THOMAS CHADZELEK

Analytische Maschinen

Dissertation zur Erlangung des Grades
Doktor der Ingenieurwissenschaften (Dr.-Ing.)
der Technischen Fakultät der Universität des Saarlandes

Saarbrücken 1998

Tag des Kolloquiums:	1998-12-22
Dekan:	PROF. DR. W. J. PAUL
Vorsitzender:	PROF. DR.-ING. H. JANOCHA
Gutachter:	PROF. DR. G. HOTZ
	PROF. DR. W. J. PAUL

Abstract

In this thesis we present some results about *analytic machines* regarding computability over $\mathbb{Q}$ and $\mathbb{R}$, solutions of differential equations, and the stability problem of dynamical systems.

We first explain the machine model, which is a kind of BLUM-SHUB-SMALE machine enhanced by infinite convergent computations. Next, we compare the computational power of such machines over the fields $\mathbb{Q}$ and $\mathbb{R}$ showing e.g. that finite computations with real numbers can be simulated by infinite converging computations on rational numbers, but the precision of the approximation is not known during the process. Analytic computations over $\mathbb{R}$ are strictly more powerful than over $\mathbb{Q}$. Our attention is then shifted to *ordinary differential equations* (ODEs) where we establish sufficient criteria for the computability of their solutions within our model. We investigate dynamical systems described by ODEs and show the undecidability of a class of stability problems for dynamical systems.

Zusammenfassung

In dieser Arbeit präsentieren wir einige Resultate über *analytische Maschinen* hinsichtlich des Berechenbarkeitsbegriffs über $\mathbb{Q}$ und $\mathbb{R}$, der Lösungen von Differentialgleichungen und des Stabilitätsproblems dynamischer Systeme.

Wir erläutern zuerst das Maschinenmodell, das eine Art von BLUM-SHUB-SMALE-Maschine darstellt, erweitert um unendliche, konvergente Berechnungen. Danach vergleichen wir die Mächtigkeit dieses Berechnungsmodells über den Körpern $\mathbb{Q}$ und $\mathbb{R}$ und zeigen z.B., daß endliche Berechnungen mit reellen Zahlen durch unendliche, konvergente Berechnungen mit rationalen Zahlen simuliert werden können, wobei die Genauigkeit der Approximation während des Prozesses nicht bekannt ist. Analytische Berechnungen über $\mathbb{R}$ sind echt mächtiger als über $\mathbb{Q}$. Unsere Aufmerksamkeit wendet sich dann *gewöhnlichen Differentialgleichungen* (DGl) zu, bei denen wir hinreichende Kriterien für die Berechenbarkeit von Lösungen innerhalb unseres Modells angeben. Schließlich untersuchen wir dynamische Systeme, die durch DGl beschrieben werden, und zeigen die Unentscheidbarkeit einer Klasse von Stabilitätsproblemen für dynamische Systeme.

Inhaltsverzeichnis

Einleitung

Wir wollen zunächst aufzeigen, wie das Forschungsgebiet entstand, dem diese Arbeit zuzuordnen ist. Anschließend geben wir einen Überblick über bisherige Arbeiten, der keinen Anspruch auf Vollständigkeit erhebt; es werden jedoch die wichtigen Autoren genannt, von deren Werken ausgehend der interessierte Leser sich das Gebiet in seiner ganzen Tiefe erschließen mag. Danach motivieren wir sowohl den allgemeinen Ansatz als auch unseren speziellen Beitrag zu diesem Themenkreis und grenzen diese voneinander und von konkurrierenden Ideen ab.

Historische Entwicklung

Die Theorie der Algorithmen, auch Rekursions- und Komplexitätstheorie genannt, beschäftigt sich mit der grundlegenden Natur von Berechnungen mit Hinblick darauf, was effizient oder überhaupt effektiv berechenbar ist. Sie nahm ihren Anfang in den zwanziger und dreißiger Jahren dieses Jahrhunderts mit den Arbeiten von HILBERT, ACKERMANN, CHURCH, GÖDEL, KLEENE, TURING und anderen, die das Phänomen der Unentscheidbarkeit entdeckten und viele unterschiedliche Modelle der Berechenbarkeit über den natürlichen Zahlen entwickelten. Trotz ihrer grundsätzlich verschiedenen Ansätze definieren diese Modelle jedoch dieselbe Klasse berechenbarer Funktionen. Die entsprechenden Begriffe lassen sich leicht auf ganze oder rationale Zahlen verallgemeinern, bleiben ihrer Natur nach aber *diskret*. Diese Theorie der diskreten Berechenbarkeit hat sich als äußerst erfolgreich und fruchtbar erwiesen, und das ist nicht zuletzt darauf zurückzuführen, daß ihre Grundbegriffe sich als derart robust und natürlich erwiesen haben. Es ist heute unumstritten, was eine berechenbare Funktion über den natürlichen Zahlen ist und was nicht, und die Churchsche These drückt unseren Glauben aus, daß dieser Formalismus den intuitiven Begriff „berechenbare Funktion“ vollständig erfaßt.

Der Versuch, das Kleinsche Erlanger Programm auf die Klassifikation von Entscheidungsproblemen zu übertragen, führte zur Entwicklung der Reduktionstheorie. COOK brachte den Aspekt der von HARTMANIS eingeführten Komplexitätsklassen ins Spiel und begründete damit die in der theoretischen Informatik außerordentlich fruchtbare Komplexitätstheorie. Anfang der siebziger Jahre entwickelten COOK, KARP und LEVIN das Konzept der NP-Vollständigkeit und formalisierten auf diese Weise, was effizient berechenbar und was *intractable* ist, also nicht wirklich machbar. Wiederum erwies sich, daß die Klasse der NP-vollständigen Probleme ein sehr natürliches und robustes Konzept ist, und so ist auch dieser Zweig der Theorie wohlfundiert und allgemein anerkannt.

Im Gegensatz dazu hat sich bis heute noch keine einheitliche Theorie der Berechenbarkeit über kontinuierlichen Strukturen wie z.B. den reellen oder komplexen Zahlen herausgebildet. Dabei sind die Unterschiede zwischen dem diskreten und dem kontinuierlichen Fall bisweilen äußerst verblüffend. GÖDELs Resultate zeigen, daß die Theorie erster Ordnung der *ganzen* Zahlen unentscheidbar ist. TARSKI hingegen hatte kurz zuvor gezeigt, daß die Theorie erster Ordnung der *reellen* Zahlen entscheidbar[1] ist.

Im Laufe der Jahre gab es immer wieder einzelne Arbeiten, die sich der effektiven Berechenbarkeit über den reellen Zahlen widmeten (siehe ABRAMSON [1]), oder algebraische Komplexitätsaussagen trafen, welche Gebrauch von reellen Zahlen machten (siehe BEN-OR [2]). Das Gebiet der algorithmischen Geometrie (*computational geometry*) stützt sich zwar auf das Modell einer reellen Registermaschine (siehe PREPARATA-SHAMOS), dennoch wird dort keine formale Theorie der Berechenbarkeit entwickelt. Ein einheitliches Maschinenmodell, in dem reelle Zahlen als Einheit aufgefaßt werden und das von einer größeren Gruppe akzeptiert und benutzt wurde, um eine Rekursions- und Komplexitätstheorie aufzubauen, entwickelte sich erst durch eine Arbeit von BLUM, SHUB, SMALE [9].

Die beiden Resultate von GÖDEL und TARSKI legen es eigentlich nahe, den diskreten und den kontinuierlichen Fall getrennt voneinander zu untersuchen. Dennoch gibt es ein angesehenes und erfolgreiches Gebiet, die rekursive Analysis, die sich auf (Orakel-)Turing-Maschinen gründet und nur (im klassischen Sinne) berechenbare reelle Zahlen betrachtet. Ein Hauptsatz dabei besagt, daß alle physikalisch realisierbaren Maschinenmodelle nur *stetige* reelle Funktionen berechnen können, wobei der Stetigkeitsbegriff nicht notwendig dem naiven entspricht. Vertreter dieser Richtung sind z.B. FRIEDMAN und KO [29, 27, 28], POUR-EL und RICHARDS [39] sowie in Deutschland vor allem WEIHRAUCH [45, 46] und seine Schüler [11, 10].

Literarisches Umfeld

BLUM, CUCKER, SHUB, SMALE haben jüngst ein umfassendes Buch [8] über „Complexity and Real Computation“ geschrieben, das praktisch alle bekannten Aspekte der Theorie der $\mathbb{R}$-Maschinen abdeckt und die ältere Literatur umfaßt. In [9] entwerfen die Autoren ein Maschinenmodell und führen die grundlegenden Begriffe von Berechenbarkeit und Komplexität ein. Es werden auch Nichtdeterminismus und NP-Vollständigkeit untersucht sowie universelle Maschinen und Unentscheidbarkeit. Ein wichtiges Ergebnis ist, daß die meisten Julia-Mengen nicht rekursiv aufzählbar sind und ihr Komplement zwar rekursiv aufzählbar, aber unentscheidbar ist. Das Entscheidungsproblem, ob ein reelles Polynom vom Grad vier in mehreren

[1] Von RENEGAR stammt ein Algorithmus, dessen Laufzeit doppelt exponentiell in der Zahl der Quantorenwechsel und exponentiell in der Zahl der Variablen ist.

Variablen eine Nullstelle hat (*4-feasibility problem*), dient als kanonisches NP-vollständiges Problem. Man beachte, daß dieses wegen TARSKI und RENEGAR über $\mathbb{R}$ in exponentieller Zeit entscheidbar ist; die entsprechende Variante über $\mathbb{Z}$ dagegen ist unentscheidbar!

Diese Themen werden in weiteren Arbeiten [5, 7, 3, 4] vertieft, zum Teil zusammen mit CUCKER; dabei interessieren unter anderem Transferresultate zur Frage „$\mathsf{P} = \mathsf{NP}$?“, z.B. daß die Antwort darauf über allen algebraisch abgeschlossenen Körpern der Charakteristik 0 identisch ist [6]. MICHAUX [36] gibt ein entsprechendes Transferresultat für reell abgeschlossene Erweiterungen von $\mathbb{R}$; EMERSON [18] relativiert die vieldiskutierte Frage „$\mathsf{P} = \mathsf{NP}$?“ mittels Orakelmaschinen.

SIEGELMANN und SONTAG [41, 42, 43] sowie MAASS [34, 35] beschäftigen sich mit der Mächtigkeit neuronaler Netze im Hinblick auf Berechenbarkeit, wobei die Eingaben natürlicherweise binär sind, d.h. aus $\mathbb{B}^*$. Es ergibt sich, daß Netze mit rationalen bzw. reellen Gewichten in Polynomzeit genau die Probleme aus P bzw. $\mathsf{P/poly}$ lösen. KOIRAN, der auf dem Gebiet der neuronalen Netze promoviert hat, wandte sich dem BSS-Modell zu und betrachtet abgeschwächte Versionen davon, wobei auch er sich besonders für den Booleschen Anteil der erkannten Sprachen interessiert, d.h. für binäre Eingaben. In [32] beschränkt er die Operationen auf Addition und Subtraktion und verzichtet teilweise auf die Ordnungsrelation „$<$“; dabei zeigt sich die Äquivalenz von reellem und digitalem Nichtdeterminismus[2] und ein einfacher Beweis von $\mathsf{P} \neq \mathsf{NP}$ für lineare Maschinen, bei denen Multiplikation nur mit Konstanten erlaubt ist. Eine Fortsetzung davon [31] erlaubt einen moderaten Gebrauch der Multiplikation in dem Sinne, daß die Kosten arithmetischer Operationen nunmehr vom Grad des entstehenden Polynoms[3] und der Größe seiner Koeffizienten abhängen. Es zeigt sich, daß dieses schwache Modell bei binären Eingaben und Polynomzeit genau die Sprachen aus $\mathsf{P/poly}$ erkennt. Mit FOURNIER argumentiert er, daß untere Schranken über den reellen Zahlen wohl nicht leichter zu beweisen sind [20].

CUCKER beschäftigt sich in [13] mit einer Hierarchie unentscheidbarer Probleme. Zusammen mit SHUB und SMALE [16] werden verschiedene Komplexitätsklassen in KOIRANs schwachem Modell separiert, insbesondere wird dafür mit Hilfe des digitalen Nichtdeterminismus $\mathsf{P} \neq \mathsf{NP}$ gezeigt; ferner interessieren parallele und alternierende Maschinen.

Gemeinsam mit KOIRAN hat er weitere Arbeiten mit wechselnden Koautoren veröffentlicht. In [14] untersuchen sie probabilistische Komplexitätsklassen, verallgemeinern $\mathsf{BPP} \subset \mathsf{P/poly}$ und studieren dabei Boolesche Anteile sowie die Mächtigkeit reeller Konstanten und des Gleichheitstests. In [15] geht es um eine verallgemeinerte Version der Frage, ob dünnbesetzte Mengen NP-vollständig sein können.

Die Idee der analytischen Maschinen wird von HOTZ in [25] im Zusammenhang mit der Berechenbarkeit fraktaler Strukturen eingeführt. VIERKE hat in seiner Diplomarbeit [44] die von HOTZ in seiner Vorlesung über $\mathbb{R}$-berechenbare Funktionen

[2] Die geratenen Werte entstammen also $\mathbb{R}^*$ bzw. $\mathbb{B}^*$.

[3] Alle Zwischenergebnisse einer Rechnung lassen sich als rationale Funktionen der Eingabe und der Programmkonstanten auffassen.

entwickelten Konzepte ausgearbeitet. Er hat die Kodierung in reellen Zahlen betrachtet, verschiedene Funktionsklassen durch Beispiele nicht-berechenbarer Funktionen getrennt und den Hierarchiesatz bewiesen. Dabei hat er die von HOTZ in seiner Vorlesung gegebene Skizze der Unentscheidbarkeit des Konvergenzproblems übernommen und auch eine Konstruktion mit drei Maschinen erfunden, die die Konvergenz einer Maschine entscheiden. Von ihm stammt auch die Idee der quasi stark δ-$\mathbb{Q}$-analytischen Maschinen, die er selbst aber nicht weiter ausgeführt hat. Zusammen mit SCHIEFFER erschien ein technischer Bericht [26] mit dem Darstellungssatz und dem Simulationssatz; hierin wird auch die nicht-fraktale Natur $\mathbb{R}$-berechenbarer Funktionen erwähnt. Ferner werden stark δ-$\mathbb{Q}$-analytische Funktionen definiert, ihre Abgeschlossenheit bewiesen und ihr Nutzen als Unterprogramme untersucht.

Motivation und Abgrenzung

Vielen Problemen von praktischer Relevanz liegt in einer natürlichen Beschreibung das Kontinuum der reellen Zahlen zugrunde. Die algorithmische Geometrie befaßt sich z.B. mit Punkten in der Euklidschen Ebene, deren Koordinaten zunächst einmal reell sind; dieses Gebiet spielt eine wichtige Rolle unter anderem in der Robotik bei Bewegungsplanung, Kollisionserkennung, Formapproximation, Bilderkennung etc. Auch in der Unternehmensforschung (*operations research*) bei der Lösung von Optimierungsproblemen, z.B. der Planung eines bestmöglichen Standorts durch Voronoi-Diagramme oder dem linearen Programmieren mittels Simplex-Algorithmus, lassen sich die verwendeten Verfahren ganz selbstverständlich mittels reeller Zahlen erklären. Dies trifft ganz allgemein auf ein großes Gebiet zu, das sich als „wissenschaftliches Rechnen“ (*scientific computing*) bezeichnen läßt.

Beim Entwurf und der Analyse solcher Algorithmen über den reellen Zahlen läßt sich großer Nutzen aus der Analysis ziehen, dem laut VON NEUMANN[4] „technisch erfolgreichsten und bestausgearbeiteten Teil der Mathematik“. Bei derselben Gelegenheit bezeichnet dieser die Logik als eines der „technisch widerspenstigsten Gebiete der Mathematik“ wegen ihres „strengen alles-oder-nichts Konzepts“, und weil sie „sehr wenig Kontakt mit dem kontinuierlichen Konzept der reellen oder komplexen Zahl, d.h. mit mathematischer Analysis“ hat. Er sieht die bisherige diskrete Automatentheorie als Teil der formalen Logik mit den gleichen Nachteilen wie diese und fordert einen mehr analytischen Zugang zur Theorie der Automaten und der Information.

BLUM, CUCKER, SHUB und SMALE erkannten, daß die klassische Rekursions- und Komplexitätstheorie für die Untersuchung der oben erwähnten Algorithmen „fundamental unangemessen“ [8] ist, und entwickelten daraufhin ihre Ideen für Maschinen und Berechnungen über den reellen Zahlen. Dies war KARP zufolge der erste Ansatz, reelle Arithmetik mit einem uniformen Berechnungsmodell zu verknüpfen, das nicht auf Eingaben fester Dimension beschränkt ist und die Grenzen polynomieller Bere-

[4] Hixon-Symposium-Vortrag 1948, übersetzt nach einem Zitat aus [8]

chenbarkeit beleuchten hilft. Zur Motivation für ihre Untersuchungen dienen den Autoren unter anderem die folgenden Beispiele:

Mandelbrot-Menge Man betrachtet die Abbildung $f_c : \mathbb{C} \to \mathbb{C}$, $z \mapsto z^2 + c$ und definiert die Mandelbrot-Menge als $M := \mathbb{C} \setminus \{c \in \mathbb{C} \mid \lim_{n\to\infty} f_c^n(0) = \infty\}$. Auf PENROSE [38, S. 124] geht die Frage zurück, ob die Menge M entscheidbar ist; dabei war er sich wohlbewußt, daß diese Frage formal nicht wohldefiniert war und eine geeignete Theorie fehlte, sie zu formulieren und zu beantworten. Im Rahmen der reellen Berechenbarkeit nach dem hier betrachteten Ansatz ergibt sich (aus dem Darstellungssatz 1.1), daß M unentscheidbar und $\mathbb{C} \setminus M$ semi-entscheidbar ist, d.h. der Haltebereich einer $\mathbb{R}$-Maschine.

Julia-Mengen Man betrachtet die obige Abbildung und definiert die Julia-Mengen $J_c := \mathbb{C} \setminus \{z \in \mathbb{C} \mid \lim_{n\to\infty} f_c^n(z) = \infty\}$. Diese spielen eine große Rolle in der Theorie komplexer dynamischer Systeme, und es zeigt sich, daß $\mathbb{C} \setminus J_c$ semi-entscheidbar und J_c bis auf triviale Fälle unentscheidbar ist.

Newton-Verfahren Dieser Suchalgorithmus zur Approximation der Nullstelle eines reellen Polynoms ist als Verfahren aus der Analysis ganz selbstverständlich über dem Kontinuum definiert. In unserer Terminologie ist die Zuordnung von Startwerten zu Nullstellen eine $\mathbb{R}$-analytische Funktion, deren Definitionsbereich – die Menge der „guten" Startwerte – eine analytisch semi-entscheidbare Menge darstellt (vgl. Beispiel 1.2).

Rucksackproblem Gegeben seien Gewichte $x_1, \ldots, x_n$ aus einem Ring $\mathcal{R}$; man entscheide, ob es eine Teilmenge $S \subset [1 : n]$ gibt mit $\sum_{i\in S} x_i = 1$. Bei diesem Beispiel fragt man sich, ob es über einem angeordneten Ring leichter zu lösen ist, ob Multiplikationen helfen, und ob es einen polynomiellen Algorithmus dafür gibt; solche Fragen werden in dieser Arbeit jedoch keine Rolle spielen.

Wir interessieren uns in dieser Arbeit für Maschinen, die unendliche konvergente Berechnungen durchführen. Dabei untersuchen wir insbesondere die Frage der Mächtigkeit von Maschinen über den reellen bzw. rationalen Zahlen und die Möglichkeit der Simulation von reellen Berechnungen durch unendliche konvergente Berechnungen mit rationalen Zahlen. Alle reellen Funktionen, die in diesem Sinne approximiert werden können, sehen wir als interessante Objekte einer Theorie dieser Maschinen an; dazu gehören ferner deren Abschlußeigenschaften und die Lösungen von in diesem Sinne berechenbaren Differentialgleichungen.

Im Hinblick auf das Gebiet der hybriden Systeme [23] und der dort geführten Korrektheitsbeweise sind wir an einer Theorie interessiert, die sowohl diskrete Entscheidungen eines Steuercomputers als auch kontinuierliche Abläufe eines physikalischen Prozesses umfassend beschreiben kann. Dabei stehen in der vorliegenden Arbeit wiederum Fragen der Simulation im Vordergrund und dann auch solche nach der Stabilität und deren Entscheidbarkeit.

Wir haben bereits erwähnt, daß z.B. KO und WEIHRAUCH sich in ihren Arbeiten strikt auf das Modell der Turing-Maschine beschränken. Dahinter steht insbesondere der Anspruch der physikalischen Realisierbarkeit, welcher bei den hier behandelten $\mathbb{R}$-Maschinen natürlich nicht erhoben wird. Die *feasible real RAM* von BRATTKA und HERTLING [11, 12] ist ein wichtiger Ansatz, das Programmiermodell der reellen Maschinen mit der eingeschränkten Mächtigkeit der Turing-Maschinen zu kombinieren. Wir begrüßen hierbei insbesondere die Bemühungen um eine Implementierung, werden aber auf diesen und ähnliche Zugänge zu diesem Gebiet nicht weiter eingehen. Unsere Untersuchungen haben zunächst einmal ein besseres Verständnis bestimmter Sachverhalte zum Ziel; erst wenn dieses erreicht ist, kann eine Umsetzung in die Praxis erfolgen. Dabei gilt, daß das *Handeln vom Verstehen geleitet* wird.

Gliederung und Resultate

Im *ersten Kapitel* werden mit Hilfe abstrakter Maschinen unendliche Berechnungen erklärt und danach verschiedene Formen von Registermaschinen über $\mathbb{R}$ und $\mathbb{Q}$ eingeführt. Ein wichtiges Thema ist die Hierarchie der Klassen berechenbarer Funktionen, der Einfluß der Rundungsfunktion und das Verhältnis zwischen Maschinen über $\mathbb{Q}$ und solchen über $\mathbb{R}$. Satz 1.2 zeigt, daß $\mathbb{R}$-berechenbare Funktionen keinen fraktalen Charakter haben können. Satz 1.10 zeigt, daß einstellige $\mathbb{R}$-analytische Funktionen durch geeignete Wahl der Rundung δ-$\mathbb{Q}$-analytisch sind. Satz 1.16 zeigt mit Hilfe algebraischer Entscheidungsbäume, daß eine gewisse $\mathbb{R}$-analytische Funktion nicht δ-$\mathbb{Q}$-analytisch ist. Mit Satz 1.19 geben wir eine Verschärfung des Simulationssatzes mit einem wesentlich eleganteren Beweis: Jede $\mathbb{R}$-berechenbare Funktion ist quasi stark δ-$\mathbb{Q}$-analytisch. Bei der Untersuchung des Konvergenzproblems wird der Hierarchiesatz verallgemeinert und es zeigt sich in Satz 1.21, wie man die Konvergenz einer Maschine durch zwei hintereinander geschaltete entscheidet. Zur Vermeidung unendlich langer Wörter als Zwischenergebnisse kann eine Kodierung in reellen Zahlen erfolgen. Zuletzt wird eine Unterprogrammtechnik eingeführt, mit der gewissermaßen die Grenzwertbildung als Elementaroperation möglich wird; dabei zeigt Satz 1.30 eine obere Schranke für die unendliche Hierarchie $\mathbb{R}^i$-analytischer Funktionen.

Im *zweiten Kapitel* betrachten wir den Verbrauch der beiden Ressourcen Zeit und Platz durch Berechnungen und das Verhältnis verschiedener Maschinentypen unter dem klassischen Aspekt der polynomiellen Verknüpftheit. Die beiden wichtigsten Resultate sind, daß auch analytische Berechnungen stets mit konstantem Platz durchgeführt werden können (Satz 2.2) und unsere speziellen Registermaschinen zu dem ursprünglichen Modell von BLUM, SHUB, SMALE gleichwertig sind (Satz 2.5).

Im *dritten Kapitel* werden Anfangswertprobleme für Systeme expliziter, gewöhnlicher Differentialgleichungen erster Ordnung untersucht, wobei die rechte Seite von einer $\mathbb{R}$-Maschine ohne Division berechnet werden kann. Satz 3.2 zeigt unter bestimmten Voraussetzungen die Eindeutigkeit der globalen Lösung, die dann gemäß Hauptsatz 1 robust δ-$\mathbb{Q}$-analytisch ist. Für stark δ-$\mathbb{Q}$-analytische rechte Seiten zeigt Satz 3.7 unter einer klassischen Voraussetzung dasselbe. Sodann betrachten wir Stabilitätsprobleme für die Lösungen solcher Differentialgleichungen und insbesondere für dynamische Systeme. Hauptsatz 2 zeigt die Unentscheidbarkeit bereits der einfachsten Formulierung des Stabilitätsproblems, nämlich „Konvergiert ein dynamisches System gegen einen stabilen Zustand im Sinne eines Fixpunktes?", selbst für das mächtige Berechnungsmodell der $\mathbb{R}$-analytischen Maschinen, das eine präzise reelle Arithmetik und unendliche Laufzeit umfaßt.

Ein *Anhang* beschreibt die verwendeten mathematischen Notationen und Begriffe. Daran schließen sich die *Ausführliche Zusammenfassung* sowie das *Literaturverzeichnis* an.

Teile dieser Arbeit erscheinen in einem Sonderband „Computability and Complexity in Analysis" der Zeitschrift *Theoretical Computer Science* als gemeinsame Veröffentlichung mit Prof. HOTZ.

Danksagung

Ich bedanke mich herzlich bei Herrn Prof. Dr. Günter Hotz für die Vergabe dieses interessanten und fruchtbaren Themas, für den Freiraum bei der Bearbeitung desselben und für die entscheidenden Anregungen, die mich auf meinem Weg leiteten. Er hat meine wissenschaftliche Ausbildung geprägt und insbesondere durch seine großzügige Förderung wesentlich zum Gelingen dieser Arbeit beigetragen.

Für hilfreiche Gespräche und wertvolle Ratschläge gebührt mein Dank den Herren Artur Fuhrmann, Dr. Björn Schieffer, Dr. Elmar Schömer, Frank Schulz und Gero Vierke. Einige von ihnen haben freundlicherweise das Korrekturlesen dieser Arbeit übernommen; nichtsdestotrotz bleibt die Verantwortung für alle noch verbliebenen Fehler selbstverständlich bei mir allein.

Allen Kollegen an diesem Lehrstuhl danke ich für das hervorragende Arbeitsklima. Insbesondere aber danke ich meiner Familie, die mein Studium stets wohlwollend unterstützt hat.

1. Maschinenmodell

Wir werden in diesem Kapitel zunächst abstrakte Maschinen definieren, welche bereits unendliche Berechnungen ausführen können, und danach konkrete Registermaschinen über einem Ring. Diese werden so erweitert, daß auch rationale Maschinen mit reellen Eingaben operieren können, dabei spielt das Konzept der Rundungsfunktion eine wichtige Rolle. Es werden dann verschiedene Klassen berechenbarer Funktionen vorgestellt, ihre Abschlußeigenschaften untersucht und ihre Beziehungen untereinander analysiert. Danach werden unentscheidbare Probleme betrachtet und als letztes eine Erweiterung des Maschinenmodells um Unterprogramme.

Eine mathematische oder *abstrakte Maschine* in unserem Sinne ist ein Tupel

$$\mathcal{M} = (K, K_a, K_e, K_z, \Delta, A, \mathrm{in}, \mathrm{out}),$$

wobei die Menge K die *Konfigurationen* der Maschine $\mathcal{M}$ bildet und K_a, K_e, $K_z \subset K$ die *Anfangs-*, *End-* und *Zielkonfigurationen* sind. Die Abbildung $\Delta : K \to K$ mit $\Delta|_{K_e} = \mathrm{id}_{K_e}$ heißt *Übergangsfunktion* von $\mathcal{M}$; $\mathrm{in} : A^* \to K_a$ und $\mathrm{out} : K \to A^*$ heißen *Ein-* und *Ausgabefunktion* von $\mathcal{M}$ über dem *Alphabet* A.

Eine Folge $b = (k_i)_{i=0}^{\infty}$ von Zuständen $k_i := \Delta^i(k_0)$ heißt *Berechnung* von $\mathcal{M}$ angesetzt auf k_0. Diese heißt *endlich*, falls $\exists\, n : \ k_n \in K_e$; damit wird die Folge ab dem n. Glied stationär, das kleinste solche n heißt *Länge* und $\mathrm{out}(k_n)$ *Resultat* der Berechnung. Gilt zusätzlich $k_0 \in K_a$, so heißt b *regulär*.

Falls auf A^* eine Metrik gegeben ist, so kann man obige Definition erweitern. Sei b eine Berechnung mit $k_0 \in K_a$, so daß $k_{i_j} \in K_z$ für unendlich viele i_j gilt; $(k_{i_j})_{j=0}^{\infty}$ sei die Teilfolge aller dieser Zielkonfigurationen. Die Berechnung heißt nun *analytisch*, falls

$$\lim_{j \to \infty} \mathrm{out}(k_{i_j})$$

existiert; dieser Grenzwert sei das Resultat von b und $\mathrm{out}(k_{i_n})$ heißt n. *Approximation* des Resultats.

Auf folgende Weise legt die Maschine $\mathcal{M}$ nun eine partielle Funktion $\Phi_{\mathcal{M}} : A^* \rightsquigarrow A^*$ fest. Falls für $x \in A^*$ die Berechnung von $\mathcal{M}$ angesetzt auf $\mathrm{in}(x)$ regulär oder analytisch ist mit Ergebnis $y \in A^*$, so sei $\Phi_{\mathcal{M}}(x) := y$; andernfalls undefiniert. Ferner läßt sich die n. Approximation $\Phi_{\mathcal{M}}^{(n)}$ dieser Funktion auf demselben Definitionsbereich festlegen durch $\Phi_{\mathcal{M}}^{(n)}(x) := \mathrm{out}(k_{i_n})$; enthält eine (reguläre) Berechnung weniger als n Zielkonfigurationen, so sei $\Phi_{\mathcal{M}}^{(n)}(x) := y$.

Den *Definitionsbereich* von $\Phi_{\mathcal{M}}$ bezeichnen wir mit $\mathbb{D}_{\mathcal{M}}$; der *Haltebereich* $\mathbb{D}_{\mathcal{M}}^{H} \subset \mathbb{D}_{\mathcal{M}}$ enthalte genau die Eingaben, für die die Berechnung von $\mathcal{M}$ regulär ist. Zwei

Maschinen $\mathcal{M}$ und $\mathcal{M}'$ heißen *äquivalent*, falls ihre Halte- und Definitionsbereiche übereinstimmen und $\Phi_{\mathcal{M}} = \Phi_{\mathcal{M}'}$ ist.

Man kann die Ein- und Ausgabefunktionen in und out auch auf unendliche Wörter über A fortsetzen, wobei nur analytische Berechnungen damit sinnvoll operieren können. Dann ergibt sich $\Phi_{\mathcal{M}} : A^\star \rightsquigarrow A^\star$ durch folgenden Zusatz: Falls für $x \in A^\omega$ die Berechnung von $\mathcal{M}$ angesetzt auf $\mathrm{in}(x)$ analytisch ist mit Ergebnis $y \in A^\star$, so sei $\Phi_{\mathcal{M}}(x) := y$; andernfalls undefiniert.

1.1. Registermaschinen

Anmerkung 1.1. Wir definieren nun eine spezielle Form von Registermaschinen, die mit Elementen eines Rings $\mathcal{R}$ rechnen; wir denken dabei vor allem an die Körper $\mathbb{Q}$ der rationalen, $\mathbb{R}$ der reellen bzw. $\mathbb{C}$ der komplexen Zahlen. Dennoch lassen sich diese Maschinen für beliebige Ringe mit Eins erklären, die nicht notwendig kommutativ oder angeordnet sein müssen. Die ganzen Zahlen sollten einen natürlichen Unterring von $\mathcal{R}$ bilden, damit die Maschinen die Länge von Ein- bzw. Ausgaben einfach zählen können. Für den Fall analytischer Maschinen muß $\mathcal{R}$ zudem ein metrischer Raum sein, damit der Begriff des Grenzwerts erklärt ist. Wenn wir in Zukunft vereinfachend von einem *Ring* sprechen, so sollen diese Bedingungen stets gelten. Die Konstruktion folgt weitgehend [26] und [44] und ist – was Berechenbarkeit angeht – äquivalent zu dem in [9] verwendeten Modell.

Diese $\mathcal{R}$-Maschinen (vgl. Abbildung 1.1) verfügen über ein endliches Programm π und ein Steuerwerk mit Akkumulator α, Befehlszähler β, Indexregister γ und Präzisionsregister δ. Ferner gibt es ein unbeschränktes Eingabeband X, von dem nur gelesen werden darf, ein unbeschränktes Ausgabeband Y, auf das nur geschrieben werden darf, sowie einen unbeschränkten Rechenspeicher Z. Das Präzisionsregister wird nur bei erweiterten $\mathbb{Q}$-Maschinen eine Rolle spielen, seine Funktion wird später im Abschnitt 1.1.1 erläutert.

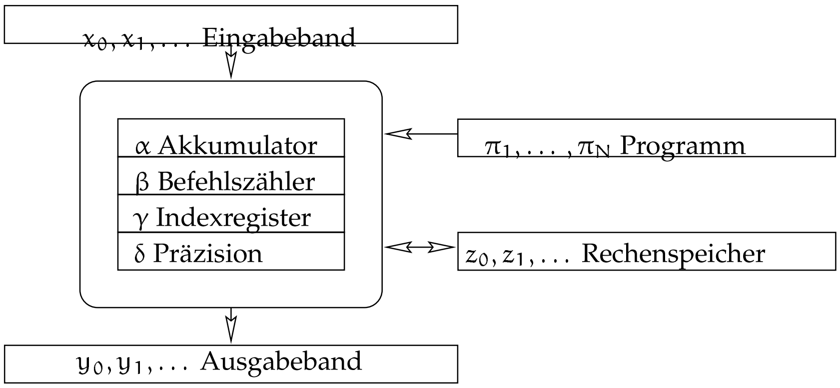


Abbildung 1.1.: Schema der Registermaschine

Eine Konfiguration einer solchen Maschine ist gegeben durch die Belegung $\pi : [1 : N] \to \Omega$ des Programmspeichers, die Werte $\alpha \in \mathcal{R}$, $\beta \in [1 : N]$, $\gamma, \delta \in \mathbb{N}$ der Register und die Belegungen $x, y, z : \mathbb{N} \to \mathcal{R}$ der Bänder und des Rechenspeichers. Dabei steht Ω für eine noch zu spezifizierende Menge von Maschineninstruktionen. Der Übersichtlichkeit halber unterscheiden wir nicht zwischen den Namen der Register und ihren Werten; den Inhalt der i. Band- bzw. Speicherzelle bezeichnen mit $\pi_i = \pi(i)$ bzw. analog x_i, y_i oder z_i. Nun definieren wir $\mathcal{R}$-Maschinen formal als abstrakte Maschinen wie folgt:

$$\begin{aligned} K &:= \{k = (\alpha, \beta, \gamma, \delta, \pi, x, y, z) \text{ wie oben erläutert}\}, \\ K_a &:= \{k \in K \mid \alpha = \gamma = \delta = 0,\ \beta = 1,\ \forall j :\ y_j = z_j = 0\}, \\ K_e &:= \{k \in K \mid \pi_\beta = \texttt{end}\}, \\ K_z &:= \{k \in K \mid \pi_\beta = \texttt{print}\}. \end{aligned}$$

Die Ein- und Ausgabefunktionen $\mathtt{in} : \mathcal{R}^* \to K_a$ und $\mathtt{out} : K \to \mathcal{R}^*$ interpretieren die erste Bandzelle als Längenangabe n und die folgenden n Zellen als Elemente der Folge. Sei also $k = (\alpha, \beta, \gamma, \delta, \pi, x, y, z) \in K$ gegeben, so setzen wir $\mathtt{out}(k) := (y_1, \ldots, y_{y_0}) \in \mathcal{R}^*$, falls $y_0 \in \mathbb{N}$, und $\mathtt{out}(k) := \varepsilon$ sonst. Sei umgekehrt $r = (r_1, \ldots, r_n) \in \mathcal{R}^*$, dann setzen wir $\mathtt{in}(r) := (\alpha, \beta, \gamma, \delta, \pi, x, y, z) \in K_a$ mit

$$x_i := \begin{cases} n & \text{falls } i = 0; \\ r_i & \text{falls } 1 \le i \le n; \\ 0 & \text{sonst.} \end{cases}$$

Diese Funktion läßt sich leicht auf $\mathcal{R}^\star$ fortsetzen, indem man als Längenangabe für unendliche Eingabewörter z.B. -1 verwendet. Bei Ausgaben wollen wir von dieser Möglichkeit keinen Gebrauch machen, so daß diese zu jedem Zeitpunkt der Berechnung endlich sind. Bildet man allerdings den Grenzwert der Ausgaben unendlicher Berechnungen, dann kann man $\lim_{n\to\infty} r^{(n)} := r$ setzen, falls alle n. Approximationen $r^{(n)} \in \mathcal{R}^*$ endlich und das Resultat $r \in \mathcal{R}^\omega$ unendlich lang sind sowie

1. $\lim_{n\to\infty} |r^{(n)}| = \infty$;
2. $\forall\, i \in \mathbb{N} : \lim_{n\to\infty} r_i^{(n)} = r_i$, d.h. die endlichen Wörter konvergieren buchstabenweise gegen das unendliche. Genaugenommen wird jeweils der Grenzwert der Teilfolge der Wörter $r^{(n_k)}$ mit Länge mindestens i gebildet.

