivismus, gleichbedeutend mit den Fragen: „Sind Schmerzen computational?“ oder „Sind Sprachproduktion und Sprachverstehen computational?“ Diese Fragen haben eine andere Analogie zu Fragen wie „Vollzieht das grüne

Blatt intrinsischermaßen Photosynthese?“ als die Frage „Ist das Hirn intrinsischermaßen ein digitaler Computer?“ Es sind nämlich ebenfalls Fragen nach bestimmten Prozessen in bestimmten physikalischen Systemen und nach ihrer angemessenen Beschreibung.

Wie in Abschnitt 3.1 gezeigt wurde, folgert Searle die universelle Realisierbarkeit aus der Vorstellung, dass die Turingmaschine und der Computer ein und dasselbe sei. Sie impliziert seiner Analyse zufolge, dass die Syntax der Physik intrinsisch ist: Endliche Automaten sind rein syntaktisch definiert (vgl. Definition und Beispiel 2.2.2: Das ist ein Endlicher Automat in formal lupenreiner Gestalt. Zeichen auf Papier – mehr wird es nicht.) Wenn Computer nichts weiter als Endliche Automaten sind, so Searles Argumentation, dann sind auch sie rein syntaktisch definiert. Da sie aber seltsamerweise auch noch physische Eigenschaften haben, müssten Kognitivisten behaupten, die den Computer konstituierende Syntax sei ihrer physischen Beschaffenheit intrinsisch.

Es könnte vielleicht hilfreich sein, probehalber zu versuchen, diese falsche Vorstellung, der Computer sei eine Turingmaschine, zu korrigieren und zu untersuchen, welche Konsequenzen eine solche Korrektur für die Fragestellungen des Kognitivismus hat:

Hier wurde schon mehrfach betont, dass der Computer kein Endlicher Automat und also auch keine Turingmaschine ist, sondern, dass das Verhältnis zwischen Computer und Turingmaschine – auf einer sehr abstrakten Ebene – ungefähr so ist wie das zwischen Maschine und Schaltplan allgemein. Eine Maschine verhält sich so, wie ihr Schaltplan es vorschreibt. Maschinen sind nicht ihr Schaltplan. Alan Turing, dessen Definition Searle zur Grundlage genommen hat, sagt zum Verhältnis zwischen Computer und Turingmaschine, die von ihm natürlich noch nicht Turingmaschine, sondern discrete state machine – Endlicher Automat – genannt wurde:

> This special property of digital computers, that they can mimic any discrete state machine, is described by saying that they are universal machines. (Turing 1950: 441)

Computer können dazu gebracht werden, sich so zu verhalten, wie eine Turingmaschine es vorschreibt. Sie können Turingmaschinen nachahmen, wie Turing sich ausdrückt. Der Unterschied zwischen einer herkömmlichen Maschine, die ihren Schaltplan nachahmt, und einem Computer scheint nach Turing der zu sein, dass man ein und den selben Computer dazu bringen kann, die Vorschriften jeder beliebigen Turingmaschine zu

befolgen (natürlich unter einer Beschreibung als Als-ob-Intentionalität). Was Turing damit, seinem Bild der discrete state machine folgend, nur gemeint haben kann, ist die Programmierbarkeit des Computers. Er meinte damit also nicht, dass ein Computer jede Turingmaschine gleichzeitig nachahmen kann, sondern immer nur diejenige, zu deren Nachahmung er im Augenblick programmiert wurde. Eine herkömmliche Maschine baut man einmal nach einem bestimmten Schaltplan und diesem entspricht sie dann, herkömmliche Maschinen sind starr. Einen Computer kann man gewissermaßen immer wieder nach immer wieder neuen Schaltplänen umbauen, Computer sind flexibler.

Ansonsten haben Computer keinen ontologischen Sonderstatus unter den Apparaten; – ebenso wie die computationale Beschreibung keinen semiotischen Sonderstatus unter den Beschreibungen hat. Das lässt sich gut an Searles Begriff des Intrinsischen verdeutlichen: Ein mechanisches Klavier zum Beispiel spielt eine Zeit lang selbständig Musik. Wenn man annimmt, dass alle Beobachter auf einmal verschwinden, während ein mechanisches Klavier spielt, was tut es dann, nach diesem plötzlichen Verschwinden, immer noch? Es spielt Searle zufolge keine Musik mehr, weil niemand mehr die Geräusche, die es macht, als Musik auffasst und genießt. Musik ist der Physik nicht intrinsisch. Aber es macht dennoch immer noch die und die Geräusche. Das mechanische Klavier hört, wenn alle Zuhörer auf einmal verschwinden, sofort damit auf Musik zu spielen, aber es hört nicht sofort damit auf Geräusche zu machen[16]. Das mechanische Klavier vollzieht also intrinsischermaßen einen Prozess, der die und die Geräusche hervorbringt, denn dazu braucht es den Beobachter nicht. Genau so wie ein Thermostat intrinsischermaßen die Heizung anschaltet, wenn die Raumtemperatur sinkt. Und genau so wie ein grünes Blatt intrinsischermaßen bei Helligkeit einen Prozess vollzieht, der das und das Gas hervorbringt. Wenn der Betrachter verschwindet, so ist auch hierbei niemand mehr da, der dieses Gas als O_2 auffasst. Die „O_2“-Zuschreibung ist ebenfalls eine Zuschreibung von außen. Aber mit Hilfe der Zuschreibung dieser Zeichen – und noch einiger anderer Zeichen mehr – wird ein Prozess beschrieben, der intrinsisch ist.

Mit dem handelsüblichen Computer lässt sich eine recht deutliche Analogie zu dem mechanischen Klavier herstellen: Angenommen, ein modernes Laptop mit großartigen Aktivboxen und Hochleistungsakkus spiele ein mp3-Klavierstück[17] in Endlosschleife. Bei einem plötzlichen Verschwinden aller Beobachter ist es auch bei diesem Computer so, dass er zwar keine Musik mehr spielt, aber immer noch, so lange bis seine Akkus leer sind, die und die Geräusche macht. Der Prozess, der diese Geräusche hervorbringt, ist dem

Computer also intrinsisch. Menschen können Systemen Prozesse installieren, die diesen Systemen intrinsisch sind.

