Nils Adermann
Jahrgangsstufe 12

Mathematik LK2
Frau Pesch
2. März 2005

Facharbeit Mathematik

Die Definition des Algorithmusbegriffs und Darstellung der wesentlichen Merkmale eines Algorithmus an ausgesuchten Beispielen

Abu Abdallah Muhammad ibn Musa al-Khwarizmi

Inhaltsverzeichnis

1 Vorwort

„*Unter einem Algorithmus versteht man allgemein eine mehr oder weniger genau definierte Handlungsvorschrift zur Lösung eines Problems oder einer bestimmten Art von Problemen.*“ (Wikipedia, 2005, [6]) Darunter fallen auch viele Vorschriften, Regeln und Anleitungen für Vorgänge aus dem täglichen Leben, wenn sie genau genug ausformuliert sind. Algorithmen könnten zum Beispiel Rezepte, Bedienungsanleitungen, Pläne für den Aufbau eines Regals oder Waschmaschinenprogramme sein.

Doch stellt sich unmittelbar die Frage, was genau man dann unter einem Algorithmus versteht. Doch dieser „*intuitiv klar erscheinende Begriff*“ (Wikipedia, 2005 [6]) ist nicht einheitlich definiert, so habe ich in verschiedener Literatur unterschiedliche Definitionen gefunden. In dieser Facharbeit möchte ich die wesentlichen Merkmale eines Algorithmus anhand von zwei Beispielen vorstellen. Mit Hilfe dieser Merkmale erläutere ich danach einen weit verbreiteten Ansatz, um den Algorithmusbegriff zu definieren.

Nachdem Kurt Gödel 1931 mit seinem Unvollständigkeitssatz, der besagt, dass jedes hinreichend mächtige formale System entweder widersprüchlich oder unvollständig ist, die Hoffnungen der Wissenschaftler, dass man jedes Problem, das ausreichend mathematisch und genügend präzise formuliert ist, durch eine Maschine gelöst werden könne, zu Nichte gemacht hatte, wurde der Algorithmusbegriff selbst zum Forschungsgegenstand (vgl. Teubner Taschenbuch der Mathematik, 2003 [3]).

2 Geschichte des Algorithmus

2.1 Begriffsherkunft

Im Jahr 825 veröffentlichte Abu Abdallah Muhammad ibn Musa al-Khwarizmi (verschiedene Schreibweisen möglich), der auf der Titelseite auf einer 1983 ausgegebenen sowjetischen Briefmarke zu sehen ist, ein Buch mit dem Titel „*Al-kitab al-muchtasar fi hisab al-dschabr wa-l-muqabala*“, das von Regeln zur Wiederherstellung und Reduktion handelt und wird „*meist als Geburtsstunde der klassischen Algebra, der Wissenschaft vom Lösen von Gleichungen, angesehen*“ (Wikipedia, 2005 [6]). Das Buch hatte großen Einfluss auf die arabische und europäische Mathematik. Es wurde auch ins lateinische übersetzt, dabei entwickelte sich aus dem Titel al-dschabr das Wort Algebra. Außerdem beginnt das Buch mit den Worten „*Dixit Algorithmi ...*“ (auf deutsch „*Also Sprach Algorithmi*“), eine Wortschöpfung, die vom Namen seines Geburtsortes al-Khwarizmi abgeleitet wurde. Außerdem ähnelt das Ende des Wortes Algorithmus dem griechischen Ausdruck für Zahl „*arithmós*“ (vgl. Schreiber, 2001 [5]).

2.2 Erste Algorithmen

Bereits in der Antike gab es die ersten mathematischen Algorithmen. Einige bekannte sind das Sieb des Eratosthenes, Verfahren zur Suche nach der Kreiszahl π und zur Bestimmung von Quadratwurzeln. Einer der antiken Algorithmen, der euklidischen Algorithmus zur Bestimmung des größten gemeinsamen Teilers, der bereits 300 v. Chr. beschrieben wurde, wird in dieser Facharbeit noch vorgestellt. Auch die Ideen vom Bau eines Automaten und der Konstruktion einer künstlichen Sprache stammen bereits aus der Antike (vgl. Schreiber, 2001 [5]).

