Professur für Kognitive Systeme


Bachelorarbeit

im Studiengang Angewandte Informatik
der Fakultät Wirtschaftsinformatik
und Angewandte Informatik
der Otto-Friedrich-Universität Bamberg

Thema:

# Ein regelbasierter Ansatz zur Induktion von Zahlenreihen

Verfasser: Alexander Strätz

Gutachter: Prof. Dr. Ute Schmid

# Inhaltsverzeichnis

# Abbildungsverzeichnis

# 1 Einleitung

Zahlenreihen-Probleme sind sehr häufig Teil von Intelligenztests, in denen man eine gegebene Sequenz an Zahlen logisch fortsetzen muss. Hierbei sollen sie die menschliche Intelligenz messbar machen. Sie sind klassische Probleme des induktiven Schließens und der Muster-Erkennung und somit interessant für die Philosophie, die Mathematik, die Psychologie und die Informatik. Zahlenreihen-Probleme sind insbesondere für die künstliche Intelligenz interessant, weil sich prinzipiell jede berechenbare Funktion in den Relationen der Zahlen verbergen kann und die Operatoren nicht nur auf die Grundrechenarten beschränkt sind. In Intelligenztests werden sie benutzt, um die Befähigung zur Erkennung von mathematischen Mustern zu messen. Dieser Aspekt Muster zu erkennen ist für jedes intelligente System wichtig.

In dieser Arbeit wird ein regelbasierter Ansatz zur Induktion von Zahlenreihen vorgestellt, der sich an der menschlichen Vorgehensweise orientiert. Daher wird nicht die typische Herangehensweise von Programmen verfolgt, bei der versucht wird, die Zahlenreihe durch eine mathematische Formel zu beschreiben. Stattdessen orientiert sich das vorgestellte System an einem Modell zum menschlichen Lösen von Zahlenreihen und soll Relationen zwischen den Zahlen erkennen und so Übereinstimmungen finden, um die Sequenzen logisch fortsetzen zu können. Dabei sollen die gleichen Rechenschritte wiedergegeben werden, die auch vom Menschen nötig wären, um das Muster einer Zahlenreihe zu erkennen.

Zunächst werden der allgemeine Aufbau und die Hintergründe von Intelligenztests als Hauptanwendungsgebiet von Zahlenreihen vorgestellt. Anschließend wird ein Modell zum menschlichen Lösen von Zahlenreihen dargestellt und auf die Fragestellung eingegangen, welche Faktoren beeinflussen, ob eine Zahlenreihe leicht oder schwer zu lösen ist. Bevor der eigene regelbasierte Ansatz und sein Algorithmus in Kapitel 3 vorgestellt werden, werden zuerst bereits existierende, alternative Ansätze zur Induktion von Zahlenreihen umrissen. Daraufhin werden diese in ihrer Leistung verglichen und die Ergebnisse beurteilt. Zuletzt wird schließlich auf mögliche Verbesserungen und Erweiterungen des Ansatzes eingegangen.

# 2 Induktion von Zahlenreihen

Obwohl die Induktion von Zahlenreihen in den verschiedensten Gebieten auftritt, ist der mit Abstand bekannteste Bereich, die Anwendung in Intelligenztests. Deswegen werden in diesem Kapitel zunächst der allgemeine Aufbau von Intelligenztests und Aufgabentypen des induktiven Schließens dargestellt. Anschließend werden sowohl die menschliche Herangehensweise an Zahlenreihen, als auch bereits existierende Computersysteme zu deren Lösung vorgestellt.

## 2.1 Aufbau von Intelligenztests

Der Begriff der Intelligenz (lateinisch intelligentia: Einsicht, Verständnis) kann sehr unterschiedlich definiert werden. Dies führt dazu, dass es sehr viele unterschiedliche Testverfahren gibt, die sich mehr oder weniger gut für die Abklärung spezifischer Fragestellungen im Bereich der Intelligenzabklärung eignen [7]. Was die meisten Intelligenztests gemeinsam haben, ist das Endergebnis, das im Normalfall als Intelligenzquotient (auch: IQ) ausgedrückt wird. Dabei wird das Ergebnis der Testperson mit einer Referenzgruppe verglichen, die sich aus der Gesamtpopulation oder Teilgruppen, wie zum Beispiel Schüler eines bestimmten Alters, zusammensetzt. Die Testergebnisse der Referenzgruppe werden auf eine Normalverteilung mit dem Mittelwert 100 und einer Standardabweichung von 15 umskaliert [4]. Dies führt dazu, dass man bei einem IQ von circa 100 ein Maximum findet und ungefähr zwei Drittel der Personen einen IQ zwischen 85 und 115 haben sollten.

Auch wenn der Begriff des Intelligenzquotienten, der von den meisten Intelligenztests gleichermaßen verwendet wird, nahe legt, dass es sich bei den verschiedenen Testverfahren um die selben erfassten Fähigkeiten handelt, ist dies nicht der Fall. Da die verschiedenen Testverfahren auf unterschiedlichen theoretischen Konzepten aufbauen, würde ein und dieselbe Person bei der Bearbeitung verschiedenartiger Intelligenztests teilweise weit abweichende Intelligenzquotienten erreichen. Worauf klassische Intelligenztests beruhen, charakterisiert Jens Asendorpf folgendermaßen: „Intelligenz ist, was Intelligenztests messen, die so konstruiert wurden, dass sie das Bildungsniveau möglichst gut vorhersagen, oder kurz: Intelligenztests messen die Befähigung zu hoher Bildung"[1].

Charles Spearman entwickelte das erste explizite Intelligenzmodell. Aus der Tatsache, dass die Leistungen verschiedener kognitiver Tests positiv korrelieren, schloss er, dass dem ein gemeinsamer Faktor zugrunde liegen muss. Diese allgemeine Intelligenz bezeichnete er als „g“ (general factor). Sein Modell bezeichnet man als Zwei-Faktoren-Theorie [11]. Bei dem zweiten Faktor neben „g“, handelt es sich um „s“. Dieser Faktor beinhaltet die zusätzlichen, spezifischen Faktoren, die die Varianz der Tests erklären sollen. Die einzelnen Aufgabentypen besitzen unterschiedlich hohe „g“- bzw. „s“-Anteile. Induktionsaufgaben zum Beispiel, wie Buchstaben- und Zahlenreihen, aber auch Ravens figurale Matrizenaufgaben weisen dabei einen hohen Anteil am „general factor“ auf.

Da sich aber empirisch zeigte, dass „g“ als einzelner Faktor nicht erklären konnte, warum bestimmte Aufgabentypen stärker miteinander korrelieren, als sie es sollten, erweiterte Thurstone diese Theorie zu einem Modell mit mehreren, gemeinsamen Faktoren [13]. Diese Faktoren, die jeweils eine „primäre“, grundlegende Fähigkeit widerspiegeln sollten,

bezeichnete er als „Gruppenfaktoren“. Wiederholt belegen ließen sich sieben von ihnen [7]:

1. verbales Verständnis
2. Wortflüssigkeit
3. schlussfolgerndes Denken, Erkennen von Regelhaftigkeiten
4. räumliches Vorstellungsvermögen
5. Merkfähigkeit, Kurzzeitgedächtnis
6. Rechenfähigkeit
7. Wahrnehmungsgeschwindigkeit

Da es sich nach Thurstone bei diesen zwar um verschiedene, aber nicht völlig unabhängige Faktoren handelte, wählte er für seine Berechnungen eine Variante der Faktorenanalyse, die Interkorrelationen zwischen den Faktoren erlaubte. Diese Theorie von Thurstone wurde zum Beispiel bei älteren Versionen des Intelligenz-Struktur-Tests zugrunde gelegt. Bezogen auf Zahlenreihen werden von den Gruppenfaktoren, sowohl schlussfolgerndes Denken/Erkennen von Regelhaftigkeiten und die Rechenfähigkeit, als auch das Kurzzeitgedächtnis (vgl. 2.2.2) abgedeckt. Im Folgenden werden verschiedene Aufgabentypen im Bereich des induktiven Schließens vorgestellt:

**Buchstabenreihen:**
Bei Buchstabenreihen müssen wie auch bei Zahlenreihen die Gesetzmäßigkeiten, nach denen eine Reihe aufgebaut ist, erkannt werden und anschließend logisch fortgesetzt werden.

Beispiel für eine Buchstabenreihe: A, D, G, J, ?

In diesem einfachen Beispiel wäre die richtige Lösung „M“. Die Sequenz erfolgt in alphabetischer Reihenfolge, beginnend mit A, wobei zwischen den Elementen jeweils zwei Buchstaben ausgelassen werden. Dieses Buchstabenbeispiel lässt sich äquivalent in eine Zahlenreihen-Problemstellung übersetzen: 1, 4, 7, 10, ? (entspricht einer Additionsreihe: +3).

**Ravens Progressive Matrizen:**
Dieses Testverfahren wurde 1938 von John C. Raven konzipiert. Das Ziel war es ein sprachfreie und kulturunabhängige Testmethode zu entwickeln, bei welcher man Verzerrungen aufgrund von kulturellen Unterschieden ausschließen konnte. Allerdings kam die Kritik auf, dass auch bei sprachfreien Tests kulturelle Unterschiede das Testergebnis beeinflussen können, da sich die Kultur auch durch unterschiedliche Denkstile und kulturelle Erfahrungen bemerkbar macht. Trotzdem sind Matrizen-Tests aktuell immer noch sehr beliebt bei der Erstellung von Intelligenztests, weil sie im Sinne Spearmans einen sehr hohen „g“- Faktor erfassen [7]. Die Faktoren bei der Bearbeitung dieser Matrizen-Tests wurden von Carpenter, Just und Shell untersucht [3]. Dabei sollten, unter anderem mit Hilfe von zwei Computer-Modellen, die Umstände untersucht werden, die Testpersonen

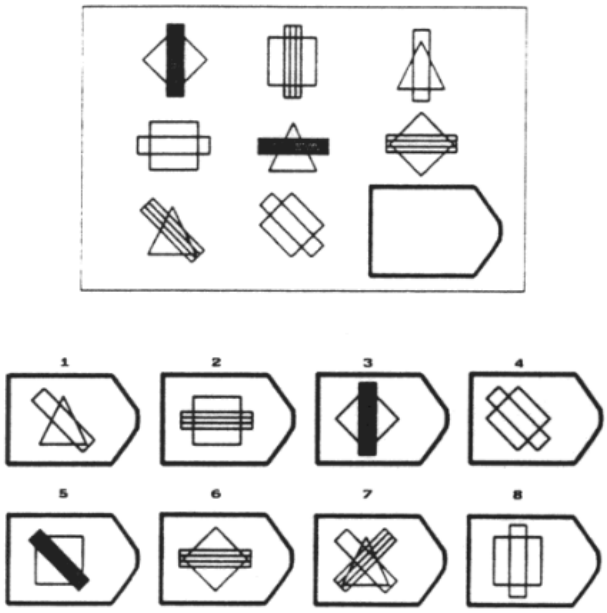



Abbildung 1: Beispiel für eine figurale Matrizenaufgabe [3]

in Bezug auf ihre Leistung voneinander unterschieden.

Ziel ist es, sich die Konstruktionsregel, nach der die Grafiken entweder in horizontaler oder vertikaler Richtung aufgebaut sind, bewusst zu machen und anschließend die logisch richtige Ergänzung aus den acht dargebotenen, verschiedenen Möglichkeiten zu wählen. Beim Beispiel in Abbildung 1 wäre die fünfte Wahlmöglichkeit richtig. In jeder Zeile befinden sich jeweils einmal ein Diamant, einmal ein Quadrat und einmal ein Dreieck. Daraus folgt, dass in der letzten Zeile ein Quadrat fehlt. Außerdem befindet sich in jeder Grafik ein Balken. Dieser ist sowohl in jeder Zeile gleich ausgerichtet (in der letzten Zeile von links oben nach recht unten), als auch unterschiedlich dargestellt (in jeder Zeile einmal weiß, einmal schwarz und einmal schraffiert). Daraus folgt, dass die Lösung 5 richtig ist, da es sich hier um ein Quadrat mit einem schwarzen Balken, von links oben nach rechts unten, handelt.

**Zahlenreihen:** Bei Zahlenreihen handelt es sich, äquivalent zu Buchstabenreihen, um eine Sequenz von Elementen, die logisch fortgesetzt werden muss. Der wichtigste Unterschied zu Buchstabenreihen besteht darin, dass die Relationen der Elemente weitaus vielfältiger sein können. So sind bei Zahlenreihenproblemen alle Arten von arithmetischen Operationen möglich. Neben den Grundrechenarten wären zum Beispiel Potenzen, Fakultäten, Quersummen oder Primzahl-Folgen denkbar. Der größte Vorteil gegenüber figuralen Matrizen im Bereich der Modellierung ist, dass keine zusätzliche Objekterkennung implementiert werden muss.

## 2.2 Menschliches Induktives Schließen

Im Folgenden wird erläutert wie der Mensch an die Lösung von Zahlenreihen herangeht und welche kognitiven Aspekte dabei eine Rolle spielen. Anschließend werden die Faktoren, die die Schwierigkeit einer Sequenz beeinflussen, vorgestellt und auf das Problem

der Eindeutigkeit von Zahlenreihen-Problemen eingegangen.