Dass dieser intrinsische Geräuscherzeugungsprozess namens mp3-Musikabspielen angemessen computational beschreibbar ist, wissen wir, denn wir haben ihn selbst nach einem computationalen Schaltplan gebaut. Dass auch die Photosynthese angemessen computational beschreibbar ist, hat Searle selbst bei seiner Argumentation für die Schwache KI schon zugestanden, indem er bestätigte, dass man sie auf einem Computer simulieren kann (vgl. Abschnitt 2.1.1). Man kann auch relativ leicht einen Endlichen Automaten zur Beschreibung der Photosynthese konstruieren. Ein – nicht sehr subtiler – Photosynthese-EA könnte wie folgt aussehen:

	H	D
Zustand $\boldsymbol{S}_1$	„O_2“ Bleibe in $\boldsymbol{S}_1$	Wechsle zu $\boldsymbol{S}_2$
Zustand $\boldsymbol{S}_2$	Wechsle zu $\boldsymbol{S}_1$	„CO_2“ Bleibe in $\boldsymbol{S}_2$

Abb. 3.1: Transitionstabelle für den Photosynthese-EA

Mit obigem EA ist eine grobe diskrete Darstellung des Tag- und Nachtzyklus der Photosynthese gegeben. Input „H“ ist zu lesen als Helligkeit und „D“ als Dunkelheit. Output „O_2“ bezeichnet den Sauerstoff der Lichtreaktion der Photosynthese und Output „CO_2“ das Kohlendioxid der Dunkelreaktion. In Zustand $\boldsymbol{S}_1$, dem Tageszustand, liefert das mit dem obigen EA beschriebene System so lange „O_2“, wie es den Input „H“ erhält. Erhält es den Input „D“, wechselt es über in den Zustand $\boldsymbol{S}_2$, den Nachtzustand, und liefert „CO_2“. Dieser EA kann für einige Zwecke, in der Schule zum Beispiel, eine hinreichend genaue und brauchbare Beschreibung der Photosynthese sein, für andere Zwecke, im Laborkontext, ist er es wahrscheinlich nicht. (Genaueres zur Photosynthese vgl. Czihak/Langer/Ziegler 1992:112ff.) Es ist also möglich und sinnvoll, intrinsische Prozesse in Systemen computational zu beschreiben, obwohl diese Systeme selbst kein Computer sind.

Die Church/Turing-Hypothese legt den folgenden Begriff der computationalen Beschreibung nahe: Jede Beschreibung mit Hilfe einer Turingmaschine oder mit Hilfe von Konzep-

ten, die denen der Turingmaschine im Sinne der Church/Turing-Hypothese äquivalent sind, ist eine computationale Beschreibung.

Am Beispiel des unsubtilen Photosynthese-EA wird spürbar, dass das Problem einer computationalen Beschreibung, wie das Problem jeder anderen Beschreibung auch, ein Problem der angemessenen Beschreibung ist. Es gibt sehr viele verschiedene Konzepte für die computationale Beschreibung (vgl. Abschnitt 2.2.3). Welcher computationale Formalismus wäre für den und den Zweck der angemessenste? Wäre vielleicht eine nicht-computationale Beschreibung angemessener, um den Sachverhalt für die gewünschten Zwecke bestmöglich wiederzugeben? Jeder Sachverhalt lässt sich auf verschiedene Art und Weise beschreiben, doch wenn sich die Beschreibenden auf bestimmte grundlegende Dinge geeinigt haben – z. B. dass sie zur Bezeichnung eines so und so gearteten Gases die Zeichenkette „O_2“ wählen – sind die Beschreibungsmöglichkeiten nicht mehr beliebig, ähnlich wie bei der Situation des Ethnologen, in Beispiel 2.2.1, der einem anderen Ethnologen seinen EA und was er damit beschreibt, erklären will.

Um es noch einmal auf den Begriff des Computers zugespitzt zusammenzufassen: Turings Unterscheidung zwischen Computer und Turingmaschine so gelesen, wie es oben geschehen ist, ergibt, dass der Computer sich dadurch von herkömmlichen Maschinen unterscheidet, dass er programmierbar ist – im praktisch bisher nie erreichten Idealfall, der Turing vorschwebte, universell programmierbar. Wenn eine herkömmliche Maschine für ihre gesamte Existenz einem einzigen herkömmlichen Schaltplan entspricht, so kann ein Computer im Laufe seiner Existenz vielen verschiedenen einer Turingmaschine äquivalenten Schaltplänen entsprechen; aber zu genau einem Zeitpunkt immer nur einem. Darüber hinaus gibt es keine ontologische Besonderheit des Computers.

Man könnte also mit Turing sagen: Innerhalb der Klasse der Maschinen gibt es konventionelle Maschinen und Computer. Computer unterscheiden sich von den konventionellen Maschinen dadurch, dass sie (universal) programmierbar sind. Die Klasse der Maschinen insgesamt ist ebenfalls nur durch syntaktische Kategorien bestimmbar. Alle Maschinen entsprechen mit Hilfe von Zeichen und Regeln zur Produktion von Zeichenketten erstellten Schaltplänen. Erst wenn man eine konkrete Maschine oder auch eine Klasse von konkreten Maschinen definiert, erhalten sie eine Definition über gewisse physische Effekte. Ein Vergaser soll die physischen Effekte hervorbringen, die hervorzubringen ihm sein Schaltplan vorschreibt. Daher sollte man, um keinen Kategorienfehler zu begehen, wenn

man eine Klasse von konkreten Maschinen mit Computern vergleichen möchte, sie mit einer Klasse von konkreten Computern vergleichen. Tut man dies, stellt sich heraus, dass eine Klasse von konkreten Computern, Computer mit einer konkreten Programmierung zu einem festgelegten Zeitpunkt, ebenfalls durch die Hervorbringung gewisser physischer Effekte definiert ist. Die Klasse der mp3-Player soll diejenigen physischen Effekte hervorbringen, die hervorzubringen ihm der mp3-Algorithmus vorschreibt.

In seinem in Abschnitt 3.1 schon einmal zitierten Einwand:

> The classes of carburetors and thermostats are defined in terms of the production of certain *physical* effects. That is why, for example, nobody says you can make carburetors out of pigeons. But the class of computers is defined syntactically in terms of the *assignment* of 0's and 1's. (Searle 1992: 207)

beachtet Searle diesen Punkt nicht. Ersetzt man im obigen Zitat „die Klasse der Computer“ durch „die Klasse der mp3-Player“, so würde man nicht mehr in der Lage sein, viele vernünftige lebende Menschen zu benennen, die behaupten, man könne einen mp3-Player aus Tauben machen, da der mp3-Player durch die Zuschreibung von Einsen und Nullen definiert sei. Ersetzt man anders herum „die Klasse der Vergaser und Thermostate“ durch „die Klasse der herkömmlichen Maschinen“ so ergibt sich anders herum das gleiche Problem, denn bestimmt kann man Tauben auch so dressieren, dass ihr Verhalten im Abstrakten dem Schaltplan irgendeiner beliebigen konventionellen Maschine entspricht. Dressiert man eine Taube zum Beispiel so, dass sie ein Häuflein Körner vom Boden ihres Käfigs auf eine höhere Etage ihres Käfigs transportiert, könnte man sagen, dass sie damit im Abstrakten der (Entwurfs-) Beschreibung eines Warenaufzugs entspricht. Hier zeigt sich ebenfalls, dass es bei der Beschreibung von Prozessen um ein Problem der angemessenen Beschreibung geht. Auf diese Weise lässt sich auch ein konkreter handelsüblicher Computer, auf dem im Augenblick nur Photoshop installiert ist, von einem konkreten handelsüblichen Computer unterscheiden, auf dem im Augenblick nur das Wordstar-Programm läuft. Der Photoshop-Computer bringt diejenigen physischen Effekte hervor, die der Photoshop-Algorithmus vorschreibt, etwa die Benutzeroberfläche von Photoshop, und der Wordstar-Computer bringt diejenigen physischen Effekte hervor, die der Wordstar-Algorithmus vorschreibt.