Nach dem Mittelalter gewannen Algorithmen immer mehr an Bedeutung. Später stand vor allem die Informatik im Mittelpunkt. Der erste für einen Computer gedachte Algorithmus wurde 1842 von Ada Lovelace entwickelt. Allerdings hat Charles Babbages die „*Analytical Engine*“ auf der der Algorithmus hätte implementiert werden sollen nie fertig gestellt, weshalb der Algorithmus nicht in die Praxis umgesetzt werden konnte. Trotzdem gilt Ada Lovelace als die erste Programmiererin (vgl. Wikipedia, 2005 [6]).

3 Algorithmen zur Veranschaulichung

3.1 Der euklidische Algorithmus

„Nimmt man abwechselnd immer das Kleinere vom Größeren weg, dann muss der Rest schließlich die vorhergehende Größe messen ...“
Euklid, Die Elemente, Zehntes Buch §3,

Der euklidische Algorithmus ist ein Verfahren zum Errechnen des größten gemeinsamen Teilers (ggT) zweier natürlicher Zahlen. Ein Beispiel für die Anwendung dieses Algorithmus ist die Ermittlung eines gemeinsamen Hauptnenners von zwei Brüchen. Im Schulunterricht wird meist eine Methode angewandt die auf Primfaktorzerlegung basiert. Historisch gesehen ist der euklidische Algorithmus allerdings von großer Bedeutung. Desweiteren ist er anschaulich zu beschreiben und wird deshalb häufig als Beispiel für einen Algorithmus herangezogen (vgl. Ziegenbalg: Der Euklidische Algorithmus [8]).

3.1.1 Funktionsweise

Es wird der größte gemeinsame Teiler von $a, b \in \mathbb{N}$ gesucht. Nach Euklid wendet man nun das folgende Verfahren an:

1. Man setze $m = a; n = b$
2. Wenn $m < n$, dann vertausche man m und n
3. Man berechne $r = m - n$
4. Man setze $m = n; n = r$
5. Wenn $n \neq m$, dann fahre man mit Schritt 2 fort.

Der größte gemeinsame Teiler ist also gefunden, wenn $m = n$. Es gibt noch eine Reihe von weiteren Methoden den euklidischen Algorithmus zu implementieren. In der heutigen Zeit wird im 3. Schritt häufig statt der Differenz von m und n der Rest der Division von m durch n verwendet. Dies hat den Vorteil, dass bei großen Differenzen von a und b weniger Subtraktionsschritte notwendig sind. Eine weitere Variante des klassischen euklidischen Algorithmus ist die rekursive Version. Letztendlich kommt man aber mit allen Varianten zum selben Ergebnis.

3.1.2 Anwendungsbeispiel

Um das Verfahren noch einmal zu veranschaulichen, möchte ich es mit zwei frei gewählten Zahlen vorführen.

In diesem Beispiel sei also $a = 35$ und $b = 21$.

Schritt	m	n	r	
1.	35	21	0	a und b werden eingesetzt
2.	35	21	0	m und n müssen nicht vertauscht werden.
3.	35	21	14	$r = 35 - 21$
4.	21	14	14	$m = n; n = r$
5.	21	14	14	$n \neq m$, fahre fort mit Schritt 2
2.	21	14	14	m und n müssen nicht vertauscht werden.
3.	21	14	7	$r = 21 - 14$
4.	14	7	7	$m = n; n = r$
5.	14	7	7	$n \neq m$, fahre fort mit Schritt 2
2.	14	7	7	m und n müssen nicht vertauscht werden.
3.	14	7	7	$r = 14 - 7$
4.	7	7	7	$m = n; n = r$
5.	7	7	7	$n = m$

Der größte gemeinsame Teiler von 35 und 21 ist dem zu Folge 7.