### 2.2.1 Modell zum Lösen von Zahlenreihen

Wie Menschen beim Lösen von Zahlenreihen vorgehen, wurde von Thomas G. Holzman untersucht. Bei seinem Experiment [6] kam er zu dem Schluss, dass der größte Faktor, der den Schwierigkeitsgrad von Zahlenreihen beeinflusst, die Informationsmenge ist, die man im Arbeitsgedächtnis behalten muss, während man die Bildungsregel der Zahlenreihe anwendet. Um das menschliche Vorgehen beim Lösen von Zahlenreihen zu erklären, erweitert er das Modell zum Lösen von Buchstabenreihen von Simon und Kotovsky. Im Gegensatz zu Buchstabenreihen, besitzen Zahlenreihen allerdings eine größere Menge an möglichen Relationen, die man auf die Elemente anwenden kann. Dabei wird der Prozess in vier Phasen aufgeteilt (siehe Abbildung 2):

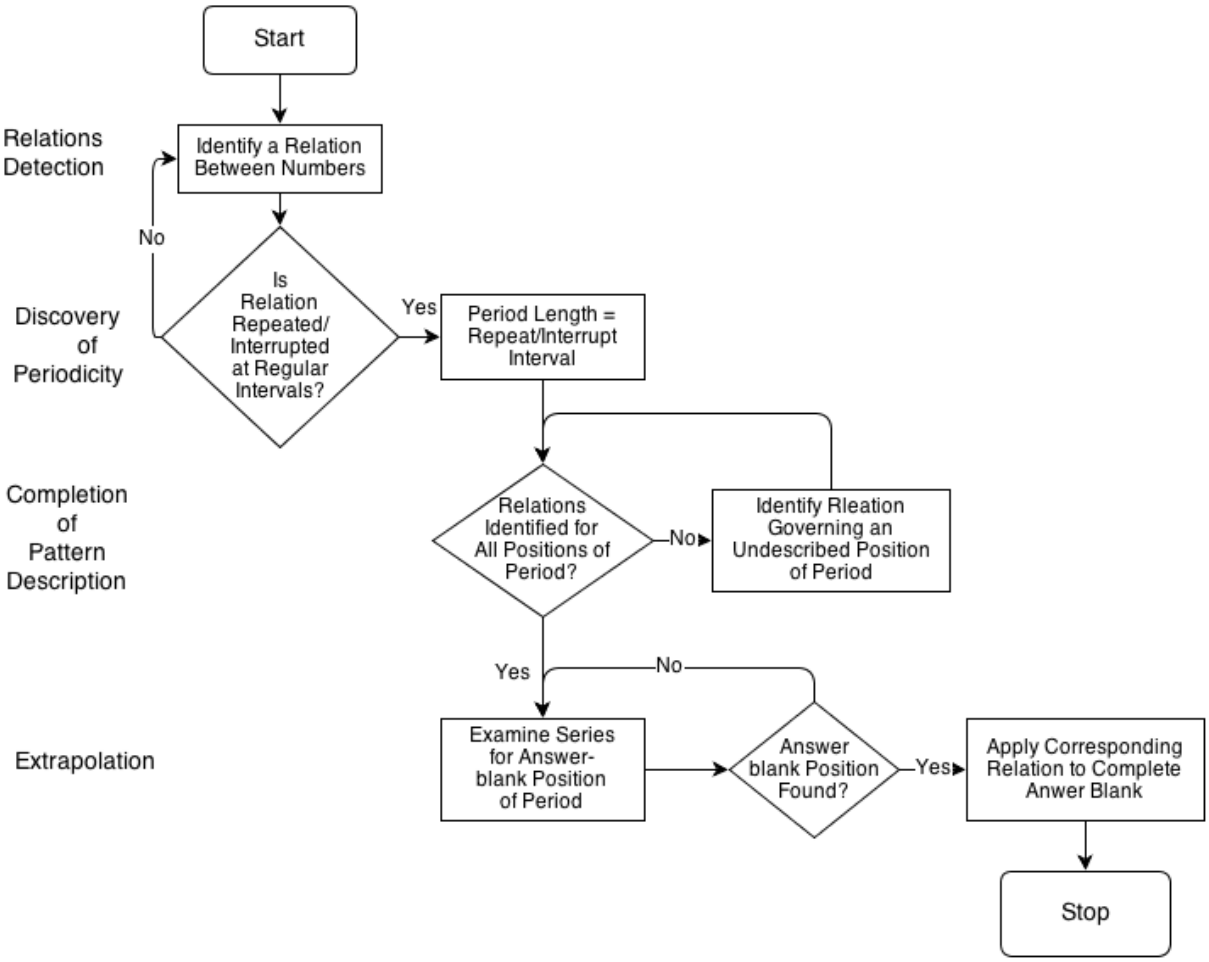



Abbildung 2: Flussdiagramm für das Lösen von Zahlenreihen

**Erkennen von Relationen** (Relations Detection):

Die Testperson sucht die Zahlenreihe ab und stellt eine erste Hypothese auf, welche Relation von einem Element zum anderen besteht. Anders als bei Buchstabenreihen, bei denen die möglichen Relationen sehr limitiert sind, kann es sich hier um

jede Form einer arithmetischen Operation handeln. Vorstellbar wären zum Beispiel Addition, Subtraktion, Multiplikation, Division oder das Anwenden einer Potenzfunktion. Zusätzlich können diese Operatoren auch in der Größe der arithmetischen Operation variieren (zum Beispiel: +1 oder +5). Des Weiteren ist es möglich, dass Zahlenreihen auch komplexere und hierarchische Relationen aufweisen. Die Sequenz 1, 2, 4, 7, 11 zum Beispiel, beinhaltet eine Additionsoperation, bei welcher aber der Summand wiederholt um 1 anwächst.

**Ermittlung der Periodenlänge** (Discovery of Periodicity):

Die Periodenlänge einer Zahlenreihe ist die Anzahl der Elemente, die einen kompletten Durchlauf der gefundenen Relationen darstellen. So ist die Periodenlänge bei einer konstanten Addition (zum Beispiel: 1, 3, 5, 7, 9) eins, weil der Summand +2 nach jedem einzelnen Element unverändert angewendet wird. Im Gegensatz dazu wäre bei der Zahlenreihe: 1,1,2,2,3,3 die Periodenlänge zwei, weil der komplette Durchlauf an Relationen (in diesem Fall: +0, +1) nach zwei Elementen wiederholt wird. Bei der Ermittlung der Periodenlänge unterscheidet Holzman zwei Methoden. Im ersten Fall wird in den gefundenen Relationen von benachbarten Elementen nach einer regelmäßigen Unterbrechung gesucht. Die Anzahl an Elementen, die zwischen diesen regelmäßigen Unterbrechungen liegen, bestimmt dann die Periodenlänge. So könnte man beim vorherigen Beispiel 1,1,2,2,3,3 feststellen, dass die (+ 0)- oder Identitäts-Relation regelmäßig nach einem Intervall von zwei Zahlen unterbrochen wird. Bei der zweiten Methode, mit der Relationen von nicht benachbarten Elementen entdeckt werden können, wird, im Gegensatz dazu, darauf geachtet, nach wie vielen Elementen sich eine gefundene Relation wiederholt. So könnte die Periodenlänge bei der Zahlenreihe 1,5,2,5,3,5 dadurch bestimmt werden, indem man erkennt, dass sich die (+1)- Relation nach jeweils zwei Zahlen wiederholt. Falls die gefundene Relation weder in regelmäßigen Abständen unterbrochen wird, noch sich wiederholt, so muss nach einer neuen Relation gesucht werden, die diese Bedingung erfüllt.

**Vervollständigung der Abbildungsrege** (Completion of Pattern Descritption):

Nachdem die Periodenlänge ermittelt wurde, muss die Testperson die vollständige Regel der Zahlenreihe konstruieren, indem sie die Relationen aller übrigen Positionen innerhalb der Periode identifiziert. Bei dem Beispiel: 2,4,8,5,7,14, 11,13,26,23 müsste man, nachdem man eine Periodenlänge von drei festgestellt hat, weil sich die (+2)- Additions-Relation nach jeweils drei Zahlen wiederholt, auch die zwei übrigen Relationen innerhalb der Periode feststellen, bevor man die Abbildungsregel vervollständigt hat (in diesem Fall: +2, *2, -3).

**Weiterführung** (Extrapolation):

Die vervollständigte Regel wird nun dazu verwendet, die Zahlenreihe fortzusetzen. Dabei muss die Position innerhalb der Periode, die die Lösungsstelle einnimmt, ermittelt werden. Danach wird der Teil der Abbildungsregel, der diese Position bestimmt, isoliert und anschließend angewendet, um die Zahl der Lösungsstelle zu generieren. Beim vorherigen Beispiel: 2,4,8,5,7,14,11,13,26,23 wäre die Position innerhalb der Periode, die die Lösungsstelle einnimmt, zwei und der Teil der Abbildungsregel, der isoliert werden muss, somit +2. Dadurch kann die Lösung 25 (23 + 2) generiert werden.

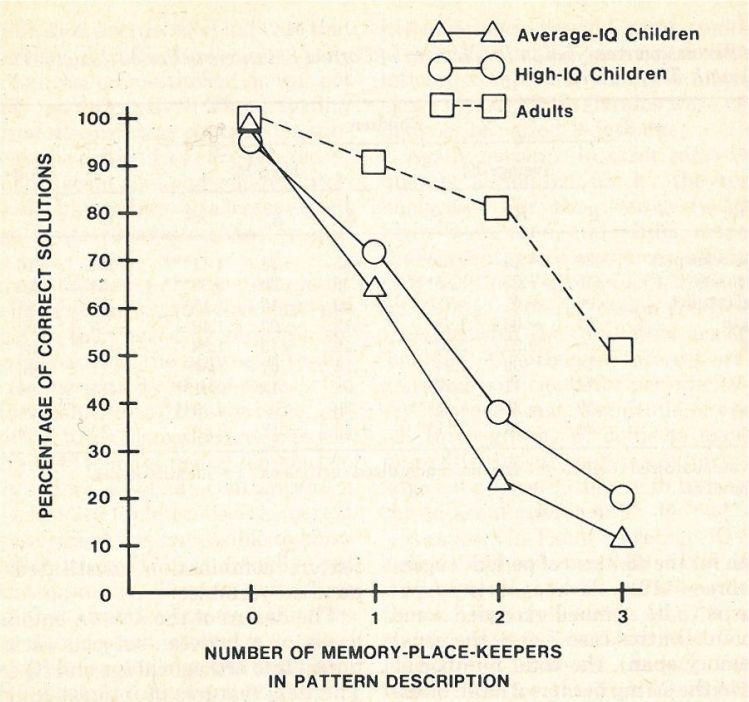



Abbildung 3: Einfluss der Gedächtnis-Platzhalter auf das Lösen von Zahlenreihen [6]

### 2.2.2 Schwierigkeitsfaktoren von Zahlenreihen

Welche Faktoren Einfluss auf die Lösbarkeit einer Zahlenreihe haben, wurde von Holzman in zwei Studien [5, 6] untersucht, in denen er mit drei Testgruppen arbeitete, die sich aus Studenten, überdurchschnittlich intelligenten Schülern (IQ circa 130) und durchschnittlich intelligenten Schülern (IQ circa 100) zusammensetzte. Aspekte, die untersucht wurden, waren unter anderem die Anzahl der Operationen, Mehrdeutigkeit von Relationen, die Größe der Operatoren, der Typ der arithmetischen Operation, die Periodenlänge und die relationale Komplexität.

Den größten Einfluss auf den Erfolg bei der Lösung einer Zahlenreihe hatte die Anzahl der Operationen, die nötig waren, um die Reihe zu erzeugen (siehe Abbildung 3). Dieser Faktor steht in direkter Verbindung mit der Informationsmenge, die die Testperson im Gedächtnis behalten muss, während sie sich mit der Auflösung der Zahlenreihe beschäftigt. Einen großen Unterschied sieht man hier zwischen den Erwachsenen und den Kindern. In der Kategorie der schweren Zahlenreihen, bei denen zwei oder drei Gedächtnis-Platzhalter benötigt waren, um die Lösungsregel zu berechnen, wurde eine sehr große Diskrepanz zwischen Kindern und Erwachsenen festgestellt. Da keine Interaktionen zwischen der Anzahl der Operationen und dem IQ-Niveau der Kinder beobachtet wurden, wird dieses Phänomen damit erklärt, dass entweder die Verwaltungskapazität von Informationsmengen mit dem Alter wächst oder dass Erwachsene bessere Strategien entwickelt haben, um diesen Speicherplatz effektiver zu nutzen.

Wie erwartet, hatte auch die Größe der arithmetischen Operation deutlichen Einfluss auf die Lösbarkeit. So wurden Zahlenreihen, bei denen die Operatoren hohe Beträge aufwiesen nur in 61% der Fälle gelöst, während Zahlenreihen mit niedrigen Werten in 81%

der Fälle gelöst wurden. Obwohl alle Testgruppen von diesem Faktor beeinflusst wurden, wurde auch hier ein signifikanter Unterschied zwischen Erwachsenen und Kindern festgestellt, da die Größe der Operatoren bei Erwachsenen deutlich weniger Einfluss ausübte.

Die Periodenlänge wurde dadurch untersucht, dass Zahlenreihen mit der Periodenlänge zwei und drei verglichen wurden. Dabei wurde nochmal unterschieden, ob sich die Periodenlänge mit Hilfe der ersten Methode, mit der Relationen von benachbarten Elementen untersucht werden, ermittelt werden konnten oder ausschließlich mit der zweiten Methode für nicht benachbarte Zahlen. Dabei war auffällig, dass Zahlenreihen, die mit der „benachbarten“ Variante bearbeitet werden konnten, deutlich besser gelöst wurden als Sequenzen, die nur mit der „nicht-benachbarten“ Variante bearbeitet werden mussten (84% gegen 59%).

Bei der Untersuchung des Operationstyps wurden die vier Grundrechenarten Addition, Subtraktion, Multiplikation und Division unterschieden. Dabei wurde ersichtlich, dass Additions- und Subtraktionsreihen eine bessere Gesamtlösungsrate als Multiplikations- und Divisionsreihen hatten. Zusätzlich wurde festgestellt, dass im Vergleich zu den durchschnittlich intelligenten Kindern, der Operationstyp bei den überdurchschnittlich intelligenten Kinder und vor allem bei den Erwachsenen weniger Einfluss auf die Leistungen hatte. Dieses Phänomen, das auch bei der Größe der arithmetischen Operation aufgetreten ist, wird dadurch erklärt, dass diese Testgruppen bessere mathematische Fähigkeiten besitzen.

Die Einflussnahme der relationalen Komplexität wurde dadurch untersucht, dass Zahlenreihen, die hierarchische Relationen beinhalteten, anderen Zahlenreihen mit zwei Gedächtnis-Platzhaltern gegenübergestellt wurden, welche man mit einfachen Relationen auflösen konnte. Während Erwachsene von der relationalen Komplexität unbeeinflusst blieben, ließ sich bei den Kindern ein signifikanter Unterschied feststellen (27% gegen 44% korrekte Lösungen). Es scheint, dass Kinder deutlich mehr Schwierigkeiten damit haben, neue oder ungewöhnliche Konzepte zu erkennen, beziehungsweise zu formulieren, obwohl sie mit ihren grundlegenden Bestandteilen vertraut sind.

Die Tatsache, dass die Gedächtnis-Platzhalter den größten Einfluss auf die Leistungen der Testpersonen hatte, stimmt auch mit der Arbeit von Kotovsky und Simon überein. Sie kamen zu dem Ergebnis, dass auch bei Buchstabenreihen die Überladung des Arbeitsgedächtnisses die Hauptquelle für Fehler ist. Zusätzlich kam Holzman zu dem Schluss, dass nicht nur Faktoren, die den Verarbeitungsprozess betreffen, sondern auch Faktoren, die sich auf das inhaltliche Wissen beziehen, Einfluss auf die korrekte Induktion von Regeln haben. Vor allem bei der Untersuchung des Operationstyps und der Größe des Operators war ersichtlich, dass Menschen, die geübter im Umgang mit arithmetischen Berechnungen waren, bestimmte Zahlenreihen leichter lösen konnten.

### 2.2.3 Eindeutigkeit von Zahlenreihen

Bei den meisten Zahlenreihen ist es der Fall, dass nachfolgende Zahlen in einer bestimmten Beziehung zu ihren Vorgängern stehen. Dabei gibt es verschiedene Methoden eine Zahlenreihe zu beschreiben [8].