Computationale Beschreibungen sind – wie jede andere Beschreibungsart auch – rein syntaktisch, wenn man mit Hilfe dieser Beschreibungsart einen handelsüblichen Com-

puter beschreibt, so beschreibt man, wenn man es richtig macht, damit einen physischen Prozess, der dem Computer intrinsisch ist. Wenn jemand dem Bildschirm des Photoshop-Computers mit Hilfe einer computationalen Beschreibung die Benutzeroberfläche des Wordstar-Programms zuschreibt, dann hat er etwas falsch gemacht. Und dieser Fehler würde sich bei einer unabhängigen empirischen Überprüfung dieser Beschreibung feststellen lassen. Ein anderer würde diese Struktur bei Wiederholung des Vorgangs unter ansonsten gleichen Bedingungen auf diesem Bildschirm nicht noch einmal wiederfinden.

Die Korrektur der Vorstellung, dass die Wörter „Turingmaschine“ und „Computer“ ein und dasselbe bedeuten, hatte also zusammengefasst die folgenden für diese Diskussion relevanten Konsequenzen:

I. Es gibt Prozesse, die einem physikalischen System intrinsisch sind und die für viele Zwecke angemessen computational beschreibbar sind.
II. Nicht jedes physikalische System, das einen intrinsischen Prozess vollzieht, der angemessen computational beschreibbar ist, ist ein Computer.
III. Dass die Syntax der Physik nicht intrinsisch ist, kann zugestanden werden. Aber mit Hilfe von Syntax können Prozesse beschrieben werden, die der Physik intrinsisch sind.
IV. Das Problem der universellen Realisierbarkeit ist nicht mehr in der von Searle intendierten Form formulierbar. Falsche oder unangemessene computationale Beschreibungen sind genau so identifizierbar wie jede andere falsche oder unangemessene Beschreibung auch.

Bei einer Anwendung dieser Ergebnisse auf die Fragestellungen des Kognitivismus lässt sich bemerken, dass sie durch die Korrektur des Verhältnisses zwischen Turingmaschine und Computer keineswegs völlig sinnlos geworden sind. Sie erfahren lediglich eine leichte Bedeutungsverschiebung. Die Frage: „Sind Sprachproduktion und Sprachverstehen computational?“ kann jetzt heißen: „Lassen sich Sprachproduktion und Sprachverstehen angemessen computational beschreiben?“ Und die Grundannahme „Das Gehirn ist ein Computer“ müsste etwas anders verstanden werden, als Searle das tut. Nicht ganz so wörtlich, vielmehr metaphorisch. Searle hat nämlich sehr Recht damit, wenn er sagt, dass der Computer, wie alle Maschinen, Eigenschaften hat, die nicht der Physik intrinsisch sind. Wenn alle menschlichen Beobachter verschwinden, ist niemand mehr da, der Computer als Computer auffasst und benutzt. Aber die Prozesse, auf die es den Kognitionswissenschaftlern mit ihrer Computer – Gehirn Analogie anscheinend eigentlich ankommt,

sind ihm immer noch intrinsisch.

Vielleicht ist es ein Prinzip der Metaphernbildung durch Analogien, dass man mit der Übertragung eines Wortes die intrinsischen Eigenschaften von beobachterrelativen Gegenständen und nicht-beobachterrelativen Gegenständen miteinander vergleicht. Ungefähr so wie bei „Italien ist ein Stiefel.“ oder „Der Gardasee ist eine Badewanne.“ Wenn der Beobachter verschwindet, gibt es keine Stiefel und keine Badewannen mehr, aber die Eigenschaften, die Italien zu einem Stiefel und den Gardasee zu einer Badewanne gemacht hätten, die gibt es nach wie vor. Die Eigenschaft des Gehirns, die eine Computer-Gehirn Analogie für Kognitionswissenschaftler besonders reizvoll machen könnte, ist das Faktum, dass die physischen Effekte, von denen wir zur Zeit annehmen, dass das Gehirn sie verursache, auch in gewissem Sinne und in gewissem Maße veränderlich zu sein scheinen.

Diese Bedeutungsveränderungen rücken den Standpunkt des korrigierten Kognitivismus nah an den Standpunkt der von Searle selbst für vertretbar gehaltenen Schwachen KI, denn die Fragestellungen „Lassen sich Sprachproduktion und Sprachverstehen angemessen computational beschreiben?“ und „Sind Sprachproduktion und Sprachverstehen angemessen auf einem Computer simulierbar?“ kommen sich bedeutungsmäßig sehr nahe, und das Letztere ist eine Frage der Schwachen KI. Der einzige Unterschied zwischen Schwacher KI und Kognitivismus lag, wie in Abschnitt 2.1.1 gezeigt, in einer unterschiedlichen Formulierung der Church/Turing-Hypothese. Searles Kognitivismus formuliert die Church/Turing-Hypothese in den Begriffen der Turingmaschine und interpretiert die Turingmaschine falsch. Searles Schwache KI formuliert die Church/Turing-Hypothese in den Begriffen des Computers und interpretiert den Computer richtig.

Searles Kognitivismus scheint also nichts anderes zu sein als eine Schwache KI mit einem falschen Begriff der Turingmaschine und des Computers. Korrigiert man diese falschen Begriffe, dann verschwindet der Unterschied auch. Es kann an dieser Stelle natürlich nicht ausgeschlossen werden, dass dieser Fehler tatsächlich mitunter in der Literatur bewusst oder unbewusst mit den beschriebenen fatalen Folgen gemacht wird. Aber da Searle selbst in der kognitionswissenschaftlichen Literatur keine Übereinstimmung bei den Definitionen finden konnte, ist es fraglich, warum er ihr ausgerechnet pauschal eine falsche Interpretation Turings klassischer Definition zuschreibt und damit pauschal der gesamten computationalen Kognitionswissenschaft den naturwissenschaftlichen Status aberkennt.

3.4 Fazit: Das Problem der universellen Realisierbarkeit ist kein spezielles Problem der Kognitionswissenschaft

Searle und Putnam haben mit ihrem Vorwurf, dass der Kognitivismus und der Funktionalismus ein unlösbares Problem mit der universellen Realisierbarkeit haben, versucht herauszustellen, dass dieses Problem ein spezielles Problem dieser beiden Ansätze sei. Es lohne sich deswegen nicht, sie wissenschaftlich ernsthaft zu vertreten und ihre Forschungsprojekte weiter zu verfolgen.

Wenn die Analysen dieser Arbeit stimmen, dann bleibt von den Argumentationen von Putnam und Searle dennoch etwas Positives zurück. Beide Argumentationen sind nicht völlig falsch. Bei Searle stimmt die Analyse, dass die Syntax der Physik nicht intrinsisch ist. Aus Putnams Beweis folgt immer noch für die Wissenschaft, dass es, wenn ein Forschungsansatz keine Hypothesen aufstellt, von denen sie die Bedingungen ihrer Falsifikation nennen kann, kein Kriterium liefert, eine Als-ob-Beschreibung von einer wirklichen, objektiven Beschreibung zu unterscheiden. Möglicherweise hat Putnam damit tatsächlich, bewusst oder unbewusst, einige im Sinne des Empirismus nicht ganz exakt wissenschaftliche Ansätze in Schwierigkeiten gebracht – vielleicht sind unter ihnen sogar einige Varianten des Funktionalismus.

