

# Bemerkungen zu Peter Wegners Ausführungen über Interaktion und Berechenbarkeit

Michael Prasse, Peter Rittgen

## **Zusammenfassung**

Peter Wegner vertritt einen Berechenbarkeitsbegriff, der vom klassischen abweicht, wie er von Church, Kleene, Markov, Post, Turing und anderen etabliert wurde. Insbesondere umfassen Wegners Ideen die Interaktion als grundlegende Eigenschaft heutiger Systeme. In dieser Arbeit erfolgt ein Vergleich der beiden verschiedenen Auffassungen sowie eine Erörterung, ob die Kritik von Wegner am klassischen Berechenbarkeitsbegriff berechtigt ist.

## **Schlüsselwörter**

Interaktionsmaschine, Turingmaschine, Churchsche These, Berechenbarkeit, Algorithmus

## **Summary**

Peter Wegner's definition of computability differs markedly from the classical term as established by Church, Kleene, Markov, Post, Turing et al.. Wegner identifies interaction as the main feature of today's systems lacking in the classical treatment of computability. We compare the different approaches and argue whether or not Wegner's criticism is appropriate.

## **Keywords**

interaction machine, turing machine, Church's Thesis, computability, algorithm

## Einleitung

Wegner betrachtet in seinem Aufsatz [13] die Beziehungen zwischen Berechenbarkeit und Interaktion. Dabei beschreibt er als weiteres Maschinenmodell die Interaktionsmaschine und versucht zu zeigen, daß Interaktion ein so mächtiges Konzept ist, daß die Churchsche These in Frage gestellt wird. Auf der Basis von Interaktion wird anschließend ein auf empirischen Ideen aufbauendes neues Bild der Softwareentwicklung begründet.

In dieser Arbeit erfolgt eine kritische Auseinandersetzung mit den in [13] und in anderen Veröffentlichungen Wegners aufgestellten Behauptungen. Unser Hauptanliegen ist die Untersuchung des Berechnungspotentials von Wegners Interaktionsmaschinen. Seine philosophischen Betrachtungen lassen wir dabei außer acht.

## Wegners Ideen

Peter Wegners Ideen basieren auf der Beobachtung, daß viele Systeme aus Komponenten bestehen, die miteinander kommunizieren. Computersysteme stellen Schnittstellen zur Interaktion mit dem Benutzer bereit. Moderne Softwaresysteme sind aus Modulen gebildet, die Schnittstellen definieren, über die sie mit anderen Modulen in Verbindung stehen. Offene Systeme reagieren auf externe Ereignisse der Systemumgebung. Das objektorientierte Programmierparadigma beschreibt Softwaresysteme als Netze miteinander interagierender Objekte. Diese Fähigkeit zur Interaktion versucht Wegner nun mit den Erkenntnissen aus der Berechenbarkeitstheorie in Einklang zu bringen.

Bezüglich der Frage, was Interaktion leistet, stellt er fest: „Interactive tasks, like driving home from work, cannot be realized through algorithms. Algorithms that execute automatically without taking notice of their surroundings cannot handle traffic and other interactive events.“ Und weiter: „Interaction can simplify tasks when algorithms exist and is the only game in town for inherently interactive tasks, like driving or reserving a seat on an airline“ ([13], S. 82).

Für solche interaktiven Aufgaben führt er Interaktionsmaschinen ein. Diese sind um Ein- und Ausgabeaktionen erweiterte Turingmaschinen, die eine dynamische Interaktion mit der externen Umgebung ermöglichen. Dadurch werden Turingmaschinen zu offenen Systemen.

Anschließend argumentiert er, daß Interaktionsmaschinen eine größere Berechnungsfähigkeit als Turingmaschinen haben. Er unterscheidet verschiedene Modelle von Interaktionsmaschinen bezüglich der Art und Weise der Eingaberealisation und ihrer Funktionalität. Die einfachste Interaktionsmaschine ist die interaktive identische Maschine, die ihre Eingaben unverändert ausgibt und beschreibbar ist durch:

$$„P = in(message).out(message).P^{*1}“ ([13], S. 84)$$

Dann vergleicht er Interaktion mit Nebenläufigkeit und Verteilung, wobei er feststellt, daß Nebenläufigkeit und Verteilung die Mächtigkeit von Turingmaschinen nicht erweitern, die Interaktion hingegen eine größere Berechnungsfähigkeit erlaube.

Das bereitgestellte Verhalten von Interaktionsmaschinen beschreibt er durch Schnittstellen. Schnittstellen bieten die Möglichkeit, die Dienste einer Interaktionsmaschine für die externe Umgebung zur Verfügung zu stellen. Sie sind die Voraussetzung zum komponentenartigen Zusammenbau von Systemen aus interaktiven Teilkomponenten.

Zuletzt fordert er, von einem rationalen Verständnis der Softwareentwicklung zu einer empirischen Sichtweise überzugehen, weil mit Turingmaschinen nicht die reale Welt modelliert werden kann.

## Berechenbarkeitstheoretische Grundlagen

Um Wegners Aussagen über Berechenbarkeit richtig einschätzen zu können, ist es notwendig, zuerst den klassischen Berechenbarkeitsbegriff zu untersuchen. Dazu gehört die Klärung der Begriffe Algorithmus, Turingmaschine und berechenbare Funktion sowie deren Abgrenzung zu nichtalgorithmischen Verfah-

---

<sup>1</sup> Man beachte, daß diese Maschine wegen der rekursiven Definition nie terminiert. Sie spezifiziert eine unendliche Schleife.

ren. Dies soll gleichzeitig verdeutlichen, daß diese Begriffe in der Informatik mit einer festen Bedeutung etabliert sind.