Beispiel: 1, 3, 5, 7, 11, ...
Diese arithmetische Sequenz kann dadurch beschrieben werden, dass man die erste Zahl angibt (hier: $a_1 = 1$) und zusätzlich den generellen Term:

$$a_i = a_1 + (i-1) * d, i = 2,3,4,\ldots$$

Wobei „d“ hier der konstante Abstand ist (d = 2). Eine andere Möglichkeit wäre es eine rekursive Gleichung anzugeben:

$$a_i = a_{i-1} + d, i = 2,3,4,\ldots$$

Es ist natürlich auch möglich, dass eine nachfolgende Zahl gleichzeitig zu mehreren Vorgängern in Beziehung steht. Dies ist bei der bekannten Fibonacci-Reihenfolge der Fall, bei der jeweils die Addition der zwei Vorgängerzahlen den Nachfolger bestimmt: 1, 1, 2, 3, 5, 8, 13, 21 ... Diese Sequenz könnte mit folgender rekursiver Beschreibung charakterisiert werden:

$$a_1 = 1, a_2 = 1, a_i = a_{i-1} + a_{i-2}, \quad i = 3,4,5$$

Bei der Verwendung von Zahlenreihen in Intelligenztests kam nun die Kritik auf, dass eine Bewertung anhand der richtigen Lösung und der benötigten Zeit, wie es in fast allen Tests üblich ist, nicht aussagekräftig ist. Denn die unterschiedlichen Konstruktionsregeln könnten alle sowohl unterschiedlich viel Bearbeitungszeit, als auch unterschiedlich viel kognitive Leistung erfordern [8]. Allerdings lösen Menschen Zahlenreihenprobleme nicht, indem sie eine vollständige mathematische Formel aufstellen und diese gegen die Problemstellung testen. Da nur nach einer Fortsetzung der Zahlenreihe gefragt ist, würden die Testpersonen normalerweise im ersten Beispiel mit der arithmetischen Folge einfach auf eine Addition von 2 schließen und dementsprechend die Sequenz fortsetzen. Mit einer formalen Repräsentation der Reihe würden sie sich nicht beschäftigen. Auch wenn es Unterschiede in verschiedenen Lösungsansätzen gibt, so ist das sicher kein spezielles Problem von Zahlenreihen. Dass es mehrere und unterschiedlich zeitaufwendige Wege gibt, die zum richtigen Ergebnis führen, betrifft sicher einen großen Anteil an Problemstellungen und es besteht kein Grund einen effizienteren Weg mit einer kürzeren Bearbeitungszeit nicht dementsprechend zu bewerten.

Von der formalen Beschreibung einer Zahlenreihe mit ihren festen Positionen wird auch im späteren Lösungsmodell abstrahiert. Es werden lediglich die Berechnungen angegeben, die auch vom Menschen zur Lösung einer Reihenfolge nötig sind. So würde das Beispiel 1, 3, 5, 7, 11, ... ausreichend mit „konstante Addition von 2“ beschrieben werden. Auch komplexere Beispiele, die hierarchische Relationen beinhalten, können so beschrieben werden.

Beispiel: 1, 2, 4, 7, 11, 16, ...

In diesem Beispiel beinhaltet die Additionsfolge eine zweite Additionsfolge (+1). Solche Problemstellungen können folgendermaßen beschrieben werden:

Berechnung der Abstände -> konstante Addition von 1
(1, 2, 3, 4, 5) -> (+1)

Somit kommt man zu dem Ergebnis, dass der nächste Abstand „6“ sein muss und die Lösung der Zahlenreihe dementsprechend „22“ ist.

Ein zweiter wichtiger Faktor in Bezug auf die Verwendung von Zahlenreihen in Intelligenztests ist natürlich die Eindeutigkeit der Lösung. In diesem Fall ist nicht die eindeutige, formale Beschreibung des Problems gemeint, sondern die Lösungszahl selbst. Diese sollte natürlich in einem Test, in dem man von einer richtigen Lösung ausgeht und alle anderen Möglichkeiten als falsch bewertet, gegeben sein. Man behilft sich damit, dass man die Testpersonen dazu auffordert, die offensichtlichste Lösungsstrategie anzuwenden. Dies ist nicht ungewöhnlich, denn auch bei Aufgaben die das verbale Verständnis prüfen, wie z.B. bei einer Wortmenge einzelne Begriffe auszusortieren, die logisch nicht dazu passen, sind meist mehrere Zusammenhänge möglich. Doch werden die Beispiele so gewählt, dass ein offensichtliches Prinzip heraussticht, das man in diesem Fall anwenden muss.

Damit es nicht mehrere plausible Lösungen gibt, reicht es im Normalfall die Zahlenreihe ausreichend lang zu gestalten. Claes Strannegard führte ein Experiment durch, bei dem er eine Gruppe von Studenten dazu aufforderte die Zahlenreihe 1, 2 zu vervollständigen [12]. Aufgrund der geringen Anzahl von Elementen ist diese Reihe natürlich höchst mehrdeutig. Die offensichtlichsten Antworten waren eine konstante Addition von 1 (nächsten Zahlen: 3, 4) und eine konstante Multiplikation mit 2 (nächsten Zahlen: 4, 8). Einige wenige Studenten kamen auch auf die alternativen Lösungen einer wiederholenden Sequenz (nächsten Zahlen: 1, 2) und einer gespiegelten Sequenz (nächsten Zahlen: 2, 1). Wie auch in den meisten anderen Fällen, muss man in diesem Beispiel die Länge der Zahlenreihe nur erhöhen, um ein einziges, offensichtliches Lösungsprinzip zu garantieren.

In dieser Arbeit wird der Standpunkt vertreten, dass eine Zahlenreihe mindestens fünf Elemente lang sein muss, um unnötige Doppeldeutigkeiten zu vermeiden. Allerdings ist dies nur der Minimalfall bei einer Periodenlänge von 1. Wenn man die Periodenlänge erhöht, so führt es dazu, dass es mehrere Reihen von Relationen gibt, die unabhängig voneinander aufgelöst werden müssen. Wenn es sich um eine Zahlenreihe mit 5 Elementen und einer Periodenlänge von 1 handelt, so muss man, in dem gebräuchlichsten Fall, dass sich eine nachfolgende Zahl ausschließlich aus einem Vorgänger berechnen lässt, eine Reihe von 4 Relationen fortsetzen. Erhöht man nun die Periodenlänge aber auf 2, so handelt es sich um zwei von einander unabhängige Relationsreihen mit jeweils zwei Elementen, die man fortführen muss. Dies führt wie im vorher angeführten Beispiel (1, 2) zu Ambiguitäten.

Beispiel: 1, 2, 2, 4, 8, ...

Hier handelt es sich um eine Reihe mit der Periodenlänge 2, die folgendermaßen fortgesetzt werden soll: +1, *1, +2, *2, +3, *3, +4, *4. Somit wären die nächsten zwei Zahlen 11 und 33. Dies ist jedoch keine eindeutige Lösung, da man die Reihe genauso mit +1, *1, +2, *2, +4, *4, +8, *8 auflösen könnte (nächsten zwei Zahlen: 12, 48). Es ist hier auch nicht der Fall, dass die erste Möglichkeit gegenüber der zweiten als offensichtliches Berechnungsprinzip heraussticht, wie es zuvor gefordert wurde. Deswegen muss die Länge der Zahlenreihe weiter erhöht werden, falls es sich um Sequenzen mit einer erhöhten Periodenlänge handelt. In der praktischen Erstellung von Testreihen sollte die Länge

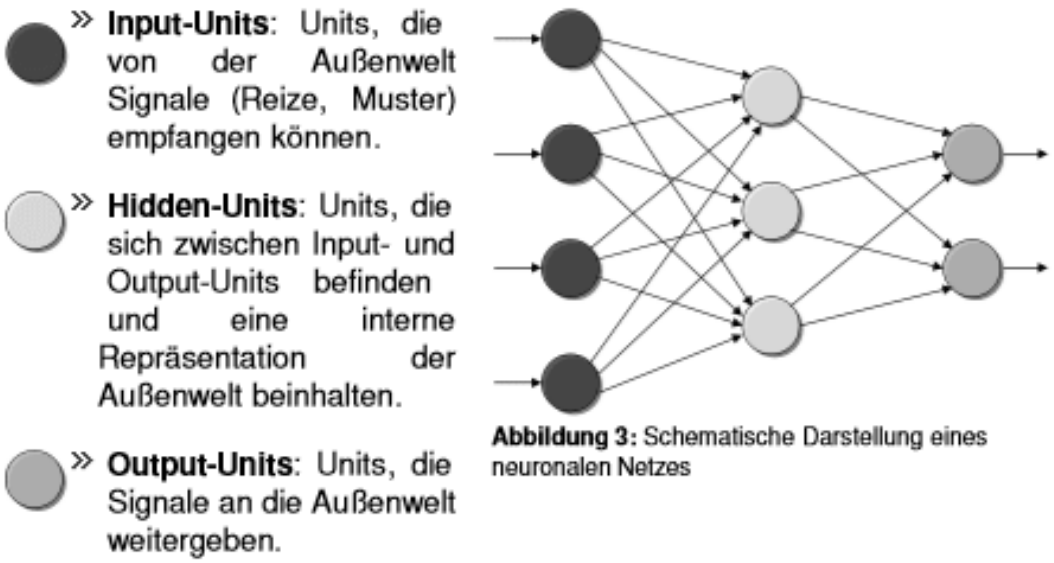



Abbildung 4: Schematische Darstellung eines neuronalen Netzes [15]

der Zahlenreihe natürlich nicht individuell angepasst werden, da Testpersonen sonst die Größe der Sequenz als zusätzlichen Hinweis auf die möglichen Anwendungsregeln missbrauchen könnten. Die Länge der Beispiele sollte vielmehr einheitlich so gewählt werden, dass die Anzahl der Elemente für jede Reihe eine ausreichende Eindeutigkeit garantiert.

## 2.3 Ansätze zur Induktion von Zahlenreihen

Die automatisierte Induktion von Daten ist schon lange ein Forschungsthema im Bereich der Informatik. Auch für die Induktion von Zahlenreihen im Speziellen wurden bereits mehrere Lösungssysteme entwickelt. Diese verschiedenen Ansätze sollen im Folgenden vorgestellt werden.

### 2.3.1 Künstliche neuronale Netze

Marco Ragni und Andreas Klein entwickelten ein System, das, basierend auf künstlichen, neuronalen Netzen und einem dynamischen Lernansatz, Zahlenreihen lösen sollte [10]. Künstliche, neuronale Netze versuchen herausragende Merkmale aus der Biologie, die klassischen Computersystemen nicht zu eigen sind, zu adaptieren. Diese sind Lernfähigkeit, Fehlertoleranz und eine Generalisierungs- bzw. Assoziationsfähigkeit, mit Hilfe der ein Neuronales Netz nach erfolgreichem Training Problemen derselben Klasse, die nicht explizit trainiert wurden, treffende Lösungen zuführen kann [9]. Neuronale Netze bestehen aus mehreren Neuronen, welche in diesem Fall auch Units oder Knoten genannt werden. Diese dienen dazu, Informationen aus der Umwelt oder anderen Knoten aufzunehmen und in modifizierter Form weiterzuleiten [15]

Eine Gruppe von Knoten, die sich auf einer vertikalen Ebene befindet, fasst man als Schicht bzw. Layer zusammen (siehe Abbildung 4). Die Kanten zwischen den Knoten werden mit einem Gewicht versehen, das die Stärke der Verbindung darstellt. Je größer der Absolutbetrag dieses Gewichts ist, desto mehr Einfluss hat ein Knoten auf einen anderen. Diese Gewichte können sowohl positiv oder negativ sein, als auch Null betragen.

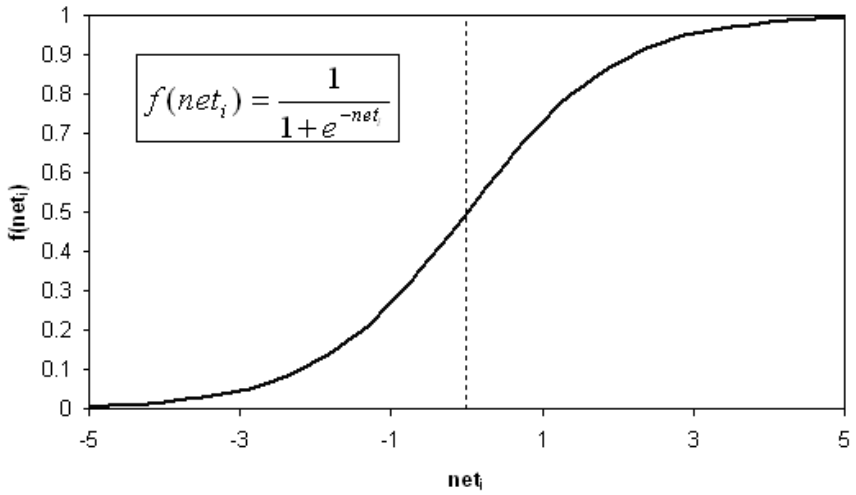



Abbildung 5: sigmoide Aktivierungsfunktion [14]

Im Normalfall wird der Lernprozess eines neuronalen Netzes durch eine Gewichtsveränderung der Verbindungen definiert.

Der Input eines Knotens beinhaltet gewöhnlich die Multiplikation des Outputs des sendenden Knotens mit dem zugehörigen Gewicht. Der gesamte Input eines Knotens, der sich normalerweise additiv aus den einzelnen Inputs zusammensetzt, wird Netzinput genannt. Den Zusammenhang zwischen dem Netzinput und dem Aktivitätslevel eines Knotens, stellt die Aktivitäts- oder Aktivierungsfunktion dar. Anschließend wird der Aktivitätslevel mit Hilfe einer Ausgabefunktion in den Output transformiert, den der Knoten anschließend selbst weitersendet. Eine mögliche Aktivitätsfunktion wäre zum Beispiel die sigmoide Funktion (siehe Abbildung 5)

Diese Funktion wird bei den meisten Modellen verwendet, die kognitive Prozesse simulieren. Der Aktivitätslevel ist hierbei von 0 (-1 bei Tangens-Hyperbolicus-Funktion), bei einem großen negativen Netzinput, bis 1, bei einem hohen positiven Netzinput, begrenzt.

Für ihren Versuch, Zahlenreihen mit Hilfe von künstlichen, neuronalen Netzen zu lösen, verwendeten Ragni und Klein ein dreischichtiges Netzwerk mit Backpropagation und dem Tangens Hyperbolicus als Aktivitätsfunktion. Bei der Backpropagation handelt es sich um ein Lernverfahren, bei dem man jede Modifikation der Gewichte in drei Schritte unterteilt:

1. Forward-Pass: Wie in der Trainings- und Testphase üblich, werden den Input-Knoten Reize präsentiert und der Output des neuronalen Netzes berechnet.

2. Fehlerbestimmung: Die gewünschten Output-Werte werden mit den im Forwardpass tatsächlich ermittelten Werten verglichen und somit wird die Fehlerbestimmung für die Output-Units erstellt. Falls die Fehler einen gewissen Schwellenwert überschreiten, erfolgt der dritte Schritt.

3. Backward-Pass: Beim letzten und wichtigsten Schritt werden die Fehlerterme nun

in entgegengesetzter Richtung betrachtet, so dass sie sich zur Input-Schicht hin ausbreiten. Mit Hilfe dieser Fehlerterme werden die Gewichte des Netzes, startend bei den Gewichten zwischen Ouput- und letzter Hidden-Schicht, so angepasst, dass die Fehlerterme kleiner werden.

In ihrem Modell verwendeten Ragni und Klein $f_i = \frac{n}{10^{len(n)}}$ als Input-Funktion für ihr Netz und $f_0 = f_i^{-1}$ als Output-Funktion. Den Gewichten wurden zunächst zufällige Werte zwischen 0 und 1 zugewiesen und es wurde ein Momentum-Faktor von 0,1 verwendet. Die Lernrate, die Anzahl der Input-Knoten, die Anzahl der verdeckten Knoten (hidden units) und die Anzahl der Trainingsdurchläufe wurden systematisch variiert, aber für alle Variationen wurde nur ein Output-Knoten verwendet.