Die Menge Ω der Operationen enthält die Befehle aus Tabelle 1.1. Man beachte, daß Ω von der Größe N des Programms und dem Ring $\mathcal{R}$ abhängt, d.h. $\Omega = \Omega_N^{\mathcal{R}}$; diese Abhängigkeit wird aber der Einfachheit halber meist nicht explizit vermerkt.

Die Semantik der Anweisungen versteht sich weitgehend von selbst und legt in naheliegender Weise die Übergangsfunktion Δ fest. Die Zuweisungen (1a) laden bzw. speichern Werte mittels fester Adressen oder dem aktuellen Inhalt des Indexregisters. Die *nichtnegative Differenz* $\dot{-}$ ist definiert durch

$$\dot{-} : \mathbb{N} \times \mathbb{N} \to \mathbb{N},\ (x, y) \mapsto x \mathbin{\dot{-}} y := \begin{cases} x - y & \text{falls } x \ge y; \\ 0 & \text{sonst.} \end{cases}$$

1. Zuweisungen
   a) $\alpha := x_i,\ \alpha := z_i,\ y_i := \alpha,\ z_i := \alpha \quad$ für $i \in \mathbb{N} \cup \{\gamma\}$
   b) $\alpha := r \quad$ für $r \in \mathcal{R}$
   c) $\alpha := \delta$
   d) $\gamma := 0$
2. Arithmetik
   a) $\alpha := \alpha + z_i,\ \alpha := \alpha \cdot z_i \quad$ für $i \in \mathbb{N} \cup \{\gamma\}$
   b) $\alpha := -\alpha,\ \alpha := \alpha^{-1}$
   c) $\gamma := \gamma + 1,\ \gamma := \gamma \dot{-} 1$
3. Verzweigungen
   `if` $\alpha > 0$ `then goto` m `else goto` $n \quad$ für $m, n \in [1:N]$
4. Sonderbefehle
   `end`, `next` δ, `print`

Tabelle 1.1.: Befehle der Registermaschine

Man beachte, daß dem γ-Register mit obigen Befehlen (1d bzw. 2c) nur natürliche Zahlen zugewiesen werden können. Der spezielle Befehl `print` markiert die Zeitpunkte einer analytischen Berechnung, zu denen ein neues Glied der Ausgabefolge bereitsteht; er bewirkt lediglich eine Veränderung des Befehlszählers. Nur zur Verwendung in erweiterten $\mathbb{Q}$-Maschinen gedacht ist „`next` δ“.

Anmerkung 1.2. Wir betrachten ein Programm nur dann als korrekt, wenn für jede Eingabe die Operation $\alpha := \alpha^{-1}$ nur ausgeführt wird, falls α^{-1} definiert ist. Über einem Ring ohne Anordnung interpretieren wir die Verzweigungsbedingung als $\alpha \neq 0$, was über dem Körper der komplexen Zahlen $\mathbb{C}$ äquivalent zu $|\alpha| > 0$ ist. Ferner sei `end` der letzte Befehl jedes Programms.

Definition 1.1. Gegeben seien ein Ring $\mathcal{R}$, eine natürliche Zahl N und ein Programm $\pi : [1:N] \to \Omega_N^{\mathcal{R}}$. Dann heißt die gemäß obiger Konstruktion dadurch eindeutig festgelegte abstrakte Maschine $\mathcal{M}_\pi^{\mathcal{R}} = (K, K_a, K_e, K_z, \Delta, \mathcal{R}, \mathtt{in}, \mathtt{out})$ die *$\mathcal{R}$-Maschine zum Programm π.* □

Bisher wurden nur $\mathcal{R}$-Maschinen mit *einem* Indexregister γ eingeführt; von diesen wollen wir auch immer ausgehen, sofern nichts anderes erwähnt ist. Unter dem Gesichtspunkt der Berechenbarkeit ist dies auch ausreichend; für Komplexitätsuntersuchungen kann es jedoch nützlich sein, $\mathcal{R}$-Maschinen *mit* k *Indexregistern* $\gamma_1, \ldots, \gamma_k$ zu betrachten. Diese gehen in naheliegender Weise aus obiger Konstruktion hervor, indem alle Befehle statt für γ für jedes γ_i definiert werden.

1.1.1. Erweiterte $\mathbb{Q}$-Maschinen

$\mathbb{Q}$-Maschinen können durch unendliche Berechnungen Ausgabefolgen erzeugen, die gegen reelle Zahlen konvergieren. Daher liegt es nahe, die Registermaschine über $\mathbb{Q}$ so zu erweitern, daß sie auch reelle Eingaben verarbeiten und damit Funktionen $f : \mathbb{R}^\star \rightsquigarrow \mathbb{R}^\star$ berechnen kann. Hierzu verwendet man das Präzisionsregister δ und fordert einfach, daß die Befehle $\alpha := x_i$ statt dem tatsächlichen Eingabewert x_i eine rationale Approximation $q \in \mathbb{Q}$ mit $|x_i - q| < 2^{-\delta}$ lesen. Ebenso verfährt man bei Zuweisungen $\alpha := r$ von *irrationalen* Konstanten. Die Präzision der Rundung wird nun schrittweise gesteigert, indem die Maschine bei jedem „`next` δ"-Befehl erneut startet und δ um eins erhöht.

Anmerkung 1.3. Wir haben damit zwischen zwei *Typen* von Konstanten bzw. Zuweisungsoperationen unterschieden, damit auch erweiterte $\mathbb{Q}$-Maschinen Konstanten aus dem ihnen eigenen Bereich der rationalen Zahlen exakt handhaben können. Es ist *zwingend*, daß zumindest eine exakte 1 verfügbar ist, um z.B. die Länge von Ausgaben zu beschreiben; dann ist aber auch jede feste Zahl aus $\mathbb{Q}$ in konstanter (d.h. von der Eingabe unabhängiger) Zeit erzeugbar, und die Idee der exakten rationalen Konstanten erscheint vernünftig. Auf reelle Konstanten für $\mathbb{R}$-Maschinen wollen wir ebenfalls nicht verzichten, damit z.B. $x \mapsto c \cdot x$ für alle $c \in \mathbb{R}$ $\mathbb{R}$-berechenbar ist; dennoch soll der Simulationssatz erhalten bleiben.

Formal bedeutet das für eine Konfiguration $k = (\alpha, \beta, \gamma, \delta, \pi, x, y, z)$ mit $\pi_\beta =$ „`next` δ", daß $\Delta(k) := (0, 1, 0, \delta + 1, \pi, x, y, z)$. Ferner erlauben wir reelle Zahlen auf dem Eingabeband und im Programm, d.h. $\mathtt{in} : \mathbb{R}^\star \to K_a$, $x : \mathbb{N} \to \mathbb{R}$ und $\pi : [1 : N] \to \Omega_N^{\mathbb{R}}$, und der Grenzwert $\lim_{j\to\infty} \mathtt{out}(k_{i_j})$ bei analytischen Berechnungen muß lediglich in $\mathbb{R}$ existieren.

Diese erweiterte Registermaschine ist nun nicht mehr durch ihr Programm π allein eindeutig festgelegt, vielmehr muß auch die Art der Rundung spezifiziert werden. Damit die Maschine deterministisch bleibt, interpretieren wir die Zuweisung $\alpha := x_i$ als $\alpha := \rho(x_i, \delta)$, wobei ρ der folgenden Definition genüge. Dabei werden wir unser Interesse in Zukunft – in den Fällen, in denen ρ vorgegeben wird – auf *berechenbare* Rundungsfunktionen konzentrieren, da sonst der Begriff der „berechenbaren Funktion" fragwürdig wird.

Definition 1.2 (Rundung). Eine Funktion $\rho : \mathbb{R} \times \mathbb{N} \to \mathbb{Q}$, $(x, n) \mapsto x_n$ heißt *Rundungsfunktion*, falls stets $|x - x_n| < 2^{-n}$. □

Definition 1.3. Gegeben seien ein Programm $\pi : [1 : N] \to \Omega_N^{\mathbb{R}}$ und eine $\mathbb{R}$-berechenbare Rundungsfunktion ρ. Dann heißt die gemäß obiger erweiterter Konstruktion dadurch eindeutig festgelegte abstrakte Maschine $\mathcal{M}_{\pi,\rho}^{\delta\text{-}\mathbb{Q}} = (K, K_a, K_e, K_z, \Delta, \mathbb{R}, \mathtt{in}, \mathtt{out})$ die *δ-$\mathbb{Q}$-Maschine zum Programm π und der Rundung* ρ. □

Diese Abhängigkeit von der Rundungsfunktion ist unschön – was wir später auch noch näher erläutern –, daher interessieren besonders solche Programme π, die unabhängig von der Art der Rundung die gleiche Funktion definieren. Präzise gesagt erweitern wir konzeptionell die Übergangsfunktion Δ zu einer Relation derart, daß nach einer Zuweisung der Form $\alpha := x_i$ (und analog $\alpha := r$ für $r \in \mathbb{R} \setminus \mathbb{Q}$) der Akkumulator *irgendeine* Zahl $q \in \mathbb{Q}$ mit $|x_i - q| < 2^{-\delta}$ enthalten darf. Wir fordern dann für jede Eingabe, daß entweder alle Berechnungen regulär bzw. analytisch sind und im Resultat übereinstimmen oder daß keine Berechnung zu einem Resultat führt. Programme mit dieser Eigenschaft nennen wir *robust* und bezeichnen mit $\mathcal{M}_\pi^{\delta\text{-}\mathbb{Q}}$ irgendeine der äquivalenten δ-$\mathbb{Q}$-Maschinen zum Programm π.

Anmerkung 1.4. Es wird später in den Beweisen deutlich werden, warum wir von einem robusten Programm verlangen, daß es *für alle Arten der Rundung* und nicht nur für alle $\mathbb{R}$-berechenbaren Rundungsfunktionen zum selben Ergebnis führt.

1.2. Berechenbare Funktionen

Wir sind nun in der Lage, den Begriff der berechenbaren Funktion über einem Ring zu formalisieren, von dem wir viele Varianten kennenlernen werden. Daher gibt die Abbildung 1.2 vorab einen Überblick über die Hierarchie der Klassen berechenbarer Funktionen und zeigt mit einem Stichwort zur Begründung an, warum die Inklusionen echt sind. Die Klassen unterhalb der gezeigten Linie sind abgeschlossen unter der Hintereinanderausführung, die darüber sind es nicht. All diese Zusammenhänge werden dann nach und nach erläutert und erarbeitet.

$\vdots$

$\mathbb{R}^2$-analytisch

$\subsetneq$ Hierarchiesatz

$\mathbb{R}$-analytisch

$\subsetneq$ $\chi_{\mathbb{Q}}^{+}$

δ-$\mathbb{Q}$-analytisch

$\subsetneq$ DIRICHLET-Fkt.

robust δ-$\mathbb{Q}$-analytisch — nicht abgeschlossen

$\subsetneq$ — abgeschlossen

stark δ-$\mathbb{Q}$-analytisch $\subsetneq$ quasi stark δ-$\mathbb{Q}$-analytisch

Stetigkeit — $\subsetneq$ KOCH-Kurve

$\mathbb{R}$-berechenbar

$\subsetneq$ Def.bereich

$\mathbb{Q}$-berechenbar = TURING-berechenbar

Abbildung 1.2.: Hierarchie der Klassen berechenbarer Funktionen

Definition 1.4. Eine Funktion $f : \mathcal{R}^\star \supset D \to \mathcal{R}^\star$ heißt *analytisch $\mathcal{R}$-berechenbar* oder kurz *$\mathcal{R}$-analytisch*, falls es eine $\mathcal{R}$-Maschine $\mathcal{M}$ gibt, so daß $f = \Phi_\mathcal{M}|_D$ (und damit auch $D \subset \mathbb{D}_\mathcal{M}$). Falls $D \subset \mathbb{D}_\mathcal{M}^H$ Teilmenge des Haltebereichs von $\mathcal{M}$ ist, so heißt f *$\mathcal{R}$-berechenbar*.

$f : \mathbb{R}^\star \supset D \to \mathbb{R}^\star$ heißt analog δ-$\mathbb{Q}$-*berechenbar* bzw. δ-$\mathbb{Q}$-*analytisch*, falls es eine entsprechende δ-$\mathbb{Q}$-Maschine $\mathcal{M} := \mathcal{M}_{\pi,\rho}^{\delta\text{-}\mathbb{Q}}$ gibt. Man beachte, daß dabei sowohl das Programm π als auch die Rundungsfunktion ρ geeignet gewählt werden dürfen.

Beschränken wir uns auf robuste Programme, so heißt f *robust δ-$\mathbb{Q}$-berechenbar* bzw. *-analytisch*. □

Die Klasse der $\mathbb{Q}$-berechenbaren Funktionen entspricht gerade den mittels Turing-Maschinen berechenbaren Funktionen. Ein wichtiges Resultat aus [26, Abschnitt 4], das einen Vorläufer in [9] und seine Wurzeln bei RABIN hat, ist der

Satz 1.1 (Darstellungssatz). *Der Definitionsbereich einer $\mathcal{R}$-berechenbaren Funktion ist eine abzählbare Vereinigung semi-algebraischer Mengen, auf denen die Funktion stückweise rational ist.*

Beispiel 1.1. Dieses Beispiel soll verdeutlichen, daß die Klasse der $\mathbb{R}$-berechenbaren Funktionen sehr mächtig ist und auch Funktionen mit intuitiv nicht zu erwartenden Eigenschaften umfaßt. Wir geben – in Anlehnung an $x \sin \frac{1}{x}$ – eine stetig differenzierbare Funktion $f : [0, 1] \to [0, 1]$ an, deren Graph eine *unendliche Kurvenlänge* hat, und zeigen, daß diese $\mathbb{R}$-berechenbar und stückweise polynomiell vom Grad drei ist.

Ausgangspunkt der Konstruktion ist das Polynom $p(x) := -2x^3 + 3x^2$, welches eine bijektive, streng monoton wachsende Funktion $p : [0, 1] \to [0, 1]$ definiert mit $p'(0) = p'(1) = 0$. Daraus gewinnt man die Funktion

$$q : \mathbb{R} \to [0, 1],\ x \mapsto \begin{cases} p(x - \lfloor x \rfloor) & \text{falls } \lfloor x \rfloor \text{ gerade;} \\ p(1 - (x - \lfloor x \rfloor)) & \text{sonst.} \end{cases}$$

welche Wellen der Höhe eins und Periode zwei beschreibt und zweimal stetig differenzierbar ist, zudem gilt $\forall\, x \in \mathbb{Z} :\ q'(x) = 0$. Die Kurvenlänge jeder dieser Wellen ist mindestens zweimal ihre Höhe.

Nun definiert man für $k = 1, 2, \ldots$ Intervalle $I_k := [2^{-k}, 2^{-k+1}[$ und eine Funktion $f : [0, 1] \to [0, 1]$ durch

$$f|_{I_k}(x) := \frac{1}{k} \cdot q(\underbrace{k2^{k+1}(x - 2^{-k})}_{\in [0,2k[}),$$

d.h. im k. Intervall finden sich k Wellen der Höhe $\frac{1}{k}$ (vgl. auch Abb. 1.3). Man sieht leicht, daß f durch $f(0) := f(1) := 0$ zu einer auf dem abgeschlossenen Einheitsintervall stetigen und in dessen Inneren stetig differenzierbaren Funktion wird. Die Kurvenlänge über dem k. Intervall ist mindestens $k \cdot 2 \cdot \frac{1}{k} = 2$, daher ist die Gesamtlänge unbegrenzt.

Eine $\mathbb{R}$-Maschine zur Berechnung von $f(x)$ überprüft zuerst die beiden Sonderfälle $x = 0$ und $x = 1$; dann bestimmt sie das Intervall $I_k \ni x$ durch Aufzählen von $k = 1, 2, \ldots$ und berechnet $f|_{I_k}(x)$. □

Indem man bei obiger Kurve die Anzahl der Wellen im k. Intervall erhöht – z.B. auf 2^{2^k} –, läßt sich ein enormes Wachstum der Kurvenlänge über den Teilintervallen erzielen. Eine interessante Frage ist daher, ob $\mathbb{R}$-berechenbare Kurven bereits *fraktalen* Charakter besitzen können. In der Tat erscheint die genannte Variante unter dem einfachen Begriff der *box count dimension* [19] als flächenfüllend, da sich ein Wert von zwei ergibt. Verwendet man allerdings den wesentlich anspruchsvolleren Begriff der Hausdorff-Dimension, so kann dies nicht der Fall sein, wie folgender Satz zeigt.

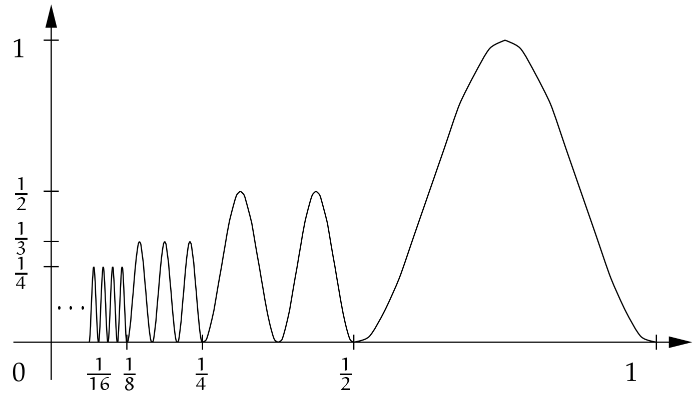


Abbildung 1.3.: $\mathbb{R}$-berechenbare Kurve unendlicher Länge

Satz 1.2. *Sei* $f : \mathbb{R}^m \supset D \to \mathbb{R}^n$ *$\mathbb{R}$-berechenbar und* $F := f(D)$ *das Bild von* f. *Dann ist die Hausdorff-Dimension* $\dim_H F \leq m$ *und ganzzahlig.*

Beweis. Gemäß dem Darstellungssatz 1.1 zerfällt der Definitionsbereich D in abzählbar viele semi-algebraische Mengen D_σ (indiziert mit dem zugehörigen Programmpfad), die o.B.d.A. zusammenhängend seien. In jeder dieser Regionen D_σ ist die Funktion rational und ohne Polstellen, also stetig differenzierbar, und hängt von $k \leq m$ Variablen ab; damit ist das Bild von $f|_{D_\sigma}$ eine k-dimensionale Submannigfaltigkeit des $\mathbb{R}^n$ und hat daher auch Hausdorff-Dimension k. Die Hausdorff-Dimension einer abzählbaren Vereinigung wiederum ergibt sich als Supremum der einzelnen Dimensionen, was die Aussage beweist. Für die verwendeten Eigenschaften von $\dim_H$ ziehe man z.B. [19][S. 29] zu Rate. □

In engem Zusammenhang mit den berechenbaren Funktionen steht der Begriff der entscheidbaren Menge, den wir nun in verschiedenen Varianten einführen. Die folgende Definition und das Lemma können auch auf Teilmengen von $\mathbb{R}^\star$ und δ-$\mathbb{Q}$-Maschinen angewandt werden.

Definition 1.5. Eine Menge $M \subset \mathcal{R}^\star$ heißt *(analytisch) $\mathcal{R}$-entscheidbar*, falls ihre charakteristische Funktion χ_M (analytisch) $\mathcal{R}$-berechenbar ist; sie heißt *(analytisch) $\mathcal{R}$-semi-entscheidbar*, falls sie Definitionsbereich einer (analytisch) $\mathcal{R}$-berechenbaren Funktion ist.

$$\chi_M : \mathcal{R}^\star \to \mathbb{B},\ x \mapsto \begin{cases} 1 & \text{falls } x \in M; \\ 0 & \text{sonst.} \end{cases}$$

□

Hieraus ergibt sich unmittelbar eine einfache Folgerung, deren Umkehrung nicht so einfach ist.

Lemma 1.3. *Jede $\mathcal{R}$-semi-entscheidbare Menge* $M \subset \mathcal{R}^*$ *ist analytisch $\mathcal{R}$-entscheidbar.*

Beweis. Sei $\mathcal{M}$ ein $\mathcal{R}$-Maschine mit Haltebereich M. Dann sei $\mathcal{M}'$ eine $\mathcal{R}$-Maschine, die ihre Ausgabe zuerst auf 0 setzt, sich dann bis auf eventuelle Ausgaben wie $\mathcal{M}$

verhält und im Falle, daß $\mathcal{M}$ hält, eine 1 ausgibt. Diese Maschine berechnet χ_M $\mathcal{R}$-analytisch und entscheidet damit M analytisch. □

Vermutung 1. Die Umkehrung von Lemma 1.3 gilt nicht.

Beispiel 1.2. Abschließend sollen noch einige Beispiele für verschiedene Varianten von Entscheidbarkeit folgen.

- Die natürlichen Zahlen sind nach Voraussetzung stets eine Teilmenge von $\mathcal{R}$ und daher stets $\mathcal{R}$-entscheidbar; hierbei wird die Anordnung des Rings benötigt. (*Hinweis:* Die $\mathcal{R}$-Maschine subtrahiert solange Eins von der Eingabe, bis ein Wert kleiner Null auftritt; die Eingabe war eine natürliche Zahl genau dann, wenn dieser Wert gleich -1 ist.)
- Ist $\mathcal{R}$ ein Körper, so ist auch $\mathbb{Q}$ Teilmenge von $\mathcal{R}$ und $\mathcal{R}$-semi-entscheidbar; hierbei wird keine Anordnung benötigt, sondern nur unsere Voraussetzung, daß $\mathbb{Z}$ einen Unterring von $\mathcal{R}$ bildet. (*Hinweis:* Die $\mathcal{R}$-Maschine zählt alle rationalen Zahlen auf und vergleicht mit der Eingabe.)
- Das Newton-Verfahren zur Bestimmung einer Nullstelle eines Polynoms konvergiert nicht für alle Startwerte; die Menge der in diesem Sinne „guten“ Startwerte bildet also eine analytisch semi-entscheidbare Menge.

□

1.2.1. Komposition von Funktionen

In diesem Abschnitt wollen wir in systematischer Weise erläutern, inwiefern verschiedene Funktionsklassen abgeschlossen sind unter Komposition, wobei insbesondere die Hintereinanderausführung interessiert. Dabei lernen wir den Hierarchiesatz kennen und sehen außerdem, wie abzählbar viele Rechnungen parallelisiert werden können. Wir bezeichnen im folgenden z.B. die Menge der $\mathbb{R}$-berechenbaren Funktionen in naheliegender Weise durch $\{\mathbb{R}\text{-berechenbar}\}$ usw.

$\mathcal{R}$-berechenbar

Die Menge der $\mathcal{R}$-berechenbaren Funktionen ist abgeschlossen unter kartesischem Produkt, Hintereinanderausführung und primitiver Rekursion. Diese und weitere Eigenschaften der $\mathcal{R}$-berechenbaren Funktionen folgen leicht aus der Theorie der Registermaschinen, die oft auch auf den Ring $\mathcal{R}$ angewendet werden kann. Die Ergebnisse gelten auch für δ-$\mathbb{Q}$-berechenbare Funktionen, selbst für nicht robuste, da alle Funktionswerte rational sind. Man kann die Bedingungen über die Anzahl der Argumente einer Funktion lockern, indem man z.B. ein Wort stets zusammen mit seiner Länge übergibt.

- Seien $D \subset \mathcal{R}^n$, $D' \subset \mathcal{R}^*$ und $f : D \to \mathcal{R}^*$ sowie $g : D' \to \mathcal{R}^*$ $\mathcal{R}$-berechenbar, so ist auch $f \times g : D \times D' \to \mathcal{R}^*$ $\mathcal{R}$-berechenbar mit $(f \times g)(x, y) = f(x) \cdot g(y)$ (hierbei ist $\cdot$ das Produkt im Wörtermonoid).

- Seien $D, D' \subset \mathcal{R}^*$ und $f : D \to D'$ sowie $g : D' \to \mathcal{R}^*$ $\mathcal{R}$-berechenbar, so ist auch $g \circ f : D \to \mathcal{R}^*$ $\mathcal{R}$-berechenbar mit $(g \circ f)(x) = g(f(x))$.
- Sei $D \subset \mathcal{R}^*$ und $g : D \to \mathcal{R}$ sowie $h : \mathbb{N} \times \mathcal{R} \times D \to \mathcal{R}$ $\mathcal{R}$-berechenbar, so ist auch $f : \mathbb{N} \times D \to \mathcal{R}$ $\mathcal{R}$-berechenbar mit

$$f(n, x) := \begin{cases} g(x) & \text{falls } n = 0; \\ h(n, f(n-1, x), x) & \text{sonst.} \end{cases}$$

$\mathbb{R}$-analytisch

Für $\mathbb{R}$-analytische Funktionen ist die Situation komplizierter. Diese sind zwar noch abgeschlossen unter dem kartesischen Produkt, aber nicht mehr unter der Hintereinanderausführung, wie der Satz 1.4 aus [44] zeigt.

Definition 1.6. Eine Funktion $f : \mathcal{R}^\star \rightsquigarrow \mathcal{R}^\star$ heißt $\mathcal{R}^i$*-analytisch* für $i \in \mathbb{N}$, falls sie durch Hintereinanderschaltung von i $\mathcal{R}$-Maschinen analytisch berechnet werden kann; d.h. falls sich f als Hintereinanderausführung von i $\mathcal{R}$-analytischen Funktionen beschreiben läßt. Analog werden δ-$\mathbb{Q}^i$-analytische Funktionen $f : \mathbb{R}^\star \rightsquigarrow \mathbb{R}^\star$ definiert. □

Satz 1.4 (Hierarchiesatz). $\forall\, i \in \mathbb{N} :$ $\{\mathbb{R}^i$*-analytisch*$\} \subsetneq \{\mathbb{R}^{i+1}$*-analytisch*$\}$.

Die beinahe stärkste Form von Hintereinanderausführung, die man nun innerhalb der Klasse der $\mathbb{R}$-analytischen Funktionen noch erreichen kann, beschreibt das

Lemma 1.5. *Seien* $D, D' \subset \mathbb{R}^*$, $f : \mathbb{R}^* \rightsquigarrow D$ $\mathbb{R}$*-berechenbar,* $g : D \to D'$ $\mathbb{R}$*-analytisch sowie* $h : D' \to \mathbb{R}^*$ $\mathbb{R}$*-berechenbar und stetig. Dann ist* $h \circ g \circ f : \mathbb{R}^* \rightsquigarrow \mathbb{R}^*$ $\mathbb{R}$*-analytisch.*

Beweis. Gegeben seien drei $\mathbb{R}$-Maschinen $\mathcal{M}_f, \mathcal{M}_g, \mathcal{M}_h$ für die jeweiligen Funktionen. Die Konstruktion einer $\mathbb{R}$-Maschine $\mathcal{M}$ für $h \circ g \circ f$ ist nicht weiter schwierig. Diese simuliert im wesentlichen $\mathcal{M}_g$; sobald $\mathcal{M}_g$ auf ein Eingabefeld zugreift, wird der nötige Wert mittels $\mathcal{M}_f$ berechnet. Bei einem `print`-Befehl wird das Ergebnis erst noch von $\mathcal{M}_h$ verarbeitet und dann ausgegeben. Die Rechnung konvergiert gegen das gewünschte Ergebnis, weil die Maschinen $\mathcal{M}_f$ und $\mathcal{M}_h$ nur endlich lange rechnen und h stetig ist. □

Daraus ergibt sich unmittelbar, daß die primitive Rekursion nach obigem Muster mit $\mathbb{R}$-analytischer Induktionsverankerung g möglich ist, sofern der Induktionsschritt h $\mathbb{R}$-berechenbar und stetig ist.

Anmerkung 1.5. Das Lemma 1.5 läßt sich zumindest noch auf solche stetigen $\mathbb{R}$-analytischen Funktionen h erweitern, für die eine Maschine $\mathcal{M}_h$ existiert mit folgender Eigenschaft: Sei $h_n = \Phi^{(n)}_{\mathcal{M}_h}$ die n. Approximation von h, wie sie $\mathcal{M}_h$ berechnet; dann müssen die h_n gleichmäßig gegen h konvergieren (für $n \to \infty$). Dies gilt z.B., falls die h_n die Partialsummen einer Potenzreihe sind.

Zum Abschluß wollen wir noch zeigen, wie abzählbar viele Rechnungen gleichzeitig von einer Maschine ausgeführt werden können; dazu bedienen wir uns einer unendlichen Ausgabe.

Lemma 1.6. *Sei* $D \subset \mathbb{R}^\star$ *und* $f : \mathbb{N} \times D \to \mathbb{R}$ *eine* $\mathbb{R}$*-analytische Funktion, dann ist auch* $f_\infty : D \to \mathbb{R}^\omega$ $\mathbb{R}$*-analytisch, wobei* $f_\infty(x) = (f(0,x), f(1,x), \dots)$ *für* $x \in D$.

Beweis. Da es berechenbare Bijektionen $\kappa : \mathbb{N}^2 \to \mathbb{N}$ gibt, kann man die verschiedenen Hauptspeicher- und Registerinhalte der Maschine $\mathcal{M}_f$ für alle Eingaben i im Hauptspeicher einer Maschine $\mathcal{M}$ ablegen; ferner kann man den j. Rechenschritt von $\mathcal{M}_f$ für Eingabe i zum Zeitpunkt $\kappa(i,j)$ auf Maschine $\mathcal{M}$ simulieren. Dann ist es leicht, die Ausgaben $f(i,x)$ auf die Bandzelle $i+1$ umzulenken und die Ausgabelänge gegen ∞ konvergieren zu lassen. $\square$

δ-$\mathbb{Q}$-analytisch

Der Hierarchiesatz 1.4 gilt auch für (robust) δ-$\mathbb{Q}$-analytische Funktionen und deren Hintereinanderausführung, d.h.

$$\forall\, i \in \mathbb{N} : \{\text{(robust) } \delta\text{-}\mathbb{Q}^i\text{-analytisch}\} \subsetneq \{\text{(robust) } \delta\text{-}\mathbb{Q}^{i+1}\text{-analytisch}\}.$$

Den Beweis für robust δ-$\mathbb{Q}$-analytische Funktionen findet man weitgehend in [44] und den allgemeinen für δ-$\mathbb{Q}$-analytische Funktionen als Satz 1.23. Das Lemma 1.5 hat ebenfalls eine entsprechende Formulierung, die aber etwas komplizierter ist.

Lemma 1.7. *Seien* $D, D' \subset \mathbb{R}^*$, $f : \mathbb{R}^* \rightsquigarrow D$ δ-$\mathbb{Q}$*-berechenbar,* $g : D \to D'$ δ-$\mathbb{Q}$*-analytisch sowie* $h : D' \to \mathbb{R}^*$ *robust* δ-$\mathbb{Q}$*-berechenbar und stetig. Dann ist* $h \circ g \circ f : \mathbb{R}^* \rightsquigarrow \mathbb{R}^*$ δ-$\mathbb{Q}$*-analytisch. Ist* f *robust, so auch* $h \circ g \circ f$.

Beweis. Man beachte, daß die Ausgabe von $\mathcal{M}_f$ rational ist, daher kann die für g nötige Rundungsfunktion von einer $\mathbb{Q}$-Maschine berechnet werden. $\square$

Lemma 1.8. *Sei* $D \subset \mathbb{R}^*$ *und* $f : \mathbb{N} \times D \to \mathbb{R}$ *eine (robust)* δ-$\mathbb{Q}$*-analytische Funktion, dann ist auch* $f_\infty : D \to \mathbb{R}^\omega$ *(robust)* δ-$\mathbb{Q}$*-analytisch, wobei* $f_\infty(x) = (f(0,x), f(1,x), \dots)$ *für* $x \in D$.

Beweis. Wie im Beweis von Lemma 1.6 verwenden wir berechenbare Bijektionen $\kappa : \mathbb{N}^2 \to \mathbb{N}$, um die verschiedenen Hauptspeicher- und Registerinhalte der Maschine $\mathcal{M}_f$ für mehrere Eingaben i im Hauptspeicher einer Maschine $\mathcal{M}$ abzulegen. Allerdings muß man nun beachten, daß δ-$\mathbb{Q}$-analytische Berechnungen ihrer Natur nach in Runden ablaufen, nach denen jeweils die Präzision δ erhöht wird. Daher werden auch die Rundungen der Eingabe x zur aktuellen Präzision im Hauptspeicher abgelegt und aufbewahrt. Die Maschine $\mathcal{M}$ simuliert dann in Runde δ mehrere Kopien von $\mathcal{M}_f$ jeweils mit Präzision $\delta - i$ – unter Verwendung der aufbewahrten Rundungen – für die Eingaben (i,x) mit $i = 0, \dots, \delta$ und arrangiert die Ausgaben geeignet. $\square$

Anmerkung 1.6. Eine Erweiterung des obigen Resultats auf $D \subset \mathbb{R}^\star$ ist möglich. Von einer unendlich langen Eingabe kann die Maschine $\mathcal{M}$ aber nicht mehr für jede Präzision eine Rundung im Hauptspeicher ablegen. Für robuste Programme stellt sich das Problem nicht, aber im allgemeinen δ-$\mathbb{Q}$-analytischen Fall muß es möglich sein, zu jedem Zeitpunkt $\rho(x_i, n)$ für $n \le \delta$ zu bestimmen. Mit einer Technik analog zu Lemma 1.11 kann eine Rundungsfunktion ρ' gefunden werden, die alle Resultate von $\rho(x_i, n)$ für $0 \le n \le \delta$ in einer geeigneten Kodierung liefert.

Korollar 1.9. Die Menge der robust δ-$\mathbb{Q}$-analytischen Funktionen ist nicht abgeschlossen unter Hintereinanderausführung.

Beweis. Die Vorzeichenfunktion

$$\mathtt{sign} : \mathbb{R} \to \{-1, 0, +1\},\ x \mapsto \begin{cases} -1 & \text{falls } x < 0; \\ 0 & \text{falls } x = 0; \\ +1 & \text{falls } x > 0. \end{cases}$$

und die „kurze Bartfunktion"

$$d_1 : \mathbb{R} \to \mathbb{Q},\ x \mapsto \begin{cases} 1/q & \text{falls } \exists\, p \in \mathbb{Z}, q \in \mathbb{N} \text{ teilerfremd}:\ x = p/q; \\ 0 & \text{sonst.} \end{cases}$$

sind robust δ-$\mathbb{Q}$-analytisch (siehe [26]), aber $\chi_{\mathbb{Q}} = \mathtt{sign} \circ d_1$ ist es nicht (siehe Satz 1.14). □

1.3. Der Einfluß der Rundung und seine Grenzen

Es erweist sich leider, daß die Menge der mittels δ-$\mathbb{Q}$-Maschinen (analytisch) berechenbaren Funktionen stark von den erlaubten Rundungen abhängt. Wir werden an Beispielen sehen, daß nicht jede δ-$\mathbb{Q}$-analytische Funktion auch robust δ-$\mathbb{Q}$-analytisch ist[1] und daß nicht jede $\mathbb{R}$-analytische Funktion auch δ-$\mathbb{Q}$-analytisch ist. Andererseits ist jede $\mathbb{R}$-analytische Funktion von nur einer Variablen bereits δ-$\mathbb{Q}$-analytisch, wobei lediglich die Rundungsfunktion geeignet zu wählen ist.