Die Version des Problems der universellen Realisierbarkeit, die nach der vorliegenden Diskussion dieser beiden Argumentationen auf jeden Fall bestehen bleibt, ist ein altes universelles Problem der empirischen Wissenschaft. Man hatte immer schon die Schwierigkeit, dass Zeichen und ihre Syntax (als Regeln zur Produktion wohlgeformter Zeichenketten) nicht der Physik intrinsisch sind, sondern damit immer eine Zuschreibung von außen vorgenommen wird. Es passiert vermutlich immer wieder, dass Wissenschaftler Fehler bei der Zuschreibung von Strukturen machen, egal, ob sie eine computationale Beschreibungsmethode wählen oder nicht. Aber oft werden diese Fehler bei einer Überprüfung der Ergebnisse gefunden, und das ist, wenn die empirische Wissenschaft überhaupt funktioniert, prinzipiell möglich.

Ein schönes Beispiel dafür, dass man mit jeder beliebigen Beschreibungsmethode jede beliebige Struktur, die mit dieser Beschreibungsmethode erfassbar ist, den Dingen zuschreiben kann, es aber gleichzeitig auch lange schon Kriterien dafür gibt, ob damit eine intrinsische Struktur beschrieben wurde oder nicht, ist das vor etwas über hundert Jahren

erschienene Buch *Mars* des amerikanischen Astronomen Percival Lowell (Lowell 1985).

Darin beschreibt Lowell in liebevoller kartografischer Kleinarbeit „Kanäle“, die er durch jahrelange Beobachtung durch sein Fernrohr auf der Oberfläche des Mars „gefunden“ hatte. Seine Karten zeigen merkwürdige, beinahe unheimliche Muster (aus: Lowell 1895):

Abb. 3.2: Dem Mars von außen zugeschriebene Strukturen

Die aufwendige Gestaltung seiner Karten und das umfängliche Buch, das er dazu geschrieben hat, lassen ahnen, dass Lowell, der ein angesehener Astronom seiner Zeit gewesen ist, es sehr ernst damit meinte.

Lowell hat daran geglaubt, dass die Strukturen, die er auf dem Mars gesehen hatte, auch wirklich existieren. Er muss es sehr bedauert haben, dass niemand seine Forschungsergebnisse jemals bestätigen konnte. Niemand anderes außer Lowell war in der Lage, ebenfalls diese Muster auf dem Mars zu sehen. Damit waren seine Ergebnisse empirisch zweifelhaft. Mit dem Wissen, dass man jede beliebige Struktur auf ein Bild des Mars zeichnen kann, wurden Lowells Kollegen zu Recht skeptisch.

Erst heute, hundert Jahre später, hat sich der Grund für Lowells Zuschreibungen aufgeklärt[18]. Er hatte sein Fernglas, mit dem er die Planeten beobachtete, umkonstruiert, um sie besser sehen zu können. Er verengte den Durchmesser der Linse von 60 auf 7,5 cm. Wodurch sich ungewollt ein damals nicht bekannter Schatteneffekt ergab. Beim Blick

durch Lowells Fernglas sah man die Strukturen seines Auges, die Regenbogenhaut der Iris, die Blutadern der Retina als schattenhafte Projektion auf dem beobachteten Gegenstand. Es ist klar, dass ein anderer, der durch Lowells Fernglas sah, andere Strukturen auf dem Mars zu sehen bekam, denn er schaute nicht mit Lowells Augen. Mit einem anderen nicht auf diese Weise manipulierten Fernglas war der gesamte Effekt, auf dem Lowells empirischer Fehler beruht, natürlich nicht zu erzielen.

Ein aktuellerer Hinweis dafür, dass es sich bei der universellen Realisierbarkeit als universeller Zuschreibbarkeit von Zeichen um ein altes allgemeines Problem der empirischen Wissenschaft handelt, das sie mit Hilfe von Forschungsmethoden schon seit längerem in den Griff zu bekommen versucht und meist in den Griff bekommt, könnte ein Dialog in PSYCHE-B sein, in dem praktizierende Kognitionswissenschaftler relativ gelassen reagierten, als sie darauf angesprochen wurden.

PSYCHE-B ist ein e-Mail orientiertes Diskussionsforum mit öffentlichem Internetarchiv an der Universität von Houston (USA), in dem sich internationale Kognitionswissenschaftler aus dem Kernbereich und verschiedenen Grenzregionen der Disziplin sowie einige ihrer Gegner über Ergebnisse und Probleme der biologisch/psychologisch orientierten Forschungen über Geist und Gehirn auseinandersetzen. Als es darin im Jahr 2000 kurzzeitig um den Begriff der Computation ging, stellte der Philosoph Michael Schmitz eher nebenbei die folgenden Fragen:

> It seems to me that in Eric Thomson's usage, the notion of computation has already evolved so far that it has lost a very distinct meaning. If any "transformation of a set of inputs to a set of outputs", that is the mediation of a response to a stimulus, is to be called a "computation", what is distinctive about a computational approach to neuroscience? Doesn't this turn any approach to cognitive neuroscience into a computational approach by definition? And why would, given this interpretation of computation, not also carburetors or trees or indeed, most or even all objects be carrying out computations? What are the constraints on the notions of "input", "output" and "transformation" that are supposed to prevent this? (Schmitz 2000)

Schmitz fragt nach Constraints – Einschränkungen – des Begriffs der Computation, die dazu geeignet sein könnten, das Problem der universellen Realisierbarkeit zu verhindern. Der angesprochene Eric Thompson, ein Vertreter der computationalen Neurowissenschaft, antwortet:

> There are none, but this isn't a problem unique to my definition, but a classical problem which traditional computational views are also stuck with. There is a paper by Chalmers (I think) called, "Does a rock implement every computation?" [Gemeint ist Chalmers 1996, TG] which tries to address this problem. I think it's best to ignore the "problem" and focus on specific types of computations that interest us as neuroscientists. As a computational neuroscientist, I think that understanding the computations being performed by brains are especially interesting as they are what ultimately cause behavior (ceteris paribus). I am quite interested in explaining behavior of animals with nervous systems, so I find this particular class of computations fun to study. (Thompson 2000)

Thompson bestreitet zu Recht, dass er Constraints für seinen Begriff der Computation braucht. Das Problem, dass man jedem beliebigen Gegenstand auf der Zeichenebene jede beliebige Computation zuschreiben kann, sei ein allgemeines Problem von computationalen Ansätzen und nicht ein spezielles seines Ansatzes. Man könnte genau so sagen, dass das Problem, dass man jede beliebige Struktur auf jeden beliebigen Gegenstand zeichnen kann, der dafür groß genug ist, kein spezielles Problem von Lowells Ansatz war; jeder Wissenschaftler, der mit Karten arbeitetet, hat es. Thompson insistiert darauf, dass er präzise Vorstellungen davon hat, welche Strukturen er in welchen Systemen mit seinen computationalen Mitteln beschreiben will, und man darf annehmen, dass er auch die Umstände angeben kann, in denen eine von ihm gelieferte Beschreibung falsifiziert ist. Thompson vertraut seinen empirischen Methoden derart, dass er der Meinung ist, dieses allgemeine philosophische Problem ignorieren zu können.