Um Trainingsmuster für das Testen der verschiedenen Netze zu generieren, wurden Tupel von Trainingswerten und ein Zielwert erstellt. Die Anzahl der Trainingswerte entsprach der Anzahl der Input-Knoten, die ein Netz benutzte. Dabei wurden zunächst die ersten m (m = Anzahl der Input-Knoten) Elemente der Zahlenreihe als Input-Werte benutzt und das darauffolgende Element als Zielwert. Anschließend wurden diese Zuweisungen um jeweils eine Stelle der Zahlenreihe nach rechts verschoben. Demnach wurden für jede Zahlenreihe genau n – m – 1 (n = Länge der Zahlenreihe) Trainingsmuster erstellt, wobei die letzte Zahl einer Sequenz immer nur zum Testen und nicht für das Training benutzt wurde (siehe Abbildung 6)

| Pattern | $n_1$ | $n_2$ | $n_3$ | $n_4$ | $n_5$ | $n_6$ | $n_7$ | $n_8$ |
|---|---|---|---|---|---|---|---|---|
| $p_1$ | $v_1$ | $v_2$ | $v_3$ | $t$ | | | | |
| $p_2$ | | $v_2$ | $v_3$ | $v_4$ | $t$ | | | |
| $p_3$ | | | $v_3$ | $v_4$ | $v_5$ | $t$ | | |
| $p_4$ | | | | $v_4$ | $v_5$ | $v_6$ | $t$ | |
| $p_5$ | | | | | $v_5$ | $v_6$ | $v_7$ | $?$ |

Abbildung 6: Erstellung der Trainingsmuster am Beispiel einer 8-stelligen Zahlenreihe [10]

Um das Modell im Vergleich zu menschlichen Leistungen zu testen, wurde ein Experiment mit 17 Teilnehmern und 20 verschiedenen Zahlenreihen durchgeführt. Für die Analyse der empirischen Daten wurde dabei die Lernrate zwischen 0,125 und 0,875 (in 0,125-Schritten), die Anzahl der Input-Knoten von 1 bis 6, die Anzahl der verdeckten Knoten von 1 bis 20 und die Anzahl der Trainingsiterationen von 500 bis 10000 (vier Stufen: 500, 1000, 5000, 10000) variiert. Drei der 20 Zahlenreihen konnten von keiner Konfiguration des neuronalen Netzes gelöst werden. Alle anderen konnten zumindest von bestimmten Konfigurationen gelöst werden (siehe Abbildung 7). C, I und U sind die Ergebnisse der Testpersonen, die für die Anzahl der korrekten Antworten (C), die Anzahl der inkorrekten Antworten (I) und die Anzahl der Testpersonen, die auf keine Lösung gekommen sind (U), stehen.

Ragni und Klein zeigten auf jeden Fall, dass es mit künstlichen, neuronalen Netzen prinzipiell möglich ist, Zahlenreihen zu lösen. Welche der verwendeten Konfigurationen am

| Number Series | C. | I. | U. | Number of solving configurations with iterations: 0.5k; 1k; 5k; 10k | Number Series | C. | I. | U. | Number of solving configurations with iterations: 0.5k; 1k; 5k; 10k |
|---|---|---|---|---|---|---|---|---|---|
| 12,15,8,11,4 | 15 | | 2 | 306; 385; 475; 530 | 4,11,15,26,41 | 8 | 1 | 8 | 6; 14; 32; 22 |
| 148,84,52,36,28 | 12 | 2 | 3 | 555; 637; 670; 689 | 5,6,7,8,10 | 10 | 1 | 6 | 83; 91; 65; 114 |
| 2,12,21,29,36 | 14 | 1 | 2 | 405; 440; 502; 539 | 54,48,42,36,30 | 16 | 1 | | 274; 299; 338; 376 |
| 2,3,5,9,17 | 13 | 1 | 3 | 3; 7; 61; 192 | 6,8,5,7,4 | 16 | | 1 | 134; 169; 198; 219 |
| 2,5,8,11,14 | 9 | 3 | 5 | 581; 618; 659; 667 | 6,9,18,21,42 | 14 | 1 | 2 | 48; 24; 94; 101 |
| 2,5,9,19,37 | 6 | 4 | 7 | 0; 0; 0; 0 | 7,10,9,12,11 | 14 | | 3 | 111; 202; 380; 404 |
| 25,22,19,16,13 | 16 | | 1 | 562; 615; 648; 667 | 8,10,14,18,26 | 13 | 1 | 3 | 57; 46; 30; 29 |
| 28,33,31,36,34 | 17 | | | 121; 183; 315; 332 | 8,12,10,16,12 | 17 | | | 37; 75; 41; 51 |
| 3,6,12,24,48 | 13 | 1 | 3 | 0; 0; 0; 0 | 8,12,16,20,24 | 15 | | 2 | 507; 546; 594; 613 |
| 3,7,15,31,63 | 12 | 3 | 2 | 0; 0; 0; 0 | 9,20,6,17,3 | 16 | | 1 | 255; 305; 397; 406 |

Abbildung 7: Ergebnisse der empirischen Analyse (Neuronales Netz) [10]

besten geeignet waren, ist schwer zu sagen, da sie sich alle in Bezug auf die verschieden hohe Iterationsanzahl und den verschiedenen Zahlenreihen in ihrer Performance unterschieden. Auffällig ist aber, dass eine relativ trivial aussehende Zahlenreihe (3, 6, 12, 24, 48), von keiner einzigen Konfiguration gelöst werden konnte, obwohl nur eine konstante Multiplikation mit 2 vonnöten gewesen wäre. Auch wird mit diesem Model nur versucht, die Zahlenreihe logisch mit ihrem Nachfolger fortzusetzen. Eine grundlegende, mathematische Funktion oder Lösungsschritte werden nicht ermittelt.

### 2.3.2 E-Generalisierung

Jochen Burghardt hat einen Ansatz [2] entwickelt, bei dem mit Hilfe von E-Anti-Unifikation Zahlenreihen analysiert und logisch fortgesetzt werden können. Dabei wird die Auffassung von Anti-Unifikation um Gleichungstheorien erweitert. Mit diesen sollen die Probleme, die mit wechselnden Repräsentationen einhergehen, bewältigt werden. Anti-Unifikation ist als die Gegenoperation zur Unifikation definiert. Während die Unifikation die generellste Spezialisierung von gegebenen Termen berechnet, berechnet die Anti-Unifikation die speziellste Generalisierung. Durch das Hintergrundwissen in Form der Gleichungstheorien können nun alle möglichen, äquivalenten Repräsentationen gleichzeitig im Abstraktionsprozess berücksichtigt werden. In Abbildung 8 sind die Gleichungstheorien für (+) und (*) und die E-Generalisierung von 0 und $s^4(0)$ dargestellt.

So kann man zum Beispiel, wenn man die gewohnte 0-Repräsentation der natürlichen Zahlen benutzt und $s(s(0))$ zu $s^2(0)$ abkürzt, die Terme $0*0$ und $s^2(0)*s^2(0)$ zu $x*x$ anti-unifizieren. Durch die E-Anti-Unifikation kann man nun Übereinstimmungen finden, die nur feststellbar sind, wenn man Gleichungstheorien als Hintergrundwissen benutzt. Wenn man beispielsweise die Terme 0 und $s^4(0)$, rein syntaktisch und ohne Berücksichtigung von Gleichungstheorien, anti-unifizieren würde, würde man den Term $y$ erhalten, der darauf hinweist, dass es keine gemeinsame Struktur gibt. Wenn man aber die Gleichungen für (+) und (-) (siehe Abbildung 8) und die alternativen Repräsentationen, die mit ihnen einhergehen, miteinbezieht, dann erhält man in Folge der Anti-Unifizierung

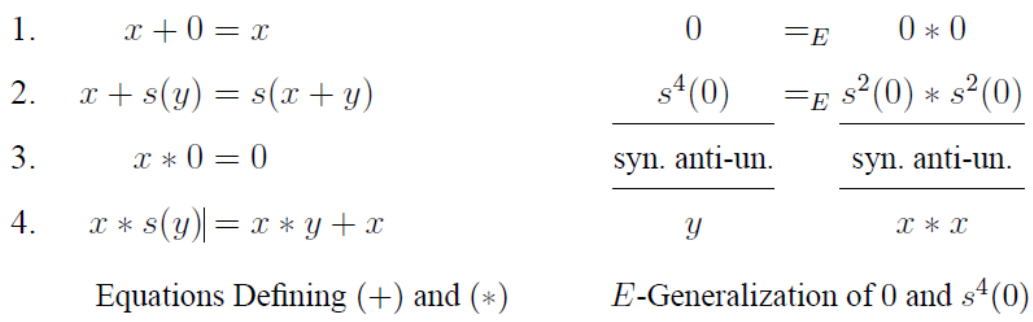



Abbildung 8: Gleichungstheroien und E-Generalisierung [2]

$x*x$ als mögliches Ergebnis. So könnte zum Beispiel das Muster einer Zahlreihe, die aus Quadratzahlen (0, 1, 4, 9, 16, ...) besteht, erkannt werden.

Die Ergebnisse einiger Zahlenreihen sind in der Abbildung 9 zusammengefasst. Die Spalte „Theory“ zeigt an, welche Gleichungstheorie benutzt wurde. Wobei die Funktion „if“ ein if-then-else umsetzt, mit den definierten Gleichungen $if(s(x), y, z) = y$ und $if(0, y, z) = z$, und die Funktion „ev“ $s(0)$ für gerade und 0 für ungerade Zahlen zurückgibt. In der Spalte „Series“ sind die getesteten Zahlenreihen aufgelistet. Die Anzahl der Elemente, die sich rechts des Strichpunkts befinden, entspricht der Anzahl an Suffixen, die übergeben wurden. Die berechneten Hypothesen müssen in der Lage sein, all diese Elemente zu erklären, nicht aber diejenigen vor dem Strichpunkt. Unter „Law“ stehen die berechneten Hypothesen, in denen $v_p$ den Platz innerhalb einer Zahlenreihe angibt (erstes Element: 0). $v_1$ und $v_2$ kennzeichnen das vorherige bzw. das vorletzte Element. Die Spalte „No“ gibt an, an welcher Stelle der Aufzählungssequenz die Hypothese aufgetreten ist und „Time“ schließlich die Laufzeit in Millisekunden.

Der Ansatz mit E-Generalisierung konzentriert sich weniger darauf, die Zahlenreihe logisch fortzusetzen, sondern vielmehr darauf, aus einer limitierten Anzahl von Operatoren heraus einen nicht rekursiven Algorithmus zu konstruieren, der die nächsten Elemente berechnen kann. Acht der neun Zahlenreihen wurden korrekt bearbeitet. Für die letzte Sequenz (1, 2, 2, 3, 3, 3, 4, 4 ,4 , 4), die von einem Menschen relativ einfach fortgesetzt werden könnte, konnte keine Hypothese berechnet werden. Auffällig ist auch, dass die Laufzeit für eine sehr simpel aussehende Zahlenreihe (0, 0, 1, 1, 0, 0, 1, 1) über einer Minute liegt. Insgesamt sind die Beispiele, zumeist nur mit simplen Wiederholungen und konstanten Additionen, sehr einfach gehalten. Inwieweit der Ansatz auch für komplexere und hierarchische Zahlenreihen Berechnungsformeln finden kann und wie das Hinzufügen von mehreren Gleichungstheorien die Laufzeit beeinflussen würde, ist nicht bekannt.

### 2.3.3 Anthropomorphe Methode

Die Methode von Cleas Strannegard kombiniert Elemente aus der künstlichen Intelligenz und der kognitiven Psychologie [12]. Sie wird als anthropomorph bezeichnet, weil sie Gebrauch von einem Modell des menschlichen Schlussfolgerns macht. Dabei nutzt das Modell kognitive Hilfsmittel, ein Reportoire an Mustern, die Zahlenreihen enkodieren, und eingeschränkte Berechnungen, um die Muster zu dekodieren. Die Lösungsstrategie

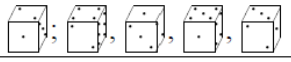
| Theory | Series | Law | No | Time |
|---|---|---|---|---|
| $+,*$ | $0;1,4,9$ | $v_p * v_p$ | 1. | 2797 |
| $+,*$ | $0;2,4,6$ | $s(s(v_1))$ | 1. | 3429 |
| $+,*$ | $0;2,4,6$ | $v_p + v_p$ | 3. | 3429 |
| $+,*$ | $1,1;2,3,5$ | $v_1 + v_2$ | 1. | 857 |
| $+,*,if,ev$ | $0,1;2,1,4,1$ | $if(ev(v_p), v_p, 1)$ | 13. | 13913 |
| $+,*,if,ev$ | $0,0,1;1,0,0,1,1$ | $ev(v_2)$ | 1. | 61571 |
| $+,*,if,ev$ | $0,0;1,0,0,1$ | $ev(v_1 + v_2)$ | 1. | 8573 |
| $+,*,if,ev$ | $0;1,3,7$ | $s(v_1 + v_1)$ | 1. | 3714 |
| $+,*,if,ev$ | $1,2,2,3,3,3,4;4,4,4$ | — | | 8143 |
| $cube,if,ev$ |  | $rg(if(ev(v_p), v_1, $  $))$ | 1. | 14713 |
| $cube,if,ev$ |  | $cl(if(ev(v_p), up(v_1), dn(v_1)))$ | 1. | 604234 |

Abbildung 9: Ausgegebene Berechnungsformeln [2]

basiert dabei auf der Beobachtung, dass Zahlenreihen, die in Intelligenztests vorkommen, von Menschen für Menschen entwickelt wurden. Dies führt zu folgenden Schlüssen:

- Es gibt unendlich viele Funktionen, die eine gegebene, begrenzte Zahlenreihe reproduzieren können
- Diese Funktionen können verschiedene Prognosen für die Lösung generieren
- Trotzdem wird von den Verfassern des Intelligenztests eine Funktion und somit auch eine Zahl als die richtige ausgewählt
- Deswegen kann ein kognitives Modell des Verfassers eines Intelligenztests einem Programm helfen, die richtige Auswahl zu treffen

Das kognitive Modell soll dabei nur den Zweck erfüllen, bei der Problemlösung zu helfen und nicht, wie in der kognitiven Psychologie, das menschliche Denken zu erklären. Es basiert auf der Arbeit von Newell und Simon über das menschliche Problemlösen, baiserend auf geistigen Zuständen und Übergängen zwischen diesen. Dabei modelliert es eine Kombination aus deduktivem und induktivem Schlussfolgern. Es beinhaltet eine limitierte Menge an Mustern, die dazu genutzt werden Zahlenreihen zu beschreiben und eine limitierte Menge an Berechnungen mit beschränkten, kognitiven Mitteln, die verwendet werden, um die Muster zu dekodieren und somit Zahlenreihen zu generieren.