Die Macht der Rundung

Man kann eine einstellige $\mathbb{R}$-analytische Funktion durch eine einfache Konstruktion δ-$\mathbb{Q}$-analytisch berechnen, indem eine spezielle Rundungsfunktion die eigentliche Arbeit übernimmt. Wir werden später sehen, daß dies mit mehrstelligen Funktionen nicht mehr funktioniert.

Satz 1.10. *Es gibt ein universelles Programm π mit folgender Eigenschaft. Sei $f : \mathbb{R} \to \mathbb{R}$ eine beliebige analytisch $\mathbb{R}$-berechenbare Funktion. Dann existiert eine $\mathbb{R}$-berechenbare Rundungsfunktion ρ_f, so daß die δ-$\mathbb{Q}$-Maschine $\mathcal{M}^{\delta\text{-}\mathbb{Q}}_{\pi,\rho_f}$ die gewünschte Funktion f analytisch berechnet.*

[1] Eine ähnliche Aussage (mit zweifelhaftem Beweis) findet sich in [26]

Anmerkung 1.7. Nach Beispiel 1.2 ist $\mathbb{Q}$ $\mathbb{R}$-semi-entscheidbar; daher ist wegen Lemma 1.3 $\chi_{\mathbb{Q}} : \mathbb{R} \to \mathbb{B}$ $\mathbb{R}$-analytisch und gemäß obigem Satz auch δ-$\mathbb{Q}$-analytisch.

Zum Beweis des Satzes beobachtet man zunächst, das im Ergebnis der Rundungsfunktion eine beliebige natürliche Zahl kodiert werden kann, und zwar durch eine $\mathbb{R}$-Maschine. Eine δ-$\mathbb{Q}$-Maschine kann diese Information später dekodieren und nutzen. Es ist offensichtlich, daß in einer rationalen Zahl nicht mehr Information untergebracht werden kann als eine unbegrenzte, aber endliche Zahl von Bits.

Lemma 1.11. *Es gibt eine $\mathbb{R}$-berechenbare Abbildung $\mathbb{R} \times \mathbb{N} \times \mathbb{N} \to \mathbb{Q}$, $(x, n, k) \mapsto x_n^{(k)}$, deren Umkehrabbildung $x_n^{(k)} \mapsto k$ $\mathbb{Q}$-berechenbar ist, wobei stets*

$$|x - x_n^{(k)}| < 2^{-n}.$$

Beweis. Die Konstruktion der Abbildung beruht darauf, daß man x durch eine rationale Zahl mit ganz speziellem Nenner approximieren kann; d.h. formal:

$$\forall\,(x, n, k) \in \mathbb{R} \times \mathbb{N} \times \mathbb{N}\; \exists\, p, q \in 2\mathbb{Z} + 1 \text{ teilerfremd}: \; |x - \frac{p}{2^k q}| < 2^{-n}.$$

Dazu wähle man $q' := 3^{n+1} > 2^n$. Dann gilt

$$\frac{1}{2^k q'} \leq \frac{1}{q'} < 2^{-n}$$

und die offene Umgebung $U_{2^{-n}}(x)$ um x vom Radius 2^{-n} enthält mindestens zwei Punkte der Form $\frac{i}{2^k q'}$ für aufeinanderfolgende i. Wähle

$$p' := \min\{i \in 2\mathbb{Z} + 1 \mid \frac{i}{2^k q'} \in U_{2^{-n}}(x)\},$$

kürze p'/q' zu p/q und habe fertig.

Damit kann $x_n^{(k)} := \frac{p}{2^k q}$ gewählt werden, und die Abbildung ist sicher $\mathbb{R}$-berechenbar. Da p und q ungerade und teilerfremd sind, ist die Darstellung auch eindeutig. Durch systematisches Aufzählen aller Tripel (p, q, k) kann k in endlicher Zeit von einer $\mathbb{Q}$-Maschine gefunden werden. $\square$

Es gibt bekanntlich Kodierungen $\kappa : \mathbb{Q} \to \mathbb{N}$, so daß κ und κ^{-1} $\mathbb{Q}$-berechenbar sind. Also gibt es eine δ-$\mathbb{Q}$-Maschine $\mathcal{M}$, die zu gegebenem $x_n^{(k)}$ wie zuvor k bestimmt und dann $\kappa^{-1}(k)$ ausgibt. Die Kodierung κ sei von nun an fest. Nun muß man nur noch f rational approximieren.

Lemma 1.12. *Zu der Funktion f von oben gibt es eine $\mathbb{R}$-Maschine $\mathcal{M}'_f$, die bei Eingabe von $(x, n) \in \mathbb{R} \times \mathbb{N}$ eine Ausgabe $y'_n \in \mathbb{Q}$ erzeugt mit $\lim_{n\to\infty} y'_n = f(x)$.*

Beweis. $\mathcal{M}'_f$ bestimmt durch Simulation die n. Approximation $y_n \in \mathbb{R}$ des Resultats von $\mathcal{M}_f$, der $\mathbb{R}$-Maschine, die f berechnet. Dann bestimmt $\mathcal{M}'_f$ ein $y'_n \in \mathbb{Q}$ mit $|y_n - y'_n| < 2^{-n}$, z.B. durch systematisches Aufzählen von $\kappa^{-1}(m)$ für $m = 0, 1, 2, \ldots$ Wegen $\lim_{n\to\infty} y_n = f(x)$ gilt auch $\lim_{n\to\infty} y'_n = f(x)$. $\square$

Insgesamt ergibt sich damit der

Beweis von Satz 1.10. Die universelle Maschine $\mathcal{M}$ ist wie zuvor beschrieben. Die Rundung ρ_f ergibt sich, indem man zuerst wie in Lemma 1.12 geschildert y'_n berechnet und dann $x_n^{(k)}$ ausgibt mit $k := \kappa(y'_n)$. □

Korollar 1.13. Es gibt ein universelles Programm π mit folgender Eigenschaft. Sei $f : \mathbb{R} \to \mathbb{R}^\star$ eine beliebige, von der Maschine $\mathcal{M}_f$ analytisch $\mathbb{R}$-berechenbare Funktion. Dann existiert eine $\mathbb{R}$-berechenbare Rundungsfunktion ρ_f, so daß die δ-$\mathbb{Q}$-Maschine $\mathcal{M}^{\delta\text{-}\mathbb{Q}}_{\pi,\rho_f}$ die gewünschte Funktion f analytisch berechnet.

Beweis. Es ist der Rundungsfunktion ρ_f leicht möglich, ein n-Tupel rationaler Zahlen inklusive der Längenangabe n an das Programm π zu übergeben. □

Robustheit ist eine Einschränkung

Wir haben bisher gezeigt, wie sehr die freie Wahl einer geeigneten Rundungsfunktion hilft. Nun soll – wie schon angekündigt – gezeigt werden, daß manche δ-$\mathbb{Q}$-analytische Funktion nicht ohne geeignete Rundung berechenbar ist.

Satz 1.14. *Die charakteristische Funktion der rationalen Zahlen $\chi_\mathbb{Q} : \mathbb{R} \to \mathbb{B}$ ist nicht robust δ-$\mathbb{Q}$-analytisch berechenbar.*

Beweis. Wir betrachten eine spezielle Rundungsfunktion ρ und zeigen, daß kein Programm mit ihrer Hilfe $\chi_\mathbb{Q}$ berechnen kann; daher gibt es keine robusten Programme für $\chi_\mathbb{Q}$. Die Rundung $\rho(x, n)$ entspricht dem Abschneiden der Binärdarstellung[2] von x nach der n. Nachkommastelle, d.h. formal

$$\rho : \mathbb{R} \times \mathbb{N} \to \mathbb{Q},\ (x, n) \mapsto \begin{cases} \lfloor x2^n \rfloor 2^{-n} & \text{falls } x \geq 0; \\ -\rho(-x, n) & \text{sonst.} \end{cases}$$

Offensichtlich ist ρ eine $\mathbb{R}$-berechenbare Rundungsfunktion und $x - 2^{-n} < \rho(x, n) \leq x$ für positive x. Nun nehmen wir an, π sei ein Programm mit der Eigenschaft, daß $\mathcal{M} := \mathcal{M}^{\delta\text{-}\mathbb{Q}}_{\pi,\rho}$ die gewünschte Funktion $\chi_\mathbb{Q}$ berechnet, und führen dies zum Widerspruch. Zu diesem Zweck konstruieren wir eine Folge $(x_i)_{i=0}^\infty$ abwechselnd rationaler und irrationaler Zahlen, bei deren Grenzwert die Ausgabe der Maschine $\mathcal{M}$ divergiert. Die Folge beginnt mit $x_0 := 0 \in \mathbb{Q}$.

Nun betrachten wir die Ausgaben im Verlauf der Berechnung $k(x_0) = (k_i)_{i=0}^\infty$ von $\mathcal{M}$ angesetzt auf die Eingabe x_0. Nach Voraussetzung liegen diese irgendwann in einer offenen Umgebung $U_{1/3}(1)$; daher sei $i_0 := \min\{i \in \mathbb{N} \mid \mathrm{out}(k_i) \in U_{1/3}(1)\}$ und δ_0 der Inhalt des Präzisionsregister zum Zeitpunkt i_0.

Wir konstruieren nun eine irrationale Zahl x_1, die sich in den ersten δ_0 Nachkommastellen ihrer Binärdarstellung nicht von x_0 unterscheidet. Daher verläuft

[2] Unter der *Binärdarstellung* einer positiven Zahl $x \in \mathbb{R}$ wollen wir eine Folge $b_n \ldots b_0.b_{-1}b_{-2}\ldots$ verstehen mit $b_i \in \mathbb{B}$ und $x = \sum_{i \leq n} b_i 2^i$. Diese ist eindeutig, falls man $b_i \to 1$ für $i \to -\infty$ ausschließt.

auch die Berechnung $k(x_1)$ bis zum Schritt i_0 genauso wie $k(x_0)$; die Konfigurationen unterscheiden sich lediglich durch den Inhalt des Eingabebands. Es sei $x_1 := (\lfloor x_0 2^{\delta_0} \rfloor + y)2^{-\delta_0} \in \mathbb{R} \setminus \mathbb{Q}$, wobei $y := \sum_{i=0}^{\infty} 2^{-2^i}$ irrational weil nichtperiodisch ist.

Die Ausgaben der Berechnung $k(x_1)$ liegen irgendwann in $U_{1/3}(0)$ und dies definiert analog zu oben $i_1 > i_0$ (dies muß man fordern) und $\delta_1 \geq \delta_0$ (dies ergibt sich dann). Die rationale Zahl $x_2 := \rho(x_1, \delta_1)$ schließt den Kreis unserer Konstruktion. Man beobachtet also, daß die Ausgabe von $\mathcal{M}$ angesetzt auf x_i i Male abwechselnd in die Nähe von Eins und Null kommt. Nun muß man nur noch zeigen, daß der Grenzwert der Folge existiert und in den ersten δ_i Nachkommastellen mit x_i übereinstimmt; dann oszilliert die Ausgabe von $\mathcal{M}$ zwischen Eins und Null und divergiert daher.

Die Teilfolge $(x_{2i})_{i=0}^{\infty}$ ist monoton wachsend und nach oben durch 1 beschränkt, also konvergent. Ferner gilt $|x_{2i+1} - x_{2i}| < 2^{-\delta_{2i}}$, da $0 < y < 1$; daher existiert $x := \lim_{i\to\infty} x_i$. Die Binärdarstellung von x ergibt sich als (punktweiser) Grenzwert der Binärdarstellungen der x_i und führt daher zu einer Berechnung $k(x) = (k_i)_{i=0}^{\infty}$, die in den ersten i_j Schritten jeweils mit $k(x_j)$ übereinstimmt. Die Teilfolge $(\mathrm{out}(k_{i_j}))_{j=0}^{\infty}$ ist also abwechselnd kleiner als $1/3$ und größer als $2/3$ und divergiert daher. □

Grenzen der Rundung

Wir haben gesehen, daß jede $\mathbb{R}$-analytische Funktion in einer Variablen im wesentlichen durch eine geeignete Rundungsfunktion berechnet werden kann. Daß diese Technik bei $\mathbb{R}$-analytischen Funktionen von mehreren Variablen im allgemeinen versagen muß, zeigen wir an folgender Beispielfunktion, die offensichtlich $\mathbb{R}$-analytisch ist.

$$\chi_{\mathbb{Q}}^{+} : \mathbb{R}^2 \to \mathbb{B},\ \chi_{\mathbb{Q}}^{+}(x, y) := \chi_{\mathbb{Q}}(x + y)$$

Dies ist die charakteristische Funktion der Menge $Q_2 := \{(x, y) \in \mathbb{R}^2 \mid x + y \in \mathbb{Q}\}$, welche $\mathbb{R}$-semi-entscheidbar ist.

Satz 1.15. *Die Menge Q_2 ist nicht δ-$\mathbb{Q}$-semi-entscheidbar.*

Um solche allgemeinen Fragen nach der Berechenbarkeit einer Funktion durch eine δ-$\mathbb{Q}$-Maschine mit beliebigem Programm und beliebiger Rundungsfunktion zu beantworten, benötigen wir eine mächtige Charakterisierung dieses Maschinenmodells. Eine genaue Betrachtung des Zusammenspiels zwischen $\mathbb{R}$-berechenbarer Rundung und dem Programm der δ-$\mathbb{Q}$-Maschine führt zu folgendem Modell.

Definition 1.7. Ein (n, k)-dimensionaler *algebraischer Entscheidungsbaum $\mathcal{B}$* mit $n, k \in \mathbb{N}$ ist ein Vektor $(\alpha_1, \ldots, \alpha_k) \in \mathbb{R}^k$ zusammen mit einem eventuell unendlichen binären Baum. Dessen innere Knoten sind beschriftet mit einem rationalen Polynom $p \in \mathbb{Q}[X_0, \ldots, X_k]$ und einem Index $i \in [1 : n]$; sie haben genau zwei geordnete ausgehende Kanten, die mit Elementen aus $\mathbb{B}$ beschriftet sind.

Die *Berechnung* von $\mathcal{B}$ für eine Eingabe $(x_1, \ldots, x_n) \in \mathbb{R}^n$ ist der maximal lange, eventuell unendliche Pfad σ, der in der Wurzel beginnt und bei inneren Knoten mit

Beschriftung (p, i) nach links verzweigt, falls $p(x_i, \alpha_1, \ldots, \alpha_k) > 0$, und nach rechts sonst. Der *Definitionsbereich* D_σ von σ ist die Teilmenge der Eingaben, für die die Berechnung diesem Pfad folgt; analog für Präfixe von σ. Das *Ergebnis* $\mathrm{out}(\sigma)$ einer solchen unendlichen Berechnung ist der Grenzwert – sofern er existiert – der Folge von Kantenbeschriftungen entlang des Pfades; für endliche Berechnungen und Präfixe sei das Ergebnis gleich der letzten Kantenbeschriftung. Auf diese Weise wird eine partielle Funktion $\Phi_{\mathcal{B}} : \mathbb{R}^n \rightsquigarrow \mathbb{B}$ bestimmt und der Begriff der entscheidbaren Menge läßt sich in naheliegender Weise übertragen. □

Wir werden nun zeigen, daß man die Berechnung einer δ-$\mathbb{Q}$-Maschine exakt angeben kann, auch wenn man Informationen über die reellen Eingaben nur durch Fragen der Form „$p(x_i, \alpha_1, \ldots, \alpha_k) > 0$?“ erhalten kann. Damit ist ein algebraischer Entscheidungsbaum eine Verallgemeinerung des Modells der δ-$\mathbb{Q}$-analytischen Berechenbarkeit, zumindest im Hinblick auf Entscheidungsprobleme.

Sei $\mathcal{M}^\rho := \mathcal{M}_\pi^{\mathbb{R}}$ eine $\mathbb{R}$-Maschine, die eine Rundungsfunktion $\rho : \mathbb{R} \times \mathbb{N} \to \mathbb{Q}$ berechnet. Ruft man $\mathcal{M}^\rho$ für eine feste Präzision n und eine variable Eingabe x_i auf, so läßt sich der Berechnungsbaum auf das Wesentliche reduzieren, nämlich auf die bedingten Sprünge `if` $\alpha > 0$ `then ...` , die von x_i abhängen. Der aktuelle Inhalt des Akkumulators läßt sich zu jedem Zeitpunkt als rationale Funktion der Eingabe x_i und der reellen Programmkonstanten darstellen; sein Vorzeichen ist daher gleich dem des Produktpolynoms aus Zähler und Nenner. Ferner bemerkt man, daß $\mathcal{M}^\rho$ für festes n den Definitionsbereich $\mathbb{R}$ von x_i disjunkt in abzählbar viele einzelne Punkte und offene Intervalle zerlegt, auf denen die Funktionswerte jeweils konstant (weil rational) sind; dies ergibt sich unmittelbar aus dem Darstellungssatz 1.1. Folglich legen die oben beschriebenen Vergleiche das Ergebnis $\rho(x_i, n)$ bereits eindeutig fest.

Sei nun $\mathcal{M} := \mathcal{M}_{\pi',\rho}^{\delta\text{-}\mathbb{Q}}$ eine δ-$\mathbb{Q}$-Maschine, die ρ benutzt und höchstens die ersten $n \in \mathbb{N}$ Elemente der Eingabefolge liest[3]; sie hat nur indirekt, mittels der Rundung, Zugriff auf jeweils eine einzelne Eingabevariable. Die Quintessenz des Berechnungsbaumes von $\mathcal{M}$ sind daher die obigen Verzweigungen von $\mathcal{M}^\rho$; deren Folge legt das Ergebnis von ρ und damit den Zustand von $\mathcal{M}$ fest, aus dem sich wiederum ergibt, ob die Berechnung terminiert oder $\mathcal{M}^\rho$ mit neuen Werten für Präzision und Index aufgerufen wird. Ferner ergibt sich daraus die aktuelle Ausgabe von $\mathcal{M}$, die als Kantenbeschriftung im Baum dient. Die Berechnungen von $\mathcal{M}$ bestimmen damit einen algebraischen Entscheidungsbaum, wobei $\alpha_1, \ldots, \alpha_k$ die reellen Programmkonstanten von π sind. Der Definitionsbereich eines Pfades im Baum ist stets ein kartesisches Produkt, da alle Verzweigungen nur je eine Eingabekomponente berücksichtigen können.

Beweis von Satz 1.15. Der Satz ist einerseits eine Folgerung der stärkeren Aussage 1.16, andererseits ist sein direkter Beweis lehrreich und einfach. Wir führen die Annahme, daß Q_2 δ-$\mathbb{Q}$-semi-entscheidbar ist, zum Widerspruch. Dazu sei $\mathcal{M}^\rho := \mathcal{M}_\pi^{\mathbb{R}}$ eine $\mathbb{R}$-Maschine, die eine Rundungsfunktion $\rho : \mathbb{R} \times \mathbb{N} \to \mathbb{Q}$ berechnet, und $\mathcal{M} := \mathcal{M}_{\pi',\rho}^{\delta\text{-}\mathbb{Q}}$ eine δ-$\mathbb{Q}$-Maschine mit Haltebereich Q_2. O.B.d.A. kann man annehmen, daß $\mathcal{M}$ während

[3] Den unbeschränkten Fall kann man analog behandeln, er spielt hier aber keine Rolle.

der Rechnung Null und nur beim Terminieren Eins ausgibt und auf keine anderen Eingaben außer x_1 und x_2 zugreift. Die Berechnungen von $\mathcal{M}$ in Verbindung mit $\mathcal{M}^\rho$ bestimmen wie oben geschildert einen (unendlichen) algebraischen Entscheidungsbaum $\mathcal{B}$.

Für einen akzeptierenden Pfad im Entscheidungsbaum ist sein Definitionsbereich $D = D_x \times D_y$ notwendig abzählbar; sonst gäbe es z.B. $x_1, x_2 \in D_x$ mit $x_1 - x_2 \notin \mathbb{Q}$ und somit $\forall\, y \in D_y :\ (x_1 + y \in \mathbb{Q} \Rightarrow x_2 + y \notin \mathbb{Q})$ im Widerspruch zur Korrektheit von $\mathcal{M}$.

Nun ist aber Q_2 sicher überabzählbar und kann nicht von den abzählbaren Definitionsbereichen der abzählbar vielen akzeptierenden und damit endlichen Berechnungspfade abgedeckt werden. □

Satz 1.16. *Die Menge Q_2 ist nicht analytisch δ-$\mathbb{Q}$-entscheidbar.*

Beweis. Wir zeigen, daß es keinen $(2, k)$-dimensionalen algebraischen Entscheidungsbaum $\mathcal{B}$ gibt, der Q_2 (analytisch) entscheidet, d.h. für den gilt

$$\Phi_{\mathcal{B}} : \mathbb{R}^2 \to \mathbb{B},\ (x, y) \mapsto \begin{cases} 1 & \text{falls } x + y \in \mathbb{Q}; \\ 0 & \text{sonst.} \end{cases}$$

Dann kann es auch keine δ-$\mathbb{Q}$-Maschine geben – nicht einmal mit geeigneter Rundung –, die χ_{Q_2} analytisch berechnet. Dazu führen wir die Annahme, daß es einen solchen Baum $\mathcal{B}$ gibt, zum Widerspruch; wir beschränken uns dabei auf den Fall $k = 0$. Falls dem algebraischen Entscheidungsbaum reelle Konstanten zur Verfügung stehen, ändert sich der Beweis nicht wesentlich; man muß lediglich $\mathbb{Q}$ jeweils durch den Erweiterungskörper $\mathbb{Q}(\alpha_1, \dots, \alpha_k)$ ersetzen und (nicht-)algebraische Zahlen bezüglich dieses Körpers betrachten.

Entscheidend für den Beweis ist ein Verständnis der Struktur des Definitionsbereichs $D_\sigma = D_x \times D_y$ eines endlichen Pfades σ. Betrachten wir dazu zuerst ein Polynom $p \in \mathbb{Q}[X]$; seine reellen Nullstellen sind endlich viele, isolierte, algebraische Zahlen. Die Menge $M := \{x \in \mathbb{R} \mid p(x) > 0\}$ ist demnach die endliche Vereinigung offener Intervalle mit algebraischen Grenzen; das Komplement $\mathbb{R} \setminus M$ enthält zusätzlich einzelne algebraische Zahlen. D_x und D_y sind nun jeweils ein endlicher Schnitt solcher Mengen und haben damit dieselbe Struktur: $\bigcup_i\,]a_i, b_i[\ \cup \bigcup_j \{c_j\}$; wobei die Grenzen a_i und b_i der offenen Intervalle und die einzelnen c_j algebraische Zahlen sind. Das kartesische Produkt zweier solcher Mengen faßt man nun zweckmäßig auf als endliche Vereinigung folgender Strukturen, wobei die Koordinaten aller dabei auftretender Eckpunkte algebraische Zahlen sind:

- offene Rechtecke, entstanden als Produkt von Intervallen;
- an den Enden offene Liniensegmente, entstanden als Produkt eines Intervalls mit einer Zahl;
- einzelne Punkte, entstanden als Produkt von zwei Zahlen.

Für zwei Pfade σ und τ von $\mathcal{B}$ gelte $\sigma \prec \tau$ genau dann, wenn σ ein Präfix von τ ist; dies definiert eine Relation. Wir werden nun – analog zum Beweis von Satz 1.14 – mittels einer konvergenten Folge ein Zahlenpaar konstruieren, bei dessen Eingabe die Berechnung von $\mathcal{B}$ eine divergente Ausgabe ergibt. Dies steht dann im Widerspruch dazu, daß $\Phi_{\mathcal{B}} = \chi_{Q_2}$ als charakteristische Funktion total ist. Die Folgenglieder haben alle die Form $(x_i, y_i) := (e + \varepsilon_i, -e)$, wobei die Eulersche Zahl e nach einem Satz von HERMITE (1873) transzendent über $\mathbb{Q}$ ist[4]. Es gilt also stets $x_i + y_i = \varepsilon_i$, und genau die ε_i mit geradem Index werden dabei in $\mathbb{Q}$ liegen; es geht los mit $\varepsilon_0 := 0$. Die Berechnung von $\mathcal{B}$ angesetzt auf (x_i, y_i) sei τ_i; für die Ausgabe gilt

$$\mathrm{out}(\tau_i) = \begin{cases} 1 & \text{falls } i \text{ gerade;} \\ 0 & \text{sonst.} \end{cases}$$

Sei $\sigma_0 \prec \tau_0$ minimal mit $\mathrm{out}(\sigma_0) = 1$. Wegen der Transzendenz muß der Punkt (x_0, y_0) in einem offenen Rechteck $R_0 \subset D_{\sigma_0}$ enthalten sein. Demnach gibt es ein irrationales ε_1, so daß $(x_1, y_1) \in R_0$ und x_1 transzendent ist[5]. Nun sei $\sigma_0 \prec \sigma_1 \prec \tau_1$ minimal mit $\mathrm{out}(\sigma_1) = 0$. Wiederum ist (x_1, y_1) in einem offenen Rechteck $R_1 \subset D_{\sigma_1}$ enthalten und es gibt nun ein rationales ε_2, so daß $(x_2, y_2) \in R_1$; damit ist auch x_2 transzendent.

Nach diesem Schema wird die Konstruktion nun fortgesetzt; dabei entsteht eine absteigende Folge von Rechtecken $R_0 \supseteq R_1 \supseteq \ldots$ und eine Folge immer längerer Pfade $\sigma_0 \prec \sigma_1 \prec \ldots$ Wählt man nun noch eine monoton wachsende Folge $0 < \varepsilon_0 < \varepsilon_1 < \ldots < 1$, so existiert $\varepsilon := \lim_{i \to \infty} \varepsilon_i$. Für die Berechnung τ von $\mathcal{B}$ angesetzt auf $(e + \varepsilon, -e)$ soll nun gelten $\forall\, i \in \mathbb{N} : \ \sigma_i \prec \tau$; dann konvergiert die Folge der Kantenbeschriftungen von τ nicht.

Die durch den Grenzwert erzeugte Eingabe hat die gewünschte Eigenschaft, falls $\forall\, i \in \mathbb{N} : \ (e + \varepsilon, -e) \in R_i$. Dies muß aber nicht der Fall sein, falls die Folge $(r_i)_{i=0}^{\infty}$ der rechten Ränder der R_i stationär wird und der Grenzwert genau auf diesem Rand zu liegen kommt. Man kann dies vermeiden, indem man in der obigen Konstruktion die Rechtecke R_i jeweils durch Teilrechtecke $R'_i \subset R_i$ ersetzt, deren rechte Ränder r'_i streng monoton fallen. Dann gilt für jedes feste $i \in \mathbb{N}$, daß $e + \varepsilon$ als Grenzwert einer Folge von $x_j \leq r'_{i+1}$ (für $j > i$) ebenfalls nicht größer als r'_{i+1} und damit echt kleiner als r'_i ist. Folglich gilt wie gewünscht $(e + \varepsilon, -e) \in R_i$. $\square$

Weitere Bemerkungen

Analog zu oben sieht man, daß jede $\mathbb{R}$-*berechenbare* Funktion in einer Variablen mit rationalem Ergebnis $f : \mathbb{R} \to \mathbb{Q}^*$ durch ein universelles Programm und eine geeignete Rundung ρ_f von einer δ-$\mathbb{Q}$-Maschine regulär berechnet werden kann. Dies geht im allgemeinen natürlich nicht robust, wie man sich am Beispiel der Vorzeichenfunkti-

[4] Im Fall $k > 0$ muß (x_0, y_0) eventuell anders gewählt werden. Die konkrete Zahl e dient hier nur der Anschaulichkeit.

[5] Da es nur abzählbar viele algebraische Zahlen in $\mathbb{R}$ gibt, existiert ε_1 sicherlich.

on sign bei Eingabe $x = 0$ verdeutlicht. Die Hintereinanderschaltung robuster δ-$\mathbb{Q}$-Maschinen hilft bei endlichen Berechnungen natürlich nicht weiter.

Wenn auch eine δ-$\mathbb{Q}$-Maschine nicht ausreicht, so werden wir doch nun sehen, daß jede $\mathbb{R}$-analytische Funktion zumindest von zwei hintereinander geschalteten δ-$\mathbb{Q}$-Maschinen analytisch berechnet werden kann. Dies geht sogar robust, d.h. unabhängig von der verwendeten Rundungsfunktion. Der folgende Satz kann als eine Fortsetzung des Simulationssatzes 1.19 verstanden werden, der im folgenden Abschnitt gezeigt wird.

Satz 1.17. *Jede $\mathbb{R}$-analytische Funktion ist robust δ-$\mathbb{Q}^2$-analytisch.*

Beweis. Sei die Funktion $f : \mathbb{R}^\star \supset D \to \mathbb{R}$ zunächst einstellig. Dann erzeugt die zugeordnete Maschine $\mathcal{M}_f$ bei Eingabe $x \in D$ eine Folge von Ausgaben $y_n \in \mathbb{R}$ mit $\lim_{n\to\infty} y_n = f(x)$. Die Maschine $\mathcal{M}_1$ (mit beliebiger Rundung) simuliere nun $\mathcal{M}_f$ analog zum Simulationssatz; die Approximation von y_n bei Präzision δ sei $y_n^{(\delta)}$, diese ist δ-$\mathbb{Q}$-berechenbar. Nun gilt

$$\forall\, n \in \mathbb{N} : \lim_{\delta\to\infty} y_n^{(\delta)} = y_n$$

und daher $\lim_{n\to\infty} \lim_{\delta\to\infty} y_n^{(\delta)} = f(x)$. Leider kann $\mathcal{M}_1$ diese beiden Grenzwerte nicht einfach gleichzeitig bilden. Allerdings ist es möglich, eine Ausgabe zu erzeugen, die gegen $(y_0, y_1, \dots) \in \mathbb{R}^\omega$ konvergiert. Dazu berechnet $\mathcal{M}_1$ mit wachsendem δ jeweils $y_0^{(\delta)}, \dots, y_\delta^{(\delta)}$, und schreibt $y_n^{(\delta)}$ in Ausgabezelle n. Die zweite Maschine $\mathcal{M}_2$ muß nun lediglich die Eingabefolge lesen und der Reihe nach die y_n ausgeben.

Falls $f : D \to \mathbb{R}^*$ mehrstellig ist, so kann $\mathcal{M}_1$ die Zwischenergebnisse $y_n^{(\delta)} \in \mathbb{Q}^*$ leicht in eine rationale Zahl packen. Will man unendliche Zwischenergebnisse vermeiden, so kann $\mathcal{M}_1$ nach einer Idee von VIERKE die Richtung jeder Verzweigung robust δ-$\mathbb{Q}$-analytisch bestimmen und alle diese Bits in eine reelle Zahl kodieren. $\mathcal{M}_2$ liest diese Information dann aus und simuliert jeweils einen endlichen Berechnungspfad von $\mathcal{M}$ zur Approximation von y_n. Für diesen kann eine Fehlerschranke wie im Beweis zu Satz 1.19 bestimmt werden; durch Erhöhen der Präzision δ wird die Fehlerschranke z.B. auf 2^{-n} begrenzt, ehe n erhöht wird. □

Anmerkung 1.8. Nachdem die oben geführten Beweise zur Separation der Klassen berechenbarer Funktionen – bzw. der verschiedenen Varianten des Berechenbarkeitsbegriffs – sich wesentlich auf die charakteristische Funktion der rationalen Zahlen stützen, könnte der geneigte Leser sich fragen, warum man nicht $\chi_\mathbb{Q}$ als Elementaroperation aufnimmt. Zum einen greifen obige Beweise natürlich nur beispielhaft geeignete Funktionen heraus, mit deren Hilfe man mehr oder weniger leicht die gewünschten Resultate zeigen kann. Es ist zu vermuten, daß auch ohne $\chi_\mathbb{Q}$ sich Probleme finden lassen, die diese Klassen separieren; daher mutet es willkürlich an, gerade diese eine charakteristische Funktion derart auszuzeichnen. Zum anderen hätte dieser Schritt nicht die erhofften Auswirkungen zur Vereinfachung der Theorie, wie die folgenden Bemerkungen zeigen sollen.

In dem um $\chi_{\mathbb{Q}}$ als Elementaroperation erweiterten Modell bleibt die Klasse der δ-$\mathbb{Q}$-analytischen Funktionen zumindest bei endlichen Eingabewörtern invariant, was man wie folgt einsieht. Da in δ-$\mathbb{Q}$-Maschinen irrationale Zahlen nur auf dem Eingabeband vorkommen, bietet sich eine neue Verzweigungsbedingung der Form `if` $x_i \in \mathbb{Q}$ `then ...` an. Die rationalen Zahlen $\mathbb{Q} \subset \mathbb{R}$ sind $\mathbb{R}$-semi-entscheidbar, d.h. eine geeignete Rundungsfunktion ρ kann ein zusätzliches Bit an Information übergeben, das ab einer bestimmten, aber unbekannten Präzision δ die Frage „$x_i \in \mathbb{Q}$?“ korrekt beantwortet. Demnach können nach endlich vielen Runden der Berechnung *alle* (endlich vielen) Abfragen korrekt simuliert werden, und das Endergebnis bleibt das gleiche.

Allerdings wäre die Klasse der quasi stark δ-$\mathbb{Q}$-analytischen Funktionen (siehe Abschnitt 1.4) nicht mehr abgeschlossen bzgl. der Hintereinanderausführung: Die Addition $\mathtt{add}$ zweier Zahlen ist quasi stark δ-$\mathbb{Q}$-analytisch, $\chi_{\mathbb{Q}}^{+} = \chi_{\mathbb{Q}} \circ \mathtt{add}$ dagegen nicht einmal δ-$\mathbb{Q}$-analytisch. Dafür wäre allerdings die „kurze Bartfunktion“ d_1 (vgl. 1.9), ehedem zwar robust aber nicht quasi stark, nunmehr quasi stark δ-$\mathbb{Q}$-analytisch. Man sieht also, daß sich die Mächtigkeit des Modells erhöht und die Grenzen zwischen den Klassen sich verschieben; die Theorie vereinfacht sich dadurch jedoch nicht.

1.4. Quasi stark analytische Funktionen

Eine einfache Simulation der Hintereinanderschaltung $\mathcal{M}_2 \circ \mathcal{M}_1$ zweier robuster analytischer δ-$\mathbb{Q}$-Maschinen durch eine einzige Maschine $\mathcal{M}$ scheitert an der Eingabe von $\mathcal{M}_2$. Diese wird durch Rundung des reellen Grenzwerts der Ausgabe von $\mathcal{M}_1$ erzeugt, aber während der Simulation von $\mathcal{M}_1$ ist die Genauigkeit der bisher erreichten Approximation dieses Grenzwerts im allgemeinen unbekannt. Man kann jedoch solche Programme betrachten, die gleichzeitig mit einem neuen Glied der Ausgabefolge stets auch eine Fehlerabschätzung errechnen.

Definition 1.8. Sei $(k_{i_j})_{j=0}^{\infty}$ die Folge aller Zielkonfigurationen einer δ-$\mathbb{Q}$-analytischen Berechnung, welche *unendlich oft* den Befehl „`next` δ“ enthält. Deren Ausgaben bezeichnen wir mit

$$\mathtt{out}(k_{i_j}) = (y_1^{(j)}, \ldots, y_{n_j}^{(j)}) \in \mathbb{Q}^*$$

und deren evtl. unendliche Länge im Grenzwert mit $n := \lim_{j\to\infty} n_j$. Für $i \leq n$ nennen wir den Grenzwert der i. Stelle der Ausgabe $y_i := \lim_{j\to\infty} y_i^{(j)}$. Dann heißt die Berechnung *quasi stark* δ-$\mathbb{Q}$-*analytisch*, falls

1. $y_1 = 0$;
2. $\exists J \, \forall 2 \leq i \leq n \, \forall j > J : \; |y_i - y_i^{(j)}| \leq y_1^{(j)}$.