### ***Algorithmusbegriff***

In der Informatik versteht man unter einem Algorithmus ein eindeutig bestimmtes Verfahren, das in endlicher Form beschreibbar ist und schrittweise bestimmte Eingangsdaten in bestimmte Ausgangsdaten umwandelt. Hierbei steht nach jedem Schritt eindeutig fest, welcher Schritt als nächster durchgeführt wird. Der Begriff Algorithmus kann weiter als Berechnungsregel präzisiert werden, die drei Bedingungen genügen muß. Die erste Bedingung ist, daß die Berechnungsregel in endlich vielen Schritten definiert ist. Zweitens gibt es eine eindeutig bestimmte Start- und Endanweisung. Die Wirkung jedes Schrittes ist eindeutig festgelegt, und auch der Nachfolger jeder Anweisung ist bestimmt. Und drittens muß jede Anweisung im gegebenen Zusammenhang effektiv ausführbar sein ([7], S. 7 ff.)<sup>2</sup>.

Betrachtet man den so eingeführten Algorithmusbegriff, so ergibt sich, daß er eng mit dem Begriff Funktion verknüpft ist. Verfahren, die der obigen Festlegung entsprechen, wandeln gewisse Eingangsdaten in eindeutig bestimmte Ausgangsdaten um und definieren somit eine Funktion. Loeckx spricht in diesem Fall von einer algorithmischen Definition der Funktion. In diesem Sinn ist der Algorithmusbegriff eng mit der Berechenbarkeit einer Funktion verbunden und wird als Synonym für die algorithmische Definition einer Funktion verstanden.

### ***Berechenbarkeit und Churchsche These***

Mit Hilfe des Begriffs Algorithmus läßt sich der Begriff berechenbare Funktion definieren. Eine Funktion heißt berechenbar, wenn es einen Algorithmus gibt, der für jedes beliebige Element den Funktionswert berechnet ([7], S. 9)<sup>3</sup>. Ist die Funktion partiell, so soll der Algorithmus an den nicht definierten Stellen nichts ausgeben, indem er z. B. nie anhält und damit kein Ergebnis zurückliefert.

Um die Klasse der berechenbaren Funktionen besser beschreiben zu können, waren Formalisierungen des Algorithmusbegriffs notwendig. Einige von ihnen sind Turingmaschinen, Markov-Algorithmen, Flußbilder, die Theorie der partiell-rekursiven Funktionen und Registermaschinen. Die Klassen von Funktionen, die durch diese unterschiedlichen Formalismen berechnet werden, sind identisch, wie verschiedene Beweise durch gegenseitiges Simulieren der einzelnen Formalismen gezeigt haben<sup>4</sup>.

Die Churchsche These verallgemeinert diese Erkenntnis zu der Aussage, daß jede in irgendeinem intuitiven Sinn algorithmisch berechenbare Funktion bereits partiell-rekursiv ist. Es ist wichtig, darauf hinzuweisen, daß die Churchsche These eine Aussage über die Berechenbarkeit von Funktionen ist<sup>5</sup>. Diese Festlegung wird dadurch bekräftigt, daß die partiell-rekursiven Funktionen als Präzisierung des Algorithmusbegriffs benutzt werden.

Neben dieser „engen“ Sichtweise der Churchschen These gibt es weitere Auslegungen<sup>6</sup>, nach der die intuitiven Begriffe der Berechenbarkeit und des Verfahrens durch Turingmaschinen voll erfaßt werden. Diese Aussagen werden insbesondere in der Philosophie und der künstlichen Intelligenz kontrovers diskutiert. Sie betreffen jedoch die hier geführte Diskussion nicht und sollen daher keine weitere Berücksichtigung finden.

### ***Andere Verfahren algorithmischer Prägung***

Es gibt eine Reihe von Verfahren oder Berechnungsregeln, die einem Algorithmus ähnlich sind, aber einige der aufgezählten Eigenschaften nicht erfüllen. Einige Beispiele hierfür sind Protokolle, Vorschriften, Gebrauchsanleitungen, unendliche Steuerungssequenzen und im weiteren Sinn auch Rollenspiele.

Ein weiteres Beispiel sind nichtdeterministische Algorithmen, wie sie in [7] kurz vorgestellt werden. Bei einem solchen Algorithmus wird nach einem Verfahrensschritt der nächste Schritt aus einer Menge von Folgeschritten beliebig ausgewählt. Ein nichtdeterministischer Algorithmus definiert daher eine zweistel-

<sup>2</sup> Eine ähnliche Definition mit vergleichbaren Forderungen findet man auch in [10].

<sup>3</sup> Vergleiche hierzu auch [6], die eine ähnliche Definition benutzen.

<sup>4</sup> Beweise für einzelne Simulationen findet man z.B. in [4].

<sup>5</sup> Vergleiche hierzu [1] oder ([4], S. 178), die alle den Funktionsbegriff in der Formulierung der Churchschen These benutzen.