Das kognitive Modell wird dabei als heuristische Grundlage benutzt, um viele Lösungsmöglichkeiten auszumustern, die zu viel Rechenleistung erfordern würden. Bestimmte Begrenzungen der menschlichen Erkenntnis, wie zum Beispiel die eingeschränkte Kapazität des Arbeitsgedächtnisses, werden sich so zu Nutzen gemacht. Das Modell basiert auf einen Vorrat an arithmetischen Fakten (zum Beispiel: 5 * 5 = 25) und algebraischen

$$
\begin{array}{l}
\dfrac{3 * 28}{} \text{ Expansion} \\
\dfrac{3 * (20 + 8)}{} \text{ Distribution} \\
\dfrac{3 * 20 + 3 * 8}{} \text{ Power multiplication} \\
\dfrac{60 + 3 * 8}{} \text{ Recall} \\
\dfrac{60 + 24}{} \text{ Expansion} \\
\dfrac{60 + (20 + 4)}{} \text{ Size matching} \\
\dfrac{(60 + 20) + 4}{} \text{ Power addition} \\
\dfrac{80 + 4}{} \text{ Compression} \\
84
\end{array}
$$

Abbildung 10: Beispiel einer eingeschränkten Berechnung eines Musters [12]

Gesetzen (zum Beispiel: a* (b + c) = a * b + b * c). Das Limit des „Arbeitsgedächtnisses“ wurde aufgrund einer vorbereitenden Untersuchung auf 8 gesetzt. Wenn das Limit zu niedrig wäre, wären relativ einfache Zahlenreihen nicht mehr lösbar und wenn es zu hoch wäre, würde das die Berechnungszeit drastisch erhöhen. Abbildung 10 zeigt ein Beispiel für eine eingeschränkte Berechnung für ein Muster. Die algebraischen Gesetze, die übergeben wurden, werden dazu verwendet, neue Ausdrücke zu generieren.

Äquivalent zu einem menschlichen Problemlöser, dem es misslingt ein bestimmtes Muster in der Sequenz s zu finden und anschließend versucht stattdessen Muster in Unter-Sequenzen von s zu finden, verhält sich das Programm. Daher werden Vereinfachungs-Funktionen benutzt um Zahlenreihen auf Unter-Sequenzen abzubilden:

- $even(a_0,...,a_{2_{m+1}}) = (a_0,a_2,...,a_{2m})$
  $even(1,44,1,27,1,92) = (1,1,1)$
- $odd(a_0,...,a_{2m}) = (a_1,a_3,...,a_{2m-1})$
  $odd(7,1,39,35,1,28) = (1,1,1)$
- $exception(a_0,...,a_k) = (a_0,a_{i-1},a_{i+1}...,a_k)$
  $exception_2(1,1,7,1) = (1,1,1)$
- $last_i(a_0,...,a_k) = (a_{k-1},...,a_k)$
  $last_3(46,147,9,1,1,1) = (1,1,1)$

Wenn zum Beispiel die Zahlenreihe (1, 44, 1, 27, 1, 92) übergeben wird, findet Asolver zunächst kein anthropomorphes Muster für die komplette Sequenz. Deswegen wird in den Vereinfachungsmodus gewechselt und die Zahlenreihe wird zu even(1, 44, 1, 27, 1, 92) = (1, 1, 1) geändert. Für diese Reihe wird das konstante Muster f(n) = 1 gefunden und für die nächste Zahl wird 1 prognostiziert.

Asolver wurde gegen andere Programme, die fähig sind Zahlenreihen zu lösen, getestet. Diese waren die Software Mahematica (mit der FindSequenceFunction), die Software Maple, die Online-Suchmaschine Wolfram Alpha und die Online Encyclopedia of Integer Series (OEIS). Getestet wurden die verschiedenen Programme mit 11 Zahlenreihen aus

dem IQ-Test PJP (Sjöberg et al., 2006). Die Tabelle 11 fasst die Ergebnisse zusammen, wobei „C“ für die korrekte, „I“ für die inkorrekte und „N“ für gar keine Fortsetzung der Reihe steht. Die Programme hatten für eine Sequenz maximal fünf Minuten Zeit.

| | Asolver | Mathematica | Maple | OEIS | WA |
|---|---|---|---|---|---|
| PJP 19 | C | C | C | C | C |
| PJP 20 | C | N | C | C | N |
| PJP 21 | C | C | C | N | C |
| PJP 22 | C | C | I | N | C |
| PJP 23 | C | N | I | I | N |
| PJP 24 | C | C | C | C | C |
| PJP 25 | C | N | I | C | N |
| PJP 26 | C | C | I | N | C |
| PJP 27 | C | C | C | N | C |
| PJP 28 | C | N | C | N | N |
| PJP 29 | C | N | I | N | N |

Abbildung 11: Ergebnisse der verschiedenen Programme [12]

Alle Programme bis auf Asolver lösten maximal sechs der getesteten Zahlenreihen und waren somit schlechter als der durchschnittliche Mensch. Asolver konnte alle Zahlenreihen lösen und erzielte somit einen umgerechneten IQ von mindestens 143. Auf einem Standard-Laptop benötigte Asolver für die 11 Problemstellungen durchschnittlich 60 Sekunden. Wenn Asolver mehrere Muster findet, die eine Zahlenreihe beschreiben, dann wird der lexikographisch kleinste Term ausgewählt. Allerdings verblieben unter den getesteten Zahlenreihen keine mehrdeutigen Sequenzen, nachdem die nicht-antropomorphen Muster aussortiert wurden.

Die Ergebnisse von Asolver sehen sehr vielversprechend aus, allerdings wurde der Algorithmus im Gegensatz zu den Vergleichsprogrammen darauf optimiert, speziell Zahlenreihen aus Intelligenztests zu lösen und auch nur gegen solche getestet. Weiterhin war die Anzahl der Testprobleme sehr limitiert, was die Aussagekraft des Experiments erheblich schmälert.

# 3 Ein regelbasierter Ansatz

In diesem Kapitel wird ein selbst entwickeltes, regelbasiertes System zum Lösen von Zahlenreihen vorgestellt. Im Gegensatz zu anderen Programmen, werden keine mathematischen Formeln herangezogen, um den Aufbau einer Zahlenreihe zu beschreiben, sondern es werden die Schritte nachvollzogen, die auch vom Menschen zu deren Lösung vonnöten wären. Das vorgestellte Programm wurde in der funktionalen Programmiersprache Haskell implementiert. Der Grund hierfür war, dass man Funktionen in Haskell leicht als mathematische Objekte handhaben kann.

## 3.1 Konzept des regelbasierten Ansatzes

Der hier vorgestellte Ansatz zum Lösen von Zahlenreihen versucht, in Anlehnung an das Lösungsmodell von Holzman, ein relativ menschenähnliches Vorgehen anzuwenden. Dabei wird, wie in Kapitel 2.2.3 vorgestellt, von absoluten Positionen innerhalb der Zahlenreihe und von formalen Repräsentationsformeln abstrahiert. Vielmehr wird eine Lösung ausschließlich durch die Relationen zwischen den Elementen der Zahlenreihe ermittelt. Dabei macht sich das System zu Nutzen, dass Zahlenreihen, die durch eine mathematische Formel beschrieben werden, die die absolute Position innerhalb der Zahlenreihe beinhaltet, zumeist auch alternativ mit Hilfe von Relationen berechnet werden können. Ein Beispiel-Zahlenreihe mit einer kompakten formalen Repräsentation, ist die Fakultäts-Reihe:

1, 1, 2, 6, 24, 120, ...
ai = ai! i = 0, 1, 2, 3, 4, ...

Mit dieser Berechnungsformel kann man sehr einfach das Element einer bestimmten Position berechnen. Die siebte Postion wäre einfach 6! = 1 * 2 * 3 * 4 * 5 * 6 = 720. In dem hier vorgestellten Ansatz, wird die absolute Position einer Zahl aber nicht beachtet, lediglich die Reihenfolge der Elemente ist bekannt. Dieses Beispiel kann jedoch auch alternativ gelöst werden, indem eine Beziehung zwischen den Zahlen identifiziert wird.

Berechnung der Faktorenabstände -> 1, 2, 3, 4, 5 -> Addition von 1

Die Faktoren-Abstände sind hier so definiert, dass man die nachfolgende Zahl durch ihren Vorgänger teilt. Indem man die gefundene Reihenfolge 1, 2, 3, 4, 5 logisch mit 6 fortsetzt, weiß man, dass der nächste Faktoren-Abstand 6 sein muss. Um die nächste Zahl zu berechnen, muss man nur noch die letzte Zahl der gegebenen Zahlenreihe mit diesem multiplizieren: 120 * 6 = 720.

## 3.2 Der Algorithmus

Der Algorithmus deckt dabei die selben Aspekte ab, die auch bei Holzman vorgestellt wurden, lediglich die Reihenfolge weist Unterschiede auf. Beim ersten Schritt (Relations Detection) wird zuerst auf einfache konstante Operationen getestet (zum Beispiel konstante Addition +3 oder konstante Multiplikation *2 zwischen den Elementen). Anschließend werden auch komplexere Relationen überprüft, wie zum Beispiel Operationen, die mehrere Vorgänger mit einbeziehen, wie es bei der Fibonacci-Reihenfolge der Fall ist.

Dabei wurde die Reihenfolge so angepasst, dass wie beim menschlichen Vorgehen auch (vgl. 2.2.1), zunächst simple Relationen überprüft werden und erst, wenn diese verworfen wurden, wird auf komplexere Beziehungen getestet. Dabei wird aber nicht wie beim Holzman-Modell anschließend, die Periodenlänge der Reihe bestimmt (Discovery of Periodicity), sondern der dritte Schritt (Completion of Pattern-Description) wird sofort mit einbezogen. Hier wird, basierend auf der Annahme, dass die Periodenlänge 1 beträgt, überprüft, ob die gefundene Relation durch alle Positionen bestätigt wird. Falls dies der Fall ist, wird zum letzten Schritt übergegangen (Extrapolation), bei dem die gefundene Regel auf die zu füllende Position angewandt wird.

Falls die Relationen der Reihenfolge nicht durch eine konstante Operation erklärt werden konnten, wird auf hierarchische Relationen getestet. Dabei werden die Abstände der Zahlenreihe berechnet (sowohl Additions-, als auch Multiplikationsabstände) und mit diesen manipulierten Sequenzen als Argument, wird der Algorithmus erneut aufgerufen. Mit Hilfe dieses Verfahrens kann zum Beispiel die Fakultätsreihe gelöst werden (vgl. 3.1). Es ist aber auch möglich, dass die Zahlenreihe mehr als einmal manipuliert werden muss, bevor eine konstante Operation gefunden werden kann. Erst wenn auch mit diesem Verfahren keine Lösung gefunden werden konnte, wird die Periodenlänge schrittweise erhöht. Dabei wird zwischen der benachbarten und der nicht-benachbarten Variante unterschieden (vgl. 2.2). Bei der benachbarten Variante wird auf sogenannte alternierenden Reihenfolgen getestet. Ein Beispiel für eine zweifach-alternierende (Periodenlänge = 2) Reihenfolge wäre:

1, 2, 4, 8, 10, 20, 22, ...

In dieser Sequenz wird abwechselnd *2 und +2 gerechnet. Der Algorithmus geht so vor, dass er wie bei hierarchischen Zahlenreihen die Funktion rekursiv mit den Abständen (auch Faktoren-Abstände) aufruft. Diesmal wird diese manipulierte Reihenfolge jedoch in mehrere Teile zerlegt, die unabhängig voneinander getestet werden. Bei einer Periodenlänge von 2, wird nur jede jeweils zweite Relation im Zusammenhang betrachtet (siehe Abbildung 12). Wenn er für beide Relationsreihen eine Lösung findet, wendet er die, zur Lösungsstelle zugehörigen, Relation auf die, in diesem Fall, vorletzte Zahl an und ergänzt so die Reihe.

In diesem Fall wird für die geraden Relationen eine konstante Addition von 2 und für die ungeraden Relationen eine konstante Multiplikation von 2 gefunden, da die jeweilig gefundenen Reihen (2, 2, 2) logisch mit 2 fortgeführt werden können. Da für die Lösungsstelle eine ungerade Relation gefunden werden muss, muss die letzte Zahl der gegebenen Reihe mit 2 multipliziert werden (22 * 2 = 44) und damit ist die Lösung gefunden. Theoretisch könnte die nächste Zahl auch ergänzt werden, wenn die Additionsrelation nicht erkannt wird, da sie für die Lösung nicht relevant ist. Lediglich die übernächste Zahl

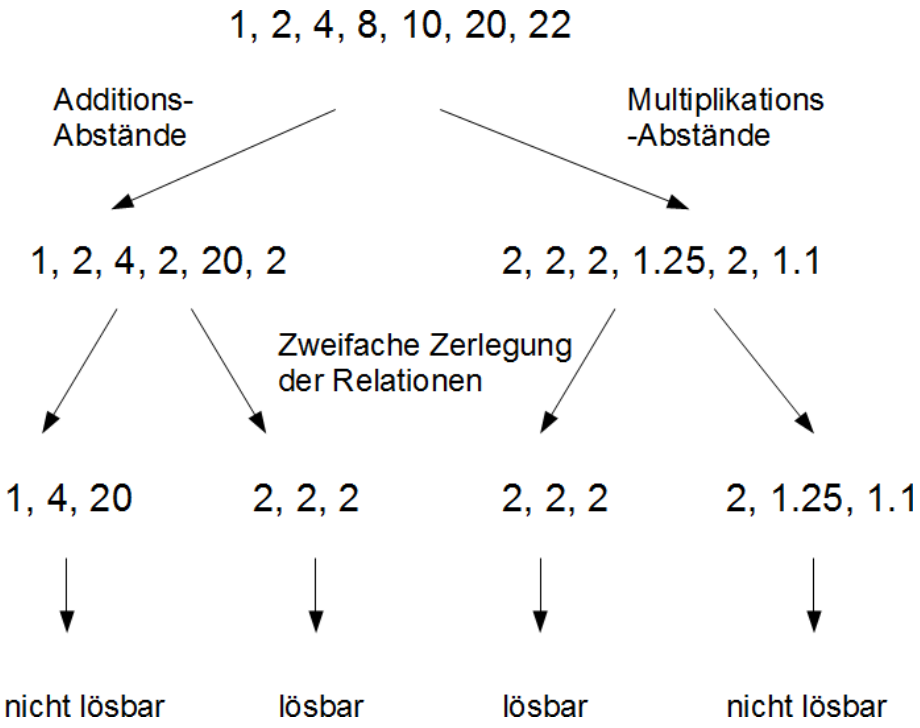



Abbildung 12: Benachbarte Variante (alternierende Operationen)

könnte dann nicht berechnet werden. Allerdings geht der Algorithmus so vor, dass er nur eine valide Lösung angibt, wenn er sich alle Relationen innerhalb der Zahlenrehe erklären kann, so dass erkennbar ist, wenn bestimmte Zusammenhänge unbekannt sind. Wenn die Periodenlänge über 2 hinaus erhöht wird, werden die Relationen in entsprechend mehr Teile zerlegt.

Die nicht-benachbarte Variante kann das System sehr viel simpler lösen. Da die benachbarten Elemente nicht in einer Beziehung stehen müssen, sondern nur bestimmte Relationen nach einer bestimmten Anzahl an Elementen wiederholt werden, kann die Zahlenreihe einfach von vorneherein in mehrere Teile zerlegt werden, die jeweils erneut rekursiv berechnet werden.

Beispiel: 2, 3, 6, 9, 10, 27, 14, 81, ...

In dieser Sequenz bestehen die Relationen immer zwischen zwei nicht direkt benachbarten Elementen. Zwischen den ungerade Zahlen liegt eine Addition von 4 vor und zwischen den geraden Zahlen eine Multiplikation von 3. Dies folgt logisch aus der Aufteilung der Zahlenreihe in zwei unabhängige Squenzen (siehe Abbildung 13).