Als Ergebnis dieser Berechnung wird der Grenzwert $(y_2, \ldots, y_n)$ der Ausgaben ohne die Fehlerschranke betrachtet. Eine Funktion $f : \mathbb{R}^\star \supset D \to \mathbb{R}^\star$ heißt *quasi stark* δ-$\mathbb{Q}$-*analytisch*, falls es ein robustes Programm π gibt, so daß $D \subset \mathbb{D}_{\mathcal{M}}$ und für $x \in D$

die Berechnung von $\mathcal{M} := \mathcal{M}_\pi^{\delta\text{-}\mathbb{Q}}$ angesetzt auf $\mathtt{in}(x)$ quasi stark δ-ℚ-analytisch mit Ergebnis $f(x)$ ist. □

Eine solche quasi stark δ-ℚ-analytische Berechnung produziert also in der ersten Stelle der Ausgabe eine absolute Fehlerschranke für alle weiteren Stellen, und diese Schranke konvergiert gegen Null. Das besondere ist nun, daß diese Fehlerschranke nur für *fast alle* Zeitpunkte der Berechnung gelten muß, d.h. erst von einem bestimmten, aber unbekannten Zeitpunkt J ab. Die bemerkenswerte Eigenschaft der quasi stark δ-ℚ-analytischen Funktionen, die sich nun unmittelbar ergibt, ist die Abgeschlossenheit unter der Hintereinanderausführung.

Lemma 1.18. *Sei* $D \subset \mathbb{R}^\star$ *und* $f : \mathbb{R}^\star \rightsquigarrow D$ *sowie* $g : D \to \mathbb{R}^\star$ *quasi stark δ-ℚ-analytisch, dann ist auch* $g \circ f$ *quasi stark δ-ℚ-analytisch.*

Beweis. Seien $\mathcal{M}_f$ und $\mathcal{M}_g$ zwei quasi stark δ-ℚ-analytische Maschinen für f und g; alle ihre Berechnungen werden durch den Befehl „`next δ`" in unendlich viele Abschnitte eingeteilt. Man simuliert nun mit einer einzigen Maschine $\mathcal{M}$ zunächst $\mathcal{M}_g$, bis eine mit Präzision δ_g gerundete Eingabe benötigt wird. Dann wird die Simulation von $\mathcal{M}_f$ auf der ursprünglichen Eingabe x soweit vorangetrieben, daß die Fehlerschranke höchstens δ_g beträgt. Die dann erreichte Approximation von $f(x)$ dient als gerundete Eingabe für die Maschine $\mathcal{M}_g$, womit deren Simulation fortgesetzt wird. Man beachte, daß die Zustände der beiden Maschine $\mathcal{M}_f$ und $\mathcal{M}_g$ beide im Hauptspeicher von $\mathcal{M}$ abgelegt werden können; das Präzisionsregister von $\mathcal{M}$ entspricht δ_f, während δ_g nur eine interne Rolle spielt als Sollwert der Fehlerschranke von $\mathcal{M}_f$.

Nun beobachtet man, daß $\mathcal{M}_g$ endlich oft mit einem falschen, da nicht ausreichend präzisen Eingabewert versorgt wird; dies spielt aber für den Grenzwert der Ausgabe keine Rolle. Die Fehlerschranke von $\mathcal{M}$ wird dadurch ebenfalls nur endlich oft falsch und die Gesamtberechnung ist quasi stark δ-ℚ-analytisch. □

HOTZ, SCHIEFFER, VIERKE haben in [26] gezeigt, daß die *stark δ-ℚ-analytischen* Funktionen, bei denen die Fehlerschranke stets gilt, stetig sind. Diese Funktionsklasse entspricht den berechenbaren reellen Funktionen im Sinne von GRZEGORCZYK [24] und KO [27], falls wir die Programmkonstanten auf rationale Zahlen beschränken. Im Gegensatz dazu umfassen die quasi stark δ-ℚ-analytischen Funktionen einen viel weiteren Bereich. Wir verschärfen nun den Simulationssatz aus [26, Theorem 1] und geben einen wesentlich eleganteren Beweis dafür.

Satz 1.19 (Simulationssatz). *Jede ℝ-berechenbare Funktion ist quasi stark δ-ℚ-analytisch.*

Beweis. Die δ-ℚ-Maschine $\mathcal{M}'$ simuliert die ℝ-Maschine $\mathcal{M}$ zur gegebenen Funktion durch Intervallarithmetik und mit wachsender Präzision δ. Dabei werden die Zellen des Rechenspeichers und Ausgabebands sowie der Akkumulator von $\mathcal{M}$ im Rechenspeicher von $\mathcal{M}'$ jeweils durch Unter- und Obergrenze nachgebildet; diese sind anfangs korrekt mit Null initialisiert. Die meisten Befehle können dann in naheliegender Weise nachgeahmt werden, mit folgenden Ausnahmen.

Zuweisungen $\alpha := x_i$ (und analog $\alpha := r$ mit $r \notin \mathbb{Q}$), bei denen eine δ-$\mathbb{Q}$-Maschine $\alpha := \rho(x_i, \delta)$ ausführt, weisen dem simulierten Akkumulator das Intervall $[\alpha - 2^{-\delta}, \alpha + 2^{-\delta}]$ zu. Die Verzweigungsbedingung `if α > 0 then ...` wird so interpretiert, daß ein Intervall genau dann positiv ist, wenn seine Untergrenze positiv ist; in diesem Sinne ist es gleich Null, solange es die Null umfaßt. Der `print`-Befehl wird nun genutzt, um die Intervallmitten des simulierten Ausgabebands auf das tatsächliche Ausgabeband von $\mathcal{M}'$ zu schreiben; die maximale Intervallänge dient dabei als Fehlerschranke. Statt eines `end` wird ein `next δ` ausgeführt und damit eine neue Runde der Simulation eingeläutet. Dabei muß der simulierte Rechenspeicher und das Ausgabeband von $\mathcal{M}$ gelöscht werden, um erneut einen Anfangszustand herzustellen.

Man beachte, daß auf diese Weise die Korrektheit des Programms insofern erhalten bleibt, als Divisionen durch Null vermieden werden. Indem man maximal δ viele Verzweigungen pro Runde ausführt, werden Endlosschleifen vermieden. Die Fehlerschranke ist höchstens solange falsch, bis $\mathcal{M}'$ den richtigen Pfad von $\mathcal{M}$ zu simulieren beginnt. Falls an einer Verzweigung des Programms der Inhalt des Akkumulators von $\mathcal{M}$ einen Wert ungleich Null hat, so gibt das von $\mathcal{M}'$ berechnete Intervall bei hinreichender Genauigkeit das Vorzeichen exakt wieder. Indem wir uns dem unentscheidbaren Fall $\alpha = 0$ vorsichtig und von der sicheren Seite her nähern, wird dieser von $\mathcal{M}'$ stets korrekt behandelt – wir nennen diese Vorgehensweise *konservatives Verzweigen*. Damit ist klar, daß jeder endliche Programmpfad von $\mathcal{M}$ nach endlicher Zeit von $\mathcal{M}'$ nachgeahmt wird; dann aber konvergiert die Simulation wie gewünscht und ist robust. □

Am Beispiel fraktaler Strukturen sieht man, daß oben bewiesene Inklusion echt ist, d.h. daß gewisse quasi stark δ-$\mathbb{Q}$-analytische Funktionen nicht $\mathbb{R}$-berechenbar sind.

Beispiel 1.3 (Koch-Kurve). Durch wiederholte Ersetzung des mittleren Drittels einer Linie durch die Schenkel eines gleichseitigen Dreiecks gemäß Abbildung 1.4 entsteht eine Folge von Kurven k_n. Deren Grenzwert $\lim_{n\to\infty} k_n$ heißt Koch[6]-Kurve und ist zwar überall stetig, aber nirgendwo differenzierbar. Wir werden sehen, daß man diese durch eine quasi stark δ-$\mathbb{Q}$-analytische stetige Funktion $k : [0,1] \to \mathbb{R}^2$ beschreiben kann, die nicht $\mathbb{R}$-berechenbar ist. Letzteres ergibt sich aus der Hausdorff-Dimension $\log_3 4 \notin \mathbb{Z}$ von $k([0,1]) \subset \mathbb{R}^2$ und Satz 1.2 (vgl. hierzu auch [25, 26]). Außerdem folgt dies auch unmittelbar aus dem Darstellungssatz für $\mathbb{R}$-berechenbare Funktionen: Diese sind nur an abzählbar vielen Stellen nicht differenzierbar!

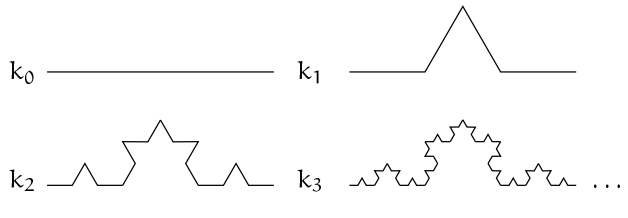


Abbildung 1.4.: Entstehung der Koch-Kurve

[6] Niels Fabian Helge von Koch (1870–1924), schwedischer Mathematiker.

Für die Fehlerabschätzung bei der Berechnung von k beobachtet man, daß die Koch-Kurve vollständig innerhalb der konvexen Hülle von k_1 verläuft[7]. Wegen des rekursiven Aufbaus[8] der Kurve ist damit auch klar, in welchem Bereich um eine Iteration k_n der Grenzwert liegt. Demnach kann eine quasi stark δ-$\mathbb{Q}$-analytische Maschine den Eingabewert mit Präzision δ lesen, den Bildpunkt auf der Kurve k_δ näherungsweise ausrechnen und eine Fehlerschranke ausgeben, die vom Durchmesser der schattierten Dreiecke in Abbildung 1.5 abhängt. □

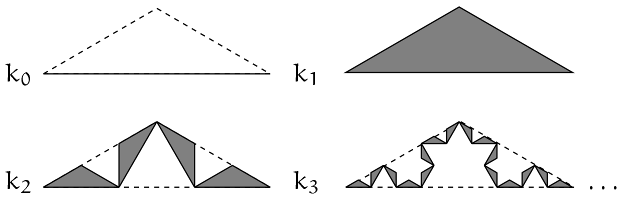


Abbildung 1.5.: Einschachtelung der Koch-Kurve

1.5. Halte- und Konvergenzproblem

Halteproblem

Aus [26, 44] sind folgende Resultate bekannt.

- Das Halteproblem für reguläre $\mathcal{R}$-Maschinen ist nicht $\mathcal{R}$-berechenbar; dies sieht man durch Diagonalisierung.
- Das Halteproblem für reguläre $\mathcal{R}$-Maschinen ist $\mathcal{R}$-analytisch; dies sieht man durch Simulation.
- Das Halteproblem für reguläre $\mathbb{R}$-Maschinen ist nicht robust δ-$\mathbb{Q}$-analytisch; dies sieht man am Beispiel $\chi_{\mathbb{Q}}$.

Da die Menge Q_2 zwar $\mathbb{R}$-semi-entscheidbar, nicht aber analytisch δ-$\mathbb{Q}$-entscheidbar ist (siehe Satz 1.16), ergibt sich unmittelbar

- Das Halteproblem für reguläre $\mathbb{R}$-Maschinen ist nicht δ-$\mathbb{Q}$-analytisch; dies sieht man am Beispiel Q_2.

Konvergenzproblem

Analog zum Halteproblem von Turing-Maschinen kann man das Konvergenzproblem analytischer $\mathcal{R}$-Maschinen betrachten als die Frage, ob eine durch ein Programm gegebene Maschine bei einer bestimmten Eingabe ein Resultat erzeugt, d.h. eine konvergente Ausgabefolge. Aus [26, 44] sind hier folgende Resultate bekannt, die auch für analytische δ-$\mathbb{Q}$-Maschinen mit robusten Programmen gelten.

[7]Dies beweist man leicht durch vollständige Induktion für alle k_n.
[8]Die Kurve k_{n+1} besteht aus 4^n Stücken, die zu k_1 kongruent sind.

- Das Konvergenzproblem für analytische $\mathbb{R}$-Maschinen ist nicht $\mathbb{R}$-analytisch (und damit auch nicht δ-$\mathbb{Q}$-analytisch); dies sieht man durch Diagonalisierung.
- Das Konvergenzproblem für analytische $\mathbb{R}$-Maschinen ist $\mathbb{R}^3$-analytisch; dies sieht man durch eine aufwendige Konstruktion.

Wir werden diese Ergebnisse nun verallgemeinern auf die verschiedenen Klassen analytischer Maschinen sowie auf deren mehrfache Hintereinanderschaltung. Ferner werden wir das Konvergenzproblem mit nur zwei analytischen Maschinen lösen, wobei allerdings unendlich lange Zwischenergebnisse auftreten; kodiert man diese in einzelne reelle Zahlen, so führt das zu der bekannten Konstruktion mit drei Maschinen. Als Einstieg beweisen wir einen allgemeinen Satz über die Unentscheidbarkeit des Konvergenzproblems.

Satz 1.20. *Das Konvergenzproblem $\mathbb{R}^i$-analytischer Maschinen ist nicht $\mathbb{R}^i$-analytisch; analoges gilt für (robust) δ-$\mathbb{Q}^i$-analytische Maschinen.*

Beweis. Sei Π die Menge aller korrekten Programme; diese werden beschrieben durch eine endliche Folge von Befehlen aus einem festen, endlichen Alphabet, jeweils ergänzt um eventuelle Indizes, Sprungadressen oder reelle Konstanten. Man kann also leicht eine präfixfreie Kodierung von $\Omega^{\mathbb{R}}$ in $\mathbb{R}^*$ finden und so $\Pi \subset \mathbb{R}^*$ verstehen. Damit ist klar, daß Programme als Eingabe von Berechnungen dienen können, und daß gilt auch für Folgen von Programmen. Ein Tupel $\pi = (\pi_1, \ldots, \pi_i)$ von Programmen bestimmt eine partielle Funktion $\Phi_\pi : \mathbb{R}^\star \rightsquigarrow \mathbb{R}^\star$, die von der Hintereinanderschaltung $\mathcal{M}_{\pi_i} \circ \cdots \circ \mathcal{M}_{\pi_1}$ der $\mathbb{R}$-Maschinen berechnet wird.

Annahme: Sei $\pi' \in \Pi^i$ ein Tupel, welches das Konvergenzproblem löst, also die Abbildung $\Pi^i \times \mathbb{R}^\star \to \mathbb{B}$ $\mathbb{R}^i$-analytisch berechnet mit

$$(\pi, x) \mapsto \begin{cases} 1 & \text{falls } \Phi_\pi(x) \text{ definiert;} \\ 0 & \text{sonst.} \end{cases}$$

Nun wandelt man π' zu π'' ab, indem zuerst die Eingabe verdoppelt und dann jede Ausgabe im Intervall $[\frac{1}{2}, \frac{3}{2}]$ abwechselnd durch 0 und 1 ersetzt wird; dann kann die Ausgabefolge nicht mehr gegen 1 konvergieren. So kann man leicht folgende partielle Funktion $\Pi^i \rightsquigarrow \mathbb{B}$ $\mathbb{R}^i$-analytisch berechnen:

$$\pi \mapsto \begin{cases} 0 & \text{falls } \Phi_\pi(\pi) \text{ undefiniert;} \\ \text{undefiniert} & \text{sonst.} \end{cases}$$

Damit ergibt sich aber der klassische Widerspruch

$$\Phi_{\pi''}(\pi'') \text{ definiert} \iff \Phi_{\pi''}(\pi'') \text{ undefiniert.}$$

Diese Argumentation funktioniert auch für robust δ-$\mathbb{Q}$-analytische Funktionen. Zur Verallgemeinerung auf nicht robuste δ-$\mathbb{Q}$-analytische Funktionen sei $P \subset \Pi$ die Menge aller Programme für $\mathbb{R}$-berechenbare Rundungsfunktionen; dann ersetze man in obigem Beweis Π jeweils durch $\Pi \times P$, da eine δ-$\mathbb{Q}$-Maschine durch Programm und Rundung festgelegt wird. □

Wir werden nun kurz erläutern, wie man das Konvergenzproblem einer δ-$\mathbb{Q}$-analytischen Maschine durch mehrere hintereinander geschaltete löst; dieser Satz gilt entsprechend auch für $\mathbb{R}$-Maschinen.

Satz 1.21. *Das Konvergenzproblem δ-$\mathbb{Q}$-analytischer Maschinen ist δ-$\mathbb{Q}^2$-analytisch.*

Beweis. Der Beweis folgt dem von VIERKE [44, Kap. 5] für $\mathbb{R}$-analytische Maschinen. Sei o.B.d.A. $f : \mathbb{R}^\star \supset D \to \mathbb{R}$ einstellig, sonst wende man Lemma 1.25 sinngemäß an oder betrachte die Ausgabe komponentenweise (dies kann parallelisiert werden). Eine wesentliche Idee ist nun, daß eine monotone Folge genau dann über alle Grenzen wächst, wenn die Kehrwerte eine Nullfolge bilden; dabei ignorieren wir der Übersichtlichkeit halber die Möglichkeit einer Division durch Null.

Für festes $x \in \mathbb{R}^\star$ sei u_n die Ausgabe der n. Zielkonfiguration der Berechnung von $\mathcal{M}_f$ angesetzt auf $\mathrm{in}(x)$. Zu entscheiden ist nun, ob $\lim_{n\to\infty} u_n$ existiert, d.h. ob die Folge beschränkt ist und genau einen Häufungspunkt aufweist. Die Folge ist genau dann unbeschränkt, wenn

$$M := \lim_{n\to\infty} (\max_{i\leq n} |u_i|)^{-1} = 0.$$

Nun betrachtet man Intervalle $I_n \supset \{u_0, \ldots, u_n\}$ – genauer gesagt symmetrische Umgebungen von Null – und deren äquidistante Verfeinerungen $J_{n,k}$ in p_n Teilintervalle. Dabei sei p_n die n. Primzahl und die Länge von I_n stets eine Zweierpotenz, so tritt keine Verfeinerungsgrenze zweimal auf. Ein Teilintervall $J_{n,k}$ hat einen Häufungspunkt genau dann, wenn

$$\chi_{n,k} := \lim_{m\to\infty} (\sum_{i=0}^{m} \chi_{i,n,k})^{-1} = 0,$$

wobei

$$\chi_{i,n,k} = \begin{cases} 1 & \text{falls } u_i \in J_{n,k}; \\ 0 & \text{sonst.} \end{cases}$$

M ist δ-$\mathbb{Q}$-analytisch, da alle u_n δ-$\mathbb{Q}$-berechenbar sind; die $\chi_{n,k}$ sind δ-$\mathbb{Q}$-analytisch, da alle Teilsummen δ-$\mathbb{Q}$-berechenbar sind. Die erste Maschine $\mathcal{M}_1$ berechne daher M und alle $\chi_{n,k}$, dies funktioniert gemäß Lemma 1.6 und ergibt ein unendliches Ausgabewort.

Die Anzahl der Häufungspunkte im Intervall I_n ist

$$\alpha_n := \sum_{k=1}^{p_n} (1 - \mathrm{sign}(\chi_{n,k})),$$

sofern keiner auf einer Verfeinerungsgrenze liegt. Die Folge $(u_n)_{n=0}^\infty$ divergiert genau dann, wenn $\mathrm{sign}(M) = 0$ oder $\alpha_n \geq 2$ für mehr als ein n. Ein einzelner „Ausrutscher“ kann auftreten, wenn der Grenzwert auf eine Verfeinerungsgrenze fällt; dies kann sich aber nicht wiederholen. Die zweite Maschine berechnet also erst $\mathrm{sign}(M)$ (dabei hilft eine geeignete Rundung) und dann nach und nach alle α_n und vermutet Konvergenz, bis das Gegenteil bewiesen ist. □

Korollar 1.22. Das Konvergenzproblem $\mathbb{R}^i$-analytischer Maschinen ist $\mathbb{R}^{i+1}$-analytisch, analog für δ-$\mathbb{Q}$-Maschinen.

Beweis. Wir stellen uns die i hintereinander geschalteten Maschinen als eine Pipeline vor, in welche vorne die Eingabe hineingeht, im Innern Zwischenergebnisse fließen und am Ende die Ausgabe herauskommt. Wir erweitern nun die Kommunikation zwischen den Maschinen um weitere Kanäle[9], welche analog zu Satz 1.21 M und $\chi_{n,k}$ weiterleiten. Die einzelnen Maschinen führen dann parallel mehrere Funktionen aus: Sie verarbeiten ihre eigentlichen Eingaben, errechnen aus ihren eigenen Ausgaben die Hilfsfunktionen M und $\chi_{n,k}$ und entscheiden mittels den entsprechenden Hilfsausgaben ihrer Vorgänger deren Konvergenz. Nur wenn alle vorherigen Maschinen Konvergenz signalisieren, kann auch die letzte, zusätzliche Maschine insgesamt auf Konvergenz entscheiden. Durch diese überlappende Anordnung sind statt $2i$ nur $i + 1$ Maschinen notwendig. □

Anmerkung 1.9. Überträgt man obige Resultate auf robust δ-$\mathbb{Q}$-analytische Funktionen, so wird jeweils eine Maschine mehr notwendig. Das Vorzeichen einer Zahl ist zwar δ-$\mathbb{Q}$-berechenbar, aber lediglich robust δ-$\mathbb{Q}$-analytisch; daher ist $\mathcal{M}_2$ mit der Berechnung der Vorzeichenbits ausgelastet. Erst $\mathcal{M}_3$ bildet dann die Summen α_n und entscheidet die Konvergenz.

Der angekündigte allgemeine Hierarchiesatz für δ-$\mathbb{Q}$-analytische Funktionen ergibt sich nun unmittelbar.

Satz 1.23.

$$\forall\, i \in \mathbb{N} : \{\textit{(robust) } \delta\text{-}\mathbb{Q}^i\textit{-analytisch}\} \subsetneq \{\textit{(robust) } \delta\text{-}\mathbb{Q}^{i+1}\textit{-analytisch}\}.$$

Beweis. Die behauptete Inklusion ist trivial; falls sie für irgendein i_0 nicht echt ist, so kollabiert die Hierarchie ab dieser Stelle. Dies steht aber im Widerspruch dazu, daß für alle i das Konvergenzproblem δ-$\mathbb{Q}^i$-analytischer Maschinen nicht δ-$\mathbb{Q}^i$-analytisch ist, wohl aber δ-$\mathbb{Q}^{i+2}$-analytisch gemäß Korollar 1.22. □

Anmerkung 1.10. Mit den obigen Beweistechniken findet man leicht, daß das Konvergenzproblem quasi stark δ-$\mathbb{Q}$-analytischer Maschinen nicht quasi stark δ-$\mathbb{Q}$-analytisch ist, wohl aber δ-$\mathbb{Q}^2$-analytisch und robust δ-$\mathbb{Q}^3$-analytisch. Dabei kann sowohl die Konvergenz der Ausgabe als auch die Korrektheit der Fehlerschranke gemäß Definition 1.8 getestet werden.

Kodierung in reellen Zahlen

Wenn man zur Lösung des Konvergenzproblems mehrere analytische Maschinen hintereinanderschaltet, so muß eine Maschine jeweils unendlich viel Information an die nächste übergeben. Falls man keine unendlichen Wörter als Ausgabe zulassen will, so muß die Information in einer (einzelnen) reellen Zahl untergebracht werden. Dazu dient das folgende

[9] Dies kann leicht durch eine geänderte Anordnung auf den Ein-/Ausgabebändern geschehen.

Lemma 1.24. *Sei* $f : \mathbb{R}^\star \supset D \to \mathbb{B}^\omega$ $\mathbb{R}$- *bzw. (robust)* δ-$\mathbb{Q}$*-analytisch, dann ist auch* $r : D \to \mathbb{R}$ $\mathbb{R}$- *bzw. (robust)* δ-$\mathbb{Q}$*-analytisch, wobei*

$$r(x) := \sum_{i=0}^{\infty} (f(x))_i 2^{-i}.$$

Umgekehrt kann eine reelle Zahl[10] $r = \sum_{i=0}^{\infty} b_i 2^{-i}$ *von einer analytischen* $\mathbb{R}$- *bzw.* δ-$\mathbb{Q}$-*Maschine in die Folge* $(b_0, b_1, \dots) \in \mathbb{B}^\omega$ *zerlegt werden; d.h. insbesondere, daß* $i \mapsto b_i$ $\mathbb{R}$- *und* δ-$\mathbb{Q}$*-berechenbar ist. Die Dekodierung ist robust* δ-$\mathbb{Q}$*-analytisch, falls* $r \notin \mathbb{D}$.

Beweis. Sei $\mathcal{M}$ eine $\mathcal{R}$-Maschine für f und $f^{(n)}(x) = (b_0^{(n)}, \dots, b_{k_n}^{(n)}) \in \mathbb{B}^*$ entstehe durch Rundung von $\Phi_{\mathcal{M}}^{(n)}(x) \in \mathcal{R}^*$. Man bilde $r^{(n)}(x) := \sum_{i=0}^{k_n} b_i^{(n)} 2^{-i}$ als n. Approximation von $r(x)$. Es gilt $\lim_{n\to\infty} r^{(n)}(x) = r(x)$, da $b_i^{(n)}$ für jedes i ab einem gewissen n korrekt ist.

Die Behauptung über die Umkehrung der Kodierung ist für $\mathbb{R}$-Maschinen offensichtlich; für δ-$\mathbb{Q}$-Maschinen wähle man als Rundungsfunktion z.B. eine abgebrochene Binärentwicklung. Damit die Dekodierung robust δ-$\mathbb{Q}$-analytisch funktioniert, darf r keine dyadische Zahl sein; dann existiert eine eindeutige Binärdarstellung, aus deren Approximationen die einzelnen Bits nach und nach ablesbar sind. □

Diese Konstruktion läßt sich erweitern, um abzählbar viele reelle Zahlen in eine zu packen. Hier muß man aber die Einschränkung machen, daß die reellen Zahlen entweder nicht dyadisch sind oder von der Berechnung monoton approximiert werden, so daß die Ausgaben nicht nur als reelle Zahlen, sondern auch als Binärdarstellungen konvergieren.

Lemma 1.25. *Sei* $f : \mathbb{R}^\star \supset D \to \mathbb{R}^\omega$ $\mathbb{R}$- *bzw. (robust)* δ-$\mathbb{Q}$*-analytisch (mit obiger Einschränkung), dann gibt es eine* $\mathbb{R}$- *bzw. (robust)* δ-$\mathbb{Q}$*-analytische Funktion* $r : D \to \mathbb{R}$*, aus deren Ausgabe* $r(x)$ *die Folge* $f(x) \in \mathbb{R}^\omega$ *von einer analytischen* $\mathbb{R}$- *bzw.* δ-$\mathbb{Q}$*-Maschine wiedergewonnen werden kann. Die Dekodierung ist robust* δ-$\mathbb{Q}$*-analytisch, falls* $r(x) \notin \mathbb{D}$.

Beweis. Man betrachte eine Binärdarstellung der Zahlen $(f(x))_i$ für festes $x \in D$ und eine bijektive, berechenbare Abbildung $\mathbb{N}^2 \to \mathbb{N}$; dann packe man alle abzählbar vielen Bits der abzählbar vielen Zahlen in eine reelle Zahl, analog zum Lemma 1.24. □

Wendet man nun obige Lemmata an, um das Konvergenzproblem ohne unendliche Zwischenergebnisse zu lösen, so ändert sich die geschilderte Konstruktion wie folgt. $\mathcal{M}_1$ berechnet weiterhin M und alle $\chi_{n,k}$ und kodiert letztere in einer reellen Zahl. $\mathcal{M}_2$ berechnet aus seiner Eingabe $\mathrm{sign}(M)$ und alle $\mathrm{sign}(\chi_{n,k})$ und kodiert diese Bits in einer reellen Zahl; diese sind jeweils $\mathbb{R}$-analytisch und damit δ-$\mathbb{Q}$-analytisch bei geeignet gewählter Rundungsfunktion (siehe Satz 1.10). Eine dritte Maschine $\mathcal{M}_3$ ist nötig, um die α_n zu berechnen und die Konvergenz analytisch zu entscheiden. Die Konstruktion ist sogar robust δ-$\mathbb{Q}$-analytisch, da M und $\chi_{n,k}$ monoton fallend approximiert werden und die Kodierung aller $\chi_{n,k}$ keine dyadische Zahl ergibt (außer

[10] Dabei darf nicht $b_i \to 1$ für $i \to \infty$ gelten, damit die Darstellung eindeutig ist.

Null, falls die Folge überall Häufungspunkte hat); letzteres gilt auch für die Vorzeichenbits. Falls die Folge divergiert, könnten zwar alle $\mathtt{sign}(\chi_{n,k}) = 1$ sein, aber das spielt dann keine Rolle, da $\mathtt{sign}(M) = 0$ erkannt wird.

Offene Frage 1. Ist das Konvergenzproblem für analytische $\mathbb{R}$-Maschinen $\mathbb{R}^2$-analytisch, wenn man auf unendliche Folgen als Zwischenergebnisse verzichtet?

1.6. Unterprogrammtechnik

Wir wollen nun die Situation untersuchen, daß z.B. ein Programm für eine $\mathbb{R}$-Maschine eine quasi stark δ-$\mathbb{Q}$-analytische Maschine wie ein Unterprogramm (UP) aufrufen darf. Man kann diesen Fall unter verschiedenen Aspekten betrachten und motivieren, und wir wollen zunächst zwei davon beschreiben. Danach stellen wir ein mögliches und einfaches Konzept vor, die Semantik des UP-Aufrufs zu präzisieren, welches vom Standpunkt der Berechenbarkeit hinreichend allgemein ist. Schließlich untersuchen wir sinnvolle Kombinationen von Maschinentypen und ordnen die jeweils entstehende Klasse berechenbarer Funktionen in die uns bereits bekannte Hierarchie ein.

Die oben geschilderte Situation kann man so deuten, daß dem Hauptprogramm eine bestimmte quasi stark δ-$\mathbb{Q}$-analytische Funktion als Elementaroperation zur Verfügung gestellt wird, z.B. die Exponentialfunktion. Man interessiert sich dann dafür, wie sich die Fähigkeiten einer Maschine erweitern in Abhängigkeit von der Klasse berechenbarer Funktionen, aus denen die Elementaroperation entnommen wird. Ein Resultat lautet dann, daß der Simulationssatz weiter gilt, auch wenn eine $\mathbb{R}$-Maschine um quasi stark δ-$\mathbb{Q}$-analytische Prozeduren erweitert wird. Diese Sichtweise legt es nahe, eine feste Sammlung von Prozeduren zu einem Hauptprogramm hinzuzufügen, die alle aus derselben Funktionsklasse stammen.

Man kann die Idee der UP-Technik aber auch ganz anders auffassen[11], indem man von einer in der Mathematik weitverbreiteten und mit großem Erfolg verwandten Operation ausgeht: der *Grenzwertbildung*. Es ist dann naheliegend, unseren $\mathbb{R}$-Maschinen die Fähigkeit zu geben, Grenzwerte über *berechenbare Folgen* reeller Zahlen zu bilden und damit weiterzurechnen. Die einzelnen Folgenglieder müssen dazu eindeutig, endlich, berechenbar und möglichst allgemein dargestellt werden – kurzum, sie sollten sich als *Folge der Ausgaben einer unendlichen Rechnung* ergeben, beschrieben durch Programm und Eingabe. Darüber einen Grenzwert zu bilden ist aber genau das Konzept der analytischen Berechnung! Ein Resultat lautet dann, daß die Grenzwertbildung über $\mathbb{R}$-berechenbare Folgen zu einer Klasse führt, die den Abschluß $\mathbb{R}$-analytischer Funktionen unter Hintereinanderausführung umfaßt. Diese Sichtweise legt es wiederum nahe, daß das UP nicht fest vorgegeben, sondern dynamisch erzeugt wird; der Typ der UP-Maschine beeinflußt die Mächtigkeit der Grenzwertbildung.

[11] Dr. Björn Schieffer verdanke ich die Anregung hierzu.

1.6.1. Semantik

Wir gehen von zwei Maschinen $\mathcal{M} := \mathcal{M}_\pi^\mathcal{R}$ und $\mathcal{M}' := \mathcal{M}_{\pi'}^\mathcal{R}$ aus, wobei das Programm π zum Zwecke des UP-Aufrufs den neuen Sonderbefehl `call(i,j)` mit $i, j \in \mathbb{N}$ enthalten darf. Dies ist eine Einschränkung, da nicht verschiedene Maschinentypen für Unterprogramme zur Verfügung stehen; zur Vereinfachung wollen wir dies allerdings annehmen. Andererseits ist es hinreichend allgemein, da $\mathcal{M}'$ z.B. eine universelle Maschine darstellen und eine beliebige, von $\mathcal{M}$ vorgegebene Funktion berechnen kann. Dabei hängen solche Eigenschaften wie Robustheit vom Programm π' ab. Ein Vergleich mit obigen Sichtweisen sollte klar machen, daß dieses Modell beiden gerecht wird.

Wir werden oft von $\mathcal{M}$ als dem *Hauptprogramm* und $\mathcal{M}'$ als dem *Unterprogramm* sprechen, d.h. die Maschine mit ihrem Programm identifizieren; dies ist solange unproblematisch, wie der Typ der Maschine klar ist. Die beiden Parameter i bzw. j des `call`-Befehls markieren die Positionen der Ein- bzw. Ausgabe des UPs im Rechenspeicher der ersten Maschine. Durch die Zusammenschaltung $\mathcal{M} \xrightarrow{\texttt{call}} \mathcal{M}'$ entsteht eine neue $\mathcal{R}$-Maschine, wobei die Semantik des Unterprogrammaufrufs wie folgt ist.

Sei $k = (\alpha, \beta, \gamma, \delta, \pi, x, y, z)$ eine Konfiguration von $\mathcal{M}$ mit $\pi_\beta = \texttt{call(i,j)}$. Dann wird $(z_i, z_{i+1}, \ldots)$ genau wie bei der Definition der Ausgabefunktion als Wort $w \in \mathcal{R}^*$ interpretiert und an das UP übergeben; es ist also $w := (z_{i+1}, \ldots, z_{i+z_i})$, falls $z_i \in \mathbb{N}$, und $w := \varepsilon$ sonst. Das Ergebnis[12] der Berechnung von $\mathcal{M}'$ angesetzt auf $\mathrm{in}(w)$ sei $r := \Phi_{\mathcal{M}'}(w)$; dieses wird analog zur Eingabefunktion in $(z_j, z_{j+1}, \ldots)$ abgelegt. Demnach ist $\Delta(k) := (\alpha, \beta + 1, \gamma, \delta, \pi, x, y, z')$ mit

$$z'_l = \begin{cases} |r| & \text{falls } l = j; \\ r_{l-j} & \text{falls } 1 \leq l - j \leq |r|; \\ z_l & \text{sonst.} \end{cases}$$

Für den Fall eines unendlichen Ergebniswortes vereinbaren wir $z'_j := -1$; allerdings erweist sich hier die Art und Weise der Übergabe als sehr unhandlich, da der gesamte Rechenspeicher ab der j. Stelle belegt wird.

In analoger Weise definiert man den Fall, daß $\mathcal{M} = \mathcal{M}_\pi^\mathbb{R}$ und $\mathcal{M}' = \mathcal{M}_{\pi'}^{\delta\text{-}\mathbb{Q}}$; da die Zahlenbereiche bei Ein- und Ausgaben beider Maschinen gleich sind, ergeben sich keine Schwierigkeiten. Wir verzichten an dieser Stelle auf weitere formale Definitionen und Bezeichnungen für diese Zusammenschaltungen und die damit berechenbaren Funktionen; letztere benennen wir anschaulich z.B. durch „$\mathbb{R}$-berechenbar $\xrightarrow{\texttt{call}}$ quasi stark δ-$\mathbb{Q}$-analytisch“ und setzen dies in Mengenklammern, wenn wir die Klasse meinen.

Man beachte, daß das Hautprogramm nur *endliche* Wörter an das UP übergeben kann, die Ergebnisse aber sehr wohl unendlich sein können; dies erscheint bei ge-

[12] Falls dieses nicht existiert, betrachten wir die Zusammenschaltung als nicht korrekt! Die Nachfolgekonfiguration ist in diesem Fall undefiniert und Δ wird zu einer partiellen Funktion; dies bläht den formalen Apparat unnötig auf.

nauerer Betrachtung aber als natürlich. Ferner haben wir keine Möglichkeit für *rekursive* Prozeduren vorgesehen, um die Situation einfach zu halten und den eingangs geschilderten Motivationen Rechnung zu tragen – wir wollen nicht rekursiv Grenzwerte bilden!