Eine zweite Antwort auf Schmitz' Frage kam von dem Wissenschaftstheoretiker Alfredo Pereira und zielte auf einen Aspekt , der auch hier angesprochen wurde: dass es bei der Wahl der Beschreibungsmethoden um die Angemessenheit geht. (vgl. Abschnitt 3.3) Es scheint einen Pluralismus der Methoden zu geben, es gibt nicht die eine alleingültige Beschreibung, und je nach Zweck ist die eine oder die andere die angemessenere:

> Many informational processes in nature can surely be well described by functionals, however it is possible that other processes are better described as non-functional relations. In this case there would be room in nature for both computationalism and non-computational dynamicism. The problem with non-functional relations is the difficulty to describe such processes mathematically, since so many useful tools are based on the notion of a functional relation. (Pereira 2000)

Computationale Beschreibungen sind für Pereira, in Übereinstimmung mit dem in dieser

Arbeit vorgeschlagenen Begriff, alle Beschreibungen mit Hilfe von Funktionen (und mit Funktionalen = Funktionen von Funktionen). Und Pereira, der anscheinend auch mehr der computationalen Methode zugeneigt ist, räumt ein, dass er sich Forschungsgegenstände, vielleicht sogar innerhalb der Kognitionswissenschaft, vorstellen könnte, die nicht mit computationalen Mitteln angemessen zu bearbeiten sind. Dass man mit seiner Beschreibung, egal welche Methode man gewählt hat, manchmal einer identifizierbaren Projektion aufgesessen sein könnte, ist universelles Problem der empirischen Wissenschaften, und so alt und bekannt, dass man heute offensichtlich wenige praktische Forscher oder Forscherinnen der empirischen Einzelwissenschaften aus der Ruhe bringt, wenn man es im Allgemeinen anspricht.

4 Anmerkungen

[1] Zur Zitierweise: Die beiden Bücher, in denen das Problem der universellen Realisierbarkeit erstmals beschrieben wird, sind im Original auf Englisch erschienen (Putnam 1998 und Searle 1992). Sie liegen aber beide in einer deutschen Übersetzung vor (Putnam 1999 und Searle 1993). Für diese Arbeit wurde entschieden, jeweils aus beiden Versionen dieser Bücher zu zitieren. Die durch die Übersetzungen in die deutsche Debatte gebrachten Fachbegriffe, die zum Teil eigens dafür neu geprägt worden sind (vgl. Abschnitt 3.1), sollen dadurch so gut wie möglich nachvollziehbar in diesen Text übernommen werden. Es werden hierin keine eigenen deutschen Begriffsprägungen benutzt. Weil es bei der Analyse häufig auf die genaue Formulierung ankommt, werden alle zentralen Zitate im englischen Original gebracht und nur etwas weniger wichtige in der deutschen Übersetzung. Dadurch sollen die deutschen Fachbegriffe einigermaßen fließend in diesen Text übergehen, ohne ihn dem Verdacht auszusetzen, bei der Analyse der Argumentationen einer möglichen Ungenauigkeit in den deutschen Versionen aufgesessen zu sein. Zur Kennzeichnung der Quellen wird die amerikanische Zitierweise verwendet. Sämtliche zitierten Texte sind alphabetisch nach Nachname des Autors und Erscheinungsjahr geordnet im Literaturverzeichnis aufgeführt. Hinter einem Zitat steht jeweils in Klammern der entsprechende Nachname, das Datum des Textes und, soweit es sinnvoll ist, nach einem Doppelpunkt die Angabe der Seitenzahl, unter der das Zitat zu finden ist. Neben den vielen Vorteilen, die zu der Entscheidung geführt haben, die amerikanische Zitierweise zu benutzen, hat sie auch einige Nachteile. Besonders den, dass mit dem Datum das Erscheinungsjahr der benutzten Fassung gegeben wird und nicht das der Erstveröffentlichung. Das kann leicht zu so merkwürdigen Effekten führen wie dem, dass soeben behauptet wurde, das Problem der universellen Realisierbarkeit stamme aus den Jahren um 1990 und hinter den Zitaten bei Putnam aber „Putnam 1998“ steht. Das liegt daran, dass aus einer Paperbackausgabe zitiert wird, die erste Hardcoverversion war schon 1988 da.

[2] Dieses Kapitel wurde nahezu wortgleich schon 1990 als Aufsatz unter dem Titel *Is the Brain a Digital Computer?* (Searle 1990) veröffentlicht.

[3] Das Beispiel ist auf Japanisch. Da Searle betont, dass sein Mann im Zimmer – er nimmt sich selbst als Beispiel – nicht in der Lage ist, chinesische Zeichen von koreanischen oder japanischen Zeichen zu unterscheiden, könnte er also in einem Japanisch-Zimmer sein und es irrtümlich für ein Chinesisch-Zimmer halten usw. Auch diesen Irrtum würde er, solange er in dem Zimmer ist, nicht aufklären können.

お元気ですか。

bedeutet „Wie geht es Ihnen?“, und

はい, 元気です。

wäre mit „Danke, mir geht es gut.“ zu übersetzen.

[4] Unmyelinisierte afferente Nervenfasern, die bis zur Haut reichen. Diese sind angeblich für Schmerzreize zuständig. „Schmerz“ wird in der philosophischen Diskussion immer wieder mit dem Feuern von C-Fasern identifiziert. Aus neurophysiologischer Sicht ist das nicht gerechtfertigt (vgl. Urchs 2002: 150). Es gibt verschiedene Sorten von Schmerzfasern, C-Fasern sind eine davon, und nicht jede muss an jedem Schmerz beteiligt sein. Umgekehrt gibt es nicht bei jeder C-Faser-Aktivität gleich eine Schmerzsensation (vgl. Thompson 1994: 186ff.).

[5] Das führte so weit, dass die Gehirnzustandstheorie so gut wie überhaupt nicht mehr vertreten wurde. So etwas ist in der Philosophie ausgesprochen selten. Inzwischen gibt es aber Stimmen, die erklären, dass damit ein Standpunkt vorschnell aufgegeben wurde (vgl. Churchland 1997).

[6] Der Aufsatz *The Nature of Mental States* (Putnam 1975e), aus dem hier hauptsächlich zitiert wird, ist erstmals 1968 unter dem Titel *Psychological Predicates* (Putnam 1968) erschienen.

[7] Die jüngste Aktualisierung des Aufsatzes *Functionalism* (Block 2001) ist nur als HTML-Datei im Internet publiziert; daher die Zitate ohne Angabe von Seitenzahlen.