Da es sich diesmal um eine Aufteilung zwei voneinander unabhängigen Zahlenreihen handelt, muss die entsprechende, gefundene Relation nicht auf die letzte, sondern die vorletzte Zahl angewendet werden. Somit ergibt sich die Lösung 14 + 4 = 18. Auch bei der nicht-benachbarten Variante wird die Sequenz in k Teile zerlegt, wobei k die Periodenlänge ist. Zusätzlich muss auch die Position, auf die die gefundene Relation angewandt wird, angepasst werden.

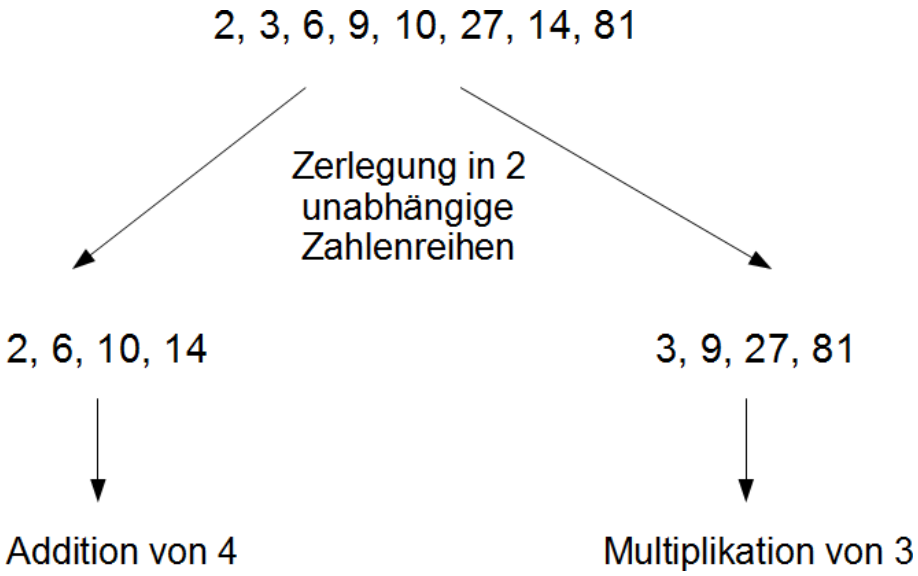



Abbildung 13: Nicht-benachbarte Variante (Aufteilung in 2 seperate Zahlenreihen)

Eine Übersicht über den Ablauf des Algorithmus wird in Abbildung 14, in Anlehnung an das Modell zum menschlichen Lösen von Zahlenreihen, dargestellt. In diesem wird ersichtlich, dass die meisten der Konzepte zwar wiederzufinden sind, das Modell aber nicht mehr in die selben, separierten Phasen unterteilt ist.

Beim Erreichen des Rekursions-Maximums ist eine Rekursion so definiert, dass mit einer Abwandlung der Zahlenreihe als Argument, der Algorithmus erneut gestartet wird. Dies ist sowohl bei einer bestätigten Erhöhung der Periodenlänge der Fall, als auch beim Aufruf der Additions- oder Faktorenabstände. Das Rekursions-Maximum wurde auf zwei zusätzliche rekursive Aufrufe festgelegt. Falls man keine derartige Begrenzung einführen würde, könnte der Algorithmus solange die jeweiligen Abstände der Abstände aufrufen, bis irgendwann zufällig eine Übereinstimmung gefunden werden würde, die längst nicht mehr vom Menschen nachzuvollziehen wäre. Das Beispiel 1, 2, 4, 8, 15, 26, 42 zeigt, dass selbst bei einer simplen Addition von 1, die Lösung, bei einer Zahlenreihe mit 2 zusätzlichen Rekursionen, längst nicht mehr trivial ist. Die Abbildung 15 veranschaulicht die Zerlegung der Zahlenreihe in die jeweiligen Abstände.

Der Punkt „eventuelle Rückrechnung der rekursiven Aufrufe“ bezieht sich darauf, dass es natürlich nicht reicht einfach die Abstände einer Zahlenreihe zu berechnen und dann diese neue Sequenz anstatt der ursprünglichen Zahlenreihe zu lösen. Manipulationen an der Zahlenreihe müssen nach einer gefundenen Lösung logisch zurück gerechnet werden. Bei Additionsabständen müsste man so die Lösung der Abstände auf das letzte Element der ursprünglichen Zahlenreihe addieren. Im vorherigen Beispiel 1, 2, 4, 8, 15, 26, 42 sähe der Lösungsweg, wie in der Abbildung 16 dargestellt, aus.

Das Maximum der Periodenlänge wurde auf 4 festgelegt. Das heißt, wenn bis zu einer Periodenlänge von 4 keine Übereinstimmungen gefunden wurden, wurde keine Lösung gefunden. Dieses Maximum wurde deswegen festgelegt, weil man auch in Intelligenztests im Normalfall keine Beispiele mit höheren Periodenlängen vorfindet. Außerdem müsste bei einer weiteren Erhöhung der Periodenlänge auch die Größe der Zahlenreihe unver-

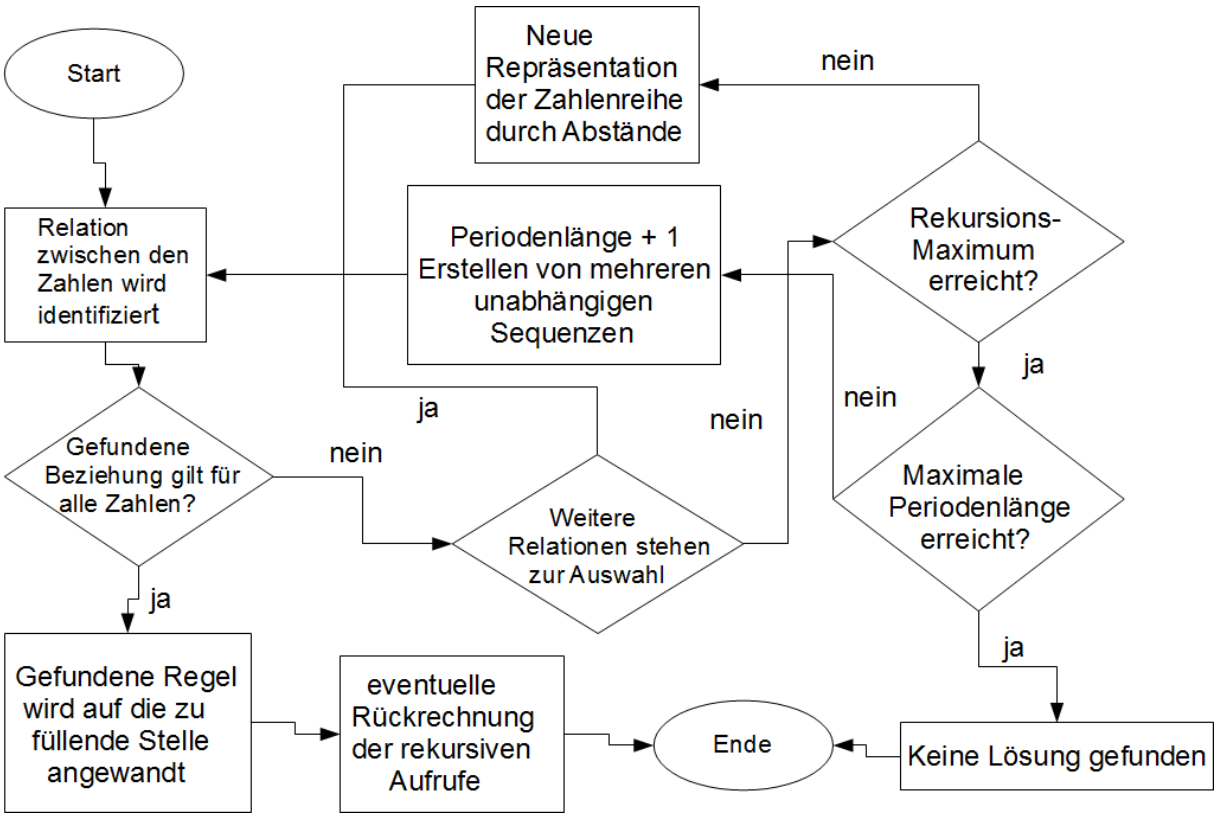



Abbildung 14: Ablauf des Algorithmus

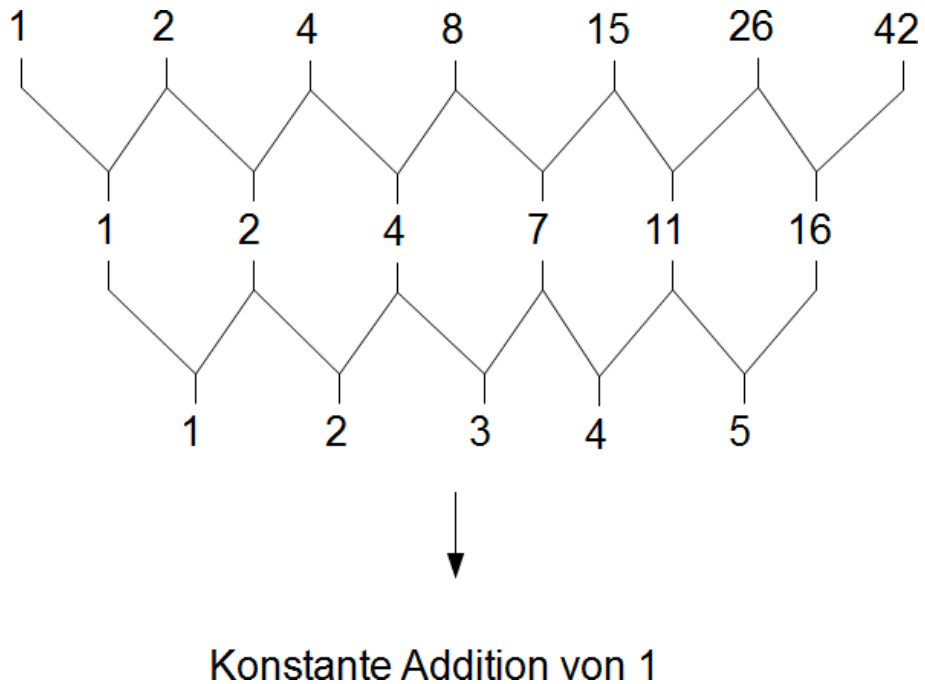



Abbildung 15: Zweifache Zerlegung einer Zahlenreihe in ihre Abstände

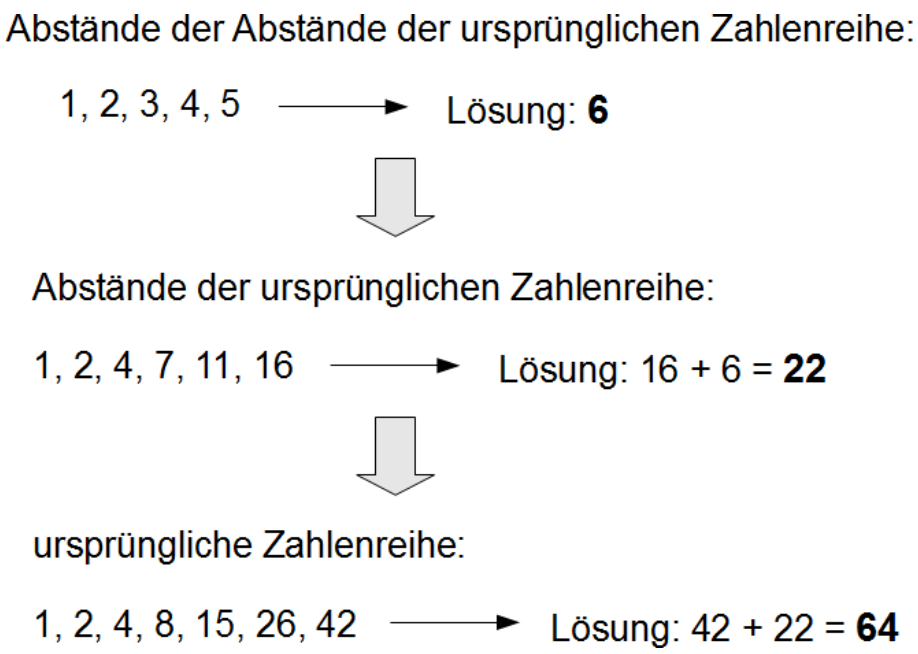



Abbildung 16: Rückrechnung der rekursiven Aufrufe

hältnismäßig weit wachsen, um unnötige Ambiguitäten auszuschließen.

## 3.3 Voraussetzungen

Um mit der Mehrdeutigkeit von Zahlenreihen umzugehen, wurden für die einzelnen Relationen minimale Größen der Sequenz festgelegt. Dies wurde, sowohl für die einzelnen Tests auf konstante, arithmetische Operationen, als auch für die Periodenlänge durchgeführt. In 2.2.3 wird der Standpunkt vertreten, dass eine Zahlenreihe mindestens aus 5 Elementen bestehen sollte, um unnötige Mehrdeutigkeiten zu vermeiden. Wenn die Periodenlänge aber erhöht wird, sollte auch die Größe der Sequenz angepasst werden, da in diesem Fall mehrere Relationsreihen unabhängig voneinander gelöst werden müssen. Da dieser Aspekt in der Praxis aber oft nicht beachtet wird und der hier vorgestellte Algorithmus trotzdem mit diesen Beispielen umgehen können sollte, wurden für die einzelnen Relationen separat Minimalgrößen festgesetzt. Durch dieses Vorgehen soll ermöglicht werden, dass auch Sequenzen, die diese Voraussetzungen nicht erfüllen, sinnvoll bearbeitet werden können.

Für die konstante Addition wurde zum Beispiel eine minimale Zahlenreihenlänge von drei festgelegt, da es bei zwei Elementen nicht möglich wäre, die, zwischen den ersten beiden Zahlen gefundene, Differenz, auf den nächsten Abstand zu testen. Daher könnte bei einer Sequenz von zwei Zahlen immer eine konstante Addition angenommen werden.

Beispiel: 16, 25 (Abstände: 9) -> konstante Addition von 9 ?

Diese Reihe ist natürlich im höchsten Maße mehrdeutig. Anstatt einer konstanten Addition könnte es sich auch um eine Potenzreihe in Form von $4^2, 5^2, 6^2, 7^2, \ldots$ handeln. Ab einer Länge von drei kann der gefundene Abstand zumindest einmal getestet werden und die Annahme einer konstanten Addition wäre somit nicht mehr komplett willkürlich. Je häufiger sich der Abstand wiederholt, desto eindeutiger ist natürlich die Relation.

Beispiel: 16, 25, 34, 43, 52 (Abstände: 9, 9, 9, 9) -> konstante Addition von 9

Die einzige Relation, die für die Anzahl von zwei Elementen zugelassen wird, ist die Identitätsrelation, da ihre Annahme allein bei zwei Zahlen nicht vollkommen willkürlich ist. So wird für eine 2-stellige Zahlenreihe nur eine Lösung ausgegeben, wenn die erste Zahl gleich der zweiten ist. Auch hier ist diese Lösung natürlich nicht eindeutig, da es sich zum Beispiel auch um die ersten zwei Zahlen einer Fibonacci-Folge handeln könnte. Jedoch ist in dem Fall, dass eine 2-stellige Sequenz gelöst werden muss, da die ursprüngliche Zahlenreihe in Folge der Periodenlänge zu stark fragmentiert wurde, nur eine Anwendung der Identitäts-Relation sinnvoll.