3.2 Der Eliminationsalgorithmus

Der Eliminationsalgorithmus, der meistens als Gaußsches Eliminationsverfahren bezeichnet wird, wurde um 1850 von Carl Friedrich Gauß veröffentlicht. Das Verfahren beschreibt ein Vorgehen zum Lösen linearer Gleichungssysteme. Somit gehört es zum Gebiet der Linearen Algebra und wird häufig im Zusammenhang mit der analytischen Geometrie gebraucht (vgl. Wikipedia, 2005 [6]).

Eine Gleichung der Form $a_1x_1+a_2x_2+...+a_nx_n = b$ mit den Variablen $x_1, ..., x_n \in \mathbb{R}$ und den Zahlen $a_1, ..., a_n, b \in \mathbb{R}$, wobei $a_1, ..., a_n$ die Koeffizienten sind, heißt lineare Gleichung.

m lineare Gleichungen der Form

$$\begin{array}{ccccccccc} a_{11}x_1 & + & a_{12}x_2 & + & ... & + & a_{1n}x_n & = & b_1 \\ a_{21}x_1 & + & a_{22}x_2 & + & ... & + & a_{2n}x_n & = & b_2 \\ \vdots & & \vdots & & \vdots & & \vdots & & \vdots \\ a_{m1}x_1 & + & a_{m2}x_2 & + & ... & + & a_{mn}x_n & = & b_m \end{array}$$

mit den Variablen $x_1, ..., x_n \in \mathbb{R}$ und den Zahlen $a_{ik}, b_i \in \mathbb{R}(i = 1, ..., m; k = 1, ..., n)$ heißen lineares Gleichungssystem mit m Gleichungen und n Variablen oder kurz $m \times n$-System.

Jedes n-Tupel $(x_1, ..., x_n)$ von reellen Zahlen, für die alle Gleichungen erfüllt sind, heißt Lösung des linearen Gleichungssystems (vgl. Möller, 1997 [4]).

3.2.1 Funktionsweise

Der Eliminationsalgorithmus lässt sich in zwei Teile aufteilen. Zunächst eine Vorwärtselimination bei der die Lösungsmatrix in die so genannte Dreiecksform gebracht wird und danach ein Rückwärtseinsetzen der Variablen, wodurch man die Gleichungen löst. Die Lösungsmatrix enthält lediglich die Koeffizienten der Variablen. Sie hat also folgende Form:

$$\left(\begin{array}{ccccc|c} a_{11} & a_{12} & a_{13} & ... & a_{1n} & b_1 \\ a_{21} & a_{22} & a_{23} & ... & a_{2n} & b_2 \\ a_{31} & a_{32} & a_{33} & ... & a_{3n} & b_3 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{m1} & a_{m2} & a_{m3} & ... & a_{mn} & b_m \end{array}\right)$$

Da jede Zeile eine lineare Gleichung repräsentiert, darf man äquivalente Umformungen vornehmen. Durch Addition des Vielfachen einer Gleichung zu einer anderen, formt man die Lösungsmatrix so um, dass alle Koeffizienten die unter der Hauptachse liegen $(a_{21}, a_{31}, a_{32}, ..., a_{m1}, a_{m2}, ..., a_{m(n-1)})$ gleich Null sind.

$$\left(\begin{array}{ccccc|c} a'_{11} & a'_{12} & a'_{13} & ... & a'_{1n} & b'_1 \\ 0 & a'_{22} & a'_{23} & ... & a'_{2n} & b'_2 \\ 0 & 0 & a'_{33} & ... & a'_{3n} & b'_3 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & ... & a'_{mn} & b'_m \end{array}\right)$$

Durch diese Umformung werden in der untersten Gleichung alle Summanden außer $a'_{mn}x_n$ gleich Null. Daraus ergibt sich die Gleichung $a'_{mn}x_n = b'_m$, die sich nach x_n auflösen lässt. Danach setzt man x_n in die darüber liegende Gleichung ein und kann diese nach x_{n-1} auflösen, dies führt man bis zur obersten Gleichung fort, danach hat man für jedes x_i mit $0 \leq i \leq n$ eine Lösung, die man in einem n-Tupel angibt. Allerdings kann es auch vorkommen, dass es statt einer eindeutigen Lösung keine oder unendlich viele Lösungen gibt (vgl. Möller, 1997 [4]).