Für den Fall einer δ-$\mathbb{Q}$-Maschine als Hauptprogramm müssen wir noch vereinbaren, wie das Ergebnis des UP abgelegt werden soll. Wir entscheiden uns, es weiterhin im Rechenspeicher zu plazieren, allerdings mittels der Rundungsfunktion gemäß der aktuellen Präzision zu einer rationalen Zahl gerundet.

Anmerkung 1.11. Diese Art des UP-Aufrufs eignet sich nicht, um aus einer δ-$\mathbb{Q}$-analytischen Maschine heraus ein $\mathbb{R}$-berechenbares UP aufzurufen mit dem Ziel, auf diese Weise den Zahlenbereich des Hauptprogramms zu erweitern; dazu müßte man dem UP (auch) direkten Zugriff auf die reellen Eingaben erlauben. Man sieht jedoch leicht, daß die entstehende Zusammenschaltung genauso mächtig wäre wie eine $\mathbb{R}$-analytische Maschine; daher interessiert uns diese Kombination nicht weiter.

1.6.2. $\mathbb{R}$-berechenbare Hauptprogramme

Man bemerkt, daß gewisse Eigenschaften von $\mathbb{R}$-Maschinen sich unmittelbar auf oben definierte Zusammenschaltungen mit $\mathbb{R}$-berechenbaren Hauptprogrammen und einem festen Maschinentyp für die Unterprogramme übertragen. Ein besonders interessantes Resultat in dieser Hinsicht (vgl. auch Abschnitt 1.2.1) ist das

Lemma 1.26. *Die Menge $\{\mathbb{R}\text{-berechenbar} \xrightarrow{\texttt{call}} \mathbb{R}\text{-analytisch}\}$ ist abgeschlossen bezüglich Hintereinanderausführung, Kreuzprodukt und primitiver Rekursion. Dies gilt ebenso für δ-$\mathbb{Q}$-analytische Funktionen und deren Varianten als UP.*

Beweis. Wenn man mehrere Maschinen der Form $\mathcal{M} \xrightarrow{\texttt{call}} \mathcal{M}'$ zusammenschaltet, besteht die einzige Schwierigkeit darin, die verschiedenen UPs zu einem zu kombinieren. Dazu erweitert man die Eingaben an das UP um eine Funktionsnummer und führt dann eine Fallunterscheidung nach der gewünschten Funktion durch; die Hauptprogramme werden entsprechend angepaßt und dann mit den bekannten Methoden kombiniert. □

$\mathbb{R}$-analytische Prozeduren

Es macht keinen Sinn, aus einem $\mathbb{R}$-berechenbaren Hauptprogramm heraus ein ebensolches UP aufzurufen; $\mathbb{R}$-analytische Prozeduren stellen jedoch geeignete Kandidaten dar. Für diesen Abschnitt sei $\mathcal{F} := \{\mathbb{R}\text{-berechenbar} \xrightarrow{\texttt{call}} \mathbb{R}\text{-analytisch}\}$. Man sieht leicht, daß $\{\mathbb{R}\text{-berechenbar}\} \subsetneq \{\mathbb{R}\text{-analytisch}\} \subset \mathcal{F}$, sofern man sich auf endliche Wörter beschränkt. Aus Lemma 1.26 folgt dann unmittelbar

Korollar 1.27. Sei $f : \mathbb{R}^* \rightsquigarrow \mathbb{R}^*$ eine $\mathbb{R}^i$-analytische Funktion, die sich ohne Verwendung unendlicher Wörter als Zwischenergebnisse berechnen läßt. Dann ist $f \in \mathcal{F}$.

Indem man die Ideen aus Satz 1.21 und Lemma 1.25 kombiniert, ergibt sich – wie bereits im Anschluß an jenes Lemma erwähnt –, daß das Konvergenzproblem einer $\mathbb{R}$-analytischen Maschine $\mathbb{R}^3$-analytisch ist; insbesondere werden dabei keine unendlich langen Zwischenergebnisse benötigt. Eine vereinfachte Variante von Korollar 1.22 ergibt dann z.B., daß $3\mathtt{i}$ hintereinander geschaltete Maschinen genügen, um das Konvergenzproblem von $\mathtt{i}$ solchen zu entscheiden. Mit obigem Korollar folgt nun, daß das Konvergenzproblem $\mathbb{R}^{\mathtt{i}}$-analytischer Maschinen durch eine Funktion aus $\mathcal{F}$ gelöst werden kann. Laut Hierarchiesatz 1.4 ist diese aber nicht $\mathbb{R}^{\mathtt{i}}$-analytisch und wir erkennen

Korollar 1.28. $\forall\, \mathtt{i} \in \mathbb{N} : \ \mathcal{F} \not\subset \{\mathbb{R}^{\mathtt{i}}\text{-analytisch}\}$

Diese Fähigkeit zur mehrfachen Hintereinanderausführung läßt sich durch das Hauptprogramm in weiten Bereichen steuern und damit variabel gestalten; mit dieser Idee erhält man sogleich das

Lemma 1.29. $\mathcal{F} \not\subset \bigcup_{\mathtt{i}\in\mathbb{N}}\{\mathbb{R}^{\mathtt{i}}\textit{-analytisch}\}$

Beweis. Die Idee ist, eine universelle Maschine zu konstruieren, die das Konvergenzproblem für eine beliebige Hintereinanderschaltung einer *variablen* Zahl $\mathbb{R}$-analytischer Maschinen lösen kann. Dabei zeigt man, daß die Maschine vom Typ „$\mathbb{R}$-berechenbar $\xrightarrow{\texttt{call}}$ $\mathbb{R}$-analytisch“ ist. Andererseits kann eine solche Funktion nicht $\mathbb{R}^{\mathtt{i}}$-analytisch für irgendein festes $\mathtt{i}$ sein, ist also auch nicht in der Vereinigung enthalten.

Dazu benötigt man eine Kodierung von Programmen durch Wörter aus $\mathbb{R}^*$, die sich leicht durch eine $\mathbb{R}$-Maschine dekodieren bzw. interpretieren läßt. Diese dehnt man dann aus auf Folgen von Programmen und Argumenten und betrachtet universelle Maschinen für die variable Hintereinanderschaltung. Zuletzt benutzt man die Ideen aus der Herleitung von Korollar 1.28 zum Bau einer Maschine, die bei Eingabe einer Folge von $\mathtt{i}$ Programmen für analytische $\mathbb{R}$-Maschinen die Folge der $3\mathtt{i}$ Programme generiert, die das Konvergenzproblem lösen. □

Zusammen genommen folgt demnach

Satz 1.30. *Wenn man sich auf $\mathbb{R}$-analytische Funktionen $\mathtt{f} : \mathbb{R}^* \rightsquigarrow \mathbb{R}^*$ beschränkt, d.h. unendlich lange Wörter vermeidet, dann gilt*

$$\bigcup_{\mathtt{i}\in\mathbb{N}}\{\mathbb{R}^{\mathtt{i}}\textit{-analytisch}\} \subsetneq \{\mathbb{R}\textit{-berechenbar} \xrightarrow{\texttt{call}} \mathbb{R}\textit{-analytisch}\}$$

und die rechte Menge ist unter der Hintereinanderausführung abgeschlossen.

Wir haben also für die unendliche Hierarchie $\mathbb{R}^{\mathtt{i}}$-analytischer Funktionen eine abgeschlossene obere Schranke gefunden, die durch ein einfaches Berechnungsmodell gegeben ist. Aufgrund der Mächtigkeit dieses Berechnungsmodells ist die obere Schranke allerdings leider nicht scharf.

Quasi stark δ-ℚ-analytische Prozeduren

Nun wollen wir die Menge der erlaubten Prozeduren soweit einschränken, daß der Simulationssatz weiterhin gilt, d.h. daß die entstehende Zusammenschaltung sich simulieren läßt durch eine δ-ℚ-Maschine. Wie schon der Hierarchiesatz zeigt, ist die Hintereinanderausführung zweier δ-ℚ-analytischer Funktionen im allgemeinen nicht mehr von dieser Form. In Abschnitt 1.4 haben wir aber gesehen, daß die Teilmenge der quasi stark δ-ℚ-analytischen Funktionen in diesem Sinne abgeschlossen ist; daher sind dies geeignete Kandidaten für unsere UP. Tatsächlich ergibt sich

Satz 1.31. *Für eine Funktion* $f : \mathbb{R}^* \rightsquigarrow \mathbb{R}^*$ *gilt*

$$f \in \{\mathbb{R}\textit{-berechenbar} \xrightarrow{\texttt{call}} \textit{quasi stark } \delta\text{-}\mathbb{Q}\textit{-analytisch}\} \iff f \in \{\textit{quasi stark } \delta\text{-}\mathbb{Q}\textit{-analytisch}\}.$$

Beweis. Die Rückrichtung ist trivial; das Hauptprogramm liest das Eingabewort und übergibt es an das UP, welches die Funktion f realisiert, um anschließend dessen Ausgabe zu drucken.

Sei jetzt also $\mathcal{M}_\pi^{\mathbb{R}} \xrightarrow{\texttt{call}} \mathcal{M}_{\pi'}^{\delta\text{-}\mathbb{Q}}$ die Zusammenschaltung (mit quasi starkem Programm π'), welche f berechnet. Jeder endliche Berechnungspfad von π läßt sich in der Form

$$\sigma = \sigma_1 \cdot \texttt{call} \cdot \sigma_2 \cdot \texttt{call} \cdot \ldots \cdot \texttt{call} \cdot \sigma_n$$

zerlegen, wobei die Teilpfade σ_i keine UP-Aufrufe mehr enthalten. Jedem Teilpfad σ_i läßt sich gemäß dem Simulationssatz 1.19 eine äquivalente quasi stark δ-ℚ-analytische Maschine $\mathcal{M}_i$ zuordnen, das UP ist bereits durch eine solche Maschine $\mathcal{M}'$ gegeben. Gemäß Lemma 1.18 läßt sich daher der gesamte Pfad σ durch eine quasi stark δ-ℚ-analytische Maschine simulieren.

Als Schwierigkeit ergibt sich hier, daß der Pfad σ nicht von vorne herein bekannt ist, sondern erst ermittelt werden muß. Dabei wird mit zunehmender Genauigkeit der Simulation die eine oder andere Verzweigungsbedingung anders ausgewertet als zuvor, so daß sich der gesamte folgende Suffix ändert. Allerdings kann dies nur endlich oft passieren, wie schon im Beweis zu Satz 1.19 erläutert wurde.

Die simulierende Maschine $\mathcal{M}$ verwaltet daher einen Stapel von quasi stark δ-ℚ-analytischen Maschinen $\mathcal{M}_i$ für den Teilpfad σ_i und $\mathcal{M}'_i$ für den i. Unterprogrammaufruf. Das Präzisionsregister von $\mathcal{M}_1$, der ersten Maschine in der Kette, bestimmt den Wert von δ von $\mathcal{M}$. Im wesentlichen gibt es zwei Phasen bei der Simulation, eine *Expansions-* und eine *Konsolidierungsphase*.

Zu Anfang und immer dann, wenn sich der Suffix von σ ändert – wobei der Rest des Stapels verworfen wird –, befindet sich die $\mathcal{M}$ in der Expansionsphase und legt neue Maschinen auf dem Stapel an. Jede neue Maschine wird für soviele Runden simuliert, wie die bisherige Approximation ihrer Eingabe dies zuläßt[13].

Wenn der Pfad bis zum `end` abgearbeitet wurde, beginnt die Konsolidierungsphase, in welcher die Präzision δ erhöht und jeweils die Simulation des Präfix von σ

[13]Man kann o.B.d.A. annehmen, daß die Fehlerschranke stets kleiner als 1 ist.

weiter vorangetrieben wird, auf den sich die höhere Genauigkeit auswirkt. Dadurch konvergiert nun auch die Ausgabe wie gewünscht und $\mathcal{M}$ ist quasi stark δ-$\mathbb{Q}$-analytisch. □

Der obige Satz liefert eine starke Charakterisierung der Klasse {$\mathbb{R}$-berechenbar $\xrightarrow{\texttt{call}}$ quasi stark δ-$\mathbb{Q}$-analytisch} als Einschränkung von {quasi stark δ-$\mathbb{Q}$-analytisch} auf Funktionen, die auf endlichen Wörtern operieren. Will man eine weitere, echte Einschränkung, die fraktale Strukturen ausschließt, so kann man sich auf Prozeduren beschränken, welche stetig differenzierbare Funktionen berechnen. Dann ergibt sich analog zu 1.2, daß die Hausdorff-Dimension wiederum ganzzahlig ist; wir wollen hier aber nicht weiter auf Details eingehen.

1.6.3. Quasi stark δ-$\mathbb{Q}$-analytische Hauptprogramme

Wie bereits angedeutet, kann in unserem Modell der UP-Technik die Zusammenschaltung einer δ-$\mathbb{Q}$- mit einer $\mathbb{R}$-Maschine, $\mathcal{M}_\pi^{\delta\text{-}\mathbb{Q}} \xrightarrow{\texttt{call}} \mathcal{M}_{\pi'}^{\mathbb{R}}$, nicht erklärt werden. In einer vernünftigen und geeigneten Erweiterung des Modells würde sich allerdings zeigen, daß eine solche Kombination genau die Mächtigkeit einer $\mathbb{R}$-analytischen Maschine hat. Das ist auch genau das, was man erwarten sollte von einer Maschine, die sowohl reelle Zahlen exakt verarbeiten als auch unendliche konvergente Rechnungen ausführen kann.

Interessanter ist da schon die Frage, wie sich das Hinzufügen eines δ-$\mathbb{Q}$-analytischen UPs auf eine analytische δ-$\mathbb{Q}$-Maschine auswirkt. Im allgemeinen läßt sich eine solche Zusammenschaltung schlecht mit den bereits bekannten Klassen berechenbarer Funktionen in Beziehung setzen; in einem wichtigen Spezialfall ergibt sich aber ein schöner Satz.

Satz 1.32.

$$\{\textit{quasi stark } \delta\text{-}\mathbb{Q}\textit{-analytisch} \xrightarrow{\texttt{call}} \textit{stark } \delta\text{-}\mathbb{Q}\textit{-analytisch}\} = \{\textit{quasi stark } \delta\text{-}\mathbb{Q}\textit{-analytisch}\}$$

Beweis. Die eine Inklusion „$\supset$“ ist klar, da das Hauptprogramm keinen `call`-Befehl enthalten muß. Für die umgekehrte Behauptung „$\subset$“ betrachten wir eine konkrete Zusammenschaltung $\mathcal{M}_\pi^{\delta\text{-}\mathbb{Q}} \xrightarrow{\texttt{call}} \mathcal{M}_{\pi'}^{\delta\text{-}\mathbb{Q}}$, wobei die beiden Programme π bzw. π' quasi stark bzw. stark seien.

Man beobachtet zunächst, daß alle Eingabe an das UP rational und daher exakt bekannt sind. Als Rückgabewert wird eine Approximation des Ergebnisses gemäß der aktuellen Präzision erwartet; diese kann jedoch in endlicher Zeit errechnet werden. Hierbei geht ein, daß die Fehlerschranken des UP stets korrekt sind und das Hauptprogramm robust ist, d.h. nicht von der speziellen Art der Rundung abhängt. Daher kann diese Zusammenschaltung leicht durch eine quasi stark δ-$\mathbb{Q}$-analytische Maschine ohne Prozeduren ersetzt werden. □

Korollar 1.33.

$$\{\text{stark } \delta\text{-}\mathbb{Q}\text{-analytisch} \xrightarrow{\texttt{call}} \text{stark } \delta\text{-}\mathbb{Q}\text{-analytisch}\} = \{\text{stark } \delta\text{-}\mathbb{Q}\text{-analytisch}\}$$

Beweis. Die Eigenschaft der Fehlerschranke, *stets* korrekt zu sein, überträgt sich unmittelbar vom Hauptprogramm auf die Zusammenschaltung. □

Es zeigt sich also, daß stark δ-$\mathbb{Q}$-analytische Funktionen nicht nur unter der endlichen Hintereinanderausführung, sondern auch bzgl. der Zusammenschaltung gemäß obiger UP-Technik abgeschlossen sind.

Fazit

Wir haben ausgehend vom abstrakten Begriff einer mathematischen Maschine verschiedene Formen von Registermaschinen hergeleitet, die im wesentlichen dem Studium der reellen Berechenbarkeit dienen. Ein wichtiger Aspekt hierbei ist das Verhältnis zwischen $\mathbb{R}$- und $\mathbb{Q}$-Maschinen und die Frage, inwieweit man präzise reelle Arithmetik durch unendliche konvergente Berechnungen über den rationalen Zahlen ersetzen kann. Dabei wird ein systematischer Überblick gegeben über die Hierarchie der betrachteten Klassen, es wird bewiesen, warum diese Inklusionen echt sind, und untersucht, wie sich diese Klassen unter Komposition verhalten.

Der Einfluß der verwendeten Rundungsfunktion spielt eine wichtige Rolle beim Verständnis dieser Zusammenhänge. Es zeigt sich, daß einstellige $\mathbb{R}$-analytische Funktionen durch geeignete Wahl der Rundungsfunktion – und im allgemeinen nur dadurch – δ-$\mathbb{Q}$-analytisch sind. Für mehrstellige Funktionen gilt dies jedoch nicht, wie das Beispiel $\chi^+_{\mathbb{Q}}$ zeigt.

Eine wichtige Klasse bilden die quasi stark δ-$\mathbb{Q}$-analytischen Funktionen, die abgeschlossen unter der Hintereinanderausführung sind und gemäß dem Simulationssatz die $\mathbb{R}$-berechenbaren umfassen. Ferner enthalten sie auch fraktale Strukturen, woraus sich unmittelbar die Trennung beider Klassen ergibt.

Analog zu dem bekannten Halteproblem der klassischen Rekursionstheorie kann man bei analytischen Maschinen Konvergenzprobleme betrachten. Es zeigt sich, daß jeweils i hintereinander geschaltete $\mathbb{R}$-Maschinen bzw. (robuste) δ-$\mathbb{Q}$-Maschinen das Konvergenzproblem für eine ebensolche Anordnung nicht entscheiden können; daraus ergibt sich der Hierarchiesatz. Andererseits können mehr als i Maschinen ein solches Problem lösen, die genaue Anzahl hängt jedoch davon ab, ob man sich auf robuste Maschinen oder endlich lange Zwischenergebnisse beschränkt oder nicht. Durch Kodierung können unendliche Zwischenergebnisse vermieden werden, doch erfordert dies im allgemeinen mehr Maschinen.

Zuletzt haben wir eine Unterprogrammtechnik für unsere Maschinen eingeführt und deren Auswirkungen studiert. Beschränkt man sich auf endliche Wörter, d.h. auf $\mathbb{R}^*$, so stellt $\{\mathbb{R}\text{-berechenbar} \xrightarrow{\texttt{call}} \mathbb{R}\text{-analytisch}\}$ eine abgeschlossene obere Schranke für die unendliche Hierarchie $\mathbb{R}^i$-analytischer Funktionen dar. Ferner ist die Klasse der quasi stark δ-$\mathbb{Q}$-analytischen Funktionen auch abgeschlossen unter der Erweiterung mit stark δ-$\mathbb{Q}$-analytischen Prozeduren.

2. Komplexität

Wir betrachten den Verbrauch der beiden Ressourcen Zeit und Platz durch Berechnungen und das Verhältnis verschiedener Maschinentypen unter dem klassischen Aspekt der polynomiellen Verknüpftheit. Als Modelle der Berechenbarkeit sollen in diesem Kapitel sowohl $\mathbb{R}$-Maschinen als auch δ-$\mathbb{Q}$-analytische Maschinen betrachtet werden. Die beiden wichtigsten Resultate sind, daß alle Berechnungen mit konstantem Platz ausgeführt werden können und daß unsere speziellen Registermaschinen zu dem ursprünglichen Modell von BLUM, SHUB, SMALE gleichwertig sind. Genauer bedeutet dies jeweils gleichen Platzbedarf und einen höchstens quadratischen Zeitverlust, der mit den indizierten Speicherzugriffen zusammenhängt.

2.1. Komplexitätsmaße für $\mathcal{R}$-Maschinen

Wir werden erklären, was unter den Kosten einer Berechnung einer $\mathcal{R}$-Maschine zu verstehen ist, und zwar für die beiden Aspekte *Zeit* und *Platz*. Daraus leitet sich die Zeit- bzw. Platzkomplexität einer $\mathcal{R}$-Maschine in Abhängigkeit von der Eingabegröße als Abschätzung für den schlechtesten Fall (*worst-case*) ab. Für Maschinen, die rein mit rationalen Zahlen arbeiten, empfiehlt sich die bekannte *logarithmische* Komplexität von Registermaschinen, die sich bereits über den natürlichen Zahlen bewährt hat; wir verweisen für eine Definition z.B. auf REISCHUK [40]. Bei Maschinen, die mit reellen (oder komplexen) Zahlen operieren, eignet sich nur das *uniforme* Kostenmaß; dies betrifft auch unsere δ-$\mathbb{Q}$-Maschinen. Im folgenden bezeichne $\max M$ für eine Teilmenge $M \subset \mathbb{N}$ das Maximum über diese Menge mit zwei Ergänzungen: $\max M := 0$, falls $M = \emptyset$, und $\max M := \infty$, falls M unbeschränkt.

Definition 2.1 (Kostenmaße für $\mathbb{R}$-Maschinen). Sei $\mathcal{M}$ eine $\mathbb{R}$-Maschine, $x \in \mathbb{D}_{\mathcal{M}}$ und $b = (k_i)_{i=0}^{l}$ die Berechnung von $\mathcal{M}$ angesetzt auf $\mathtt{in}(x)$ mit Länge $l \in \widehat{\mathbb{N}}$. Ferner seien $\{z_{i_1}, \ldots, z_{i_m}\}$ mit $m \in \widehat{\mathbb{N}}$ die Speicherzellen von $\mathcal{M}$, auf die im Verlauf der Berechnung zugegriffen wird.

Dann bezeichne $\mathtt{time}_{\mathcal{M}}(x) := l$ die *uniformen Zeitkosten* und $\mathtt{space}_{\mathcal{M}}(x) := m$ die *uniformen Platzkosten* von $\mathcal{M}$ bei Eingabe x. Die *uniforme Zeit-* und *Platzkomplexität* von $\mathcal{M}$ sei definiert wie folgt:

$$\mathtt{time}_{\mathcal{M}} : \mathbb{N} \to \widehat{\mathbb{N}},\ n \mapsto \max\{\mathtt{time}_{\mathcal{M}}(x) \mid x \in \mathbb{R}^n \cap \mathbb{D}_{\mathcal{M}}\},$$
$$\mathtt{space}_{\mathcal{M}} : \mathbb{N} \to \widehat{\mathbb{N}},\ n \mapsto \max\{\mathtt{space}_{\mathcal{M}}(x) \mid x \in \mathbb{R}^n \cap \mathbb{D}_{\mathcal{M}}\}.$$

□

Anmerkung 2.1. Wir unterscheiden im Namen nicht zwischen der *Kosten*funktion $\mathbb{D}_{\mathcal{M}} \to \widehat{\mathbb{N}}$ und der *Komplexitäts*funktion $\mathbb{N} \to \widehat{\mathbb{N}}$. Aus dem Zusammenhang und insbesondere dem Typ des Arguments sollte jedoch stets klar sein, welche Variante gemeint ist.

In analoger Weise lassen sich Kostenmaße für δ-$\mathbb{Q}$-Maschinen definieren, wobei bei analytischen Berechnungen natürlich stets $\mathtt{time}_{\mathcal{M}}(x) = \infty$ gilt. Allerdings kann $\mathtt{space}_{\mathcal{M}}(x)$ selbst bei unendlich langen Eingaben $x \in \mathbb{R}^{\omega}$ eine endliche Größe sein wegen unseres *off-line*-Modells von Maschinen mit eigenem Ein- und Ausgabeband.

Beispiel 2.1 (Lineares Programmieren). BLUM diskutiert in [3] eingehend dieses Problem (LP), das formal wie folgt lautet. Sei $A \in \mathbb{R}^{m \times n}$ eine Matrix und $b \in \mathbb{R}^m$ sowie $c \in \mathbb{R}^n$ Vektoren. Maximiere $c^T x$ (Zielfunktion) für $x \in \mathbb{R}^n$, wobei $x \geq 0$ und $Ax \leq b$ (Nebenbedingungen). Der bekannte *Simplex*-Algorithmus für LP wurde 1947 von DANTZIG erfunden; seine uniforme Komplexität als Funktion der Eingabegrößen m und n ist exponentiell. □

Beispiel 2.2 (Entier-Funktion). Die bekannte Funktion $\lfloor . \rfloor : \mathbb{R} \to \mathbb{Z}$ mit

$$\mathbb{R} \ni x \mapsto \lfloor x \rfloor := \max\{z \in \mathbb{Z} \mid z \leq x\}$$

rundet eine reelle Zahl ab auf eine ganze. Die uniformen Zeit*kosten* bei Eingabe x sind bei geschickter Implementierung $\Theta(\log(|x| + 1))$, und dies ist laut [3] auch optimal; daher ist die uniforme Zeit*komplexität* als Funktion der Eingabegröße (hier: 1) unbegrenzt. Die Platzkomplexität ist hingegen konstant. Das Problem in diesem Falle ist, daß die Funktion mit Hilfe der ganzen Zahlen definiert ist; deshalb ist das „uniforme Eingabekostenmaß" hier nicht angemessen.

Damit in Zusammenhang steht die charakteristische Funktion der ganzen Zahlen

$$\chi_{\mathbb{Z}} : \mathbb{R} \to \mathbb{B}, \; x \mapsto \begin{cases} 1 & \text{falls } x \in \mathbb{Z}; \\ 0 & \text{sonst.} \end{cases}$$

Es ist klar, daß $\chi_{\mathbb{Z}}(x)$ auf das Prädikat $x \stackrel{?}{=} \lfloor x \rfloor$ zurückgeführt und nach dem oben gesagten in Zeit $\Theta(\log(|x| + 1))$ berechnet werden kann. Wie man aus den Resultaten von BEN-OR [2] leicht schließt, ist diese Zeit auch optimal. □

Diese Zeit- und Platzkomplexitäten sind so definiert, daß die folgenden einfachen Beziehungen gelten.

Lemma 2.1. *Sei $\mathcal{M}$ eine beliebige $\mathbb{R}$- bzw. δ-$\mathbb{Q}$-Maschine, $x \in \mathbb{D}_{\mathcal{M}}$ und $\mathtt{cost} \in \{\mathtt{time}, \mathtt{space}\}$. Dann gilt für die Beziehung zwischen Kosten und Komplexität*

$$\mathtt{cost}_{\mathcal{M}}(x) \leq \mathtt{cost}_{\mathcal{M}}(|x|).$$

Für den Zusammenhang zwischen dem Platzbedarf bzw. der Länge der Ausgabe und der Dauer einer Rechnung gilt

$$\mathtt{space}_{\mathcal{M}}(x) \leq \mathtt{time}_{\mathcal{M}}(x) \geq |\Phi_{\mathcal{M}}(x)|.$$

□

VIERKE [44, Seite 18, Satz 3] und BLUM, SHUB, SMALE [9, Seite 28, Fußnote 4] beschreiben und beweisen unabhängig voneinander einen Satz, der sich in unserer Terminologie wie folgt auf analytische Berechnungen verallgemeinern läßt. Er zeigt, daß das uniforme Platzmaß alleine auch bei $\mathbb{R}$-Maschinen nicht sehr aussagekräftig ist.

Satz 2.2. *Sei $\mathcal{M}$ eine beliebige $\mathbb{R}$- bzw. δ-$\mathbb{Q}$-Maschine. Dann gibt es eine zu $\mathcal{M}$ äquivalente Maschine $\mathcal{M}'$ mit $\mathtt{space}_{\mathcal{M}'} = \Theta(1)$.*

Beweis. Unser Beweis folgt der Idee von VIERKE und korrigiert die Lücken in dessen Ausarbeitung. Die (reguläre oder analytische) Berechnung von $\mathcal{M}$ wird (gegebenenfalls rundenweise) durch $\mathcal{M}'$ simuliert, wobei lediglich die Folge der Inhalte von Programmzähler $\beta \in [1 : N]$ und Indexregister $\gamma \in \mathbb{N}$ in je einem Register protokolliert werden. Zur Simulation von $\mathtt{if}\ \alpha > 0\ \mathtt{then\ goto}\ m\ \mathtt{else\ goto}\ n$ und $y_i := \alpha$ muß der Wert des Akkumulators bestimmt werden, der sich durch eine rationale Funktion aus den Eingaben ergibt. Zu diesem Zweck wird der bisherige Programmpfad zurückverfolgt und iterativ eine Beschreibung dieser Funktion als Quotient zweier Polynome in einem Register aufgebaut. Deren einzige Variablen sind am Ende die x_i mit festen $i \in \mathbb{N}$, und so kann der Quotient leicht mit drei Registern ausgewertet werden.

Nun muß man nur noch einsehen, daß Wörter über einem abzählbaren Alphabet in einem Register kodiert werden können und daß die notwendigen Stringoperationen mit konstantem Speicherbedarf möglich sind. Dies folgt unmittelbar daraus, daß jede Turing-berechenbare Funktion mit zwei Registern ausgewertet werden kann [37, Seite 103]. $\square$

Wir wollen nun noch den Zusammenhang zwischen $\mathcal{R}$-Maschinen mit verschieden vielen Indexregistern untersuchen und diese dann verwenden, um den von einer Maschine verwendeten Adreßraum zu transformieren. Das ist z.B. in vielen Simulationsbeweisen notwendig, wo zusätzlich zum Adreßraum der simulierten Maschine weitere Daten unterzubringen sind.

Lemma 2.3. *Eine $\mathcal{R}$-Maschine mit k Indexregistern kann bei quadratischem Zeitverlust durch eine äquivalente $\mathcal{R}$-Maschine mit* einem *Indexregister simuliert werden.*

Beweis. Sei $\mathcal{M}$ eine $\mathcal{R}$-Maschine mit k Indexregistern. Wir konstruieren eine äquivalente $\mathcal{R}$-Maschine $\mathcal{M}'$ mit einem Indexregister, die $\mathcal{M}$ simuliert. Die Indexregister γ_i werden in z_i' abgelegt, der eigentliche Rechenspeicher ab Adresse c, die sich aus dem benötigten Hilfsspeicherplatz ergibt. Alle konstanten Programmadressen werden angepaßt etc. Das Kernstück der Simulation besteht darin, bei jedem indizierten Speicherzugriff (auch auf das Ein- bzw. Ausgabeband) zuerst das gewünschte Indexregister γ_i nach γ' zu laden – dies benötigt Zeit proportional zu γ_i – und gegebenenfalls um c zu erhöhen.

Aufgrund der eingeschränkten Möglichkeiten, Indexregister zu verändern, gilt nach n Schritten von $\mathcal{M}$, daß $\gamma_i \in [0 : n]$ für alle $1 \leq i \leq n$. Daher kann jeder Schritt von $\mathcal{M}$ in Zeit $O(\mathtt{time}_{\mathcal{M}})$ simuliert werden; der zusätzliche Speicherbedarf ist konstant. $\square$

Lemma 2.4 (Lineare Adreßtransformation). *Sei $\mathcal{M}$ eine $\mathbb{R}$- bzw. δ-$\mathbb{Q}$-Maschine mit k Indexregistern. Dann kann $\mathcal{M}$ mit konstantem Zeit- und ohne Platzverlust durch eine $\mathbb{R}$- bzw. δ-$\mathbb{Q}$-Maschine $\mathcal{M}'$ mit $2k$ Indexregistern simuliert werden, welche nur die Rechenspeicherzellen z_{ai+b} benutzt, wobei $a, b, i \in \mathbb{N}$ mit $a > 0$.*

Beweis. Wir beschreiben die Simulation nur für $k = 1$. Indexregister γ_1 ersetzt γ und wird für Ein-/Ausgabeoperationen verwendet; γ_2 variiert über die transformierten Adressen und dient Rechenspeicherzugriffen. Alle konstanten Adressen z_i im Programm werden angepaßt. Ferner wird $\gamma_1 := 0$ ergänzt durch $\gamma_2 := b$ und $\gamma_1 := \gamma_1 + 1$ durch $\gamma_2 := \gamma_2 + a$ (d.h. durch die naheliegende Befehlsfolge mit der entsprechenden Wirkung). $\gamma_1 := \gamma_1 \dot{-} 1$ wird in diesem Sinne durch die Folge $\gamma_2 := \gamma_2 \dot{-} a$; $\gamma_2 := \gamma_2 \dot{-} b$; $\gamma_2 := \gamma_2 + b$ erweitert. Dadurch wird sozusagen der Fall $0 \dot{-} 1$ abgefangen. Diese Simulation macht höchstens $(a + 2b + 1)$-mal soviele Schritte wie die ursprüngliche Maschine. $\square$

Anmerkung 2.2. Für *gedächtnislose* δ-$\mathbb{Q}$-Maschinen, welche bei jedem `next δ` ihren Rechenspeicher löschen, kann man maximale und asymptotische (im Sinne eines *limes superior*) Zeit- und Platzkosten pro Runde betrachten. Es ergibt sich dann, daß für konstruierbare Zeit- bzw. Platzschranken die beiden Maße äquivalent sind. Ferner kommt der Simulationssatz mit gedächtnislosen Maschinen aus, deren Kosten mit denen der ursprünglichen Maschine *polynomiell verknüpft* sind.

2.2. Vergleich mit dem Modell von Blum, Shub, Smale

Die Autoren von [9] definieren *Maschinen über einem Ring $\mathcal{R}$*, wobei $\mathcal{R}$ ein angeordneter, kommutativer Ring mit Eins sei[1], der $\mathbb{Z} \subset \mathcal{R}$ als natürlichen Unterring enthält. Der Kontrollfluß der Maschinen wird von einem Graphen bestimmt; die Rechenschritte bestehen im Wesentlichen aus polynomiellen bzw. rationalen Funktionen; Berechnungen werden als Iterationen eines Endomorphismus des Konfigurationsraums aufgefaßt. Wir werden nun versuchen, diese Maschinen in einer möglichst einfachen Normalform als abstrakte Maschinen zu präsentieren, und beschränken uns dabei auf die sogenannten *unendlich dimensionalen* Maschinen, d.h. solche mit unbeschränktem Speicher, die Funktionen $\mathcal{R}^* \rightsquigarrow \mathcal{R}^*$ berechnen.

Der *Zustandsraum (state space)* S ist die unendliche direkte Summe $\mathcal{R}^\infty$ von $\mathcal{R}$ mit sich selbst; seine Elemente $s = (s_1, s_2, \dots)$ – die wir *Zustände* nennen – sind unendliche Wörter über $\mathcal{R}$, bei denen nur endlich viele Komponenten von Null verschieden sind. Die beiden ersten Koordinaten s_1, s_2 spielen die Rolle von Indexregistern, sie dürfen nur in der Form $s_i := 1$ bzw. $s_i := s_i + 1$ verändert werden. Eine *polynomielle* oder *rationale* Funktion auf S ist die Fortsetzung einer solchen Funktion auf $\mathcal{R}^n$ für ein $n \in \mathbb{N}$, die alle weiteren Komponenten unberücksichtigt und unverändert läßt und die Bedingung hinsichtlich s_1, s_2 erfüllt. Wir sagen, daß diese genau auf

[1] Die Bedingung „kommutativ“ scheint nicht notwendig zu sein.

die Koordinaten *zugreift*, welche in nichttrivialer Weise verändert werden oder das Funktionsergebnis beeinflussen[2].

Anmerkung 2.3. Die unendliche direkte Summe $\mathcal{R}^\infty$ von $\mathcal{R}$ mit sich selbst hat starke Ähnlichkeiten mit der Menge $\mathcal{R}^*$ der endlichen Folgen über $\mathcal{R}$. Der wichtigste Unterschied liegt darin, daß $(1) \neq (1, 0)$ in $\mathcal{R}^*$, während es in $\mathcal{R}^\infty$ für beide nur eine Darstellung als $(1, 0, 0, \dots)$ gibt. Daraus resultiert auch ein anderer Längenbegriff, da in $\mathcal{R}^\infty$ abschließende Nullen ohne Bedeutung sind. Dennoch kann man durch eine geeignete Kodierung, die z.B. ans Ende eines Wortes eine zusätzliche 1 anfügt, beliebige Wörter über $\mathcal{R}$ in $\mathcal{R}^\infty$ darstellen und manipulieren.