<sup>6</sup> Vergleiche hierzu [3] oder [5].

lige Relation<sup>7</sup>, indem er einem Element  $x$  die Menge  $\{y \mid y \text{ kann durch Anwendung des Algorithmus auf } x \text{ erhalten werden}\}$  zuordnet. Aufbauend auf dieser Definition fährt Loeckx fort, daß nichtdeterministische Algorithmen benutzt werden können, um den Begriff berechenbare Relation einzuführen und auf diese Art zu einer Verallgemeinerung der Theorie der berechenbaren Funktionen führen können. Diese nichtdeterministischen Algorithmen beschreiben keine Funktionen und können daher nicht durch Turingmaschinen simuliert werden. Allerdings gibt es eine funktional äquivalente Formulierung, die die charakteristische Funktion  $c$  der Relation akzeptiert.

Die Churchsche These ist von diesen weiteren „Algorithmen“<sup>8</sup> nicht betroffen, da sie keine Funktionen beschreiben. Kann man solche Verfahren jedoch mit äquivalenten berechenbaren Funktionen assoziieren, dann leisten sie bezüglich ihrer funktionalen Mächtigkeit auch nicht mehr als Turingmaschinen. Sie erfüllen einfach einige technische Anforderungen zur Berechnung von Funktionen nicht.

## Diskussion

### *Interaktive Berechnungen und Systeme*

Ein interaktives System ist ein System, das durch ein Ein- und Ausgabeverhalten mit der Systemumgebung in Wechselwirkung steht. Die Systemumgebung bezeichnet im allgemeinen die Komponenten, die nach Identifikation und Abgrenzung des Systems außerhalb von diesem existieren. Ein interaktives System wird auch als offenes System bezeichnet. Offene Systeme sind Systeme, die zur Erfüllung ihrer Aufgabe auf externe, nicht vom System bereitgestellte Informationen angewiesen sind. Integriert man diese externen Ressourcen mit dem System, dann erfolgt keine Wechselwirkung mit der Systemumgebung mehr und man erhält ein neues, nun abgeschlossenes System. Die Unterscheidung in offenes oder geschlossenes System ist daher vom Betrachter abhängig. Sie hat sich aber bei der Beschreibung komponentenartiger Systeme bewährt.

Eine Vielzahl heutiger Systeme sind interaktiver Natur. Diese Systeme sind vielfach Gegenstand der Softwareentwicklung und umfassen betriebswirtschaftliche Informationssysteme, Büroanwendungen, Entwicklungsumgebungen, Abfragesysteme, Dialogsysteme, Netzwerkanwendungen und verteilte Systeme. Wie erwähnt, können auch Objekte als eigenständige Maschinen betrachtet werden, die mit anderen Objekten interagieren. Interaktion ist eine wichtige Charakteristik von objektorientierten Systemen. Es kann zwischen einer internen Sicht, die die Interaktion zwischen den Objekten umfaßt, und einer externen Sicht, die das externe Systemverhalten und die Benutzersicht beinhaltet, unterschieden werden. Da für solche Systeme Computerlösungen existieren, stellt sich die Frage, welche Beziehungen zwischen diesen Systemen und Turingmaschinen bestehen.

### *Turingmaschinen als Maschinen ohne Ein- und Ausgabe*

Turingmaschinen beschreiben nur den Berechnungsprozeß. Wie bei einer Funktion stehen alle Eingabeinformationen am Anfang der Berechnung zur Verfügung. Als Ausgabe wird der Bandinhalt einer Endkonfiguration bezeichnet. Insbesondere verfügen Turingmaschinen über keinerlei Eingabe- und Ausgaberroutinen. Turingmaschinen sind geschlossene Systeme und beschreiben Funktionen.

Computer verfügen im Gegensatz zu Turingmaschinen über Ein- und Ausgabekanäle, sind jedoch ressourcenbeschränkt. Diese Kanäle wurden hinzugefügt, um die Kommunikation mit dem menschlichen Benutzer und der Systemumgebung zu ermöglichen.

Es ist Wegners Verdienst, der Existenz zusätzlicher Ein- und Ausgabeoperationen größere Bedeutung zuzumessen und zu untersuchen, inwieweit dadurch das Verständnis von der Leistungsfähigkeit heutiger Computer beeinflußt wird.

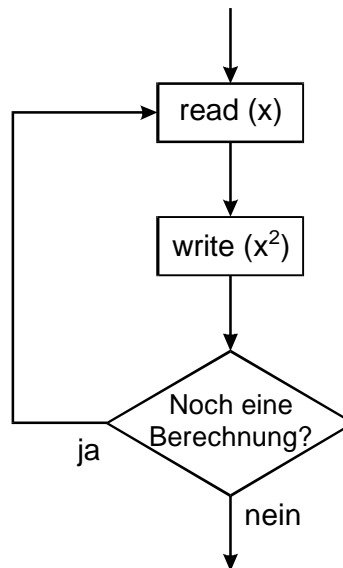
---

<sup>7</sup> Nichtdeterministische Algorithmen können auch Funktionen berechnen, indem verlangt wird, daß alle Berechnungspfade für ein  $x$  dasselbe Ergebnis liefern.

<sup>8</sup> Verwendet man den Begriff Algorithmus nur im Zusammenhang mit Funktionen, dann sollte hier besser der allgemeinere Begriff Verfahren benutzt werden.