Ob ein lineares Gleichungssystem lösbar ist und wie viele Lösungen es gibt lässt sich von der untersten Zeile der Matrix in Dreiecksform ablesen.

$0x_1 + 0x_2 + ... + 0x_n = b'_m (b'_m \neq 0)$

Das lineare Gleichungssystem hat keine Lösung.

$0x_1 + 0x_2 + ... + 0x_n = 0$

Das lineare Gleichungssystem hat unendlich viele Lösungen.

$0x_1 + 0x_2 + ... + 0x_{n-1} + a'_{mn}x_n = b'_m (a'_{mn})$

Für x_n gibt es genau eine eindeutige Lösung. Allerdings muss man jede Zeile überprüfen um eine eindeutige Aussage über die Lösbarkeit machen zu können.

3.2.2 Anwendungsbeispiel

Auch dieser Algorithmus soll an einem Beispiel veranschaulicht werden. Ich wähle hierfür drei Gleichungen mit drei Variablen.

$$\begin{array}{rcrcrcr} -3x_1 & + & 2x_2 & - & 3x_3 & = & 6 \\ 9x_1 & - & 2x_2 & + & 10x_3 & = & -10 \\ 6x_1 & + & 8x_2 & + & 14x_3 & = & 22 \end{array}$$

Daraus ergibt sich folgende Lösungsmatrix:

$$\Rightarrow \left(\begin{array}{ccc|c} -3 & 2 & -3 & 6 \\ 9 & -2 & 10 & -10 \\ 6 & 8 & 14 & 22 \end{array}\right)$$

Im nächsten Schritt addiert man zur zweiten Zeile das 3-fache der ersten, so dass $a'_{21} = 0$ wird. Außerdem addiert man zur dritten Zeile das 2-fache der ersten Zeile. Dadurch wird auch $a'_{31} = 0$.

$$\Leftrightarrow \left(\begin{array}{ccc|c} -3 & 2 & -3 & 6 \\ 0 & 4 & 1 & 8 \\ 0 & 12 & 8 & 34 \end{array}\right)$$

Als nächstes wird das (-3)-fache der zweiten Zeile zur dritten addiert, damit auch $a'_{32} = 0$ wird und somit die Dreiecksform hergestellt ist. In diesem Schritt addiert man in der Regel nicht die erste zur dritten Zeile um sicher zu stellen, dass $a_{31} = 0$ bleibt.

$$\Leftrightarrow \left(\begin{array}{ccc|c} -3 & 2 & -3 & 6 \\ 0 & 4 & 1 & 8 \\ 0 & 0 & 5 & 10 \end{array}\right)$$

Aus dieser Matrix kann nun die Gleichung zum Finden einer Lösung für x_3 ausgelesen werden.

$$\begin{aligned} \Rightarrow \quad & 5x_3 && = 10 \\ \Leftrightarrow \quad & x_3 && = 2 \end{aligned}$$

Durch Einsetzen in die zweite Zeile wird es möglich, die nächste Gleichung nach x_2 aufzulösen.

$$\begin{aligned} in(2): \quad & 4x_2 + 2 && = 8 \\ \Leftrightarrow \quad & 4x_2 && = 6 \\ \Leftrightarrow \quad & x_2 && = 1\tfrac{1}{2} \end{aligned}$$

Und auch die Gleichung der obersten Zeile mit der letzten Variable kann jetzt durch Einsetzen aufgelöst werden.

$$\begin{aligned} in(1): \quad & -3x_1 + 2 \cdot 1\tfrac{1}{2} - 3 \cdot 2 && = 6 \\ \Leftrightarrow \quad & -3x_1 + 3 - 6 && = 6 \\ \Leftrightarrow \quad & -3x_1 - 3 && = 6 \\ \Leftrightarrow \quad & -3x_1 && = 9 \\ \Leftrightarrow \quad & x_1 && = -3 \end{aligned}$$