Das *Programm* der Maschine ist gegeben durch einen gerichteten, zusammenhängenden Graphen G, dessen Knoten $1, \dots, N$ mit einem Typ und eventuell einer Abbildung markiert sind; auch manche Kanten sind markiert. Gemäß der Typen der Knoten ergeben sich folgende weitere Bedingungen:

1. Der einzige *Eingabeknoten* ist 1; er hat keine eingehende und eine ausgehende Kante. Die mit diesem Knoten ursprünglich verbundene Eingabefunktion wird bei unseren abstrakten Maschinen getrennt behandelt; die Wirkung bleibt aber erhalten.

2. Der einzige *Ausgabeknoten* ist N; er hat eine eingehende und keine ausgehende Kante. Auch hier gilt, daß die Ausgabefunktion gesondert beschrieben wird.

3. *Berechnungsknoten* haben eine ausgehende Kante und sind mit einer polynomiellen oder rationalen Funktion auf S markiert.

4. *Verzweigungsknoten* haben zwei ausgehende Kanten, die mit *wahr* bzw. *falsch* beschriftet sind, sowie die implizit zugeordnete und für alle Verzweigungsknoten gleiche Bedingung $s_3 \geq 0$.

5. *Indexknoten (fifth nodes)* zur indizierten Adressierung des Zustandsraums S haben eine ausgehende Kante. Ihnen ist eine spezielle Funktion zugeordnet, die das Kopieren der von s_2 bezeichneten Komponente in die durch s_1 adressierte bewirkt:
$$s_i := \begin{cases} s_{s_2} & \text{falls } i = s_1; \\ s_i & \text{sonst.} \end{cases}$$

Die Menge der Konfigurationen dieser abstrakten Maschine ergibt sich dann wie

[2] Präzise: Sei $f : S \rightsquigarrow S$ die Fortsetzung von $f^{(n)} = (f_1^{(n)}, \dots, f_n^{(n)}) : \mathcal{R}^n \rightsquigarrow \mathcal{R}^n$ mit minimalem n. f greift genau auf die Koordinaten zu, welche nicht lediglich identisch kopiert werden, d.h. welche nicht nur zur Definition von $f_m^{(n)}(s_1, \dots, s_n) := s_m$ benutzt werden.

folgt, wobei analytische Berechnungen nicht vorgesehen sind:

$$\begin{aligned} K &:= [1 : N] \times S, \\ K_a &:= \{(n, s) \in K \mid n = 1,\ s_1 = s_2 = 0\}, \\ K_e &:= \{(n, s) \in K \mid n = N\}, \\ K_z &:= \emptyset. \end{aligned}$$

Die Eingabefunktion $in : \mathcal{R}^* \to K_a$ bildet ein Wort $r := (r_1, \ldots, r_n)$ ab auf $in(r) := (1, s)$ mit

$$s_i := \begin{cases} n & \text{falls } i = 4; \\ r_j & \text{falls } i = 2j + 1 \text{ mit } 1 \leq j \leq n; \\ 1 & \text{falls } i \in \{1, 2\}; \\ 0 & \text{sonst.} \end{cases}$$

und $out : K \to \mathcal{R}^*$ interpretiert die Teilfolge $(s_3, s_5, \ldots)$ als Element von $\mathcal{R}^\infty$, d.h. das Ergebnis ist der längste Präfix ohne abschließende Nullen bzw. eine einzelne Null. Durch die Angabe der Länge können beliebige Wörter als Eingabe verwendet werden, allerdings findet sich dieses Konzept nicht für die Ausgabe.

Der Nachfolger eines Knoten ist durch den Graphen und die Verzweigungsbedingung eindeutig festgelegt; nur Berechnungs- und Indexknoten verändern den Zustand der Maschine. Es wird wieder angenommen, daß der Kontrollfluß durch Verzweigungen stets so gesteuert wird, daß keine rationale Funktion an einer undefinierten Stelle ausgewertet wird. Damit ist die Übergangsfunktion Δ hinreichend beschrieben.

Definition 2.2. Gegeben seien ein Ring $\mathcal{R}$ und ein Graph G mit den zuvor genannten Eigenschaften. Dann heißt die gemäß obiger Konstruktion dadurch eindeutig festgelegte abstrakte Maschine $\mathcal{M}_G^{\mathcal{R}} = (K, K_a, K_e, K_z, \Delta, \mathcal{R}, in, out)$ die *$\mathcal{R}$-Graphmaschine zum Graphen* G. □

Anmerkung 2.4. Die von Blum, Shub, Smale beschriebenen Maschinen über einem Ring, die wir hier zur Unterscheidung Graphmaschinen nennen wollen, weisen in vielen Details bereits Ähnlichkeiten zu Registermaschinen auf. Diese werden besonders deutlich, wenn man die entsprechenden Begriffe wie indizierte Adressierung statt *fifth node computation* verwendet. Durch ihre Rolle bei Verzweigungsbedingungen hat die Zustandskomponente s_3 auch eine gewisse Verwandtschaft mit einem Akkumulator. Dennoch erscheint das von der mathematischen Sichtweise geprägte und von der Theorie dynamischer Systeme beeinflußte Beschreibungsmodell vom Standpunkt der Informatik als eher schwerfällig. Daher bevorzugen wir zur Definition unseres Begriffs von Berechenbarkeit und Komplexität den Zugang über Registermaschinen.

Blum, Shub, Smale definieren die Zeitkosten ihrer Maschinen für beliebige Ringe mit Hilfe des Begriffs der *Höhe (height)* eines Ringelements, der jedoch nur für $\mathbb{Z}$ und $\mathbb{R}$ erklärt wird. Er ergibt für diese beiden Fälle das logarithmische bzw. uniforme Kostenmaß. Wir beschränken uns wie schon zuvor auf den Fall $\mathcal{R} = \mathbb{R}$.

Definition 2.3 (Kostenmaße für $\mathbb{R}$-Graphmaschinen). Sei $\mathcal{M}$ eine $\mathbb{R}$-Graphmaschine, $x \in \mathbb{D}_{\mathcal{M}}$ und $b = (k_i)_{i=0}^{l}$ die Berechnung von $\mathcal{M}$ angesetzt auf $\mathtt{in}(x)$ mit Länge $l \in \widehat{\mathbb{N}}$. Ferner seien $\{s_{i_1}, \ldots, s_{i_m}\}$ mit $m \in \widehat{\mathbb{N}}$ die Zustandskomponenten von $\mathcal{M}$, auf die im Verlauf der Berechnung zugegriffen wird.

Dann bezeichne $\mathtt{time}_{\mathcal{M}}(x) := l$ die *uniformen Zeitkosten* und $\mathtt{space}_{\mathcal{M}}(x) := m$ die *uniformen Platzkosten* von $\mathcal{M}$ bei Eingabe x. Die *uniforme Zeit-* und *Platzkomplexität* von $\mathcal{M}$ sei definiert wie folgt:

$$\mathtt{time}_{\mathcal{M}} : \mathbb{N} \to \widehat{\mathbb{N}},\ n \mapsto \max\{\mathtt{time}_{\mathcal{M}}(x) \mid x \in \mathbb{R}^n \cap \mathbb{D}_{\mathcal{M}}\},$$
$$\mathtt{space}_{\mathcal{M}} : \mathbb{N} \to \widehat{\mathbb{N}},\ n \mapsto \max\{\mathtt{space}_{\mathcal{M}}(x) \mid x \in \mathbb{R}^n \cap \mathbb{D}_{\mathcal{M}}\}.$$

□

Beispiel 2.3 (Identität). Die $\mathcal{R}$-Graphmaschine $\mathcal{M}$, die nur aus einem Ein- und einem Ausgabeknoten besteht, berechnet die Identität $\mathtt{id} : \mathcal{R}^\infty \to \mathcal{R}^\infty$ und hat Zeitkosten $\mathtt{time}_{\mathcal{M}} = 1$ und Platzkosten $\mathtt{space}_{\mathcal{M}} = 0$. Dieses unintuitive Resultat ergibt sich, weil Ein- und Ausgaben im Zustandsraum an derselben Stelle liegen; eine Maschine, die nun nichts tut, „berechnet" demnach die Identität aufgrund der Konstruktion des Ein-/Ausgabeverhaltens. □

Der Vergleich von Kostenmaßen, z.B. zwischen Turing- und Registermaschinen, erfolgt üblicherweise wegen der besonderen Bedeutung polynomieller Kosten im Sinne der folgenden

Definition 2.4. Zwei Funktionen $t, t' : M \to \widehat{\mathbb{N}}$, wobei M eine Menge sei, heißen *polynomiell verknüpft*, wenn es ein Polynom $p(X) \in \mathbb{Z}[X]$ gibt mit

$$\forall\, x \in M :\ t(x) \leq p(t'(x))\ \wedge\ t'(x) \leq p(t(x)).$$

□

Wegen des Beispiels 2.3 betrachten wir ab jetzt nur solche $\mathcal{R}$-Graphmaschinen, die ihre Ausgabe vollständig selbst schreiben. Da die Graphmaschinen nicht über getrennte Ein-/Ausgabemöglichkeiten verfügen, nehmen wir weiterhin an, daß der Speicherbedarf einer $\mathcal{R}$-Maschine die Länge der Ein- bzw. Ausgabe nicht unterschreitet.

Satz 2.5. *Sei $\mathcal{R}$ ein Ring. Dann gibt es zu jeder $\mathcal{R}$-Graphmaschine eine äquivalente $\mathcal{R}$-Maschine, deren Zeit- und Platzkosten polynomiell verknüpft sind (man beachte obige Annahmen), und umgekehrt.*

Beweis.
- Sei $\mathcal{M} := \mathcal{M}_G^{\mathcal{R}}$ eine beliebige $\mathcal{R}$-Graphmaschine zum Graphen G. Wir werden ein Programm π konstruieren, so daß $\mathcal{M}' := \mathcal{M}_\pi^{\mathcal{R}}$ eine zu $\mathcal{M}$ äquivalente $\mathcal{R}$-(Register-)Maschine mit drei Indexregistern ist, die $\mathcal{M}$ effizient simuliert. Dazu speichern wir die Zustandskomponenten s_i in den Registern z_{c+i}, wobei c eine noch unbestimmte Konstante ist, und ersetzen die Knoten des Graphen wie folgt durch Programmstücke. Wir verwenden vorerst nur zwei Indexregister, nämlich während der Ein- und Ausgabephase als Laufvariablen und dazwischen zur Simulation von s_1 und s_2.

1. Der Eingabeknoten wird durch eine Befehlsfolge ersetzt, die die Eingabe in der für Graphmaschinen angegebenen Weise im Speicher ablegt. Ferner wird ein Zähler Z auf Null gesetzt, den alle anderen Programmstücke inkrementieren.
2. Der Ausgabeknoten wird dadurch simuliert, daß das Ausgabewort $(s_3, s_5, \ldots, s_k)$ aufs Ausgabeband kopiert wird; anschließend erfolgt ein `end`. Dabei ist $k \leq \max\{C, Z\}$, wobei C sich aus den konstanten Adressen ergibt, die in Berechnungsknoten verwendet werden.
3. Berechnungsknoten werden durch ein Programmstück nachgebildet, das die polynomielle bzw. rationale Funktion auswertet. Dies kann leicht mit drei Speicherzellen und in konstanter Zeit geschehen, d.h. unabhängig von der Größe der Eingabe. Länge und Laufzeit des Programmstücks sind linear in der Beschreibungskomplexität der Funktion. Veränderungen an s_1 und s_2 werden sowohl in z_{c+1} und z_{c+2} als auch in γ_1 und γ_2 vollzogen, wobei die Indexregister die um c erhöhte Adresse enthalten.
4. Ein Verzweigungsknoten wird realisiert durch $\alpha := z_{c+3}$ und einen bedingten Sprung.
5. Indexknoten erfordern eine indizierte Lade- und eine Speicheroperation mit zwei verschiedenen Indizes in der Form $\alpha := z_{\gamma_1}; z_{\gamma_2} := \alpha$, wobei der Akkumulator gerettet und wiederhergestellt werden muß.

Diese Programmstücke werden nun gemäß den Kanten des Graphen zu maximalen Abschnitten zusammengefaßt, die höchstens am Ende die Realisierung eines Verzweigungsknotens haben. Solche Abschnitte kann man dann, mit Ausnahme des ersten (Eingabeknoten) und des letzten (Ausgabeknoten), beliebig anordnen, wobei nur die Sprungziele angepaßt werden müssen. Damit wird der Kontrollfluß des Graphen vom Programm nachgebildet.

Der Speicherbedarf von $\mathcal{M}'$ ist bis auf die c Hilfsspeicherzellen und die Eingabe gleich dem von $\mathcal{M}$. Berechnungs-, Verzweigungs- und Indexknoten können in konstanter Zeit simuliert werden, Eingabe- und Ausgabeknoten erfordern Zeit $O(\mathrm{time}_{\mathcal{M}})$.

Um nicht Eingabewerte im Speicher zu halten, die nie gelesen werden, simuliert man das oben angegebene Programm so, daß die Befehlsfolge für den Eingabeknoten nur bei Bedarf und stückweise ausgeführt wird. Genauer gesagt merkt man sich zu jeder Speicherzelle, ob auf sie schon einmal zugegriffen wurde, und holt einen Eingabewert dann in den Speicher, wenn er zum ersten Mal verlangt wird. Dies funktioniert, analog zu Lemma 2.4 mittels der Adreßtransformation $i \mapsto 2i$, mit konstantem Zeit- und Platzverlust und einem weiteren Indexregister zum Zugriff aufs Eingabeband. Somit gilt nach Simulation auf einer Maschine mit einem Indexregister

$$\mathrm{space}_{\mathcal{M}'} = O(\mathrm{space}_{\mathcal{M}} + 1), \qquad \mathrm{time}_{\mathcal{M}'} = O((\mathrm{time}_{\mathcal{M}})^2).$$

- Sei nun umgekehrt $\mathcal{M} := \mathcal{M}_\pi^\mathcal{R}$ eine beliebige $\mathcal{R}$-(Register-)Maschine, welche o.B.d.A. keine arithmetischen Operationen mit indiziertem Speicherzugriff ausführt. Wir werden einen Graphen G konstruieren, so daß $\mathcal{M}' := \mathcal{M}_G^\mathcal{R}$ eine zu $\mathcal{M}$ äquivalente $\mathcal{R}$-Graphmaschine ist, die $\mathcal{M}$ effizient simuliert; dazu ersetzen wir die Befehle des Programms wie folgt durch Teilgraphen. Die Speicherbelegung gestaltet sich kompliziert, da drei unendliche Bänder in eins gepackt werden müssen. Das Eingabeband wird gemäß

$$x_i \mapsto \begin{cases} s_4 & \text{falls } i = 0; \\ s_{2i+1} & \text{sonst.} \end{cases}$$

an die vorgesehene Position gelegt und damit von der Funktion $\mathtt{in}$ initialisiert; dies bedingt eine Fallunterscheidung beim indizierten Zugriff hierauf. Die Register α und γ sowie einige Hilfsspeicherstellen belegen $s_6, s_8, \ldots, s_c$, wobei sich die Konstante c aus der Konstruktion ergibt. Ausgabeband und Rechenspeicher teilen sich die restlichen geraden Adressen, d.h. $y_i \mapsto s_{4i+c+2}$ und $z_i \mapsto s_{4i+c+4}$ für $i \in \mathbb{N}$.

1. Fast jede Zuweisung oder arithmetische Operation mit konstanten Adressen wird durch einen einzelnen Berechnungsknoten nachgebildet. Lediglich $\gamma := \gamma \dot{-} 1$ erfordert noch eine Fallunterscheidung.
2. Indizierte Lesezugriffe auf den Speicher erfordern $(s_1, s_2) = (6, a\gamma + b)$, wobei die Konstanten $a, b \in \mathbb{N}$ von der Adreßtransformation herrühren. Durch einen Graphen konstanter Größe, der eine Schleife enthält, kann der gewünschte Zustand mit konstantem Speicherbedarf in Zeit $O(\mathtt{time}_\mathcal{M})$ hergestellt werden[3]. Dann folgt noch ein Indexknoten, der den eigentlichen Zugriff simuliert. Indizierte Schreibzugriffe erfolgen analog.
3. Bedingte Sprünge erfordern es, den Inhalt von s_3 zu retten, den Akkumulator dorthin zu transportieren, die eigentliche Verzweigung vorzunehmen und s_3 wiederherzustellen; also zwei Berechnungs- und einen Verzweigungsknoten.
4. Der Sonderbefehl `print` kann ignoriert werden, `next` δ darf nicht auftreten (ebenso $\alpha := \delta$). Den Befehl `end`, der o.B.d.A. nur einmal auftritt, ersetzen wir durch einen Teilgraphen, der die Ausgabe an ihren Platz bringt und mit einem Ausgabeknoten abschließt. Dazu bedarf es lediglich einer einfachen Schleife mit einem Indexknoten; die Zustandskomponenten s_1 und s_2 zählen monoton aufwärts.

Diese Teilgraphen werden nun gemäß der Abfolge der Befehle im Programm zu G zusammengefaßt; lediglich beim Ersatz von Verzweigungen müssen die Kanten zum wahr- bzw. falsch-Nachfolger entsprechend gesetzt werden. Damit

[3] Man beachte, daß nach k Schritten das Indexregister der $\mathcal{R}$-Maschine höchstens den Wert k hat.

wird der Kontrollfluß des Programms vom Graphen nachgebildet; toten Programmcode sollte man noch eliminieren, damit der Graph zusammenhängend wird.

Der Speicherbedarf von $\mathcal{M}'$ ist bis auf die c Hilfsspeicherzellen sowie die Ein- und Ausgaben gleich dem von $\mathcal{M}$; gemäß unserer einführenden Annahme ist dies $O(\mathtt{space}_{\mathcal{M}} + 1)$. Fast alle einzelnen Befehle können in konstanter Zeit simuliert werden, lediglich die indizierten Speicherzugriffe und der `end`-Befehl erfordern Zeit $O(\mathtt{time}_{\mathcal{M}})$. Somit gilt

$$\mathtt{space}_{\mathcal{M}'} = O(\mathtt{space}_{\mathcal{M}} + 1), \qquad \mathtt{time}_{\mathcal{M}'} = O((\mathtt{time}_{\mathcal{M}})^2).$$

□

Fazit

Wir haben die Kosten einer Berechnung und die Komplexität einer Maschine für die beiden Aspekte Zeit und Platz hergeleitet, und zwar sowohl für $\mathbb{R}$- als auch für δ-$\mathbb{Q}$-Maschinen. Es ergibt sich die Verallgemeinerung auf analytische Rechnungen des bekannten Resultats, daß jede berechenbare Funktion mit einer konstanten Anzahl von Registern ausgewertet werden kann. $\mathcal{R}$-Maschinen mit unterschiedlich vielen Indexregistern sind polynomiell verknüpft, woraus leicht folgt, daß der Simulationssatz effizient ist. Ebenso ist der Ressourcenverbrauch unserer Registermaschinen und der Graphmaschinen von Blum, Shub, Smale im wesentlichen gleich.

Damit sind einerseits die grundlegenden Begriffe der Komplexitätstheorie auf unser Maschinenmodell übertragen, worauf aufbauend man z.B. Fragen der NP-Vollständigkeit studieren könnte. Andererseits rechtfertigt dies voll und ganz unsere etwas andere, aber äquivalente Beschreibung reeller Berechenbarkeit und der zugrundeliegenden Maschinen.

3. Differentialgleichungen

Differentialgleichungen bilden die Grundlage des naturwissenschaftlich-mathematischen Weltbildes. WLADIMIR IGOREWITSCH ARNOLD

Sehr viele Prozesse in Natur und Technik lassen sich durch Differentialgleichungen (DGl) beschreiben, wobei zwischen *gewöhnlichen* und *partiellen* DGl unterschieden wird. Diese beschreiben eine Funktion von einer (zeitlichen) bzw. mehreren (räumlichen) Variablen durch ihre gewöhnlichen bzw. partiellen Ableitungen erster oder höherer Ordnung nach diesen Variablen. Nach [22] läßt sich diese Unterscheidung auch so auffassen, daß Systeme mit *endlich vielen Freiheitsgraden* (z.B. endlich viele Punktmassen in der Newtonschen Mechanik, etwa Planetensysteme) gewöhnlichen DGl entsprechen, solche mit *unendlich vielen Freiheitsgraden* (z.B. Flüssigkeiten, Gase oder elektromagnetische Felder) dagegen partiellen DGl.

Wir wollen nun unter anderem zeigen, daß die analytisch berechenbaren Funktionen eine sehr mächtige Klasse darstellen, die insbesondere die Lösung von DGl umfaßt. Die allgemeine Idee dabei ist, daß eindeutige Lösungen berechenbar sind, und das wird unter gewissen hinreichenden Voraussetzungen bewiesen. Ferner betrachten wir Stabilitätsprobleme für die Lösungen von DGl und zeigen sehr allgemeine Resultate über deren Unentscheidbarkeit.

3.1. Lösung von Anfangswertproblemen

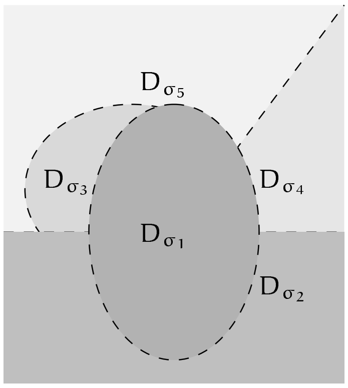
In diesem Abschnitt werden wir zeigen, wie Lösungen von DGl als analytisch berechenbare Funktionen dargestellt werden können. Zu diesem Zweck beschränken wir uns zunächst auf *Anfangswertprobleme* für *Systeme expliziter, gewöhnlicher DGl erster Ordnung* mit einer rechten Seite, die von einer $\mathbb{R}$-Maschine *ohne Division*[1] *berechnet* werden kann; später erlauben wir auch stark δ-$\mathbb{Q}$-analytische rechte Seiten. Sei also $N \in \mathbb{N}$ die Dimension des Systems, $\mathbf{f} : \mathbb{R} \times \mathbb{R}^N \to \mathbb{R}^N$, $(t, \boldsymbol{\xi}) \mapsto \mathbf{f}(t, \boldsymbol{\xi})$, eine über dem Ring $\mathbb{R}$ berechenbare Funktion und $(t_0, \mathbf{x}_0) \in \mathbb{R} \times \mathbb{R}^N$. Dann betrachten wir Anfangswertprobleme der Form

$$\mathbf{x}'(t) = \mathbf{f}(t, \mathbf{x}(t)),\ \mathbf{x}(t_0) = \mathbf{x}_0. \tag{3.1}$$

Für diese sucht man klassischerweise eine differenzierbare Lösungsfunktion $\mathbf{x} : \mathbb{R} \supset I \to \mathbb{R}^N$, $t \mapsto \mathbf{x}(t)$, die auf einem Intervall $I \ni t_0$ den komponentenweisen Gleichungen $x_i'(t) = f_i(t, \mathbf{x}(t))$ mit $i = 1, \ldots, N$ sowie der Anfangsbedingung genügt.

[1] Ohne den Befehl $\alpha := \alpha^{-1}$.

Wir werden nun in mehreren Schritten die Definition, Existenz, Eindeutigkeit und Berechenbarkeit einer Lösungsfunktion klären. Da die rechte Seite der DGl als $\mathbb{R}$-berechenbare Funktion mittels Fallunterscheidungen definiert ist, müssen wir zunächst präzisieren, was genau wir unter einer Lösung eines solchen abschnittsweise definierten Anfangswertproblems verstehen wollen. Wichtig dafür ist ein Verständnis der Struktur der Abbildung **f**, wie sie der Darstellungssatz beschreibt.



Durch die Verzweigungsbedingungen entlang eines Programmpfades σ wird der Definitionsbereich von $\mathbf{f}$, $\mathbb{R}^{1+N}$, disjunkt in semi-algebraische Mengen[2] D_σ zerlegt, die den Einzugsbereich des Pfades darstellen; diese wollen wir fortan *Regionen* nennen. Ein solche Region D_σ ist also die Lösungsmenge eines Systems polynomieller Ungleichungen mit $>$ und $\geq$ in $1+N$ Variablen. Formuliert man dasselbe System ausschließlich mit strengen Ungleichungen ($>$), so definiert es das offene Innere $\mathtt{int}(D_\sigma)$. Verwendet man nur $\geq$, so erhält man den Abschluß $\overline{D_\sigma}$, dessen Rand ∂D_σ aus den Punkten besteht, welche mindestens eine der Bedingungen mit Gleichheit erfüllen; ob ein solcher Punkt zu D_σ gehört, hängt von der ursprünglichen Ungleichung ab. Der Rand ist damit im allgemeinen zusammengesetzt aus N-dimensionalen Mannigfaltigkeiten, die jeweils genau eine polynomielle Gleichung erfüllen und die wir *Facetten* nennen, und deren Nahtstellen kleinerer Dimension. Aus Gründen der Darstellbarkeit beschränkt sich die nebenstehende Abbildung auf ein zweidimensionales Beispiel.

Die i. Komponente der entlang des Pfades σ berechneten Funktion kann als polynomielle (oder – im allgemeinen Fall des Körpers $\mathbb{R}$ – rationale) Funktion $f_{\sigma,i}$ der Variablen[3] mit rationalen Koeffizienten beschrieben werden; dies liefert die Darstellung

$$\mathbf{f} = \sum_\sigma \chi_{D_\sigma}(f_{\sigma,1} \times \cdots \times f_{\sigma,N}).$$

Definition, Existenz und Eindeutigkeit Wegen der Beschränkung auf den *Ring* $\mathbb{R}$ ist $f_{\sigma,i} : \mathbb{R}^{1+N+K} \to \mathbb{R}$ ein Polynom und damit eine C^∞-Funktion, d.h. beliebig oft stetig differenzierbar. Daher existiert für das (lokale) Problem $\mathbf{x}'(t) = \mathbf{f}_\sigma(t, x(t))$ mit $\mathbf{f}_\sigma(\tau, \boldsymbol{\xi}) := f_{\sigma,1}(\tau, \boldsymbol{\xi}, \boldsymbol{\kappa}) \times \cdots \times f_{\sigma,N}(\tau, \boldsymbol{\xi}, \boldsymbol{\kappa})$ und beliebiger Anfangsbedingung $(\tau, \boldsymbol{\xi}) \in \mathbb{R}^{1+N}$ gemäß dem *globalen Existenz- und Eindeutigkeitssatz* (vgl. z.B. [47, 1.12.9.1.]) eine eindeutige Lösung auf einem maximalen, offenen Intervall $I_\sigma \ni \tau$.

Man stellt sich nun vor, daß die globale Lösung des ursprünglichen Problems analog zur Darstellung von **f** stückweise zusammengesetzt ist aus Lösungen der lokalen Probleme für die Pfade σ. Innerhalb der Regionen D_σ soll die Lösung differenzierbar

[2] Sogenannte *basic semi-algebraic sets*, da nur der Durchschnitt über die Lösungsmengen polynomieller Ungleichungen gebildet wird, nicht auch die Vereinigung.

[3] Diese umfassen sowohl die $1 + N$ Eingabevariablen $(\tau, \boldsymbol{\xi})$ als auch die K irrationalen Programmkonstanten $\boldsymbol{\kappa}$.

sein und der DGl genügen, die Übergänge dazwischen sollen lediglich stetig erfolgen. Damit die Lösung global stetig differenzierbar ist, muß die rechte Seite $\mathbf{f}$ stetig sein; diese recht starke Einschränkung wollen wir jedoch nicht voraussetzen. Damit ergibt sich folgende

Definition 3.1 (Lösung). Gegeben sei ein Anfangswertproblem gemäß Gleichung 3.1. Wir nennen eine auf einem Intervall $I \subset \mathbb{R}$ stetige Funktion $\mathbf{x} : I \to \mathbb{R}^N$ mit $\mathbf{x}(t_0) = \mathbf{x}_0$ *(globale) Lösung* davon, falls $\mathbf{x}$ für alle Pfade σ und alle Zeitpunkte $t \in \mathtt{int}(T_\sigma)$ differenzierbar ist mit Ableitung $\mathbf{x}'(t) = \mathbf{f}_\sigma(t, \mathbf{x}(t))$. Dabei sei $T_\sigma := \{t \mid (t, \mathbf{x}(t)) \in D_\sigma\}$. Die Lösung heißt *maximal*, falls es keine Fortsetzung auf ein umfassenderes Intervall gibt. □

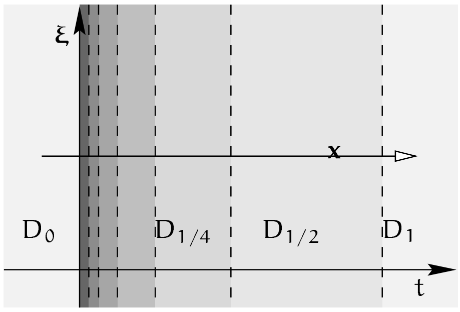
Anmerkung 3.1. Wegen der Stetigkeit der $\mathbf{f}_\sigma$ erfüllen die einseitigen Ableitungen einer Lösung $\mathbf{x}$ zu den Zeitpunkten $t \in \partial T_\sigma$, also beim Wechsel einer Region, die jeweiligen lokalen DGl.

Zunächst stellt sich die Frage nach der *Existenz* solcher maximaler globaler Lösungen.

Lemma 3.1 (Existenz). *Für jedes Anfangswertproblem gemäß Gleichung 3.1 existiert eine maximale globale Lösung gemäß Definition 3.1.*

Beweis. Sei σ_0 der Programmpfad bei der Berechnung von $\mathbf{f}$ mit $(t_0, \mathbf{x}_0) \in D_{\sigma_0}$. Dann ist die lokale Lösung des Problems $\mathbf{x}' = \mathbf{f}_{\sigma_0}(t, \mathbf{x})$ für ein geeignet gewähltes Intervall $t_0 \in I \subset T_{\sigma_0}$ bereits eine globale Lösung im Sinne der Definition; man beachte, daß T_{σ_0} nicht zusammenhängend sein muß. Demnach existiert auch sicherlich eine (nicht notwendig eindeutige) maximale Fortsetzung auf ein (nicht notwendig abgeschlossenes) Intervall I_{MAX}. □

Interessanter sind nun die Bedingungen für die *Eindeutigkeit* einer maximalen globalen Lösung. Aus Symmetriegründen betrachten wir im folgenden nur das Verhalten der Lösung *nach* dem Anfangszeitpunkt t_0 in Richtung der positiven Zeitachse. Unter den folgenden beiden Voraussetzungen an die rechte Seite bzw. den Startpunkt werden wir beweisen, daß Verzweigungen nicht auftreten.

1. 

Die nebenstehende Abbildung illustriert einen entarteten Fall, den wir unbedingt vermeiden möchten. Das Programm zur Berechnung von $\mathbf{f}(t, \boldsymbol{\xi})$ fragt zuerst $t \leq 0$ ab und dann im Falle des jeweiligen Scheiterns weiter $t \geq 1/k$ für $k = 1, 2, \ldots$. Dadurch häufen sich die Ränder der Regionen im Bereich der $\boldsymbol{\xi}$-Achse derart, daß eine Lösungskurve in endlicher Zeit unendliche viele Regionen durchläuft[4]. Dies führt unter anderem

[4]Man fühlt sich hierbei an den Wettlauf von Achilles und der Schildkröte erinnert – das Paradox des ZENO.

dazu, daß es keine Region gibt, in welche die Lösung von D_0 aus überwechselt. Wir wollen solche degenerierten Fälle im folgenden ausschließen und verlangen daher von einer *zulässigen Funktion*, daß jede kompakte Teilmenge des Definitionsbereichs nur endlich viele Regionen schneidet.

Von einer zulässigen Funktion können wir ohne weitere Einschränkung annehmen, daß jede Region im Abschluß ihres offenen Inneren enthalten ist: $D_\sigma \subset \overline{\mathtt{int}(D_\sigma)}$. Randpunkte von D_σ, die nicht zum Inneren benachbart sind, werden von einer Lösungsfunktion durchlaufen, ohne deren Richtung zu beeinflussen; daher können sie auch einer anderen (benachbarten) Region zugeschlagen werden.

Vermutung 2. Für eine Region gilt $D_\sigma \subset \overline{\mathtt{int}(D_\sigma)} \iff D_\sigma \neq \partial D_\sigma$.

2. Wählt man den Startpunkt $(t_0, \mathbf{x}_0)$ so, daß die lokal eindeutige Lösung die Region in einem mehrfachen Randpunkt[5] verläßt, so ist die Fortsetzung nicht notwendig eindeutig – siehe Abbildung 3.1. Dies gilt entsprechend auch für zusammengesetzte Lösungen.

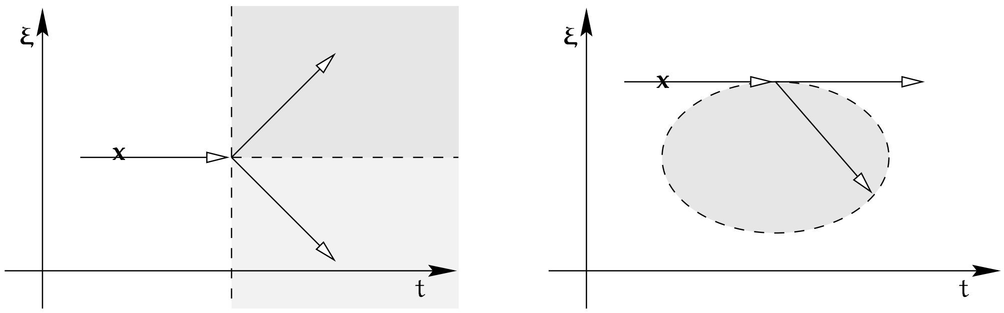


Abbildung 3.1.: Lösungsverzweigung

Ebenso sollte keine Lösung den Rand einer Region berühren, ohne ihn zu schneiden; wenn der Graph einer globalen Lösungsfunktion einen Rand berührt oder gar ein Stück weit darin verläuft, so kann die Lösung sich verzweigen – siehe Abbildung 3.1. Wir nennen daher nur solche Anfangswerte *geeignet*, für die keine (maximale) globale Lösung einen mehrfachen Randpunkt trifft oder den Rand einer Region lediglich tangiert. Aus der Eindeutigkeit der Lösung wird folgen, daß abhängig vom Anfangswert entweder jede oder keine maximale globale Lösung auf einen solchen Punkt stößt.

Satz 3.2 (Eindeutigkeit). *Für jedes Anfangswertproblem gemäß Gleichung 3.1 mit zulässiger rechter Seite* $\mathbf{f}$ *und geeignetem Anfangswert* $(t_0, \mathbf{x}_0)$ *existiert eine eindeutige maximale globale Lösung gemäß Definition 3.1.*

Beweis. Wir nehmen an, daß zwei Lösungen $\mathbf{x}_1 : [t_0, t_1[\to \mathbb{R}^N$ und $\mathbf{x}_2 : [t_0, t_2[\to \mathbb{R}^N$ gegeben sind, die sich nach einem frühsten Zeitpunkt

$$t^* := \inf\{t \geq t_0 \mid \mathbf{x}_1(t) \neq \mathbf{x}_2(t)\} < \min\{t_1, t_2\}$$

[5] In diesem Zusammenhang steht *mehrfach* für *mehr als zweifach*.

unterscheiden; diese Verzweigungsstelle nennen wir $P := (t^*, \mathbf{x}_1(t^*)) = (t^*, \mathbf{x}_2(t^*))$, ihre eindeutige Region sei D_σ. Da nach dem bereits Gesagten die lokalen Lösungen eindeutig sind, kann P nicht im Innern der Region liegen, sondern nur auf deren Rand. Andererseits ist P auch kein mehrfacher Randpunkt, und so gibt es genau eine weitere Region D_τ mit $P \in \partial D_\tau$; man beachte die obige Diskussion um die verbotene Häufung von Regionen im Endlichen.

