Technische Universität Dresden
Institut für Theoretische Informatik

Universalitätsresultate für Watson-Crick-Automaten

Bachelorarbeit

Paul Nitzsche
Matrikel-Nummer 3569936

Betreuerin Dr.-Ing. Monika Sturm
Betreuender Hochschullehrer Prof. Dr.-Ing. Franz Baader

Beginn: 04.06.2012
Abgabe: 17.09.2012

Inhaltsverzeichnis

1 Einleitung

In der modernen Gesellschaft nimmt die Informationsverarbeitung einen immer größeren Stellenwert ein und somit auch immer größere Ausmaße an. Es sollen Berechnungen in kürzester Zeit und mit so wenig Speicherbedarf wie möglich durchgeführt werden. Hierbei stoßen jedoch die konventionellen Rechenkonzepte auf technologische und physikalische Grenzen. Folglich entwickelte sich in den letzten Jahren ein Trend, Naturwissenschaften wie Biologie, Chemie und Physik mit Gebieten der Informatik interdisziplinär zu verknüpfen und so alternative Rechenkonzepte zu erschaffen. Dadurch bildeten sich neue Forschungszweige, wie die Quanteninformatik, die Neuroinformatik, evolutionäres Rechnen und molekulares Rechnen.

Das Thema dieser Arbeit liegt in dem interdisziplinären Gebiet des molekularen Rechnens, welches sich aus einer Kombination der Teilgebiete Biologie, Chemie und Theoretische Informatik zusammensetzt. Dabei werden Berechnungen mit Molekülen als Datenträger und Speichermedium ausgeführt. Das molekulare Rechnen lässt sich wiederum in die Teilgebiete des Rechnens „in vitro“ und „in vivo“ unterteilen. Während zum Rechnen „in vivo“ formale Modelle gehören, die Prozesse in der Natur beschreiben, wie z.B. Membrane-Computing oder Gene-Assembly, so gehören zum Rechnen „in vitro“ Modelle, die versuchen Berechnungen durch Operationen auf DNA-Strängen auszuführen, wie z.B. beim DNA-Computing. Beim DNA-Computing wird die DNA (Desoxyribonukleinsäure) als Datenträger und Speichermedium genutzt. Die Vorteile sind offensichtlich:

- DNA-Moleküle sind äußerst kleine und kompakte Speichermedien. DNA-Moleküle erlauben eine Speicherdichte von bis zu 10^{21} Basenpaaren pro Liter, das entspricht etwa 1 bit pro nm^3.
- DNA-Moleküle sind ein persistentes Speichermedium.
- DNA-Moleküle bieten eine redundante, verlustfreie und dezentrale Informationsspeicherung.
- DNA-Moleküle haben eine gewisse Fehlerresistenz, es lassen sich einfache Fehler in einem bestimmten Umfang effizient beheben.

Im Gegensatz dazu steht die weit entwickelte und auch praktisch oft genutzte Automatentheorie. Die Watson-Crick-Automaten, die in dieser Arbeit vorgestellt werden, vereinen Automatentheorie und DNA-Computing.

Es soll in dieser Arbeit untersucht werden, ob das Modell der Watson-Crick-Automaten universell ist und ob ein Watson-Crick-Automat angegeben werden kann, der in der Lage ist, alle anderen Watson-Crick-Automaten zu simulieren. Dazu werden zuerst die molekularbiologischen und mathematischen Grundlagen erklärt und wichtige Begriffe erläutert, wonach im nächsten Kapitel die verschiedenen Arten von Watson-Crick-Automaten definiert werden. Im vorletzten Kapitel werden die Watson-Crick-Automaten schließlich auf ihre Universalität untersucht.

2 Grundlegende Betrachtungen

In diesem Kapitel werden wichtige Begriffe definiert, die für das Verständnis der folgenden Kapitel notwendig sind. Dabei sollen im ersten Abschnitt molekularbiologische Grundlagen vermittelt werden, um dann im zweiten Abschnitt zu den Definitionen aus der Theoretischen Informatik überzugehen. Die Definitionen und Begriffe im Folgenden orientieren sich an [Ba10], [HS04] und [PRS98].

2.1 Molekularbiologische Grundlagen

Wie schon in der Einführung erläutert, liegen die Anregungen und Ideen des DNA-Computing im molekularbiologischen Bereich. Folglich soll in diesem Abschnitt auf die molekularbiologischen Grundlagen eingegangen werden, welche für die praktische Realisierung, die Entwicklung des formalen Modells der DNA und der Watson-Crick-Automaten wichtig sind. Dabei werden laborpraktische Verfahren der DNA-Manipulation erläutert, mit deren Hilfe Berechnungen ausgeführt werden können.