[8] Die eigens für diese philosophische Debatte ins Deutsche eingeführten Anglizismen „Computation“ und „computational“ sind für den Zusammenhang dieser Arbeit sehr ungünstig. Die ontologische Vermischung von Computation und Computer passiert leicht, wenn sich schon die entsprechenden Wörter lautlich so sehr nahe kommen. Weil hier aber der Entschluss gefasst wurde, keine eigenen Übersetzungen der englischen Texte anzubieten, sondern sich an die Begriffe zu halten, die es in der deutschen Literatur nun einmal gibt, sei eine Erklärung des Übersetzers von Searles *Rediscovery of the Mind* (Searle 1992) für seine Entscheidung, Anglizismen zu prägen, zitiert: „Die englischen Termini ‚computation’ und ‚computational’ werden hier und im folgenden mit ‚Computation’ bzw. ‚computational’ wiedergegeben. Dieser häßliche Neologismus sei kurz gerechtfertigt. Er ist deshalb so schwer vermeidbar, weil dieser angelsächsische Fachterminus in den Diskussionen der Philosophen, Psychologen, Linguisten usw. sehr schillernd verwandt wird. Ursprünglich heißt ‚computation’ natürlich nichts anderes als ‚Rechnung’ oder ‚Berechnung’; schon dazu gibt es übrigens im Deutschen kein gebräuchliches Adjektiv. Nun ist es aber ein sehr besonderer Sinn von Rechnen, um den es da geht, wenn von ‚computational’ die Rede ist, denn um das Addieren, Subtrahieren, Multiplizieren und dergleichen geht es dabei nicht. Vielmehr geht es ganz allgemein um das Operieren mit Symbolen in der Manier eines Computers. Dafür ist aber im Deutschen kein Wort verfügbar. Darüber hinaus haben sich im angelsächsischen Fachjargon

auch noch viele Zusammensetzungen mit dem Adjektiv ‚computational' eingebürgert, für die es im Deutschen keine sprachlich einheitliche Übersetzung gibt; manchmal würde ‚symbolmanipulativ' ganz gut passen, manchmal ‚algorithmisch', manchmal ‚computerwissenschaftlich', manchmal ‚computerartig' und manchmal noch anderes. Tauchte in der deutschen Übersetzung jedes Mal ein anderes Wort auf, wo im amerikanischen Text immer dasselbe Wort steht, dann wäre so mancher Argumentationsschritt, der sich im Original sehr glatt und suggestiv ausnimmt, in der Übersetzung holprig und dubios. Deshalb also der neologistische Notbehelf ‚Computation'/‚computational'." (H. P. Gavagai in Searle 1993: 275) Eigentlich ist nichts daran auszusetzen, wenn holprige und dubiose Argumentationen auch holprig und dubios klingen, aber natürlich hat Gavagai Recht, wenn er versucht, Searles Text so glatt wie möglich ins Deutsche zu bringen. Für den vorliegenden Text aber gilt grundsätzlich, dass, wenn kein Zitat oder Referat vorliegt, mit „Computation" das Operieren mit Symbolen in der Manier eines Endlichen Automaten gemeint ist und niemals, an keiner Stelle, in der Manier eines Computers, denn genau das ist die Verwirrung, die aufgelöst werden soll.

[9] Dieser Eindruck wird noch dadurch außerordentlich verstärkt, dass Searle zwischen den Wörtern „Turing machine" und „digital computer" abwechselt, als seien sie synonym. So schreibt er an einer Stelle: „Well, what made it shivers up and down the spines of a whole generation of young workers in artificial intelligence was the following thought: Suppose the brain is a universal *Turing machine*." (Searle 1992: 202) Und wenige Seiten weiter schreibt er: „But now if we are trying to take seriously the idea that the brain is a *digital computer*, we get the uncomfortable result that we could make a system that does just what the brain does out of pretty everything." (Searle 1992: 207) Interessanterweise übersetzt Gavagai diese Passage so: „Wenn wir jedoch die Idee ernstzunehmen versuchen, das Hirn sei eine *Turingmaschine*, dann gelangen wir zu dem unbequemen Resultat, wir könnten aus so gut wie allem ein System bauen, das genau das tut, was das Hirn tut." (Searle 1993: 228) Wenn man das einen Übersetzungsfehler nennen möchte, dann ist es einer, den Searle mit seinem synonymen Gebrauch dieser Wörter und seiner wirklich sehr suggestiven Schreibweise mit zu verantworten hat. (Kursive Hervorhebungen TG)

[10] In den folgenden Abschnitten ist mit „Computer" ein derzeit handelsüblicher PC gemeint. Etwa ein Pentium IV mit zwanzig GB Festplatte, 256 MB RAM, 17 Zoll Flachbildschirm, Infrarottastatur/ -maus und Windows XP.

[11] Auf ganz ähnliche Weise wie für das Musikprogramm eines mechanischen Klaviers, das auf einem gelochten Band ist, Noten in eine Struktur aus Loch und Nicht-Loch übersetzt wurden. Von diesem Band muss der Mensch keine Noten lesen; er muss die Löcher nicht als Noten interpretieren, damit das Klavier Musik macht. Die ersten Computer der Welt speicherten ihre Programme

auch als Struktur von Loch und Nicht-Loch auf einem Fotofilm. Im Prinzip hat sich daran nicht viel geändert, nur die Materialien sind verschieden.

[12] Nebenbei bemerkt ist es seltsam, wo Searle hier plötzlich die Zustandswechsel hernimmt. In seiner am Anfang des Abschnittes zitierten Definition der Turingmaschine kommen sie nicht vor.

[13] Das Lemma besagt, dass es dem Prinzip der Kontinuität widerspricht, wenn ein gleichgroßes System *S'* innerhalb eines Systems *S* definierbar ist, das immer den Zustand annimmt, den *S* eine Zeiteinheit vorher hatte. Dieses Lemma braucht Putnam in seinem Beweis um zu zeigen, dass Zustand *A* den Übergang zu Zustand *B* tatsächlich kausal verursacht. Nur wenn ausgeschlossen ist, dass das System *S* gleichzeitig als *S'* wieder den Zustand realisiert, den es eigentlich gerade verlassen haben sollte, ist seine Behauptung über kausale Verursachung möglich. Denn andernfalls hätte man nutzlose Aussagen der Form: „*S* geht von Zustand *A* über in Zustand *B* und bleibt in Zustand *A**". In Chrisleys Aufsatz zum Problem der universellen Realisierbarkeit (Chrisley 1994) finden sich einige plausible Gedanken darüber, weshalb Putnams Begriff der Kausalität ohnehin ein zu schwacher ist.

[14] Putnams Abkürzungen sind leider im Vergleich zu den hier sonst benutzten etwas verwirrend. *S* bedeutet bei Putnam System während es sonst immer Zustand bedeutet (State). Für Zustand benutzt Putnam die Abkürzung *St* und die Zustände selbst werden von Putnam mit *A* und *B* bezeichnet. Für den Putnam-Abschnitt, 3.3, geht dieser Text mit Putnams Konventionen mit.