### ***Bedeutung von Interaktion***

In Bild 1 ist ein einfaches interaktives Programm zum wiederholten Berechnen des Quadrats verschiedener natürlicher Zahlen  $x$  abgebildet. Dieses Verfahren wird solange ausgeführt, wie der Benutzer noch eine Quadratzahl berechnen möchte.



**Bild 1: Eine einfache Interaktion**

Aus Bild 1 ist sofort ersichtlich, daß Interaktion nicht an bestimmte Programmierparadigmen gebunden ist. Der Ursprung der Interaktion liegt in der Bereitstellung von Kommunikationsmöglichkeiten, die helfen können, den Programmablauf zu steuern. Eine weitere Eigenschaft des beschriebenen Verfahrens ist die Endlichkeit der Darstellung. Der obige Programmablaufplan beschreibt den Vorgang vollständig und eindeutig. Jede einzelne Iteration ohne die Ein- und Ausgabeoperationen kann als Teilberechnung aufgefaßt und durch eine Turingmaschine dargestellt werden. Offen ist dagegen die Frage, welche Gesamtfunktion berechnet wird und ob das Verfahren als Berechnung aufgefaßt werden kann. Für das interaktive Programm allein ist es schwierig, die berechnete Funktion anzugeben, da die Benutzeraktionen unbestimmt sind. Betrachtet man jedoch die Kombination von Benutzer und Programm, dann kann die Gesamtfunktion angegeben werden durch:

$$f: N^* @ N^* \text{ mit } f(x_1 x_2 x_3 \dots x_n) := x_1^2 x_2^2 x_3^2 \dots x_n^2$$

Die Eingabe wird dabei erst zur Laufzeit vollständig festgelegt. Beim Aufstellen der Gesamtfunktion wird der Benutzer automatisch in den Berechnungsprozeß eingegliedert, so daß ein geschlossenes System entsteht.

Es ist verständlich, daß Turingmaschinen wegen der fehlenden Ein- und Ausgabeoperationen diese Verfahren nicht direkt beschreiben können, sondern nur die Gesamtfunktion. Für die Beschreibung heutiger Computersysteme sind Modelle notwendig, die es ermöglichen, externe Eingaben zur Laufzeit zu berücksichtigen.

### ***Wegners Interaktionsmaschinen***

Wegner definiert Interaktionsmaschinen informal als Erweiterung von Turingmaschinen, die zusätzlich Ein- und Ausgabeoperationen besitzen. Diese Erweiterung bewirkt, daß Interaktionsmaschinen offene Systeme sind, die von externen Faktoren abhängig sind. Durch das Auslagern von Aufgaben an externe Partner wird eine Berechnung nur durch das Zusammenspiel mit diesen Partnern ermöglicht.

Allerdings läßt Wegner eine ganze Reihe von Fragen unbeantwortet:

- Wie erfolgt diese Erweiterung?
- Was berechnen bzw. akzeptieren solche Maschinen?
- Wie erfolgt die Ein- und Ausgabe?

- Wie ist die Arbeitsweise definiert?
- Wie erfolgen Interaktion und Kommunikation?
- Was ist die Ausgabe und wie ist sie zu interpretieren?

Vergleicht man im Gegensatz dazu die exakten Festlegungen bei Turingmaschinen, dann ist offensichtlich, daß die Interaktionsmaschinen auf Basis von Wegners Definitionen als weiteres Berechnungsmodell nicht akzeptabel sind. Durch die ungenauen Angaben fehlen weiterhin wichtige formale Grundlagen zur Überprüfung von Wegners Theorien.

### Formale Definitionen von Interaktionsmaschinen

Die Idee Wegners, Turingmaschinen mit Interaktion zu verbinden, ist nicht neu. In ([2], S. 235 ff.) werden interaktive Beweissysteme und Arthur-gegen-Merlin-Spiele<sup>9</sup> behandelt, die aus interaktiven Turingmaschinen aufgebaut sind. Diese Systeme und ihr Akzeptierungsverhalten sind im Gegensatz zu Wegners Maschinen exakt und eindeutig definiert.

Eine interaktive Turingmaschine (ITM) ist eine Turingmaschine mit einem Nur-Lese-Eingabeband, einem Nur-Lese-Zufallsband, einem Arbeitsband, einem Nur-Lese-Kommunikationsband, einem Nur-Schreibe-Kommunikationsband und einem Nur-Schreibe-Ausgabeband. Die eigentlichen Berechnungen erfolgen durch interaktive Protokolle. Ein interaktives Protokoll ist ein geordnetes Paar  $(M_p, M_v)$  von ITMs, die sich ein Eingabeband und die Kommunikationsbänder zugriffvertauscht teilen.



**Bild 2: Interaktives Protokoll**

Die Arbeitsweise eines interaktiven Protokolls ist durch folgendes Schema beschrieben. Die Berechnung erfolgt in Kommunikationsrunden und startet mit  $M_v$ . Das Senden einer Botschaft an eine andere ITM deaktiviert die eigene Berechnung und aktiviert die Berechnung der anderen ITM. Das Akzeptieren erfolgt durch  $M_v$ .