Nach Lösung der drei Gleichungen ergibt sich also folgende Lösungsmenge:

$$\mathbb{L} = \{(-3|1\frac{1}{2}|2)\}$$

4 Charakteristische Merkmale eines Algorithmus

Im folgenden Kapitel werden die für einen Algorithmus charakteristischen Merkmale vorgestellt und anhand der vorhergehenden Beispiele erläutert. Anzumerken ist, dass sich der Algorithmusbegriff in der heutigen Zeit nicht mehr „*nur auf das Rechnen mit Zahlen*“ (Ziegenbalg, 1996 [7]) beschränkt, was zwar zur etymologischen Herkunft des Begriffs passt, aber nur einer „*sehr reduzierte[n] Schicht*“ (Ziegenbalg, 1996 [7]) dessen, was man heute unter Rechnen und Berechenbarkeit versteht, entspricht. So sind auch die meisten Computerprogramme Algorithmen, die in einer formalen Sprache formuliert sind, was weit über das reine Zahlen-Rechnen hinaus geht.

4.1 Finitheit

4.1.1 Statische Finitheit

Unter statischer Finitheit versteht man die „*Endlichkeit des Textes durch den der Algorithmus beschrieben wird ([...] endliche Folge von Elementaranweisungen [...]*“ (Ziegenbalg, 1996 [7]). Eine statische Finitheit war in beiden vorgestellten Algorithmen gegeben, andernfalls hätten sie im Rahmen dieser Facharbeit gar nicht erklärt werden können. Der euklidische Algorithmus besteht aus 5 Elementaranweisungen und auch der Gaußsche Eliminationsalgorithmus enthält endlich viele Anweisungen. Ein Text, der eine endliche Folge von Elementaranweisungen enthält wird auch Quelltext genannt.

4.1.2 Dynamische Finitheit

Zu jedem Zeitpunkt darf ein Algorithmus nur endlich viel Speicherbedarf haben. Der euklidische Algorithmus benötigt zu jedem Zeitpunkt lediglich Platz für drei Zahlen (m, n, r). Der Gaußsche Eliminationsalgorithmus benötigt Speicher für eine Matrix, außerdem Platz für eine Zahl pro Schritt zum Errechnen der neuen Koeffizienten und im letzten Schritt Speicher für drei weitere Gleichungen. Die jederzeit gegebene Endlichkeit des Speicherbedarfs nennt man dynamische Finitheit.

4.2 Terminiertheit

Ein terminierender Algorithmus bricht für jede beliebige Eingabe nach endlich vielen Schritten gesteuert ab. Allerdings gibt es keine einheitliche Meinungen in der Frage, ob ein nicht terminierender Algorithmus überhaupt ein Algorithmus ist. Vorallem bei Computerprogrammen wie Betriebssystemen und anderen Programmen, die auf Benutzerinteraktion warten stellt sich diese Frage, da sie, wenn der Benutzer keine Eingabe liefert nicht terminieren. „*Donald Knuth schlägt in diesem Zusammenhang*

vor, nicht terminierende Algorithmen als rechnergestützte Methoden (‚Computational Methods') zu bezeichnen." (Wikipedia, 2005 [6]).

Der euklidische Algorithmus terminiert, wenn $n = r$, da dieser Fall spätestens erreicht ist, wenn $n = 1$ und $a, b \in \mathbb{N}$, ist er für jede Eingabe terminierend. Auch der Gaußsche Eliminationsalgorithmus ist terminierend, da man die Lösungsmatrix in $n - 1$ Schritten in die Dreiecksform bringt und danach m lineare Gleichungen in endlicher Zeit löst.

4.3 Determiniertheit

Ein Algorithmus ist determiniert genau dann, wenn er bei gleichen Parametern und Startwerten zum gleichen Ergebnis kommt. Sowohl der euklidische, als auch der Gaußsche Eliminationsalgorithmus sind offensichtlich determiniert, denn zwei Zahlen können nur einen größten Gemeinsamen Teiler haben und ein lineares Gleichungssystem kann nicht zwei verschiedene Lösungsmengen besitzen.