Beispiele: 3, 3 -> Identitäts-Relation (Lösung: 3)
3, 7 -> keine Lösung

In 2.2.3 wurde darauf hingewiesen, dass die Länge der Zahlenreihe sinnvoll angepasst werden sollte, wenn die Periodenlänge erhöht wird, da die Relationsreihe in diesem Fall aufgeteilt wird und die entsprechenden Teile separat gelöst werden müssen. Bei einer Periodenlänge von 1 und bei einer üblichen Relation, die sich nur auf ihre Vorgängerzahl bezieht, erhält man n-1 Relationen, wenn n die Anzahl der Elemente der ursprünglichen Zahlenreihe ist. Um die selbe Länge der Relationsreihen beizubehalten und somit auch den selben Grad an Eindeutigkeit zu garantieren, müsste folgende Formel angewendet werden:

(Länge der Zahlenreihe - 1) * Periodenlänge + 1

So müsste man eine Zahlenreihe, die aus fünf Elementen besteht, bei einer Erhöhung der Periodenlänge folgendermaßen anpassen, um die Länge der Relationsreihen beizubehalten:

Periodenlänge von 1: 5 Elemente -> 1 x 4 Relationen
Periodenlänge von 2: 9 Elemente -> 2 x 4 Relationen
Periodenlänge von 3: 13 Elemente -> 3 x 4 Relationen
Periodenlänge von 4: 17 Elemente -> 4 x 4 Relationen

Diese Berechnungen gelten nur für die benachbarten Relationen. Wenn es sich nämlich um die nicht- benachbarte Variante handelt, müsste die Größe der Zahlenreihe noch weiter erhöht werden, da bei deren Lösungsvorgang die komplette Zahlenreihe aufgeteilt wird. Somit müsste die Länge der Zahlenreihe einfach mit der Periodenlänge multipliziert werden.

Da solch große Sequenzen in der Praxis ungewöhnlich sind und man auch in Intelligenztests selten Zahlenreihen mit mehr als 9 Elementen vorfindet, wurden die Voraussetzungen entsprechend angepasst. Ein generelles Minimum der Zahlenreihe bei einer Periodenlänge von 1 existiert nicht, lediglich die Voraussetzungen der Relationen bestimmen die Lösbarkeit einer Zahlenreihe, somit wären 2 Elemente das Minimum für eine gefundene Lösung (bei Identitäts-Relation). Ab 5 Elementen wird zusätzlich auf eine Periodenlänge von 2 getestet, ab 7 Elementen auf eine Periodenlänge von 3 und ab 9 Elementen schließlich, auf eine Periodenlänge von 4.

# 4 Umsetzung und Evaluation

Im Folgenden wird das Programm (siehe beiliegende CD) mit den Zahlenreihen aus den vorgestellten, alternativen Lösungsansätzen getestet. Da Zahlenfolgen aus Intelligenztests nicht abgebildet werden dürfen, wurde der Algorithmus nur mit den Ergebnissen des künstlichen, neuronalen Netzes (siehe 2.3.1) und der E-Anti-Unifikation (siehe 2.3.2) verglichen. Dabei wurde sowohl die Lösung, als auch der Rechenweg des vorgestellten Algorithmus dargestellt. Bei der Zerlegung der ursprünglichen Zahlenreihe in unabhängige Sequenzen (Periodenlänge > 1), wurden dabei nur die neu formierten Zahlenreihen dargestellt, die für die Lösungsstelle vonnöten waren. Dabei ist aber sichergestellt, dass derartige Lösungen nur ausgegeben werden, wenn auch die Reihen, die nicht abgebildet wurden, erfolgreich auf eine Lösung getestet wurden.

## 4.1 Vergleich mit dem künstlichen, neuronalen Netz

Tabelle 1 beinhaltet eine Übersicht der getesteten Zahlenreihen aus [10]. Dabei bezeichnet „K“ die Anzahl der korrekten Antworten aus dem empirischen Experiment mit den 17 Teilnehmern und „F“ die Anzahl der falschen Antworten. „F“ beinhaltet dabei sowohl die inkorrekten Antworten als auch die Teilnehmer die keine Lösung angegeben haben. Der Lösungsweg zeigt, auf welche Weise der regelbasierte Ansatz eine Übereinstimmung finden konnte. Der Pfeil weist dabei darauf hin, dass ein rekursiver Aufruf mit einer, zuvor neu erstellten, Sequenz stattfindet. Die Lösung gibt schließlich die Zahl an, mit der die Sequenz fortgesetzt wurde.

Da alle Zahlenreihen einheitlich aus fünf Elementen bestehen und somit einige zu klein waren, um die nötige Eindeutigkeit für den Algorithmus zu garantieren, wurden diese Zahlenfolgen erweitert. Dabei wurde dem System keinerlei weitere Hilfestellung geleistet, sondern die Sequenzen wurden lediglich nach der gleichen Konstruktionsregel mit weiteren Elementen fortgesetzt. Die Daten der Testpersonen beziehen sich allerdings nur auf die jeweils ersten fünf Elemente. Die Reihe „5, 6, 7, 8, 10“ wurde wegen ihrer zu großen Mehrdeutigkeit nicht berücksichtigt. Folgende Fortsetzungen wären im Bereich des Möglichen:

5, 6, 7, 8, 10, 12, 14, 17, 20, 23 (+1, +1, +1, +2, +2, +2, +3, +3, +3, ...)
5, 6, 7, 8, 10, 11, 14, 15, 19, 20 (+1, +1, +1, +2, +1, +3, +1, +4, +1, ...)
5, 6, 7, 8, 10, 11, 12, 13, 15, 16 (+1, +1, +1, +2, +1, +1, +1, +2, +1, . . . )

Vier der 19 Zahlenreihen mussten erweitert werden, aber danach wurden alle Reihen richtig vom System gelöst. Auch die drei Zahlenreihen, die keine Konfiguration des neuronalen Netzes lösen konnte, wurden dabei korrekt bearbeitet. Auffallend ist aber, dass manche Beispiele, die alternierende Operationen (Periodenlänge: 2) beinhalteten, auf eine alternative Weise bearbeitet wurden, die so nicht zu erwarten war. Zum Beispiel zur Zahlenreihe „12, 15, 8, 11, 4“ wurde folgender Lösungsweg ausgegeben:

Abstände der Reihe: 3, -7, 3, -7
-> Abstände der Reihe: -10, 10, -10 -> Multiplikation mit -1

Einleuchtender wäre eine Lösung gewesen, die sich auf alternierende Operationen stützt:

zweifach alternierende Abstände [3,3] -> Identitätsrelation

Der Grund, dass der erste Lösungsweg ausgegeben wird, ist, dass der Algorithmus so aufgebaut ist, dass er zuerst alle möglichen Lösungen mit einer Periodenlänge von 1 testet, bevor er diese erhöht. Da er hier auf einen alternativen Lösungsweg mit einem zweifach rekursiven Aufruf der Abstände der Reihe kommt, wird die Periodenlänge 2 nie erreicht. Was die Fortsetzung der Reihe anbelangt, sind aber beide Lösungswege gleichwertig und korrekt.

Insgesamt hat der vorgestellte Algorithmus wesentlich besser gearbeitet als das künstliche, neuronale Netz, bei dem man noch zusätzlich in Betracht ziehen muss, dass es jeweils nur bestimmte Konfigurationen gab, die diese lösen konnten. Es gab keine bestimmte, ideale Konfiguration, die 17 der 20 Problemstellungen richtig analysieren konnte. In Anbetracht der großen Anzahl an Konfigurationen und dem logisch begrenzten Wertebereich, muss man davon ausgehen, dass viele Lösungen auch vom Zufall bestimmt waren. Zwar mussten für den regelbasierten Ansatz 4 Zahlenreihen um einige Elemente erweitert werden, allerdings lag das vor allem daran, dass die ausgewählten Beispiele zu mehrdeutig waren. Das sieht man zum Beispiel an der Zahlenreihe 8,10,14,18,26, die um zwei Elemente (20, 14) erweitert wurde. Diese ist in höchstem Maße mehrdeutig und die vorgestellte Fortsetzung ist nur eine der Möglichkeiten. Zwar wurde die Reihe auch im ursprünglichen Zustand von mehreren Konfigurationen richtig fortgesetzt, doch weist die Abnahme der richtig liegenden Konfigurationen (57;46;30;29) bei einer Zunahme der Iterationen auch hier auf zufällige Treffer hin.

## 4.2 Vergleich mit der E-Anti-Unifikation

Tabelle 2 zeigt die getesteten Zahlenreihen aus [2]. Gesetz und Zeit beziehen sich dabei auf die E-Generalisierung und Lösungsweg, sowie Lösung auf den regelbasierten Ansatz. Drei der Zahlenreihen mussten um jeweils ein Element erweitert werden, um die nötige Eindeutigkeit für den Algorithmus zu gewährleisten. Im Gegensatz zur E-Anti-Unifikation lag die Laufzeit des regelbasierten Systems bei jedem Beispiel unter einer Sekunde.

Alle acht verschiedenen Zahlenreihen wurden vom regelbasierten Ansatz erfolgreich bearbeitet. Selbst die letzte Sequenz, für die der E-Generalisierungs-Ansatz keine Berechnungsformel gefunden hat, wurde richtig fortgesetzt. Allerdings sieht man an dem Berechnungsweg, der eine Aufteilung in 3 Zahlenreihen und somit eine Periodenlänge von 3 beinhaltet, dass die Problemstellung nicht vollständig erfasst wurde. Dies erkennt man dann, wenn man zur Zahlenreihe weitere Elemente hinzufügt und den Algorithmus mit diesen neuen Beispielen aufruft:

1,2,2,3,3,3,4,4,4,4,5 -> Aufteilung in 3 Zahlenreihen [2,3,4] -> Addition von 1
Lösung: [5]

Die Lösung entspricht dem Beispiel davor und auch hier ist die Fortsetzung mit 5 natürlich korrekt. Wenn man aber ein weiteres Element hinzufügt, erhält man ein unerwartetes Ergebnis:

Tabelle 1: Menschliche Leistungen und Ausgaben des Programms

| Zahlenreihe | K | F | Lösungsweg | Lösung |
|---|---|---|---|---|
| 12, 15, 8, 11, 4 | 15 | 2 | Abstaende der Reihe: [3,-7,3,-7]<br>-> Abstaende der Reihe: [-10,10,-10]<br>-> Multiplikation mit -1.0 | [7] |
| 148, 84, 52, 36, 28 | 12 | 5 | Abstaende der Reihe: [-64,-32,-16,-8]<br>-> Multiplikation mit 0.5 | [24] |
| 2, 12, 21, 29, 36 | 14 | 3 | Abstaende der Reihe: [10,9,8,7]<br>-> Addition von -1 | [42] |
| 2, 3, 5, 9, 17 | 13 | 4 | Abstaende der Reihe: [1,2,4,8]<br>-> Multiplikation mit 2 | [33] |
| 2, 5, 8, 11, 14 | 9 | 8 | Addition von 3 | [17] |
| 2,5,9,19,37,75,149, 299, 597 | 6 | 8 | Aufteilung in 2 Zahlenreihen: [5,19,75,299]<br>-> Abstaende der Reihe: [14.0,56.0,224.0]<br>-> Multiplikation mit 4 | [1195] |
| 25, 22, 19, 16, 13 | 16 | 1 | Addition von -3 | [10] |
| 28, 33, 31, 36, 34 | 17 | 0 | Abstaende der Reihe: [-7,7,-7]<br>-> Multiplikation mit -1 | [39] |
| 3, 6, 12, 24, 48 | 13 | 4 | Multiplikation mit 2 | [96] |
| 3, 7, 15, 31, 63 | 12 | 5 | Abstaende der Reihe: [4, 8, 16, 32]<br>-> Multiplikation mit 2 | [127] |
| 4, 11, 15, 26, 41 | 8 | 9 | Addition der 2 Vorgaengerzahlen | [67] |
| 54, 48, 42, 36, 30 | 10 | 7 | Addition von -6 | [24] |
| 6, 8, 5, 7, 4 | 16 | 1 | Abstaende der Reihe: [2,-3,2,-3]<br>-> Abstaende der Reihe: [-5,5,-5]<br>-> Multiplikation mit -1.0 | [6] |
| 6, 9, 18, 21, 42 | 14 | 3 | zweifach alternierende Abstaende: [3,3]<br>-> Identität | [45] |
| 7, 10, 9, 12, 11 | 13 | 4 | Abstaende der Reihe: [3,-1,3,-1]<br>-> Abstaende der Reihe: [-4,4,-4]<br>-> Multiplikation mit -1.0 | [14] |
| 8, 10, 14, 18, 26, 34, 50 | 13 | 4 | zweifach alternierende Abstaende: [2,4,8]<br>-> Multiplikation mit 2 | [66] |
| 8, 12, 10, 16, 12, 20, 14 | 17 | 0 | zweifach alternierende Abstaende: [4,6,8]<br>-> Addition von 2 | [24] |
| 8, 12, 16, 20, 24, 28 | 15 | 2 | Addition von 4 | [28] |
| 9, 20, 6, 17, 3 | 16 | 1 | Abstaende der Reihe: [11,-14,11,-14]<br>-> Abstaende der Reihe: [-25,25,-25]<br>-> Multiplikation mit -1.0 | [14] |

Tabelle 2: Vergleich der E-Generalisierung mit dem regelbasierten Ansatz

| Zahlenreihe | Gesetz | Zeit | Lösungsweg | Lösung |
|---|---|---|---|---|
| 0,1,4,9,16 | $v_p * v_p$ | 2797 | Abstaende der Reihe: [1,3,5,7] -> Addition von 2 | [25] |
| 0,2,4,6 | $s(s(v_1))$ / $v_p + v_p$ | 3429 | Addition von 2 | [8] |
| 1,1,2,3,5 | $v_1 + v_2$ | 857 | Addition der 2 Vorgängerzahlen | [8] |
| 0,1,2 1,4,1 | $if(ev(v_p), v_p, 1)$ | 13913 | Aufteilung in 2 Reihen: [0,2,4] -> Addition von 2 | [6] |
| 0,0,1,1,0, 0,1,1 | $ev(v_2)$ | 61571 | 2fach alt. Abstaende: [1,-1,1] -> Multiplikation mit -1 | [0] |
| 0,0,1,0,0, 1,0 | $ev(v_1 + v_2)$ | 8573 | 3fach alt. Abstaende: [0,0] -> Identität | [0] |
| 0,1,3,7,15 | $s(v_1 + v_1)$ | 3714 | Abstaende der Reihe: [1,2,4,8] -> Addition von 1 | [31] |
| 1,2,2,3,3, 3,4,4,4,4 | — | 8143 | Aufteilung in 3 Reihen: [2, 3, 4] -> Addition von 1 | [31] |

1,2,2,3,3,3,4,4,4,4,5,5 -> Aufteilung in 3 Zahlenreihen 1,3,3,4 ->
Abstände der Reihe: 2, 1, 0 -> Addition von -1
Lösung: [3]

Normalerweise sollte die Reihe natürlich wieder mit 5 fortgesetzt werden und nach fünf 5ern sollte die 6 folgen, und so weiter. Die Lösung, die der Algorithmus berechnet, ist zwar mathematisch korrekt, aber in diesem Fall unbrauchbar. Weil die Zahlenreihe in dem Sinne eindeutig ist, dass die Fortsetzung mit 5 als offensichtliche Lösungsstrategie heraussticht, wie es in Kapitel 2.2.3 gefordert wurde.