Eine Sprache  $L$  besitzt ein interaktives Beweissystem, wenn eine polynomialzeitbeschränkte interaktive Turingmaschine  $M_v$  (Verifier) existiert und folgendes Akzeptierungsverhalten gilt: Es gibt ein  $M_p$  (Prover), so daß für das interaktive Protokoll  $(M_p, M_v)$  und hinreichend große  $x \in L$  gilt,  $M_v$  akzeptiert  $x$  mit einer Wahrscheinlichkeit  $> 2/3$ . Diese Bedingung sichert, daß wenigstens ein  $M_p$  einen gültigen Beweis vorschlägt. Für jedes mögliche  $M_p$  gilt weiterhin, daß für das zugehörige interaktive Protokoll  $(M_p, M_v)$  und hinreichend große  $x \notin L$  gilt,  $M_v$  akzeptiert  $x$  mit einer Wahrscheinlichkeit  $< 1/3$ . Dies sichert, daß kein  $M_p$  einen gültigen Beweis für ein  $x \notin L$  vorschlägt. Ein interaktives Beweissystem arbeitet nach folgendem Schema. Der Beweiser (Prover) schlägt dem Überprüfer (Verifier) einen Beweis vor, ob ein Wort  $x \in L$  ist. Der Überprüfer nimmt diesen Beweis und testet ihn. Ist er überzeugt von dem Beweis, dann akzeptiert er, andererseits lehnt er ab.

Diese Kurzarstellung zeigt die prinzipielle Möglichkeit, interaktive Maschinen durch formale und exakte Definitionen zu spezifizieren. Aufgrund der Komplexitätsangaben sowie des probabilistischen Grundmodells sind diese Maschinen mit denen von Wegner jedoch nicht vergleichbar. Die Grundlage

<sup>9</sup> Interaktive Beweissysteme und Arthur-gegen-Merlin-Spiele sind Modelle zur Beschreibung spezieller Komplexitätsklassen.

der in dieser Arbeit betrachteten interaktiven Turingmaschinen bildet die Definition aus [2], wobei allerdings keine probabilistischen Turingmaschinen als Ausgangspunkt genommen werden. Für die folgende Argumentation ist es ausreichend, daß eine interaktive Turingmaschine zusätzliche Arbeitsbänder und Zustände besitzt, die nur für den Informationsaustausch benutzt werden und keine zusätzliche Berechnungsfähigkeit bieten. Es ist nicht Ziel dieser Arbeit, die fehlende Formalisierung von Wegners Interaktionsmaschine nachzuholen.

Die Problematik bei der Definition liegt weniger in der Angabe der Maschinen, als in der Angabe eines Protokolls, daß das Akzeptierungsverhalten und das Zusammenarbeiten einer Gruppe von Interaktionsmaschinen mit eventuellen weiteren externen Informationen festlegt. Je nachdem wie diese Protokolle definiert werden, können mit Interaktionsmaschinen verschiedene Klassen von Verfahren abgedeckt werden. Dies bedeutet, daß unabhängig von der Interaktionsmaschine das verwendete Protokoll festlegt, ob es sich um eine weitere Konkretisierung des Algorithmusbegriffs handelt. Um Interaktionsmaschinen mit Turingmaschinen vergleichen zu können, muß das gewählte Protokoll Funktionen berechnen oder Mengen akzeptieren.

Neben diesen Modellen gibt es eine Reihe weiterer formaler Modelle, die Interaktion bzw. Kommunikation benutzen, aber nicht auf Turingmaschinen basieren. Dazu zählen Kommunikationsprotokolle, Prozeßkalküle oder neuronale Netze. Diese Modelle sollen jedoch nicht weiter betrachtet werden, da Wegner Turingmaschinen als Grundlage für seine Interaktionsmaschinen wählt.

### ***Interaktion und Churchsche These***

Nachdem sowohl die Grundlagen der Berechenbarkeit als auch die Idee der Interaktionsmaschine von Peter Wegner vorgestellt sind, kann der Zusammenhang zwischen Churchscher These und Interaktion untersucht werden.

Peter Wegner schreibt: „The hypothesis that the formal notation of computability by Turing machines corresponds to the intuitive notation of what is computable has been accepted as obviously true for 50 years. However, when the intuitive notion of what is computable is broadened to include interactive computations, Church’s thesis breaks down. Though the thesis is valid in the narrow sense that Turing Machines express the behavior of algorithms, the broader assertion that algorithms capture the intuitive notation of what computers compute is invalid“ ([13], S. 83).

Wegner argumentiert, daß Interaktion eine Eigenschaft ist, die den derzeitigen Berechenbarkeitsbegriff in Frage stellt. Maschinen, die über Ein- und Ausgabe (Interaktion) verfügen, sind seiner Ansicht nach mächtiger als Turingmaschinen. In diesem Zusammenhang stellt sich die Frage, welche zusätzlichen Arten von Problemen durch Interaktionsmaschinen gelöst werden können und ob es zum Beispiel Interaktionsmaschinen gibt, die das Halteproblem berechnen können. Die Untersuchung der Leistungsfähigkeit von Interaktionsmaschinen kann daher in zwei Schritten erfolgen. Zuerst kann untersucht werden, ob Verfahren, die keine Funktionen berechnen, mittels Interaktionsmaschinen beschrieben werden können. Bei diesen Verfahren, wie z.B. „Nach Hause fahren“ [13], muß jedoch zuerst geklärt werden, was berechenbar in diesem Zusammenhang bedeutet. Peter Wegner spricht auch von nichtalgorithmischen Berechnungen, ohne jedoch zu klären, was darunter zu verstehen ist. Die zweite und wesentlich interessantere Frage ist, ob es Funktionen gibt, die durch Interaktionsmaschinen, aber nicht durch Turingmaschinen berechnet werden können. Nur wenn diese Frage bejaht werden kann, kann man zustimmen, daß ein Berechnungsmodell gefunden wurde, welches mächtiger als Turingmaschinen ist. In diesem Fall wäre die Churchsche These widerlegt. Interessanterweise bietet Wegner in seinen Veröffentlichungen keine Beispiele für den zweiten Fall an.