Nicht determinierte Algorithmen sind Algorithmen, in denen absichtlich ein Zufallsfaktor mit einbezogen wird. Solche Algorithmen werden stochastische, randomisierte oder prohabilistische Algorithmen genannt.

4.4 Determinismus

Im Abschnitt, der die statische Finitheit von Algorithmen behandelte wurden bereits Elementaranweisungen erwähnt. Ein Algorithmus ist deterministisch, wenn es bei jeder Elementaranweisung nur maximal eine Möglichkeit zur Fortsetzung gibt. Falls es mehrere Möglichkeiten zur Fortsetzung gibt, so spricht man, wie bei nicht determinierten Algorithmen, von stochastischen, randomisierten oder prohabilistischen Algorithmen.

Beide vorgestellten Algorithmen sind deterministisch, da sie determiniert sind und in jedem Schritt an die entsprechenden eindeutigen mathematischen Regeln gebunden sind. Alle deterministischen Algorithmen sind auch determiniert. Allerdings sind nicht alle determinierten Algorithmen automatisch auch deterministisch.

Es gibt außerdem einen Nichtdeterminismus, der hauptsächlich in der theoretischen Informatik zum Einsatz kommt. Die Ausführung solcher Algorithmen würde aber durch Quantencomputer ermöglicht. Bei einem nichtdeterministischen Algorithmus gibt es für eine Elementaranweisung verschiedene Möglichkeiten in den nächsten Zustand überzugehen.

5 Formale Definition des Algorithmusbegriffs

5.1 Formalisierungen des Berechenbarkeitsbegriffs

Die Frage nach der Berechenbarkeit ist eng verbunden mit der Definition des Algorithmus, denn ein Algorithmus darf nur aus Elementaranweisungen bestehen, die berechenbar sind, da der Algorithmus ansonsten nicht ausführbar wäre. Mitte der 30er Jahre entwickelten Alonzo Church, Stephen Cole Kleene, Alan Mathison Turing und einige weitere Wissenschaftler verschiedene Berechenbarkeitsmodelle.

Kleene stellte 1936 eine funktionsorientierte Formalisierung der Berechenbarkeit vor. Er beschrieb „*Grundfunktionen*“, „*Erzeugungsschemata*“ und Verfahren zur Berechnung der Funktionswerte. Hierbei spielen vorallem rekursive Funktionen eine wichtige Rolle (vgl. Teubner Taschenbuch der Mathematik, 2003 [3]). Alonzo Church entwickelte einen unter dem Namen Lambda Kalkül bekannten Ansatz, dabei wird durch die Lambda-Notation eindeutig, ob man die ganze Funktion mit allen ihren Funktionswerten meint oder nur die Funktion an der Stelle x. Weitere Berechenbarkeitsmodelle sind Markov Systeme, Chomsky Grammatiken, logische Formeln, rekursive Gleichungssysteme, Registermaschinen und while-Programme (vgl. Felscher, 1993 [1]).

Das von Alan Turing entwickelte mathematische Modell, dessen Grundidee es ist, „*Eingabewörter Schrittweise durch elementare Programme [...] in Ausgabewörter zu überführen*“ (Teubner Taschenbuch der Mathematik, 2003 [3]), wird Turing Maschine genannt. Sie errechnet aus dem Eingabealphabet und einem Anfangszustand eine Menge von Endzuständen. Dabei muss eine zulässige Zustandsmenge und ein Bandalphabet definiert sein. Die Turing Maschine terminiert, wenn das Blanksymbol erreicht ist (vgl. Teubner Taschenbuch der Mathematik, 2003 [3]).

Letztendlich konnte aber unter maßgeblicher Mitarbeit der oben genannten Personen gezeigt werden, dass alle diese Modelle gleichwertig sind. Es gibt für jedes der Modelle eine Turing Maschine, die das Verhalten des anderen Modells emulieren kann und auch jedes andere Modell kann eine Turing Maschine emulieren.