[15] Diese seltsame Terminologie mit „intrinsisch" und „Als-ob", die in dieser Arbeit von Putnam und Searle übernommen wurde, um eine halbwegs immanente Darstellung und Kritik zu erreichen, ist bestimmt nicht die optimalste und wünschenswerteste. Man könnte statt „intrinsisch" auch einfach „objektiv" oder „real" sagen.

[16] Sollte das Wort „Geräusch" noch immer als zu beobachterrelativ erscheinen, wäre das Gemeinte noch deutlicher wiedergegeben mit dem Satz: „Das mechanische Klavier hört, wenn alle Zuhörer auf einmal verschwinden, sofort damit auf Musik zu spielen, aber es hört nicht sofort damit auf, die und die Schwingungen zu erzeugen." Eine Argumentation, worin unterschiedliche Beschreibungen eine ähnlich gelagerte wichtige Rolle spielen, findet sich im Übrigen schon seit längerem in der Handlungstheorie. Zum Beispiel, wenn es um die Beschreibung einer Handlung als absichtlich oder unabsichtlich geht, in G.E.M. Anscobes *Absicht* (Anscobe 1986) und in D. Davidsons *Handlungen, Ursachen, Gründe* (Davidson 1990), der schreibt: „Ich knipse den Schalter an, mache das Licht an und beleuchte das Zimmer. Ohne es zu wissen, alarmiere ich einen Einbrecher, der merkt, daß ich zu Hause bin. Hier brauche ich keine vier Dinge getan zu haben, sondern nur eines, von

dem vier Beschreibungen gegeben worden sind.“ (Davidson 1990: 21) Und nicht unter jeder dieser Beschreibungen ist die Handlung absichtlich.

[17] mp3 ist zurzeit eines der gebräuchlichsten Formate für Audio-, insbes. Musikdateien. *mp3* ist die Abkürzung für *MPEG Audio Layer-3*. *MPEG* steht für *Moving Picture Experts Group*. Diese Organisation entwickelt Standards, um Töne und bewegte Bilder und deren Kombination zu berechnen, zu komprimieren und sie zu codieren bzw. zu decodieren. Layer-3 ist die Bezeichnung für die Audio-Komponente dieser Verschlüsselung.

[18] In der Frankfurter Allgemeinen Zeitung vom 02. 10. 2002 fand sich die folgende Notiz: „Für eine merkwürdige Struktur auf der Oberfläche der Venus, über die der amerikanische Astronom Percival Lowell vor hundert Jahren berichtet hat, scheint jetzt eine Erklärung gefunden worden zu sein. Lowell, der damals viele Jahre damit zubrachte, mit seinem 60-Zentimeter-Teleskop bei Flagstaff/Arizona die angeblichen Kanäle auf dem Mars zu kartieren, hatte auch auf der Venus ein System dünner, dunkler Linien beobachtet. Das Muster scheint seinen Aussagen zufolge den Speichen eines Rades geähnelt zu haben, dessen Nabe immer zur Erde wies. Außer ihm hat es niemand gesehen. Der Grund dürfte in einer „Manipulation“ seines Fernrohrs zu finden sein. Lowell hat die Öffnung des Instruments für seine Beobachtungen wegen der Helligkeit der Venus künstlich von 60 auf 7,5 Zentimeter oder weniger verkleinert, wobei die Brennweite erhalten blieb. Das Verfahren regte Augenspezialisten, die eine entsprechende Darstellung in der Zeitschrift „Sky and Telescope“ lasen, zum Nachdenken an. Dabei fanden sie die wahrscheinlichste Lösung des Rätsels. Die Venus, die Lowell sah, ähnelt einer hellen Strahlungsquelle, die man durch das enge Loch in einer dicht vor dem Auge gehaltenen Platte sieht. Vermutlich hat der Astronom deshalb wie beim Blick durch ein Ophthalmoskop die Schatten der Blutadern und anderer Strukturen seiner Retina wahrgenommen und sie für ein Muster auf der Oberfläche des Planeten gehalten.“ Da sich Lowells Skizzen von Venus, Merkur und Mars sehr auffällig ähneln, liegt es nahe, dass ihm dieser Fehler auch schon bei der Kartografie des Mars unterlaufen ist.

5 Literaturverzeichnis

Anscombe, Gertrude E. M.

(1986) Absicht. Freiburg [u. a.], Alber, 1986

Armstrong, David M.

(1980) The Nature of Mind. Hrsg.: Ned Block: Readings in Philosophy of Psychology. Bd. 1. Cambridge Massachusetts, Harvard UP, 1980, S.191-199

Block, Ned

(1980a) What is Functionalism. Hrsg.: ders.: Readings in Philosophy of Psychology. Bd. 1. Cambridge Massachusetts, Harvard UP, 1980, S.171-184

Block, Ned

(1980b) Troubles With Functionalism. Hrsg.: ders.: Readings in Philosophy of Psychology. Bd. 1. Cambridge Massachusetts, Harvard UP, 1980

Block, Ned

(1990) The Computer Model of the Mind. Hrsg.: D. Osherson und E. Smith: An Invitation to Cognitive Science. Bd. 3. Cambridge Massachusetts, MIT Press, 1990, S.247-289

Block, Ned

(1995) The Mind as the Software of the Brain. Hrsg.: D. Osershon, L. Gleitman, S. Kosslyn, E. Smith und S. Sternberg: An Invitation to Cognitive Science. Bd. 4. Cambridge Massachusetts, MIT Press, 1995, http://www.nyu.edu/gsas/dept/philo/faculty/block/papers/msb.html

Block, Ned

(2001) Functionalism (revised version). World Wide Web, Januar 2002, http://www.nyu.edu/gsas/dept/philo/faculty/block/papers/functionalism.html

Boolos, George S. / Jeffrey, Richard C.

(1989) Computability and Logic. Cambridge [u. a.], Cambridge UP, 1989

Borst, C. V. [Hrsg.]

(1970) Hrsg.: C. V. Borst: The Mind-Body Identity Theory. London [u. a.], Macmillan, 1970

Chalmers, David

(1996) Does a Rock Implement Every Finite-State-Automaton?. Synthese. *108*, 1996, S.391-402, http://www.u.arizona.edu/~chalmers/papers/rock.html

Chrisley, Ronald L.

(1994) Why Everything Doesn't Realize Every Computation. Minds and Machines: Journal for Artificial Intelligence, Philosophy and Cognitive Science. *Vol. 4, No. 4*, 1994

Churchland, Patricia S.

(1997) Can Neurobiology teach us anything about Consciousness?. Hrsg.: N. Block, O. Flanagan und G. Güzeldere: The Nature of Consciousness. Cambridge Massachusetts, MIT Press, 1997

Czihac, G. [Hrsg.]

(1992) Hrsg.: G. Czihac, H. Langer und H. Ziegler: Biologie: Ein Lehrbuch. 5. korr. Aufl., Berlin [u. a.], Springer-Verl., 1992

Davidson, Donald

(1990) Handlungen, Gründe, Ursachen. Donald Davidson: Handlung und Ereignis. Frankfurt a. M., Suhrkamp, 1990, S.19-42

Hauser, Larry S.