Das Problem, warum der Algorithmus dieses Zahlenreihen-Problem nicht richtig analysieren kann, ist, dass hier die Periodenlänge selbst ansteigt. Die Abstände, in denen die Identitätsrelation (Wiederholung der Zahl) unterbrochen wird, ist in diesem Fall nicht, wie in dem Lösungsmodell von Holzmann (vgl. 2.2.1) regelmäßig, sondern steigen selbst konstant um 1. Auch der regelbasierte Ansatz geht von einer regelmäßigen Unterbrechung der Relation aus und versucht daher immer zuerst einheitlich mit der Periodenlänge 1 eine Lösung zu finden, bevor er diese schrittweise um 1 erhöht. Muster, die mit einer dynamischen Veränderung der Periodenlänge einhergehen, kann er daher nicht vollständig analysieren.

## 4.3 Vergleich mit der anthropomorphen Methode

Sowohl das hier vorgestellte, eigene System, als auch die anthropomorphe Methode beruhen auf einem regelbasierten Ansatz. Dadurch, dass in [12] kein genauer und vollständiger Ablauf des Algorithmus angegeben wird und auch die Beschaffenheit der Zahlenreihen aus dem Test nicht bekannt ist, wird ein detaillierter Vergleich der zwei Systeme erschwert. Ein Unterschied zwischen den beiden Ansätzen besteht allerdings darin, dass bei der anthropomorphen Methode der Algorithmus auf Zahlenreihen aus Intelligenztests spezialisiert ist. Kompliziertere Zahlenreihen, als solche, die auch vom Menschen in akzeptabler Zeit gelöst werden können, sollen gar nicht erfolgreich bearbeitet werden.

Deswegen und auch um die Laufzeit des Programms in vertretbaren Schranken zu halten, wurde das Limit für das Arbeitsgedächtnis auf 8 gesetzt. Das heißt, dass Berechnungen nur durchgeführt werden können, wenn sie maximal 8 Ziffern beinhalten. Daher steht es fest, dass für „12!“ zum Beisiel keine beschränkten Berechnungen bestehen können, da 12!(= 479001600) 9 Ziffern beinhaltet. Bei der Fakultätsreihe könnte schon ab f(9) = 9! keine beschränkte Berechnung mehr durchgeführt werden, da diese ohne eine Erweiterung des Arbeitsgedächtnisses nicht fortgesetzt werden kann:

$$\frac{\frac{f(9)}{f(8)*9}}{40320*9}$$

Der regelbasierte Ansatz hat keine solche Einschränkung und wird vom Algorithmus in seinen rechnerischen Fähigkeiten nicht eingeschränkt. So könnte im Gegensatz zur anthropomorphen Methode hier auch die Zahlenreihe „720, 5040, 40320, 362880, 3628800“ gelöst werden:
Faktoren der Reihe: 7,8,9,10 -> Addition von 1
Lösung: [39916800]

Zwar wird auch beim hier vorgestellten Ansatz die Bearbeitung der Sequenzen eingeschränkt, allerdings geschieht das nur anhand der relationalen Komplexität. So ist der Lösungsvorgang auf zwei zusätzliche, rekursive Aufrufe beschränkt, um hierarchisch zu komplexe Lösungswege, die vom Menschen nicht mehr nachzuvollziehen sind, auszuschließen. Da die Zahlenreihen, mit denen die anthropomorphe Methode getestet wurde, aufgrund ihres Vorkommens in einem Intelligenztest nicht veröffentlicht wurden, kann die Leistung der zwei verschiedenen Ansätze nicht verglichen werden. Nur die Bearbeitungszeit des anthropomorphen Ansatzes für die 11 Zahlenreihen ist bekannt, die bei ungefähr 60 Sekunden lag. Beim, hier vorgestellten, regelbasierten Ansatz, ist die Laufzeit, trotz keiner Einschränkung der arithmetischen Komplexität, hingegen vernachlässigbar. Denn die Bearbeitungszeit für die getesteten Zahlenreihen, deren Komplexität vergleichbar mit denen aus Intelligenztests und somit auch mit denen aus [12] ist, lag jeweils unter einer Sekunde.

## 4.4 Zusammenfassung

Alle 27 getesteten Zahlenreihen konnten richtig fortgesetzt werden, auch wenn im Falle des, zuvor erläuterten Beispiels, die Berechnungsformel nicht richtig analysiert wurde. Allerdings muss man auch in Betracht ziehen, dass 7 der 27 Zahlenreihen um einige, wenige Elemente erweitert werden mussten, damit das Programm sie bearbeiten konnte. Die Lösungswege wurden in einer Form ohne Berechnungsformeln, sondern wie beabsichtigt mit Rechenschritten, die vom Menschen zum Finden einer Übereinstimmung nötig wären, dargestellt. Lediglich manche Sequenzen mit alternierenden Operatoren wurden, anders als erwartet, mit einem mehrfachen Aufruf der Abstände bearbeitet.

# 5 Diskussion und Ausblick

Im vorherigen Kapitel konnte gezeigt werden, dass der selbst entwickelte, regelbasierte Ansatz sehr gut dazu geeignet ist, Zahlenreihen zu lösen und auch der Vergleich zu anderen Systemen fiel sehr günstig aus. Ein großes Manko, nämlich, dass manche Zahlenreihen in ihrer ursprünglichen Form zu kurz waren, um die Problemstellung komplett zu erfassen, liegt auch an der Funktionsweise des Algorithmus. Bei Zahlenreihen, die sich aus mehreren Sequenzen zusammensetzen (Periodenlänge > 2), müssen nämlich alle Teilsequenzen erschlossen werden, damit eine gültige Lösung zurückgegeben wird. Daher musste bei der Evaluation, auch wenn die Bearbeitung theoretisch ohne die Lösung einer Teilsequenz möglich gewesen wäre, die Zahlenreihe eventuell erweitert werden, um alle Unklarheiten zu beseitigen. Ein Beispiel wäre folgende Reihe:

1,4,1,13,1,25

Trotz der unbekannten Teilsequenz „4, 13, 25“, könnte diese Zahlenreihe ohne weiteres mit einer 1 fortgeführt werden, da, beginnend mit der 1.Stelle, jede zweite Zahl eine 1 ist. Allerdings fordert das Programm in diesem Fall eine Erweiterung der Reihe, damit auch die Teilsequenz „4, 13, 25“ gelöst werden kann. Eine mögliche Verbesserung bestünde darin, dass die Zahlenreihe zwar gelöst wird, aber gleichzeitig in der Ausgabe darauf hingewiesen wird, dass eine Teilsequenz nicht bearbeitet werden konnte.

Zusätzlich haben auch die Restriktionen, dass Muster erst ab einer Mindestanzahl an Elementen berücksichtigt werden, Einfluss auf die Mindestlänge der Zahlenreihe. Diese mussten vor allem deswegen eingeführt werden, weil der Algorithmus so vorgeht, dass er, sobald er eine Übereinstimmung gefunden hat, die entsprechende Lösung zurückgibt und terminiert. Ohne Restriktionen würde somit bei einer mehrdeutigen Zahlenreihe ausschließlich die Reihenfolge, in der die verschiedenen Muster getestet werden, die Lösung bestimmen. Deswegen musste mit der Mindestanzahl an Elementen sichergestellt werden, dass die Lösungen einen gewissen Grad an Eindeutigkeit garantieren. So wird zum Beispiel eine Fibonacci-Reihenfolge erst ab einer Elementanzahl von 4 berücksichtigt, da 3 Elemente sehr oft nur zufällig eine solche Konstellation aufweisen. Sonst würde schon eine Reihe, die nur aus „1, 2, 3“ besteht, eventuell als Fibonacci-Reihenfolge erkannt werden und weitere Lösungen würden nicht weiter in Betracht gezogen werden.

Natürlich würde auch der regelbasierte Ansatz bei diesem Beispiel nicht auf eine Fibonacci-Folge schließen, sondern auf eine Additionsreihe +1. Dies liegt daran, dass der Algorithmus entsprechend dem menschlichen Lösen von Zahlenreihen so angepasst ist, dass erst auf einfache Muster getestet wird und erst wenn diese ausgeschlossen wurden, werden auch komplexere Lösungswege berücksichtigt. Trotzdem ist der hier vorgestellte Ansatz nicht optimal, da es auch trotz dieser Anpassung vorkommen könnte, dass nach dem Finden einer Übereinstimmung noch eine bessere Lösungsmöglichkeit folgt, die in diesem Fall nicht mehr berücksichtigt werden könnte. Außerdem ist die Reihenfolge, in der die Muster getestet werden, auch noch nicht optimal angepasst worden. Bei der Bearbeitung von zweifach alternierenden Reihenfolgen wurde beobachtet, dass nicht ein Lösungsweg angegeben wurde, der sich auf alternierende Operationen stützt, sondern ein zweifacher Aufruf der Abstände ausgeführt wurde. Zwar hat das in diesem Fall die Fortsetzung nicht beeinflusst, da die beiden Lösungen äquivalent waren, aber dennoch sieht man daran, dass

Optimierungspotenzial besteht.

Der Lösungsweg wird dadurch bestimmt, dass zuerst alle möglichen Lösungen mit einer Periodenlänge von 1 getestet werden, bevor diese erhöht wird. Eine gute Anpassung bestünde darin, dass unter Einbezug der Richtlinie, dass leichte Muster vor schwereren getestet werden, der Algorithmus so modifiziert wird, dass die Reihenfolge zuerst durch die Anzahl der rekursiven Aufrufe bestimmt wird. So würde zunächst ein einfacher Aufruf der Abstände getestet werden, bevor dann einfache alternierende Operationen berücksichtigt werden und schließlich zuletzt komplexere hierarchische Muster, wie der zweifache Aufruf der Abstände.

Die einzige Zahlenreihe, die vom Programm nicht korrekt analysiert werden konnte, war „1, 2, 2, 3, 3, 3, 4, 4, 4, 4“. Der Grund hierfür ist, dass der Algorithmus nicht die Fähigkeit besitzt mit einer wachsenden Periodenlänge innerhalb der Zahlenreihe umzugehen. Zwar könnte das Programm um eine solche Funktion erweitert werden, allerdings erscheint das wenig zweckmäßig, da bis auf dieses sehr simple Beispiel wenige Zahlenreihen in den Sinn kommen, die trotz einer wachsenden Periodenlänge, eindeutig und gleichzeitig vom Menschen lösbar sind. Auch in dem Modell zum menschlichen Lösen von Holzman (vgl. 2.2.1) wird von konstanten Unterbrechungen/Wiederholungen der Relationen ausgegangen und wachsende Periodenlängen werden somit nicht berücksichtigt.

Das größte Verbesserungspotenzial wird darin gesehen, dass der Algorithmus im Moment auf eine einzige Rückgabe mit zugehörigem Lösungsweg beschränkt ist, da dieser nach dem Finden der ersten Übereinstimmung terminiert. Möglich wäre es, die Suche weiterlaufen zu lassen und alle möglichen Fortsetzungen mit eventuell jeweils verschiedenen Lösungswegen ausgeben zu lassen. Diese könnten dann nach der strukturellen Einfachheit oder anderen Parametern sortiert werden, so dass eine dieser Fortsetzungen als die Offensichtlichste gewählt wird. Eine solche Erweiterung würde auch dafür sorgen, dass weniger oder gar keine Zahlenreihen mehr aufgrund ihrer Mehrdeutigkeit erweitert werden müssten. Denn dadurch könnten die Restriktionen für die Mindestanzahl an Elementen bei der Erkennung von Mustern abgeschafft werden, weil mehrere mögliche Ergebnisse ausgegeben werden könnten und eine strikte Eindeutigkeit nicht mehr vonnöten wäre. Die Anzahl der Elemente könnte dann wiederum dafür verwendet werden, die Reihenfolge der Ergebnisse mitzubestimmen. Vor einer derartigen Erweiterung des Algorithmus wäre es sinnvoll, die Faktoren genauer zu untersuchen, die darauf Einfluss nehmen, welche Lösungen vom Menschen als die Offensichtlichen angesehen werden.

# Literatur

[1] Jens Asendorpf. *Persönlichkeitspsychologie*. Springer Medizin Verlag, Heidelberg, 2009. S. 80–81.

[2] Jochen Burghardt. E-generalization using grammars. *Artificial Intelligence*, 165:1–35, 2005.

[3] P.A. Carpenter, M.A. Just, and P. Shell. What one intelligence test measures: A theoretical account for processing in the raven progressive matrices test. *Psychological Review*, 97:404–431, 1990.

[4] Jürgen Guthke. *Ist Intelligenz messbar? Einführung in Probleme der psychologischen Intelligenzforschung und Intelligenzdiagnostik*. Deutscher Verlag der Wissenschaften, Berlin, 1980.

[5] Thomas G. Holzman, James W. Pellegrino, and Robert Glaser. Cognitive dimensions of numerical rule induction. *Journal of Educational Psychology*, pages 360–373, 1982.

[6] Thomas G. Holzman, James W. Pellegrino, and Robert Glaser. Cognitive variables in series completion. *Journal of Educational Psychology*, pages 603–618, 1983.

[7] Ulrike Kipman, Gabriele Köhlböck, and Walburga Weilguny. *Psychologische Testverfahren zur Messung intellektueller Begabung*. ÖZBF, Salzburg, 2012.

[8] Klaus Korossy. Solvability and uniqueness of linear-recursive number sequence tasks. *Methods of Psychological Research*, 3:43–68, 1998.

[9] David Kriesel. Ein kleiner Überblick über neuronale netze. http://www.dkriesel.com/_media/science/neuronalenetze-de-zeta2-2col-dkrieselcom.pdf, 2005. [Online; accessed 15-March-2013].

[10] Marco Ragni and Andreas Klein. Predicting numbers : An ai approach to solving number series. *KI 2011: Advances in artificial intelligence*, pages 255–259, 2011.

[11] Charles Spearman. General intelligence, objectively determined and measured. *American Journal of Psychology*, 15:201–293, 1904.

[12] Claes Strannegard. An anthropomorphic method for number sequence problems. *Cognitive Systems Research*, 22-23:27–34, 2013.

[13] Louis L Thurstone and Elma Thurstone. *Factorial Studies of Inteliigence*. The University of Chicago Press, Chicago, 1941.

[14] Osnabrück Uni. Methoden der Datenanalyse und -prognose. http://www.informatik.uni-osnabrueck.de/prakt/pers/dipl/doc/jhannema/MethodenderDatenanalyseun1.htm. [Online; accessed 15-March-2013].

[15] www.neuronalesnetz.de. Neuronale Netze - Eine Einführung. http://www.neuronalesnetz.de/downloads/neuronalesnetz_de.pdf. [Online; accessed 15-March-2013].

# A Erklärung

Ich erkläre hiermit gemäß §17 Abs. 2 APO, dass ich die vorstehende Bachelorarbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

_____ _____

Datum Unterschrift