5.2 Definition mit Hilfe der Turing Maschine

Mit Hilfe des Berechenbarkeitsmodells der Turing Maschine gelangt man zur folgenden Definition des Algorithmusbegriffs:

> „*Eine Berechnungsvorschrift zur Lösung eines Problems heißt Algorithmus genau dann, wenn eine zu dieser Berechnungsvorschrift äquivalente Turingmaschine existiert, die für jede Eingabe stoppt.*“

Diese Definition berücksichtigt aufgrund der Eigenschaften einer Turing Maschine mehrere bereits vorgestellte Charakteristiken eines Algorithmus. Ein Algorithmus

muss dynamisch und statisch finit sein, berechenbar sein und terminieren (vgl. Wikipedia, 2005 [6]).

5.3 Church-Turing-These

Die Church'sche oder Church-Turing These lautet:

„*Jede im intuitiven Sinne berechenbare Funktion ist turingmaschinen berechenbar.*“

Eine Alternative formulierung:

„*Der intuitive Begriff ‚Algorithmus‘ wird mathematisch durch die Turing Maschine erfasst.*“

Das bedeutet für die im Vorwort beschriebenen Algorithmen, dass sie, wenn sie präzise genug formuliert werden können im mathematischen Sinne eine Turing Maschine sind. Diese These wird allgemein anerkannt, da es bis jetzt nicht gelungen ist, Funktionen aufzustellen, die nicht durch eine Turing Maschine emuliert werden können. Allerdings ist es bis jetzt auch nicht möglich diese These zu beweisen (vgl. Teubner Taschenbuch der Mathematik, 2003 [3]).

Literatur

[1] Felscher, Walter: Berechenbarkeit. Rekursive und programmierbare Funktionen. Springer-Verlag, Berlin/Heidelberg 1993

[2] Grauert, Hans: Lineare Algebra und Analytische Geometrie. Oldenbourg Wissenschaftsverlag, München/Wien 1999

[3] Grosche, G., Zeidler, E., Ziegler, D., Ziegler, V. (Hrsg.): Teubner Taschenbuch der Mathematik. Teil II. 8. Auflage. B.G. Teubner Verlag, Wiesbaden 2003

[4] Möller, Herbert: Algorithmische Lineare Algebra. Eine Einführung für Mathematiker und Informatiker. Verlag Vieweg, Braunschweig/Wiesbaden 1997

[5] Schreiber, Alfred: „Was ist ein Algorithmus“, Universität Flensburg, Stand: 18. Januar 2001, ⟨http://www.uni-flensburg.de/mathe/zero/veranst/algorithmen/algo_abschn11/algo_abschn11.html⟩, Abfrage: am 14. Februar 2005

[6] Wikipedia, die freie Enzyklopädie: „Algorithmus“, Wikipedia, Stand: 18. Februar 2005, ⟨http://de.wikipedia.org/wiki/Algorithmus⟩ und ⟨http://de.wikipedia.org/wiki/Gaußsches_Eliminationsverfahren⟩, Abfrage: am 22. Februar 2005, Bild: ⟨http://de.wikipedia.org/wiki/Muhammad_ibn_Musa_al-Chwarizmi⟩

[7] Ziegenbalg, Jochen: Algorithmen. Von Hammurapi bis Gödel. Spektrum, Akademischer Verlag, Heidelberg/Berlin/Oxford 1996

[8] Ziegenbalg, Jochen: Der Euklidische Algorithmus. Institut für Mathematik und Informatik, Pädagogische Hochschule Karlsruhe ohne Datum, siehe Anhang

6 Anhang

Im Anhang finden befinden sich Ausdrucke verwendeter Internet Seiten und „*Der Euklidische Algorithmus*“ von Jochen Ziegenbalg.

Ich erkläre hiermit, dass ich die Facharbeit ohne fremde Hilfe angefertigt und nur die im Literaturverzeichnis aufgeführten Quellen und Hilfsmittel benutzt habe.

Essen, 2. März 2005