Interaktionsmaschinen werden als einfache Erweiterungen von Turingmaschinen definiert. Die Ein- und Ausgabemechanismen dienen dem Informationsaustausch und bieten keine eigene Berechnungsfähigkeit. Eine Interaktionsmaschine kann somit intern nicht mehr leisten als eine äquivalente Turingmaschine. Sie kann aber die Berechnungsfähigkeit ihrer Interaktionspartner nutzen. Das Hinzufügen von Ein- und Ausgabemechanismen führt zu einem relativen Maschinenmodell, vergleichbar mit Orakelmaschinen, da Interaktionen als Unterprogrammaufruf oder als Anfrage interpretiert werden können. Mit folgendem Beweis versucht Wegner zu zeigen, daß Interaktionsmaschinen ausdrucksstärker als Turingmaschinen sind: „Interaction machines that passively interact with input sequences generated by oracles or processes in nature can have richer behavior in a more direct sense since their input may not have a recur-

sively enumerable specification“ ([11], S. 28). Dieser „Beweis“ basiert auf einem Vergleich mit Orakeln. Die größere Berechnungsfähigkeit ergibt sich dabei nicht aus den Ein- und Ausgabeoperationen, sondern aus der Berechnungsfähigkeit des externen Interaktionspartners. Aber auch Orakelmaschinen können nur dann nicht-berechenbare Probleme lösen, wenn das Orakel unentscheidbar ist. Ist das Orakel dagegen entscheidbar, kann jede Orakelanfrage durch einen Unterprogrammaufruf einer Turingmaschine gelöst werden, die das Orakel entscheidet. Damit existiert für das Gesamtproblem aber eine Turingmaschine und dieses ist berechenbar.

In [9] wird Interaktion als Möglichkeit verstanden, das Berechnungsverhalten von Turingmaschinen zu verbessern, indem eine Kommunikation mit der externen Umgebung zugelassen wird. Schönig verweist darauf, daß Orakelmaschinen in diesem Zusammenhang als ein Berechnungsmodell mit Interaktion aufgefaßt werden können. Orakelmaschinen berechnen Probleme relativ zu einem Orakel. Dabei „interagiert“ die Orakelmaschine durch Anfragen mit dem Orakel. Erst unter Festlegung eines bestimmten Orakels kann eine Orakelmaschine eine Funktion berechnen. Eine Orakelmaschine ist damit vergleichbar mit einer Schablone, aus der mit Hilfe eines bestimmten Orakels eine konkrete Orakelmaschine für ein spezielles Problem instanziiert wird. Nach Schönig kann eine Orakelmaschine  $M^I$  als einstellige Funktion (Operator) aufgefaßt werden.  $M^{BI}$  ordnet dann dem Orakel  $B$  die Menge  $A=L(M^{BI})$  zu.

Diese Sichtweise kann auch auf Interaktionsmaschinen übertragen werden. Eine wesentliche Charakteristik der Interaktionsmaschinen von Wegner ist die Unbestimmtheit der Ausgabe, weil zusätzliche externe Informationen für eine Starteingabe immer andere Ausgaben generieren können. Im Gegensatz zu Orakelmaschinen, wo das Orakel für eine Berechnung angegeben werden muß, läßt Wegner die Festlegung der Interaktionspartner offen. Damit ist eine Interaktionsmaschine aber eher eine Schablone als eine konkrete Maschine zum Lösen eines Problems. Erst durch Festlegen der Interaktionspartner und des Interaktionsverhaltens ergibt sich eine konkrete Maschine für ein konkretes Problem.

Interaktionsmaschinen führen damit zu einem relativen Berechnungsmodell. Berechnungen von Interaktionsmaschinen müssen relativ zu den beteiligten Interaktionspartnern betrachtet werden. Wenn man davon ausgeht, daß die Interaktionspartner einer Interaktionsmaschine nur berechenbare Probleme behandeln können, dann kann die resultierende Maschine auch nur berechenbare Probleme lösen!