Man kann nun für jede Kurve $\mathbf{x}_i$ angeben, durch welche Region $D_{\pi_i^-}$ bzw. $D_{\pi_i^+}$ sie unmittelbar vor bzw. nach dem Zeitpunkt t^* verläuft, dabei ist $\pi_i^\pm \in \{\sigma, \tau\}$ eindeutig mit:

$$\exists\, \varepsilon > 0\, \forall\, t^* - \varepsilon < t < t^* : \ (t, \mathbf{x}_i(t)) \in D_{\pi_i^-}$$

und analog für π_i^+ und Zeitpunkte $t^* < t < t^* + \varepsilon$.

Da lokale Lösungen eindeutig durch den Anfangswert bestimmt sind, auch wenn dieser ein nicht zur Region gehörender Randpunkt ist, gilt $\pi_1^+ \neq \pi_2^+$, sonst wäre eine Verzweigung ausgeschlossen. Andererseits ist $\pi_1^- = \pi_2^-$, da die beiden Kurven vor dem Zeitpunkt t^* übereinstimmen; demnach verbleibt o.B.d.A $\mathbf{x}_1$ innerhalb der Region $D_{\pi_1^-}$ und $\mathbf{x}_2$ wechselt von dort nach $D_{\pi_2^+}$. Dieses Verhalten von $\mathbf{x}_1$ steht allerdings im Widerspruch zu der Anforderung an geeignete Startwerte! Damit ist die ursprüngliche Annahme als falsch entlarvt. □

Schränkt man eine maximale globale Lösung $\mathbf{x}$ gerade soweit ein, daß sie keinen mehrfachen Randpunkt durchläuft und nirgendwo einen Rand lediglich tangiert, so nennen wir die entstandene Lösung den *Kern* von $\mathbf{x}$. Dieser genügt der Definition einer globalen Lösung, aber ist im allgemeinen natürlich nicht mehr maximal.

Korollar 3.3. Für jedes Anfangswertproblem gemäß Gleichung 3.1 mit zulässiger rechter Seite $\mathbf{f}$ gibt es eine eindeutige globale Lösung gemäß Definition 3.1, die Kern aller maximalen globalen Lösungen ist.

Definition 3.2. Sei $\mathbf{f}$ eine zulässige Funktion und $(t_0, \mathbf{x}_0)$ ein geeigneter Anfangswert im obigen Sinne. Dann bezeichne $\mathbf{x}_{\mathbf{f},t_0,\mathbf{x}_0} : I_{\mathbf{f},t_0,\mathbf{x}_0} \to \mathbb{R}^{1+N}$ die gemäß Satz 3.2 eindeutig bestimmte maximale globale Lösung des Anfangswertproblems nach Gleichung 3.1. Dabei ist $I_{\mathbf{f},t_0,\mathbf{x}_0} \subset \mathbb{R}$ ein Intervall. □

Berechenbarkeit Nachdem nun einiges über die Struktur von Lösungen einer DGl in unserem Sinne bekannt ist, werden wir sehen, daß die unter den obigen Voraussetzungen eindeutigen Lösungen auch bereits $\mathbb{R}$-analytisch berechenbar sind.

Satz 3.4 (Berechenbarkeit). *Unter obigen Voraussetzungen und Bezeichnungen ist* $\mathbf{x}_{\mathbf{f},t_0,\mathbf{x}_0}$ *$\mathbb{R}$-analytisch und kann ohne Division berechnet werden.*

Beweis. Sei $\mathcal{M}_\mathbf{f}$ eine $\mathbb{R}$-Maschine zur Berechnung von $\mathbf{f}$, $\mathbf{x} := \mathbf{x}_{\mathbf{f},t_0,\mathbf{x}_0}$ und $I := I_{\mathbf{f},t_0,\mathbf{x}_0}$. Wir konstruieren eine $\mathbb{R}$-Maschine $\mathcal{M}$, die zu einem gegebenen Zeitpunkt $t \in I$ den Funktionswert der Lösung $\mathbf{x}(t)$ ermittelt.

Dazu benutzen wir das explizite Euler-Verfahren zur Schrittweite $h := (t - t_0)(\frac{1}{2})^n$. Dieses approximiert die Lösung zu den Gitterzeitpunkten $\tau_k := t_0 + kh$, $0 \leq k \leq$

2^n, durch $\mathbf{y}(\tau_k, h) \approx \mathbf{x}(\tau_k)$, welches leicht ohne Division berechnet werden kann: $\mathbf{y}(\tau_0, h) := \mathbf{x}_0$ und

$$\mathbf{y}(\tau_{k+1}, h) := \mathbf{y}(\tau_k, h) + h\,\mathbf{f}(\tau_k, \mathbf{y}(\tau_k, h)) \tag{3.2}$$

Die Maschine $\mathcal{M}$ gibt dann $\mathbf{y}(t, h)$ aus für $n \to \infty$ und damit auch $h \to 0$.

Wir beschränken uns aus Symmetriegründen wiederum auf den Fall $t > t_0$ und zeigen nun, daß $\lim_{h\to 0} \mathbf{y}(t, h) = \mathbf{x}(t)$. Sei σ_0 der Pfad mit $(t_0, \mathbf{x}_0) \in D_{\sigma_0}$, und wir nehmen zunächst an, daß sich alles innerhalb dieser Region D_{σ_0} abspielt. Ferner sei $B \subset \mathbb{R}^{1+N}$ eine Kugel um $(t_0, \mathbf{x}_0)$, in der alle $\mathbf{y}(\tau_k, h)$ liegen, und $L > 0$ eine Lipschitz-Konstante von $\mathbf{f}_{\sigma_0}$ auf B bezüglich $\boldsymbol{\xi}$, d.h.

$$\forall\,(\tau, \boldsymbol{\xi}_1), (\tau, \boldsymbol{\xi}_2) \in B: \ |\mathbf{f}_{\sigma_0}(\tau, \boldsymbol{\xi}_1) - \mathbf{f}_{\sigma_0}(\tau, \boldsymbol{\xi}_2)| \le L|\boldsymbol{\xi}_1 - \boldsymbol{\xi}_2|.$$

Dann ist der globale Fehler des Euler-Verfahrens $|\mathbf{y}(t, h) - \mathbf{x}(t)| = O(\frac{h}{L} e^{(t-t_0)L})$ und die Näherung konvergiert für $h \to 0$ gegen den korrekten Wert.

Nun betrachten wir den allgemeinen Fall, daß die Lösung die Grenze einer Region überschreitet. Die lokale Lösung schneide in $(t_1, \mathbf{x}_1)$ den Rand von D_{σ_0} beim Übergang nach D_{σ_1}; nach Voraussetzung ist dies kein mehrfacher Randpunkt und die Lösung verläuft nicht tangential zum Rand. Die Approximation dieses Punktes sei $(t_{1,h}, \mathbf{x}_{1,h})$, der erste vom Euler-Verfahren generierte Punkt außerhalb D_{σ_0}. Nach dem oben gezeigten gilt $\lim_{h\to 0}(t_{1,h}, \mathbf{x}_{1,h}) = (t_1, \mathbf{x}_1)$.

Für eine hinreichend kleine Schrittweite h gilt schließlich $(t_{1,h}, \mathbf{x}_{1,h}) \in D_{\sigma_1}$. Wegen der stetigen Abhängigkeit der Lösung von den Anfangsbedingungen konvergiert die von $(t_{1,h}, \mathbf{x}_{1,h})$ ausgehende Näherungslösung ebenfalls gegen $\mathbf{x}$. Damit läßt sich das Verfahren also auch über die Grenzen der Regionen hinaus korrekt fortsetzen. □

Hauptsatz 1. *Sei* $\mathbf{f}$ *eine zulässige Funktion und* $(t_0, \mathbf{x}_0)$ *ein geeigneter Anfangswert im obigen Sinne. Dann ist die eindeutig bestimmte maximale globale Lösung* $\mathbf{x}_{\mathbf{f},t_0,\mathbf{x}_0}$ *des Anfangswertproblems nach Gleichung 3.1 robust* δ-$\mathbb{Q}$*-analytisch und kann ohne Division berechnet werden.*

Beweis. Wir verfahren analog zum Beweis von Satz 3.4 und simulieren zusätzlich die $\mathbb{R}$-Maschine $\mathcal{M}_\mathbf{f}$ wie im Simulationssatz (1.19). Das Hauptproblem besteht darin, $\mathbf{f}$ mit hinreichender Genauigkeit und in endlicher Zeit durch eine δ-$\mathbb{Q}$-Maschine auszuwerten. Zunächst führen wir den Beweis wieder für die Startregion $D_{\sigma_0} \ni (t_0, \mathbf{x}_0)$, danach schließt man wie zuvor auf den allgemeinen Fall.

Aus Gleichung 3.2 folgt, daß $\mathbf{y}(t, h) = p_n(t_0, \mathbf{x}_0, t)$, wobei p_n ein Polynom mit rationalen Koeffizienten ist, weil auch $\mathbf{f}_{\sigma_0}$ ein solches ist. Die Auswertung von $p_n(t_0, \mathbf{x}_0, t)$ mit Präzision δ ergibt eine Approximation $p_n^{(\delta)}(t_0, \mathbf{x}_0, t)$ von $\mathbf{x}(t)$ unbekannter Güte zusammen mit einer Fehlerschranke $\varepsilon_n^{(\delta)}(t_0, \mathbf{x}_0, t)$ *für die Auswertung selbst*, d.h.

$$|p_n^{(\delta)}(t_0, \mathbf{x}_0, t) - p_n(t_0, \mathbf{x}_0, t)| < \varepsilon_n^{(\delta)}(t_0, \mathbf{x}_0, t).$$

Es ist klar, daß $\lim_{\delta\to\infty} \varepsilon_n^{(\delta)}(t_0, \mathbf{x}_0, t) = 0$, und wir würden gerne

$$\lim_{n\to\infty} \lim_{\delta\to\infty} p_n^{(\delta)}(t_0, \mathbf{x}_0, t) = \lim_{n\to\infty} p_n(t_0, \mathbf{x}_0, t) = \lim_{h\to 0} \mathbf{y}(t, h) = \mathbf{x}(t)$$

berechnen. Leider kann eine δ-$\mathbb{Q}$-Maschine nur einen einzelnen Grenzwert für $\delta \to \infty$ berechnen, und daher brauchen wir einen kleinen Trick. Wir betrachten die Ungleichung $\varepsilon_n^{(\delta)}(t_0, \mathbf{x}_0, t) < 2^{-n}$, die bei jedem n für $\delta \geq \delta_n$ gilt. Umgekehrt suchen wir zu gegebenem δ das maximale $n \leq \delta$, so daß die Ungleichung noch erfüllt ist, und nennen diesen Wert n_δ. Man beachte, daß die Folge dieser n_δ mit δ monoton und über alle Grenzen wächst. Daher konstruieren wir die δ-$\mathbb{Q}$-Maschine $\mathcal{M}$ so, daß sie in jeder Phase $p_{n_\delta}^{(\delta)}(t_0, \mathbf{x}_0, t)$ ausgibt; gemäß obigen Ausführungen sollte klar sein, daß diese Ausgabe wie gewünscht konvergiert.

Nun betrachten wir den allgemeinen Fall, daß die Lösung in $(t_1, \mathbf{x}_1)$ die Grenze zwischen D_{σ_0} und D_{σ_1} überschreitet. Die Approximation dieses Punktes sei $(t_{1,h,\delta}, \mathbf{x}_{1,h,\delta})$, der erste vom Euler-Verfahren generierte Punkt, welcher bei einer Auswertung von $\mathbf{f}$ mit Präzision δ außerhalb von D_{σ_0} liegt. Bei hinreichend kleiner Schrittweite h und großer Präzision δ gilt $(t_{1,h,\delta}, \mathbf{x}_{1,h,\delta}) \in D_{\sigma_1}$. Wie zuvor wird dieser Punkt beschrieben durch ein Polynom in den Eingaben t_0, $\mathbf{x}_0$ und t; wieder wählen wir n so, daß dieses mit Genauigkeit 2^{-n} ausgewertet werden kann. Wegen der stetigen Abhängigkeit der Lösung von den Anfangswerten konvergiert die Näherung, die in $(t_{1,h,\delta}, \mathbf{x}_{1,h,\delta})$ startet, ebenfalls korrekt für $\delta \to \infty$, wodurch $n \to \infty$ und $h \to 0$ impliziert werden. Damit läßt sich das Verfahren also auch über die Grenzen der Regionen hinaus korrekt fortsetzen. ☐

Korollar 3.5. Sei $\mathbf{f}$ wie zuvor; dann ist auch die Abbildung $\mathbb{R} \times \mathbb{R}^N \times \mathbb{R} \rightsquigarrow \mathbb{R}^N$ mit $(t_0, \mathbf{x}_0, t) \mapsto \mathbf{x}_{\mathbf{f},t_0,\mathbf{x}_0}(t)$ robust δ-$\mathbb{Q}$-analytisch. Diese ist zumindest für alle geeigneten Startwerte $(t_0, \mathbf{x}_0)$ und Zeitpunkte $t \in I_{\mathbf{f},t_0,\mathbf{x}_0}$ definiert.

Beweis. Die Startwerte können der Maschine aus dem Beweis von Satz 1 leicht als zusätzliche Eingaben übergeben werden. ☐

δ-$\mathbb{Q}$-analytische rechte Seiten In praktischen Anwendungen ist man oft an rechten Seiten interessiert, die z.B. die Exponentialfunktion beinhalten. Derartige Funktionen sind zwar nicht $\mathbb{R}$-berechenbar, wohl aber δ-$\mathbb{Q}$-analytisch. Wir untersuchen daher hinreichende Voraussetzungen, unter denen man die Lösung einer DGl mit δ-$\mathbb{Q}$-analytischer rechter Seite mit Hilfe des Euler-Verfahrens berechnen kann. Die Hauptschwierigkeit besteht darin, daß man in jedem einzelnen Schritt dieses Verfahrens die rechte Seite *in endlicher Zeit* auswerten – und daher die analytische Rechnung vorzeitig abbrechen –, den so entstandenen Approximationsfehler aber kontrollieren muß.

Aus der klassischen Numerik [33] ist bekannt, daß das explizite Euler-Verfahren *konsistent* mit Ordnung eins ist, d.h. der *lokale Diskretisierungsfehler* ist $O(h)$. Dies allein genügt jedoch nicht, um die *Konvergenz* zu gewährleisten, d.h. den *globalen Diskretisierungsfehler* zu kontrollieren. Hierzu wird im allgemeinen gefordert, daß die rechte Seite $\mathbf{f}$ Lipschitz-stetig bzgl. des zweiten Arguments sein soll. Wir werden nun zeigen, daß unter dieser klassischen Voraussetzung an eine stark δ-$\mathbb{Q}$-analytische rechte Seite die Lösung der DGl robust δ-$\mathbb{Q}$-analytisch ist; dazu steigen wir etwas tiefer als zuvor in den Formalismus der Numerik ein. Zuvor beweisen wir jedoch ein technisches Lemma, das sich später als nützlich erweist.

Lemma 3.6. *Genügen die Zahlen $\gamma_0, \dots, \gamma_k \in \mathbb{C}$ einer Abschätzung der Form*

$$|\gamma_{i+1}| \le (1+\delta)|\gamma_i| + B \quad \textit{für} \quad \delta > 0, B \ge 0, 0 \le i < k,$$

dann gilt

$$|\gamma_k| \le e^{k\delta}|\gamma_0| + \frac{e^{k\delta}-1}{\delta}B.$$

Beweis. $|\gamma_k| \le (1+\delta)^k|\gamma_0| + B\sum_{i=0}^{k-1}(1+\delta)^i = (1+\delta)^k|\gamma_0| + B((1+\delta)^k - 1)/\delta$ und $0 < 1+\delta \le e^\delta$. □

Gegeben ist also ein Anfangswertproblem $\mathbf{x}'(t) = \mathbf{f}(t, \mathbf{x}(t))$, $\mathbf{x}(t_0) = \mathbf{x}_0$, mit $\mathbf{f} : \mathbb{R} \times \mathbb{R}^N \to \mathbb{R}^N$. Die Lösung dieser DGl zum Zeitpunkt t wird approximiert durch ein Einzelschrittverfahren (ESV) zur Schrittweite $h := (t - t_0)(\frac{1}{2})^n$, das die Lösung auf den Gitterpunkten $t_k = t_0 + kh$, $0 \le k \le 2^n$, durch $\boldsymbol{\xi}(t_k, h) \approx \mathbf{x}(t_k)$ annähert:

$$\boldsymbol{\xi}(t_0, h) := \mathbf{x}_0, \quad \boldsymbol{\xi}(t_{k+1}, h) := \boldsymbol{\xi}(t_k, h) + h\Phi_{\mathbf{f}}(t_k, \boldsymbol{\xi}(t_k, h), h).$$

In unserem Fall ist die *Verfahrensfunktion* $\Phi_{\mathbf{f}}(\tau, \boldsymbol{\xi}, h) = \mathbf{f}(\tau, \boldsymbol{\xi}) + \varepsilon_{\mathbf{f}}(\tau, \boldsymbol{\xi}, h)$, wobei $\varepsilon_{\mathbf{f}}$ den Fehler bei der abgebrochenen Auswertung der rechten Seite beschreibt.

Wir zeigen nun, daß die Bedingung $\forall \tau, \boldsymbol{\xi}, h : \|\varepsilon_{\mathbf{f}}(\tau, \boldsymbol{\xi}, h)\| = O(h)$ ausreichend für die Konvergenz ist; dabei steht $\|.\|$ für eine der äquivalenten Normen auf dem $\mathbb{R}^N$. Diese Bedingung kann leicht erfüllt werden, falls $\mathbf{f}$ stark δ-$\mathbb{Q}$-analytisch ist, da man dann den Approximationsfehler kennt und die Auswertung erst abbricht, wenn er klein genug ist. Zunächst betrachten wir den *lokalen Diskretisierungsfehler* an einer Stelle t_k, wobei $\boldsymbol{\xi}(t_k, h) = \mathbf{x}(t_k)$ angenommen wird:

$$\begin{aligned}
h\lambda(t_k, h) &:= \|\mathbf{x}(t_{k+1}) - \boldsymbol{\xi}(t_{k+1}, h)\| \\
&= \|\mathbf{x}(t_{k+1}) - \boldsymbol{\xi}(t_k, h) - h\Phi_{\mathbf{f}}(t_k, \mathbf{x}(t_k), h)\| \\
&\le \|\underbrace{\mathbf{x}(t_{k+1}) - \mathbf{x}(t_k)}_{h\mathbf{x}'(t_k)+O(h^2)} - h\underbrace{\mathbf{f}(t_k, \mathbf{x}(t_k))}_{\mathbf{x}'(t_k)}\| + h\underbrace{\|\varepsilon_{\mathbf{f}}(t_k, \mathbf{x}(t_k), h)\|}_{O(h)} \\
\Rightarrow \quad \lambda(t_k, h) &= O(h)
\end{aligned}$$

Obige Abschätzung verwendet den Satz von TAYLOR, der zunächst einmal nicht für vektorwertige Funktionen gilt, aber leicht komponentenweise angewendet werden kann und dann das gewünschte leistet. Damit ist gezeigt, daß trotz des Auswertungsfehlers $\varepsilon_{\mathbf{f}}$ das ESV weiterhin konsistent mit Ordnung eins ist. Nun betrachten wir den *globalen Diskretisierungsfehler*, wobei nur $\boldsymbol{\xi}(t_0, h) = \mathbf{x}_0$ angenommen wird:

$$\begin{aligned}
\gamma(t_{k+1}, h) &:= \|\mathbf{x}(t_{k+1}) - \boldsymbol{\xi}(t_{k+1}, h)\| \\
&= \|\mathbf{x}(t_{k+1}) - \boldsymbol{\xi}(t_k, h) - h\Phi_{\mathbf{f}}(t_k, \boldsymbol{\xi}(t_k, h), h)\| \\
&\le \overbrace{\|\mathbf{x}(t_{k+1}) - \mathbf{x}(t_k) - h\Phi_{\mathbf{f}}(t_k, \mathbf{x}(t_k), h)\|}^{h\lambda(t_k,h)} + \overbrace{\|\mathbf{x}(t_k) - \boldsymbol{\xi}(t_k, h)\|}^{\gamma(t_k,h)} \\
&\quad + h\underbrace{\|\Phi_{\mathbf{f}}(t_k, \mathbf{x}(t_k), h) - \Phi_{\mathbf{f}}(t_k, \boldsymbol{\xi}(t_k, h), h)\|}_{\|\mathbf{f}(t_k,\mathbf{x}(t_k)) - \mathbf{f}(t_k,\boldsymbol{\xi}(t_k,h))\| + O(h)}
\end{aligned}$$

Wegen der Lipschitz-Stetigkeit von $\mathbf{f}$ im zweiten Argument existiert eine Konstante $L > 0$ mit $\|\mathbf{f}(\tau, \xi_1) - \mathbf{f}(\tau, \xi_2)\| \leq L \|\xi_1 - \xi_2\|$, d.h. insbesondere

$$\|\mathbf{f}(t_k, \mathbf{x}(t_k)) - \mathbf{f}(t_k, \xi(t_k, h))\| \leq L \|\mathbf{x}(t_k) - \xi(t_k, h))\| = L\gamma(t_k, h).$$

Damit ergibt sich mit einer geeigneten Konstante $C > 0$ zunächst

$$\gamma(t_{k+1}, h) \leq (1 + hL)\gamma(t_k, h) + Ch^2$$

und daraus mit Lemma 3.6 – $\gamma_k := \gamma(t_k, h)$, $\delta := hL$ und $B := Ch^2$ – die folgende Abschätzung sowie wegen $kh \leq (t - t_0)$ die Konvergenz des ESV:

$$\gamma(t_k, h) \leq \frac{e^{khL} - 1}{hL} Ch^2 = O(h).$$

Satz 3.7. *Die Lösung eines Anfangswertproblems ist robust δ-ℚ-analytisch, falls die rechte Seite stark δ-ℚ-analytisch und auf jedem beschränkten Gebiet Lipschitz-stetig im zweiten Argument ist. Sie kann ohne Division berechnet werden, falls das auf* **f** *zutrifft.*

Beweis. Wir benutzen das explizite Euler-Verfahren zur Schrittweite $h := (t - t_0)(\frac{1}{2})^\delta$; dabei werten wir in jedem Schritt die rechte Seite $\mathbf{f}$ bis auf einen Fehler $\|\varepsilon_\mathbf{f}\| \leq h$ aus. Nach dem oben gezeigten konvergiert das ESV dann bei reeller Arithmetik gegen die korrekte Lösung; Divisionen werden höchstens für die Auswertung von $\mathbf{f}$ gebraucht. Wegen der Stetigkeit der Lösung und ihrer stetigen Abhängigkeit von den Anfangsbedingungen konvergiert das Verfahren auch auf einer δ-ℚ-Maschine wie gewünscht; dabei wird die Maschine $\mathcal{M}_\mathbf{f}$ für die rechte Seite mit einem eigenen δ-Register simuliert[6], um den Auswertungsfehler wie oben zu begrenzen. Man beachte, daß die im Beweis verwendete Lipschitz-Konstante nur für das Gebiet zu existieren braucht, in dem die Polygonzüge des Euler-Verfahrens verlaufen; auch braucht sie nicht berechenbar zu sein. □

Beispiel 3.1 (Exponentialfunktion). Wir wollen nun kurz erläutern, wie man die Funktion $\exp : \mathbb{R} \to \mathbb{R}$, $x \mapsto e^x$ stark δ-ℚ-analytisch *ohne Division* berechnen kann; einige schöne Hinweise und Ideen hierzu verdanke ich Herrn Frank Schulz. Wir gehen von der bekannten Darstellung $e^x = \lim_{n\to\infty}(1 + \frac{x}{n})^n$ zur Teilfolge der $n = 2^k$ über, da sich $2^{-k} = (\frac{1}{2})^k$ leicht ohne Division berechnen läßt. Zur Abschätzung des Approximationsfehlers betrachtet man

$$\forall\, n, m \in \mathbb{N} : \left(1 + \frac{x}{n}\right)^n \leq e^x \leq \left(1 + \frac{x}{m}\right)^{m + \lceil x \rceil},$$

wobei die linke Seite monoton wachsend und die rechte monoton fallend ist; ferner ist die rechte Seite stets mindestens so groß wie die linke und konvergiert gegen denselben Grenzwert. Wir wollen die entsprechenden Beweise nicht näher ausführen, sie

[6] Da die Eingaben bei der Auswertung von $\mathbf{f}(\tau, \xi)$ *rationale* Zahlen sind, kann die Rundung simuliert werden!

verwenden als zentrales Hilfsmittel die Ungleichung von BERNOULLI. Man schließt dann für $n = m$ leicht

$$0 \leq e^x - \left(1 + \frac{x}{n}\right)^n \leq \left(1 + \frac{x}{n}\right)^n \left[\left(1 + \frac{x}{n}\right)^{\lceil x \rceil} - 1\right],$$

was sich für $n = 2^k$ wiederum ohne Division berechnen läßt. Insgesamt haben wir damit bereits ein Verfahren für eine analytische $\mathbb{R}$-Maschine entwickelt; daraus ergibt sich unmittelbar ein stark δ-$\mathbb{Q}$-analytischer Algorithmus, indem man $k = \delta$ wählt. Da die Konvergenz bei Verwendung gerundeter Eingaben jedoch nicht offensichtlich ist, führen wir diesen Beweis aus; dazu sei $x_\delta := \rho(x, \delta)$ die rationale Approximation der reellen Eingabe mit Präzision δ. Mit $n \equiv 2^\delta$ und wegen $|x - x_\delta| < \frac{1}{n}$ gilt

$$\left(1 + \frac{nx - 1}{n^2}\right)^n < \left(1 + \frac{x_\delta}{n}\right)^n < \left(1 + \frac{nx + 1}{n^2}\right)^{n + \lceil x \rceil},$$

wobei die linke Seite mit δ monoton wächst, die rechte monoton fällt und beide durch e^x getrennt sind. Wir zeigen, daß ihre Differenz gegen Null geht, woraus dann sofort $\lim_{\delta \to \infty} \left(1 + \frac{x_\delta}{n}\right)^n = e^x$ folgt.

$$\begin{aligned}
0 &< \left(\frac{n^2 + nx + 1}{n^2}\right)^{n + \lceil x \rceil} - \left(\frac{n^2 + nx - 1}{n^2}\right)^n \\
&= \underbrace{\left(\frac{n^2 + nx + 1}{n^2}\right)^n}_{\leq e^{x+1}} \left[\underbrace{\left(\frac{n^2 + nx + 1}{n^2}\right)^{\lceil x \rceil}}_{\to 1 \text{ für } n \to \infty} - \left(\frac{n^2 + nx - 1}{n^2 + nx + 1}\right)^n\right]
\end{aligned}$$

Jetzt muß man nur noch den Grenzwert des letzten Terms bestimmen; dieser konvergiert wie benötigt gegen 1, da für große n nach BERNOULLI gilt

$$1 \geq \left(1 - \frac{2}{n^2 + nx + 1}\right)^n \geq 1 - \frac{2n}{n^2 + nx + 1} \geq 1 - \frac{2}{n + x} \overset{n \to \infty}{\longrightarrow} 1$$

Indem man für $\frac{2}{n+x}$ eine scharfe obere Schranke der Form 2^{-l} sucht, was ohne Division geht, kann man auch die Fehlerabschätzung divisionsfrei auswerten. □

3.2. Stabilität

Wir betrachten weiterhin Anfangswertprobleme wie in Gleichung 3.1 und setzen nun die Existenz und Eindeutigkeit der Lösung voraus. Ein wichtiger Aspekt bei der Untersuchung solcher DGl ist die Frage nach dem qualitativen Verhalten der Lösung über große Zeiträume hinweg und insbesondere der *Stabilität* bei leicht gestörten Anfangsbedingungen. Dabei werden in der Literatur (siehe z.B. [17] und besonders [21]) viele verschiedene Arten von Stabilität betrachtet, von denen wir einige wichtige näher untersuchen wollen.

Unser Augenmerk richtet sich dabei auf Fragen der Berechenbarkeit solcher qualitativer Eigenschaften einer Lösung, wobei wir davon ausgehen, daß uns ein Anfangswert und ein Programm für die $\mathbb{R}$-berechenbare rechte Seite $\mathbf{f}$ gegeben sind. Nachdem wir zuvor den Schritt von der DGl zur analytischen Rechnung beschrieben haben, nämlich die Ermittlung der Lösung, wollen wir nun den umgekehrten Weg beschreiten und aus Berechnungen Differentialgleichungen konstruieren. Wir erhalten dann das Ergebnis, daß alle betrachteten Stabilitätseigenschaften zumindest für nicht analytische $\mathbb{R}$-Maschinen unentscheidbar sind, indem wir die entsprechenden Fragen übertragen auf das Halte- oder Konvergenzproblem und so ihre Schwierigkeit beleuchten.

Konvergenz Eine fundamentale und sehr einfach zu beschreibende Eigenschaft der Lösung eines Anfangswertproblems ist es, für große Zeiten gegen einen festen Wert zu konvergieren und in diesem Sinne stabil zu werden. Es ist bei näherer Betrachtung jedoch nicht weiter verwunderlich, daß der Versuch, diese Eigenschaft zu entscheiden, auf ein Konvergenzproblem führt und damit in einem starken Sinn nicht berechenbar ist.

Wir werden nun eine analytische Rechnung konstruktiv durch eine DGl beschreiben, deren Lösung zu ganzzahligen Zeitpunkten den Maschinenkonfigurationen entspricht und diese dazwischen linear interpoliert. Sei $\mathcal{M} := \mathcal{M}_\pi^\mathbb{R}$ eine $\mathbb{R}$-Maschine mit Programm π, $x \in \mathbb{R}^*$ eine Eingabe, $N \in \mathbb{N}$ wie zuvor die Dimension des DGl-Systems und $R \subset \{\alpha, \beta, \gamma, y_i, z_j \mid i, j \in \mathbb{N}\}$ eine Menge von $|R| = N$ zu beobachtenden Registern. Dann gibt es eine $\mathbb{R}$-berechenbare rechte Seite $\mathbf{f} = \mathbf{f}_{\pi,x,R}$ und Anfangswerte $(t_0, \mathbf{x}_0)$ für das Problem 3.1, so daß dessen Lösung zum Zeitpunkt $t \in \mathbb{N}$ mit $\Delta^t(\mathrm{in}(x))$, der Konfiguration von $\mathcal{M}$ bei Eingabe x nach t Schritten, auf den Registern aus R übereinstimmt. Dazu wählt man $t_0 = 0$, $\mathbf{x}_0$ gemäß dem Startzustand der Maschine ($\beta = 1$, alle anderen Register 0) und $\mathbf{f}(t, \boldsymbol{\xi}) := \mathbf{f}_{\lfloor t \rfloor + 1} - \mathbf{f}_{\lfloor t \rfloor}$; wobei $\mathbf{f}_i$ für $i \in \mathbb{N}$ die durch Simulation ermittelte Konfiguration $\Delta^i(\mathrm{in}(x))$ eingeschränkt auf R ist. Für negative Zeitpunkte sei $\mathbf{f}_i := \mathbf{x}_0$.

Lemma 3.8. *Die Abbildung, die einem Programm π, einer Eingabe x und einer Registermenge R ein Programm für $\mathbf{f}_{\pi,x,R}$ zuordnet, ist $\mathbb{R}$-berechenbar.* □

Daraus können wir nun folgenden Satz schließen, der sich problemlos vom Körper auf den Ring $\mathbb{R}$ übertragen läßt.

Satz 3.9. *Die Frage, ob die Lösung des Anfangswertproblems 3.1 für große Zeiten konvergiert, ist nicht $\mathbb{R}$-analytisch entscheidbar.*

Beweis. Das Anfangswertproblem sei gegeben durch ein Programm π, so daß $\mathcal{M}_\pi^\mathbb{R}$ die rechte Seite $\mathbf{f}$ regulär berechnet, sowie den Anfangswert $(t_0, \mathbf{x}_0) \in \mathbb{R}^{1+N}$. Wir nehmen an, daß $\mathcal{M}_N$ eine $\mathbb{R}$-Maschine sei, welche die Abbildung

$$(\pi, t_0, \mathbf{x}_0) \mapsto \begin{cases} 1 & \text{falls } \lim\limits_{t\to\infty} \mathbf{x}_{\mathbf{f},t_0,\mathbf{x}_0}(t) \text{ existiert;} \\ 0 & \text{sonst.} \end{cases}$$

für festes $N > 0$ analytisch berechnet. Diese Annahme führen wir zum Widerspruch durch Konstruktion einer $\mathbb{R}$-Maschine $\mathcal{M}_0$, die das Konvergenzproblem analytischer $\mathbb{R}$-Maschinen mit leerer Eingabe und eindimensionaler Ausgabe löst. Nach Satz 1.20 ist dies unmöglich.

Die Maschine $\mathcal{M}_0$ berechnet nach der Methode von Lemma 3.8 zur Eingabe π ein Programm π' für die rechte Seite $\mathbf{f} := \mathbf{f}_{\pi,\varepsilon,R}$, wobei ε die leere Eingabe ist und $R := \{y_1, \ldots, y_N\}$. Man beachte, daß $\mathcal{M}_\pi^\mathbb{R}$ nach Voraussetzung nur eine eindimensionale Ausgabe erzeugt; die anderen Zellen des Ausgabebands werden nur wegen der Dimension des Systems benötigt. Sodann verhält sich $\mathcal{M}_0$ wie die Maschine $\mathcal{M}_N$ auf Eingabe $(\pi', 0, \mathbf{0})$ und entscheidet so analytisch die Konvergenz. □

Fehlerdämpfung und -verstärkung Man interessiert sich im Zusammenhang mit DGl noch für weitere Stabilitätsaspekte, z.B. den der *Fehlerdämpfung*. Dabei wird ein Fehler durch die DGl gedämpft, wenn kleine Abweichungen im Anfangswert die Lösung für große Zeiten nicht ändern; [17] spricht hier auch von *asymptotisch stabil*[7]. Formal heißt das

$$\exists\, \delta > 0\ \forall\, (\tau, \boldsymbol{\xi}) \in U_\delta(t_0, \mathbf{x}_0) :\ \lim_{t\to\infty} |\mathbf{x}_{\mathbf{f},t_0,\mathbf{x}_0}(t) - \mathbf{x}_{\mathbf{f},\tau,\boldsymbol{\xi}}(t)| = 0. \tag{3.3}$$

Ferner wird untersucht, ob kleine Störungen sich für große Zeiten verstärken oder in ihren Auswirkungen beschränkt bleiben. Dabei heißt die Lösung einer DGl *Ljapunov*[8]*-stabil*, falls man durch hinreichend kleine Abweichungen im Anfangswert die Abweichung der Lösung für alle künftigen Zeiten begrenzen kann. Formal heißt das

$$\forall\, \varepsilon > 0\ \exists\, \delta > 0\ \forall\, (\tau, \boldsymbol{\xi}) \in U_\delta(t_0, \mathbf{x}_0)\ \forall\, t \geq t_0 :\ \mathbf{x}_{\mathbf{f},\tau,\boldsymbol{\xi}}(t) \in U_\varepsilon(\mathbf{x}_{\mathbf{f},t_0,\mathbf{x}_0}(t)) \tag{3.4}$$

Man kann nun zu einer gegebenen $\mathbb{R}$-Maschine $\mathcal{M}_\pi^\mathbb{R}$ eine DGl konstruieren, die Fehler genau dann dämpft (rechte Seite $\mathbf{f}_\pi^-$) bzw. verstärkt ($\mathbf{f}_\pi^+$), wenn die Maschine nicht hält. Dazu sei $\mathbf{f}_\pi^\pm : \mathbb{R} \times \mathbb{R}^N \to \mathbb{R}^N$ definiert durch

$$\mathbf{f}_\pi^\pm(t, \boldsymbol{\xi}) := \begin{cases} \pm\boldsymbol{\xi} & \text{falls } t \leq t_{\text{MAX}}; \\ 0 & \text{sonst.} \end{cases}$$

wobei $t_{\text{MAX}} \in \mathbb{N}$ die Länge der Berechnung von $\mathcal{M}_\pi^\mathbb{R}$ bei Eingabe ε ist. Als Anfangswert wählt man $(0, \mathbf{0})$; dann sieht man leicht, daß $\mathbf{x}_{\mathbf{f}_\pi^\pm,0,0} \equiv 0$ und allgemein

$$\mathbf{x}_{\mathbf{f}_\pi^\pm,0,\mathbf{x}_0} = \begin{cases} e^{\pm t}\mathbf{x}_0 & \text{falls } t \leq t_{\text{MAX}}; \\ e^{\pm t_{\text{MAX}}}\mathbf{x}_0 & \text{sonst.} \end{cases}$$

Das Aussehen der Lösungsfunktionen für $N = 1$ in Abhängigkeit von t_{MAX} illustriert die folgende Abbildung 3.2. Man schließt dann leicht auf die Unentscheidbarkeit der Fragen nach Fehlerdämpfung bzw. Stabilität einer DGl.