(1994) Searle's Chinese Box: The Chinese Room Argument and Artificial Intelligence. World Wide Web, Januar 2002, http://members.aol.com/wutsamada/disserta.html

Hodges, Andrew

(1994) Alan Turing, Enigma. 2. Aufl., Wien [u. a.], Springer-Verl., 1994

Hofstadter, Douglas R.

(1992) Gödel, Escher, Bach: ein Endloses Geflochtenes Band. 2. Aufl., München, Deutscher Taschenbuch Verl., 1992

Hopcroft, John E. / Ullman, Jeffrey D.

(1990) Einführung in die Automatentheorie, formale Sprachen und Komplexitätstheorie. Bonn [u. a.], Addison-Wesley, 1990

Johnson-Laird, P. N.

(1988) The Computer and the Mind. Cambridge Massachusetts, Cambridge UP, 1988

Kim, Jaegwon

(1980) Physicalism and the Multiple Realizability of the Mind. Hrsg.: Ned Block: Readings in Philosophy of Psychology. Bd. 1. Cambridge Massachusetts, Harvard UP, 1980 S.234-236

Lewis, David

(1980) Psychophysical and Theoretical Identifications. Hrsg.: Ned Block: Readings in Philosophy of Psychology. Bd. 1. Cambridge Massachusetts, Harvard UP, 1980, S.207-215

Lowell, Percival

(1895) Mars. World Wide Web, Oktober 2002, http://www.bootlegbooks.com/NonFiction/Lowell/Mars/

Nagel, Thomas

(1993) Physikalismus. Hrsg.: Peter Bieri: Analytische Philosophie des Geistes. Bodenheim, Athenäum [u. a.], 1993, S.56-72

Pereira, Alfredo

(2000) PSYCHE-B Beitrag: Computation and Functions, Thursday 03 Aug 2000 10:24:24 -0300. World Wide Web, Januar 2002, http://listserv.uh.edu/cgi-bin/wa?A2=ind0008&L=psyche-b&F=&S=&P=1631

Putnam, Hilary

(1968) Psychological Predicates. Hrsg.: W. H. Captain, D. D. Merrill: Art, Mind and Religion. Pittsburgh, Pittsburgh UP, 1968, S.37-48

Putnam, Hilary

(1975a) Philosophy and Our Mental Life. Hilary Putnam: Mind, Language and Reality. Bd. 2. Cambridge [u. a.], Cambridge UP, 1975, S.291-303

Putnam, Hilary

(1975b) Brains and Behavior. Hilary Putnam: Mind, Language and Reality. Bd. 2. Cambridge [u. a.], Cambridge UP, 1975, S.325-341

Putnam, Hilary

(1975c) Minds and Machines. Hilary Putnam: Mind, Language and Reality. Bd. 2. Cambridge [u. a.], Cambridge UP, 1975, S.362-385

Putnam, Hilary

(1975d) The Mental Life of Some Machines. Hilary Putnam: Mind, Language and Reality. Bd. 2. Cambridge [u. a.], Cambridge UP, 1975, S.362-385

Putnam, Hilary

(1975e) The Nature of Mental States. Hilary Putnam: Mind, Language and Reality. Bd. 2. Cambride [u. a.], Cambridge UP, 1975, S.429-440

Putnam, Hilary

(1982) Vernunft, Wahrheit und Geschichte. Frankfurt a. M., Suhrkamp, 1982

Putnam, Hilary
(1993) Die Natur mentaler Zustände. Hrsg.: Peter Bieri: Analytische Philosophie des Geistes. Bodenheim, Athenäum [u. a.], 1993, S.123-135

Putnam, Hilary
(1998) Representation and Reality. Cambridge Massachusetts, MIT Press, 1998

Putnam, Hilary
(1999) Repräsentation und Realität. Frankfurt a. M., Suhrkamp, 1999

Pylyshyn, Zenon W.
(1984) Computation and Cognition. Cambridge Massachusetts, MIT Press, 1984

Schmitz, Michael
(2000) PSYCHE-B Beitrag: Re: NEWS: Aleksander: How to Build a Mind, Thursday August 03 2000 11:05 AM. World Wide Web, Januar 2002 http://listserv.uh.edu/cgi-bin/wa?A2=ind0008&L=psyche-b&P=R718

Schöning, Uwe
(1992) Theoretische Informatik kurz gefasst. Mannheim [u. a.], BI-Wiss.-Verl., 1992

Searle, John R.
(1980) Minds, Brains and Programs. Behavioral and Brain Sciences. *3*, 1980, S.417-424

Searle, John R.
(1990) Is the Brain a Digital Computer?. Proceedings and Addresses of the American Philosophical Association. *64 (3)*, 1990, http://cogsci.soton.ac.uk/~harnad/Papers/Py104/searle.comp.html

Searle, John R.
(1992) The Rediscovery of the Mind. Cambridge Massachusetts, MIT Press, 1992

Searle, John R.
(1993) Die Wiederentdeckung des Geistes. München, Artemis u. Winkler, 1993

Thompson, Eric
(2000) PSYCHE-B Beitrag: Re: NEWS: Aleksander: How to Build a Mind, Thursday 03 Aug 2000 21:33:03 -0700. World Wide Web, Januar 2002, http://listserv.uh.edu/cgi-bin/wa?A2=ind0008&L=psyche-b&F=&S=&P=1847

Thompson, Richard F.
(1994) Das Gehirn: Von der Nervenzelle bis zur Verhaltenssteuerung. 2, Heidelberg, Spektrum Akad. Verl., 1994

Turing, Alan M.

(1950) Computing Machinerey and Intelligence. Mind. *59*, 1950, S.433-460

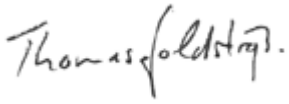
Urchs, Max

(2002) Maschine, Körper, Geist: Eine Einführung in die Kognitionswissenschaft. Frankfurt a. M., Vittorio Klostermann, 2002

Eigenständigkeitserklärung

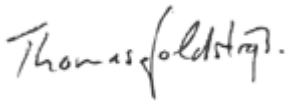
Hiermit erkläre ich an Eides Statt, dass ich die vorliegende Arbeit selbständig ohne fremde Hilfe verfasst und die dafür verwendete Literatur vollständig verzeichnet habe.

Berlin, 07.11.2002



Hinweis: Bei dem vorliegenden Text handelt es sich um eine formal leicht veränderte Version des im November 2002 abgegebenen Originals. Ich habe aus einigen Bandwurmsätzen mehrere Sätze gemacht, einige Kommas ergänzt, andere Kommas gestrichen und einige Tippfehler korrigiert. In seltenen, schlimmsten Fällen habe ich versucht, missglückte Formulierungen zu verbessern, ohne inhaltlich etwas daran zu verändern.

Berlin, 01.09.2003



(Auch: Thomas Goldstrasz)

Diese Arbeit enthält keine sensiblen personenbezogenen Daten.