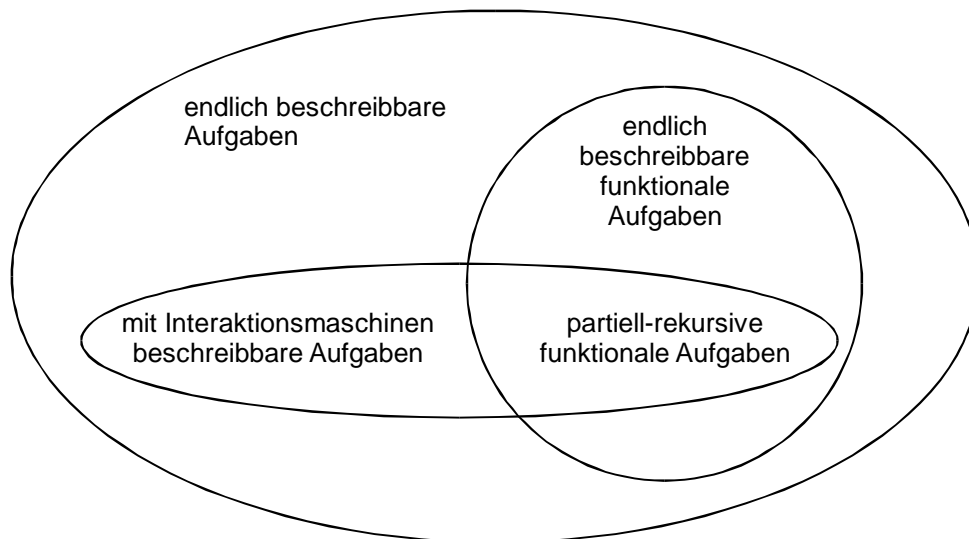
Interaktion kann aber auch als eine weitere Kompositionsform von Verfahren interpretiert werden. Diese Kompositionsform ist wie die Hintereinanderausführung von Turingmaschinen bzgl. der partiell-rekursiven Funktionen abgeschlossen, weil jede Interaktion durch ein Unterprogramm simuliert werden kann. Interaktionsmaschinen sind nur Komponenten für interaktive Systeme (Protokolle). Diese Protokolle müßten an Stelle von Interaktionsmaschinen als Berechnungsmodell betrachtet werden, wobei der Interaktionsprozeß und das Akzeptierungsverhalten exakt definiert werden müssen. Eine Interaktionsmaschine kann im allgemeinen wegen ihrer unvollständigen Spezifikation und der Abhängigkeit von anderen Interaktionspartnern allein keine Funktion berechnen<sup>10</sup>. Sie ist keine Konkretisierung des Begriffs Algorithmus im Sinn der Berechenbarkeitstheorie, sondern eine Komponente, aus denen interaktive Systeme gebildet werden! Interaktion und Kommunikation setzen immer mehrere Partner voraus! Nur im Zusammenspiel mit anderen Partnern ermöglichen Interaktionsmaschinen die Berechnung von Funktionen. Werden Personen oder externe Prozesse in die Interaktion einbezogen, dann ist sofort ersichtlich, daß, falls sich diese nicht partiell-rekursiv verhalten, nicht garantiert werden kann, daß das Gesamtverfahren berechenbar ist. Es ist bekannt, daß ein System, bei dem eine Komponente nicht berechenbar ist, selbst die Stufe der Berechenbarkeit überschreiten kann. Dies ist aber keine spezielle Eigenschaft von Interaktion, sondern jedes Berechenbarkeitsmodell, das Komposition unterstützt, verfügt über diese Möglichkeit.

Somit sind Modelle, die auf Interaktionsmaschinen basieren und in denen alle Interaktionspartner partiell-rekursiv sind, nicht in der Lage, nicht-berechenbare Funktionen zu berechnen. Sie verfügen über die gleiche funktionale Mächtigkeit wie Turingmaschinen. Interaktion führt damit nicht zu einer größeren Berechenbarkeit von Funktionen. In Bild 3 ist die Beziehung zwischen den einzelnen Problemklassen dargestellt. Dabei kann es sich bei den Verfahren, die sich durch Interaktionsmaschinen darstellen lassen und denen keine Funktion zugeordnet werden kann, um eine potentiell leere Menge handeln.

---

<sup>10</sup> Der Fall, daß eine Interaktionsmaschine die Ein- und Ausgabeoperationen nicht verwendet, sei ausgeschlossen, da es sich dann um eine einfache Turingmaschine handelt.

Auch Wegner selbst hat wohl einige der hier formulierten Entgegnungen und Bedenken erkannt. So lautet eine seiner früheren Behauptungen: „Interaction machines, defined by extending Turing machines with input actions (read statements), are shown to be more expressive than computable functions, providing a counterexample to the hypothesis of Church and Turing that the intuitive notion of computation corresponds to formal computability by Turing machines“ ([11], S. 2). In späteren Veröffentlichungen formuliert er schon schwächer: „Though Turing machines capture the intuitive semantics of algorithms, the broader Church-Turing thesis (that Turing machines capture the intuitive notion of computing) is invalid“ ([12], S. 1).



**Bild 3: Berechnungsfähigkeit von Interaktionsmaschinen**

Es läßt sich die Schlußfolgerung ziehen, daß Wegners Berechenbarkeitsbegriff sich eher an realen Computern und Systemen orientiert als an der theoretischen, funktionsorientierten Sichtweise von Church, Kleene, Markov, Post und Turing.

Zum Abschluß soll noch eine kritische Anmerkung zur Leistungsfähigkeit heutiger Computer erfolgen. Heutige Rechner und Rechnernetze sind ressourcenbeschränkt. Aus diesem Grund versagen sie alle bei der Berechnung von Funktionen mit unendlichem Ressourcenbedarf. Es ist daher problematisch, den Begriff Berechenbarkeit auf der Basis heutiger Computer festzulegen. Deshalb kommen nur theoretischen Modelle für die Definition des Begriffs Berechenbarkeit in Frage. Die Einbeziehung von Ein- und Ausgabe in gedankliche Modelle dagegen ist zu begrüßen.