[7] Die Sprechweise ist in der Literatur jedoch nicht einheitlich, [21] benutzt den Begriff *quasi-asymptotisch stabil* und nennt die Kombination mit Ljapunov-stabil dann asymptotisch stabil.

[8] Der russische Mathematiker ALEXANDER MICHAILOVITSCH LJAPUNOV (1857–1918) begründete die Stabilitätstheorie dynamischer Systeme.

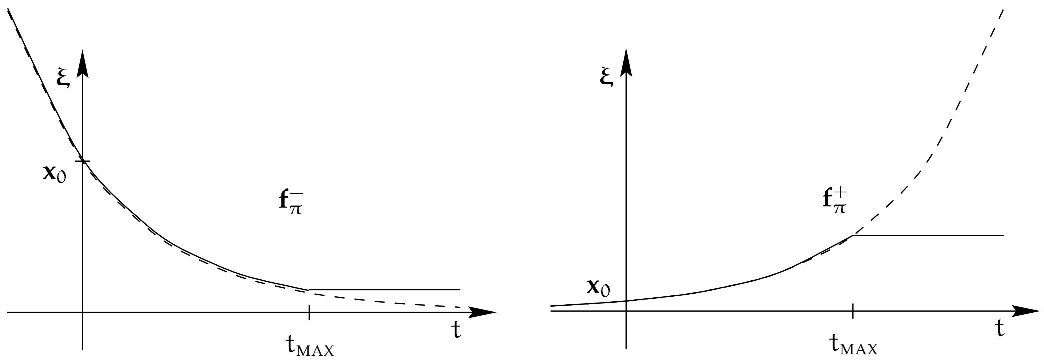


Abbildung 3.2.: Keine Fehlerdämpfung bzw. -verstärkung, falls $t_{MAX} < \infty$

Satz 3.10. *Eine* $\mathbb{R}$*-Maschine kann zu gegebenen* $\mathbf{f}$, t_0, $\mathbf{x}_0$ *nicht regulär entscheiden, ob die DGl mit rechter Seite* $\mathbf{f}$ *und Anfangswert* $(t_0, \mathbf{x}_0)$ *gemäß Gleichung 3.3 Fehler dämpft.*

Beweis. Die Annahme, daß eine $\mathbb{R}$-Maschine $\mathcal{M}$ das genannte Problem löst, führt wie folgt zum Widerspruch. Eine Maschine $\mathcal{M}'$ kann zu jedem π leicht ein Programm für die rechte Seite $\mathbf{f}_\pi^-$ berechnen und dann mittels $\mathcal{M}$ die Fehlerdämpfung der DGl entscheiden. Es gilt aber $\lim_{t\to\infty} \mathbf{x}_{\mathbf{f}_\pi^-,0,\mathbf{x}_0}(t) = 0 \iff (\mathbf{x}_0 = \mathbf{0} \vee t_{MAX} = \infty)$, d.h. die DGl dämpft Anfangsfehler im Bereich $(0, \mathbf{0})$ genau dann, wenn die Maschine $\mathcal{M}_\pi^{\mathbb{R}}$ bei leerer Eingabe nicht anhält. Somit kann $\mathcal{M}'$ das Halteproblem entscheiden, was unmöglich ist. □

Satz 3.11. *Eine* $\mathbb{R}$*-Maschine kann zu gegebenen* $\mathbf{f}$, t_0, $\mathbf{x}_0$ *nicht regulär entscheiden, ob die Lösung einer DGl mit rechter Seite* $\mathbf{f}$ *und Anfangswert* $(t_0, \mathbf{x}_0)$ *gemäß Gleichung 3.4 stabil ist.*

Beweis. Die Annahme, daß eine $\mathbb{R}$-Maschine $\mathcal{M}$ das genannte Problem löst, führt wie folgt zum Widerspruch. Eine Maschine $\mathcal{M}'$ kann zu jedem π leicht ein Programm für die rechte Seite $\mathbf{f}_\pi^+$ berechnen und dann mittels $\mathcal{M}$ die Fehlerverstärkung der DGl entscheiden. Die Lösung der DGl ist für Anfangswerte im Bereich $(0, \mathbf{0})$ genau dann stabil, wenn die Maschine $\mathcal{M}_\pi^{\mathbb{R}}$ bei leerer Eingabe nicht hält; sonst wird jeder noch so kleine Anfangsfehler über alle Schranken verstärkt. Somit kann $\mathcal{M}'$ das Halteproblem entscheiden, was unmöglich ist. □

3.2.1. Dynamische Systeme

Ist das Sonnensystem stabil? Streng genommen ist die Antwort noch unbekannt, und diese Frage hat zu sehr tiefen mathematischen Resultaten geführt, die vermutlich wichtiger sind als die Antwort auf die ursprüngliche Frage.
JÜRGEN MOSER (1975)

Viele Prozesse in Natur und Technik – so auch die Bewegung der Planeten unseres Sonnensystems – hängen wesentlich von der Zeit ab. Das Ziel der allgemeinen Theorie der dynamischen Systeme besteht nach [22] darin, solche zeitabhängigen Prozesse mathematisch zu modellieren, ihre wesentlichen *qualitativen* Eigenschaften zu beschreiben und diese vorherzusagen. Dynamische Systeme im engeren Sinn entsprechen zeitabhängigen Prozessen, die *homogen* bezüglich der Zeit sind. Dabei hängt der Prozeßverlauf – wie bei der Planetenbewegung – nicht vom Anfangszeitpunkt, sondern nur vom Anfangszustand ab.

Man kann nun kontinuierliche dynamische Systeme durch *autonome* DGl modellieren, bei denen die rechte Seite $\mathbf{f}$ nicht von der Zeit abhängt. Wir werden daher unsere bisherigen Unentscheidbarkeitsresultate bezüglich Stabilitätsfragen auf autonome DGl übertragen. Da die Entwicklung der Lösung nicht vom Anfangszeitpunkt abhängt, wird ein Startwert bereits durch die Komponente $\mathbf{x}_0$ hinreichend festgelegt und wir wählen o.B.d.A. stets $t_0 = 0$.

Konvergenz Um die Frage nach der Konvergenz des Zustands eines dynamischen Systems für große Zeiten wie zuvor auf das Konvergenzproblem analytischer Maschinen zu reduzieren, bedarf es eines Tricks. Die rechte Seite $\mathbf{f}$ der DGl ist nunmehr autonom, also unabhängig von der Zeit; diese spielt jedoch bei der Simulation einer $\mathbb{R}$-Maschine eine wesentliche Rolle und muß deshalb nachgebildet werden. Für den Fall $N > 1$ kann man einfach einer Komponente von $\boldsymbol{\xi} = (\xi_1, \ldots, \xi_N)$ die Rolle der Zeit zuweisen, wobei man darauf achten muß, daß diese „Zeit" nicht divergiert. Daher wählen wir als DGl nicht – wie es naheliegend wäre – $\xi_N' = 1$ mit Startwert 0, sondern vielmehr $\xi_N' = -\ln 2 \cdot \xi_N$ mit Startwert 1. Dann gilt $\xi_N(t) = 2^{-t}$, $\lim_{t\to\infty} \xi_N = 0$ und insbesondere ist $\xi_N \mapsto \lfloor t \rfloor = \lfloor \log 1/\xi_N \rfloor$ $\mathbb{R}$-berechenbar. Anschließend verfährt man wie zuvor in Lemma 3.8 mit der Definition von $\mathbf{f}_{\pi,x,R}$ für die restlichen Komponenten von $\boldsymbol{\xi}$, wobei nur $\lfloor t \rfloor$ und nicht t selbst benötigt wird. Daraus ergibt sich unmittelbar der folgende Analogon zu Satz 3.9.

Hauptsatz 2. *Die Frage, ob die Lösung des Anfangswertproblems*

$$\mathbf{x}' = \mathbf{f}(\mathbf{x}),\ \mathbf{x}(0) = \mathbf{x}_0$$

mit autonomer, $\mathbb{R}$-berechenbarer rechter Seite $\mathbf{f} : \mathbb{R}^N \to \mathbb{R}^N$ und Startwert $\mathbf{x}_0 \in \mathbb{R}^N$ für große Zeiten konvergiert, ist im Fall $N > 1$ nicht $\mathbb{R}$-analytisch entscheidbar.

Für den verbleibenden Fall $N = 1$ können wir nach dem Schema der Abschnitte über Fehlerdämpfung und -verstärkung leicht zeigen, daß Konvergenz zumindest für $\mathbb{R}$-Maschinen mit endlicher Rechenzeit unentscheidbar ist. Dazu definieren wir zu gegebener $\mathbb{R}$-Maschine $\mathcal{M}_\pi^{\mathbb{R}}$

$$\mathbf{f}_\pi(\boldsymbol{\xi}) := \begin{cases} 1 & \text{falls } \boldsymbol{\xi} < t_{\mathrm{MAX}}; \\ 0 & \text{sonst.} \end{cases}$$

und betrachten einen beliebigen Startwert $\mathbf{x}_0 \in \mathbb{R}$. Dann gilt $\lim_{t\to\infty} \mathbf{x}_{\mathbf{f}_\pi,0,\mathbf{x}_0}(t) = \max\{\mathbf{x}_0, t_{\mathrm{MAX}}\}$ existiert in $\mathbb{R}$ genau dann, wenn $t_{\mathrm{MAX}} < \infty$.

Satz 3.12. *Eine $\mathbb{R}$-Maschine kann zu gegebenen $\mathbf{f}$ und $\mathbf{x}_0$ nicht regulär entscheiden, ob die Lösung der DGl mit autonomer, $\mathbb{R}$-berechenbarer rechter Seite $\mathbf{f} : \mathbb{R} \to \mathbb{R}$ und Startwert $\mathbf{x}_0 \in \mathbb{R}$ für große Zeiten konvergiert.*

Fehlerdämpfung und -verstärkung Wir definieren $\mathbf{f}^{\pm} : \mathbb{R}^N \to \mathbb{R}^N$ durch

$$\mathbf{f}^{\pm}_{\pi}(\boldsymbol{\xi}) := \begin{cases} \pm\boldsymbol{\xi}, & \text{falls } \boldsymbol{\xi} \neq 0 \text{ und } |\boldsymbol{\xi}|^{-1} \leq t_{\text{MAX}}; \\ 0 & \text{sonst.} \end{cases}$$

und erhalten für $t_{\text{MAX}} = \infty$ die Funktion $\mathbf{f}^{\pm}_{\pi}(\boldsymbol{\xi}) = \pm\boldsymbol{\xi}$ mit Lösung $e^{\pm t}\mathbf{x}_0$. Diese dämpft bzw. verstärkt alle Anfangsfehler in einer Umgebung von $\mathbf{0}$. Im Falle $t_{\text{MAX}} < \infty$ bleiben Anfangsfehler mit $|\mathbf{x}_0| < t^{-1}_{\text{MAX}}$ konstant; außerhalb dieser Umgebung werden sie bis zum kritischen Betrag gedämpft bzw. über alle Grenzen hinweg verstärkt. Wie im nicht autonomen Fall ist Fehlerdämpfung also äquivalent zu $t_{\text{MAX}} = \infty$. Ein Unterschied ergibt sich bei der Fehlerverstärkung und damit der Stabilität der Lösungen; diese ist genau für endliche Berechnungen gegeben. Man sieht leicht, daß zu jedem $\varepsilon > 0$ die Wahl von $\delta := \min\{\varepsilon, t^{-1}_{\text{MAX}}\}$ gewährleistet, daß Anfangsfehler mit Betrag kleiner δ zu einer (konstanten) Lösung führen, die um weniger als ε von der ungestörten Lösung $\mathbf{x}_{\mathbf{f}^{\pm}_{\pi},0,0} \equiv \mathbf{0}$ abweicht. Daraus ergibt sich unmittelbar der

Satz 3.13. *Eine $\mathbb{R}$-Maschine kann zu gegebenen $\mathbf{f}$ und $\mathbf{x}_0$ nicht regulär entscheiden, ob die DGl mit autonomer, $\mathbb{R}$-berechenbarer rechter Seite $\mathbf{f} : \mathbb{R}^N \to \mathbb{R}^N$ und Anfangswert $(0, \mathbf{x}_0) \in \mathbb{R} \times \mathbb{R}^N$ gemäß Gleichung 3.3 Fehler dämpft oder ob ihre Lösung gemäß Gleichung 3.4 stabil ist.*

Fazit

Wir haben uns mit Anfangswertproblemen für Systeme expliziter, gewöhnlicher DGl erster Ordnung beschäftigt, wobei die rechte Seite von einer $\mathbb{R}$-Maschine ohne Division berechnet werden kann bzw. stark δ-$\mathbb{Q}$-analytisch ist. Deren globale Lösungen sind unter den von uns betrachteten hinreichenden Voraussetzungen eindeutig und robust δ-$\mathbb{Q}$-analytisch. Dies zeigt die Mächtigkeit des Modells der analytischen δ-$\mathbb{Q}$-Maschinen und ihre Eignung zur Beschreibung zahlreicher Prozesse aus Natur und Technik.

Sodann wurden verschiedene Stabilitätsbedingungen für solche Lösungen betrachtet, und es wurde gezeigt, daß diese nicht regulär bzw. analytisch entscheidbar sind. Schließlich haben wir diese Ergebnisse auf autonome DGl übertragen und damit auf dynamische Systeme anwendbar gemacht. Es zeigt sich, daß bereits die einfachste Formulierung des Stabilitätsproblems, nämlich „Konvergiert ein dynamisches System gegen einen stabilen Zustand im Sinne eines Fixpunktes?“, selbst für das mächtige Berechnungsmodell der $\mathbb{R}$-analytischen Maschinen, das eine präzise reelle Arithmetik und unendliche Laufzeit umfaßt, unentscheidbar ist.

A. Notation

Die Menge der natürlichen Zahlen *einschließlich* der Null bezeichnen wir mit $\mathbb{N}$; $\widehat{\mathbb{N}} := \mathbb{N} \cup \{\infty\}$ sei ergänzt um die uneigentliche Zahl „unendlich". Für zwei ganze Zahlen $i, j \in \mathbb{Z}$ sei $[i : j] := \{k \in \mathbb{Z} \mid i \leq k \leq j\}$ das abgeschlossene ganzzahlige Intervall von i bis j. Mit $2\mathbb{Z} + 1$ ist die Menge der ungeraden Zahlen gemeint. $\mathbb{B} := \{0, 1\}$ bezeichne die Menge der Bits oder Booleschen Werte. Die Teilmenge $\mathbb{D} := \{p \cdot 2^{-q} \mid p \in \mathbb{Z},\ q \in \mathbb{N}\}$ der rationalen Zahlen heißt Menge der *dyadischen Zahlen*.

Für einen beliebigen Ring $\mathcal{R}$ und Unbestimmte X_i, $i \in I$, sei $\mathcal{R}[X_i \mid i \in I]$ der *Polynomring* über $\mathcal{R}$ mit Variablen X_i. Ist $\mathcal{R}$ ein Körper, so sei $\mathcal{R}(X_i \mid i \in I)$ der *Erweiterungskörper*, der aus $\mathbb{Q}$ durch Adjunktion der X_i entsteht.

Für eine beliebige Menge M bezeichne id_M die identische Abbildung auf M. Für eine beliebige Funktion $f : M \to M$ und eine natürliche Zahl n sei f^n die n-fache Hintereinanderausführung von f; dabei sei $f^0 = \mathrm{id}_M$. Für eine Teilmenge $N \subset M$ bezeichne $f|_N$ die Einschränkung von f auf N und $\chi_N : M \to \mathbb{B}$ mit $\chi_N(x) = 1 \iff x \in N$ die *charakteristische Funktion*.

Eine partielle Funktion f von einer Menge M nach N bezeichnen wir mit $f : M \rightsquigarrow N$ bzw. $f : M \supset D \to N$, falls wir ihren Definitionsbereich $D \subset M$ benennen wollen. Sind zwei Funktionen $f, g : M \to N$ sowie eine (partielle) Ordnung auf N gegeben, so sei $f \sim g$ für $\sim \in \{<, \leq, =, \dots\}$ als *punktweiser Vergleich* $\forall\, x \in M : \ f(x) \sim g(x)$ zu verstehen.

Es stehe M^n für die Menge aller Folgen von n Elementen aus M, d.h. $M^n := \{(m_1, \dots, m_n) \mid m_1, \dots, m_n \in M\}$; dabei sei ε_M oder kurz ε das *leere Wort* über M, d.h. die Folge der Länge Null. Für ein Wort $m \in M^n$ sei $|m| := n$ seine Länge. Mit M^* werde die Menge aller endlichen Folgen über M bezeichnet, also $M^* := \bigcup_{n=0}^{\infty} M^n$. Die Menge M^* bildet mit der Konkatenation $\cdot$ von Wörtern ein Monoid mit neutralem Element ε_M. $M^\omega := \{m : \mathbb{N} \to M\}$ stellt die unendlichen Folgen über M dar; die Vereinigung aller Folgen über M sei $M^\star := M^* \cup M^\omega$. Ist M zumindest ein Monoid mit neutralem Element 0, so bezeichne dazu M^∞ die unendliche direkte Summe von M mit sich selbst, also die Menge jener unendlichen Folgen über M, bei denen nur *endlich viele* Elemente ungleich 0 sind.

Für zwei reelle Zahlen $x \leq y$ bezeichnen $[x, y]$, $]x, y[$ und $[x, y[$ das abgeschlossene, offene und halboffene Intervall von x bis y. Mit $\lfloor x \rfloor$ bzw. $\lceil x \rceil$ bezeichnen wir den zu einer ganzen Zahl ab- bzw. aufgerundeten Wert von x. Für $\varepsilon > 0$ sei $U_\varepsilon(x) := \{y \in \mathbb{R} \mid |x - y| < \varepsilon\}$ die offene Umgebung von x mit Radius ε.

Für eine Teilmenge A eines topologischen Raums, z.B. des $\mathbb{R}^n$, sei $\mathrm{int}(A)$ das offene Innere, $\overline{A}$ der Abschluß und $\partial A = \overline{A} \setminus \mathrm{int}(A)$ der Rand.

Zur Beschreibung des asymptotischen Verhaltens von Funktionen bedienen wir uns der Landauschen Symbole, die wir hier noch einmal kurz erläutern. Für eine Funktion $f : \mathbb{N} \to \widehat{\mathbb{N}}$ definieren wir

$$O(f) := \{g : \mathbb{N} \to \widehat{\mathbb{N}} \mid \exists\, 0 < C \in \mathbb{R}, N \in \mathbb{N}\, \forall n \geq N : \; g(n) \leq Cf(n)\}.$$

Ferner sei $g = \Theta(f) \iff g = O(f) \wedge f = O(g)$; diese Begriffe beschreiben qualitativ das Wachstum von $f(n)$ für $n \to \infty$. Wie üblich schreiben wir $g = O(f)$ statt $g \in O(f)$ und z.B. $g = O(n^2)$, wenn wir $g \in O(f)$ mit $f : \mathbb{N} \to \widehat{\mathbb{N}}$, $f(n) := n^2$ meinen; insbesondere schreiben wir $\Theta(1)$ für eine Konstante. Es sollte stets aus dem Zusammenhang klar sein, was tatsächlich gemeint ist.

Im reellen Fall benötigen wir ferner den Begriff $O(h)$ für den Grenzübergang positiver reeller h gegen Null, wobei wir $O(f)$ meinen mit $f : \mathbb{R} \to \mathbb{R}$, $f(h) := h$. Wir definieren dazu für eine geeignete Funktion $f : \mathbb{R} \to \mathbb{R}$

$$O(f) := \{g : \mathbb{R} \to \mathbb{R} \mid \exists\, 0 < C \in \mathbb{R} : \; \lim_{x \to 0} \left| \frac{g(x)}{f(x)} \right| \leq C\}.$$

Mit log ohne Angabe der Basis bezeichnen wir den Logarithmus zur Basis zwei, mit ln hingegen den natürlichen Logarithmus zur Basis e.

Ausführliche Zusammenfassung

In dieser Arbeit präsentieren wir einige Resultate über *analytische Maschinen* hinsichtlich des Berechenbarkeitsbegriffs über $\mathbb{Q}$ und $\mathbb{R}$, der Lösungen von Differentialgleichungen und des Stabilitätsproblems dynamischer Systeme.

Maschinenmodell Eine $\mathcal{R}$-Maschine im Sinne von BLUM, SHUB, SMALE ist im wesentlichen eine Registermaschine mit unbegrenztem Speicher, die Elemente des Ringes $\mathcal{R}$ mittels *exakter* Arithmetik verarbeitet. Analytische Berechnungen sind unendliche Berechnungen, die eine konvergente Folge von Ausgaben produzieren. Wir erweitern das Modell der $\mathbb{Q}$-Maschine um ein Präzisionsregister δ, welches die Genauigkeit der Rundung reeller Zahlen beim Einlesen in ein rationales Register steuert; damit können nun analytische δ-$\mathbb{Q}$-Maschinen ebenfalls Funktionen über $\mathbb{R}$ berechnen.

Durch die Varianten des Modells ergibt sich eine Hierarchie von Klassen berechenbarer Funktionen, die wir eingehend studieren im Hinblick auf strikte Inklusion und Abschlußeigenschaften. Ein Vergleich der Mächtigkeit dieser Berechnungsmodelle über den Körpern $\mathbb{Q}$ und $\mathbb{R}$ zeigt z.B., daß endliche Berechnungen mit reellen Zahlen durch unendliche, konvergente Berechnungen mit rationalen Zahlen simuliert werden können, wobei die Genauigkeit der Approximation während des Prozesses nicht bekannt ist. Der Einfluß der Rundungsfunktion, welche bei δ-$\mathbb{Q}$-Maschinen reelle Zahlen in rationale abbildet, geht so weit, daß einstellige $\mathbb{R}$-analytische Funktionen durch geeignete Wahl der Rundung δ-$\mathbb{Q}$-analytisch sind. Es werden Beispiele gegeben für $\mathbb{R}$-analytische Funktionen, die *nicht mit jeder* bzw. *mit keiner* Rundungsfunktion δ-$\mathbb{Q}$-analytisch berechenbar sind; d.h. insbesondere, daß analytische Berechnungen über $\mathbb{R}$ echt mächtiger sind als über $\mathbb{Q}$.

Wir betrachten die Hintereinanderschaltung mehrerer analytischer Maschinen, insbesondere im Zusammenhang mit dem Konvergenzproblem, dem analytischen Äquivalent des Halteproblems. Dabei zeigt sich, daß zwei Maschinen die Konvergenz einer Maschine entscheiden können und allgemein $i + 1$ Maschinen mächtiger sind als i Stück; letzteres ist eine Verallgemeinerung des Hierarchiesatzes. Die spezielle Klasse der quasi stark δ-$\mathbb{Q}$-analytischen Maschinen, welche mit jeder Ausgabe auch eine *fast immer* richtige Schranke für die Abweichung vom Grenzwert berechnen, ist abgeschlossen unter der Hintereinanderschaltung und umfaßt die $\mathbb{R}$-Maschinen; letzteres ist unsere Verschärfung des Simulationssatzes. Zuletzt wird eine Unterprogrammtechnik eingeführt, mit der gewissermaßen die Grenzwertbildung als

Elementaroperation möglich wird; dabei ergibt sich eine obere Schranke für die unendliche Hierarchie $\mathfrak{i}$ hintereinander geschalteter analytischer $\mathbb{R}$-Maschinen.

Komplexität Wir definieren die Kosten einer Berechnung und die Komplexität einer Maschine für die beiden Aspekte Zeit und Platz, und zwar sowohl für $\mathbb{R}$- als auch für δ-$\mathbb{Q}$-Maschinen. Es ergibt sich die Verallgemeinerung auf analytische Rechnungen des bekannten Resultats, daß jede berechenbare Funktion mit einer konstanten Anzahl von Registern ausgewertet werden kann. $\mathcal{R}$-Maschinen mit unterschiedlich vielen Indexregistern sind polynomiell verknüpft, woraus man folgern kann, daß der Simulationssatz effizient ist. Ebenso ist der Ressourcenverbrauch unserer Registermaschinen und der Graphmaschinen von BLUM, SHUB, SMALE im wesentlichen gleich.

Damit sind einerseits die grundlegenden Begriffe der Komplexitätstheorie auf unser Maschinenmodell übertragen, worauf aufbauend man z.B. Fragen der NP-Vollständigkeit studieren könnte. Andererseits rechtfertigt dies voll und ganz unsere etwas andere, aber äquivalente Beschreibung reeller Berechenbarkeit und der zugrundeliegenden Maschinen.

Differentialgleichungen Wir beschäftigen uns mit Anfangswertproblemen für Systeme expliziter, gewöhnlicher Differentialgleichungen erster Ordnung, wobei die rechte Seite von einer $\mathbb{R}$-Maschine ohne Division berechnet werden kann. Die allgemeine Idee dabei ist, daß eindeutige Lösungen mittels des klassischen Euler-Verfahrens berechenbar sind, und das wird unter gewissen hinreichenden Voraussetzungen für die Eindeutigkeit bewiesen.

Dazu wird zunächst untersucht, was man in diesem Zusammenhang unter einer globalen Lösung verstehen kann, gezeigt, daß eine solche stets existiert, und illustriert, daß sie nicht immer eindeutig bestimmt ist. Falls man gewisse Entartungen der rechten Seite ausschließt und einen geeigneten Startpunkt wählt, ist die globale Lösung eindeutig und $\mathbb{R}$-analytisch berechenbar. Unser Hauptsatz verschärft dieses Resultat, indem er mit rationaler Arithmetik auskommt, genauer mit einer robust δ-$\mathbb{Q}$-analytischen Berechnung. Unter klassischen Voraussetzungen der numerischen Mathematik läßt sich ein entsprechendes Ergebnis auch für stark δ-$\mathbb{Q}$-analytische rechte Seiten zeigen.

Sodann werden verschiedene Stabilitätsbedingungen für die Lösungen solcher Differentialgleichungen betrachtet, und es wird gezeigt, daß diese Bedingungen nicht regulär bzw. analytisch entscheidbar sind. Schließlich übertragen wir diese Ergebnisse auf autonome Differentialgleichungen, deren rechte Seite nicht von der Zeit abhängt, und machen sie damit auf dynamische Systeme anwendbar. Es zeigt sich, daß bereits die einfachste Formulierung des Stabilitätsproblems, nämlich „Konvergiert ein dynamisches System gegen einen stabilen Zustand im Sinne eines Fixpunktes?“, selbst für das mächtige Berechnungsmodell der $\mathbb{R}$-analytischen Maschinen, das eine präzise reelle Arithmetik und unendliche Laufzeit umfaßt, unentscheidbar ist.

Literaturverzeichnis

[1] Fred G. Abramson. Effective computation over the real numbers. In *Conference Record 1971 Twelfth Annual Symposium on Switching and Automata Theory*, pages 33–37, East Lansing, Michigan, 13–15 October 1971. IEEE.

[2] Michael Ben-Or. Lower bounds for algebraic computation trees (preliminary report). In *Proceedings of the 15th Annual ACM Symposium on Theory of Computing*, pages 80–86, Boston, Massachusetts, 25–27 April 1983. ACM.

[3] Lenore Blum. Lectures on a theory of computation and complexity over the reals (or an arbitrary ring). Technical Report TR-89-065, International Computer Science Institute, Berkeley, California, December 1989.

[4] Lenore Blum. Lectures on a theory of computation and complexity over the reals (or an arbitrary ring). In *Proceedings of the International Congress of Mathematicians Kyoto 1990*, volume 2, pages 1491–1507, Tokyo, 1991. Math. Soc. Japan, Springer-Verlag. cf. [3].

[5] Lenore Blum, Felipe Cucker, Mike Shub, and Steve Smale. Complexity and real computation: A manifesto. Technical Report TR-95-042, International Computer Science Institute, Berkeley, California, 1995. cf. [7].

[6] Lenore Blum, Felipe Cucker, Mike Shub, and Steve Smale. Algebraic settings for the problem P =? NP. http://math.berkeley.edu/~smale/biblio/p_equal_np.ps, 1996.

[7] Lenore Blum, Felipe Cucker, Mike Shub, and Steve Smale. Complexity and real computation: A manifesto. *International Journal of Bifurcation and Chaos*, 6:3–26, 1996.

[8] Lenore Blum, Felipe Cucker, Mike Shub, and Steve Smale. *Complexity and Real Computation*. Springer-Verlag, New York, 1997.

[9] Lenore Blum, Mike Shub, and Steve Smale. On a theory of computation and complexity over the real numbers: NP-completeness, recursive functions and universal machines. *Bulletin (New Series) of the American Mathematical Society*, 21(1):1–46, July 1989. ICSI technical report TR-88-012.

[10] Vasco Brattka. Recursive characterizations of computable real-value functions and relations. *Theoretical Computer Science*, 162(1):45–77, 5 August 1996.

[11] Vasco Brattka and Peter Hertling. Feasible real random access machines (extended abstract). In Ko and Weihrauch [30].

[12] Vasco Brattka and Peter Hertling. Feasible real random access machines. http://www.informatik.fernuni-hagen.de/thi1/vasco.brattka/papers/ram.html, 3 April 1998.

[13] Felipe Cucker. The arithmetical hierarchy over the reals. *Journal of Logic and Computation*, 2(3):375–395, June 1992.

[14] Felipe Cucker, Marek Karpinski, Pascal Koiran, Thomas Lickteig, and Kai Werther. On real Turing machines that toss coins. In *Proceedings of the 27th Annual ACM Symposium on Theory of Computing*, pages 335–342, Las Vegas, Nevada, 29 May–1 June 1995. ACM.

[15] Felipe Cucker, Pascal Koiran, and Martín Matamala. Complexity and dimension. *Information Processing Letters*, 62(4):209–212, 28 May 1997.

[16] Felipe Cucker, Mike Shub, and Steve Smale. Separation of complexity classes in Koiran's weak model. *Theoretical Computer Science*, 133(1):3–14, 10 October 1994. Selected Papers of The Workshop on Continous Algorithms and Complexity, Barcelona, Spain, October 1993.

[17] Peter Deuflhard and Folkmar Bornemann. *Integration gewöhnlicher Differentialgleichungen*, volume 2 of *Numerische Mathematik*. de Gruyter, 1994.

[18] T. Emerson. Relativizations of the P =? NP question over the reals (and other ordered rings). *Theoretical Computer Science*, 133(1):15–22, 10 October 1994. Selected Papers of The Workshop on Continous Algorithms and Complexity, Barcelona, Spain, October 1993.

[19] Kenneth J. Falconer. *Fractal Geometry: Mathematical Foundations and Applications*. John Wiley & Sons Ltd., 1990.

[20] Hervé Fournier and Pascal Koiran. Are lower bounds easier over the reals? Research Report 97-38, Ecole Normale Supérieure de Lyon, 69364 Lyon Cedex 07, France, 10 October 1997.

[21] Paul Glendinning. *Stability, Instability, and Chaos: An Introduction to the Theory of Nonlinear Differential Equations*. Cambridge University Press, cambridge texts in applied mathematics edition, 1994.

[22] G. Grosche, E. Zeidler, D. Ziegler, and V. Ziegler, editors. *Teubner-Taschenbuch der Mathematik*, volume 2. B.G. Teubner Leipzig, 1995.

[23] Robert L. Grossman, Anil Nerode, Anders P. Ravn, and Hans Rischel, editors. *Hybrid Systems*. Number 736 in Lecture Notes in Computer Science. Springer-Verlag, 1993.

[24] A. Grzegorczyk. On the definition of computable real continuous functions. *Fundamenta Mathematicae*, 44:61–71, 1957.

[25] Günter Hotz. Über Berechenbarkeit fraktaler Strukturen. Abhandlungen der mathematisch-naturwissenschaftlichen Klasse 1, Akademie der Wissenschaften und der Literatur, Mainz, 1994.

[26] Günter Hotz, Björn Schieffer, and Gero Vierke. Analytic machines. Technical Report TR95-025, Electronic Colloquium on Computational Complexity, 1995. http://www.eccc.uni-trier.de/eccc.

[27] Ker-I Ko. *Complexity Theory of Real Functions*. Progress in theoretical computer science. Birkhäuser, Boston, 1991.

[28] Ker-I Ko. Recent progress on complexity theory of real functions. In Ko and Weihrauch [30].

[29] Ker-I Ko and Harvey Friedman. Computational complexity of real functions. *Theoretical Computer Science*, 20(3):323–352, July 1982. Fundamental study.

[30] Ker-I Ko and Klaus Weihrauch, editors. *Computability and Complexity in Analysis*, number 190-9/1995 in Informatikberichte, D-58084 Hagen, 1995. FernUniversität. Workshop proceedings published as technical report.

[31] Pascal Koiran. A weak version of the Blum, Shub & Smale model. In *34th Annual Symposium on Foundations of Computer Science*, pages 486–495, Palo Alto, California, 3–5 November 1993. IEEE.

[32] Pascal Koiran. Computing over the reals with addition and order. *Theoretical Computer Science*, 133(1):35–47, 10 October 1994. Selected Papers of The Workshop on Continous Algorithms and Complexity, Barcelona, Spain, October 1993.

[33] Alfred K. Louis. Höhere Numerik. Vorlesungsmitschrift, SS 1993.

[34] Wolfgang Maass. Bounds for the computational power and learning complexity of analog neural nets (extended abstract). In *Proceedings of the 25th Annual ACM Symposium on Theory of Computing*, pages 335–344, San Diego, California, 16-18 May 1993. ACM. cf. [35].

[35] Wolfgang Maass. Bounds for the computational power and learning complexity of analog neural nets. *SIAM Journal on Computing*, 26(3):708–732, June 1997.

[36] Christian Michaux. $P \neq NP$ over the nonstandard reals implies $P \neq NP$ over $\mathbb{R}$. *Theoretical Computer Science*, 133(1):95–104, 10 October 1994. Selected Papers of The Workshop on Continous Algorithms and Complexity, Barcelona, Spain, October 1993.

[37] Wolfgang J. Paul. *Komplexitätstheorie*, volume 39 of *Leitfäden der angewandten Mathematik und Mechanik*. Teubner, Stuttgart, 1 edition, 1978.

[38] Roger Penrose. *The Emperor's New Mind*. Penguin, 1991.

[39] Marian Boykan Pour-El and J. Ian Richards. *Computability in analysis and physics*. Perspectives in mathematical logic. Springer-Verlag, Berlin, 1989.

[40] Karl Rüdiger Reischuk. *Einführung in die Komplexitätstheorie*. Leitfäden und Monographien der Informatik. B.G. Teubner, Stuttgart, 1990.

[41] Hava T. Siegelmann and Eduardo D. Sontag. On the computational power of neural nets. In *Proceedings of the Fifth ACM Workshop on Computational Learning Theory*, pages 440–449, Pittsburgh, July 1992. ACM. cf. [43].

[42] Hava T. Siegelmann and Eduardo D. Sontag. Analog computation via neural networks. *Theoretical Computer Science*, 131(2):331–360, 12 September 1994.

[43] Hava T. Siegelmann and Eduardo D. Sontag. On the computational power of neural nets. *Journal of Computer and System Sciences*, 50(1):132–150, February 1995.

[44] Gero Vierke. Berechenbarkeit reellwertiger Funktionen und analytische Berechnungen. Diplomarbeit, Universität des Saarlandes, Saarbrücken, Germany, July 1995.

[45] Klaus Weihrauch. *Computability*. Number 9 in EATCS Monographs on Theoretical Computer Science. Springer-Verlag, 1987.

[46] Klaus Weihrauch. A foundation of computable analysis. In Ko and Weihrauch [30].

[47] E. Zeidler, editor. *Teubner-Taschenbuch der Mathematik*, volume 1. B.G. Teubner Leipzig, 1996. Begründet von I.N. Bronstein und K.A. Semandjajew.