## Synthese

Peter Wegner hat mit seinen Ideen ein interessantes Problem für die theoretische Informatik, aber auch für die Softwareentwicklung aufgeworfen. Die Frage, wie interaktive Berechnungen sich in die Berechenbarkeitstheorie einordnen, ist faszinierend. Seine Schlußfolgerung, daß damit die Churchsche These widerlegt sei, kann allerdings nicht geteilt werden.

Peter Wegner hat weiterhin eine andere Vorstellung von dem Begriff Berechenbarkeit, die über Funktionen hinausgeht und auch andere Verfahrensklassen einschließt. Interaktionsmaschinen sind aber selbst auch nicht befähigt, nicht-rekursive Funktionen zu berechnen. Sie bieten für Funktionen keine größere Berechnungsfähigkeit, und damit ist die Churchsche These weiterhin gültig.

Peter Wegners Berechenbarkeitsbegriff läßt sich eher als „Systemadäquanz“ charakterisieren, die Interaktion und die unendliche Ausführbarkeit von Steuerschleifen beinhaltet. Systemadäquanz bezieht sich dabei auf die Leistungsfähigkeit von interaktiven Systemen und charakterisiert das Maß in dem ein Modell für die Darstellung von (interaktiven) Systemen geeignet ist. Nach Wegners Auffassung erlauben seine Interaktionsmaschinen eine adäquate Beschreibung solcher Systeme. Dies berücksichtigt auch die

Tatsache, daß ein System im allgemeinen nicht als Funktion (Algorithmus) ausgedrückt werden kann. Betrachtet man Peter Wegners Untersuchungen unter diesem Blickwinkel, so bieten sie einen wichtigen Ansatz zur Beschreibung moderner Softwaresysteme mit interaktiven graphischen Benutzungsoberflächen, der durch Turingmaschinen nicht geleistet wird. Insbesondere die Betonung von Schnittstellen und die Betrachtung von Interaktionsmaschinen als Komponenten erlauben eine gute Beschreibung objektorientierter modularer Systeme. Eine Interaktionsmaschine ohne Kommunikationspartner entspricht einer Schablone mit fester Grundfunktionalität, die aber nicht immer befähigt ist, allein Funktionen zu berechnen. Erst die Kombination verschiedener solcher Maschinen ermöglicht die Berechnung. Diese Idee kann gut beim objektorientierten Paradigma angewendet werden, wo erst Objektnetze die Systemfunktionalität gewährleisten. Im Zusammenhang mit anderen Formalisierungstechniken in der Objektorientierung<sup>11</sup> bietet sich somit eine interessante Möglichkeit für formale Beschreibungen objektorientierter Systeme.

Ein Kritikpunkt ist jedoch die fehlende formale und exakte Darstellung der Interaktionsmaschinen. Damit fehlt eine wichtige Grundlage zur Überprüfung seiner Theorien. Auch sind viele der aufgestellten Behauptungen und Beweise nicht exakt, schwer nachvollziehbar oder nur Plausibilitätserklärungen.

## Literatur

- [1] Adyan, S. I.: Church Thesis. In: Hazewinkel, M.: Encyclopaedia of Mathematics. Kluwer Academic Publishers. Dordrecht. 1987. ISBN 1-55608-010-7.
- [2] Balcazar, J. L.; Diaz, J.; Gabarro, J.: Structural Complexity II. Springer Verlag. 1990. ISBN 3-540-52079-1.
- [3] Copeland, J.: The Church-Turing Thesis. In: Online Stanford Encyclopedia of Philosophy. 1996.  
<http://plato.stanford.edu/>
- [4] Hopcroft, J. E.; Ullman, J. D.: Einführung in die Automatentheorie, formale Sprachen und Komplexitätstheorie. Addison Wesley. 1990. ISBN 3-89319-181-x.
- [5] Kearns, J. T.: Thinking Machines: Some Fundamental Confusions. Minds and Machines. Nr. 7. 1997. S. 269-287.
- [6] Lavrov, I. A.; Taimanov, A. D.: Computable Function. In: Hazewinkel, M.: Encyclopaedia of Mathematics. Kluwer Academic Publishers, Dordrecht. 1987. ISBN 1-55608-010-7.
- [7] Loeckx, J.: Algorithmentheorie. Springer Verlag. 1976.
- [8] Meyer, B.: Object-Oriented Software Construction. 2. Edition. Prentice Hall. 1997.
- [9] Schöning, U.: Complexity Theory and Interaction. In: Herken, R.: The Universal Turing Machine - A Half-Century Survey. Springer. Wien, New York. 1988. ISBN 3-211-82637-8.
- [10] Uspenskii, V. A.: Algorithm. In: Hazewinkel, M.: Encyclopaedia of Mathematics. Kluwer Academic Publishers. Dordrecht. 1987. ISBN 1-55608-010-7.
- [11] Wegner, P.: Tutorial Notes: Models and Paradigms of Interaction. OOPSLA 1995.  
<http://www.cs.brown.edu/people/pw/>
- [12] Wegner, P.: Interactive Foundations of Computing. Unveröffentlichtes Manuskript. December 1996.  
<http://www.cs.brown.edu/people/pw/>
- [13] Wegner, P.: Why interaction is more powerful than algorithms. CACM. May 1997. S. 80-91.

---

<sup>11</sup> Vergleiche hierzu insbesondere [8], in dem mit dem „Programmieren nach Vertrag“ eine interessante Technik zur Spezifikation des Klassen- und Objektverhalten vorgestellt wird.