2.1.1 DNA-Moleküle und ihre Struktur

Der Fokus wird hierbei natürlich zuerst auf das DNA-Molekül selbst gelegt, welches eines der wichtigsten Moleküle in lebenden Zellen ist. DNA ist ein Polymer, welches aus einer Sequenz von Nukleotiden besteht. Nukleotide enthalten drei Komponenten: Die Zuckerart Desoxyribose, Phosophorsäure und eine Base. Die Unterscheidung der Nukleotidarten erfolgt anhand der an sie gebundenen Basen. So wird das Nukleotid mit der Base Adenin mit A, der Base Cytosin mit C, der Base Thymin mit T und der Base Guanin mit G beschriftet.

Die Desoxyribose ist aus fünf Kohlenstoffatomen aufgebaut, die ringförmig angeordnet sind und mit $1'$ bis $5'$ bezeichnet werden. Weitere Nukleotide können mittels Phosphodiesterbindung an das $3'-$ und $5'-$Kohlenstoffatom angelagert werden. Durch die zwei verschiedenen Anlagerungsmöglichkeiten an ein Nukleotid existieren zwei verschiedene Leserichtungen für Nukleotidketten, in $5'-3'-$Richtung (*sense*) und in $3'-5'$-Richtung (*antisense*). Ein DNA-Strang

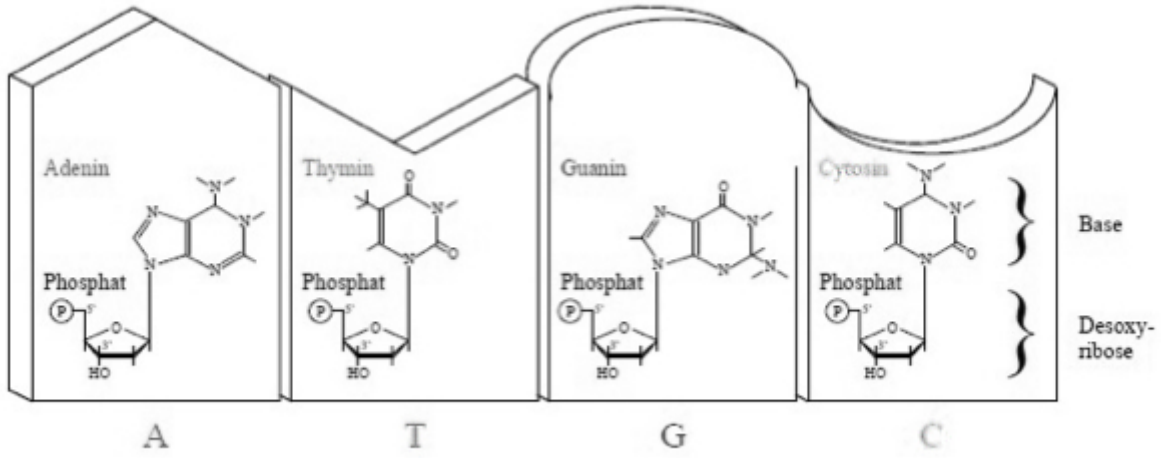

Abbildung 2.1: Die Nukleotide A, T, G, C. Abbildung aus [St12] entnommen.

kann somit durch die Struktur seiner Nukleotidkette und die Markierungen an den Enden des Stranges eindeutig identifiziert werden. Natürlich vorkommende DNA-Endmarkierungen sind Phosphatgruppen $-PO_4$ oder Hydroxylgruppen $-OH$.

Ein DNA-Einzelstrang ist eine Folge von Nukleotiden A, C, G und T, deren Enden mit $5'$ und $3'$ angegeben werden. An jedem Ende eines DNA-Einzelstrangs findet man eine Strangendenmarkierung vor. Strangendenmarkierungen sind chemische Gruppen oder Moleküle, die an den $5'$- oder $3'$-Enden angelagert werden können. Bisher wurde DNA nur in ihrer Primärstruktur betrachtet, im Folgenden soll die Sekundärstruktur untersucht werden. Bestimmte Voraussetzungen ermöglichen die Basenpaarung zwischen benachbarten Einzelsträngen. Da Wasserstoffbrücken ausgebildet werden, kann sich Basenpaarung nur zwischen komplementären Basen vollziehen. Komplementär sind die Basen Adenin und Thymin, Guanin und Cytosin. Folglich sind die Nukleotide A und T, sowie G und C komplementär.

Ein DNA-Doppelstrang besteht aus zwei DNA-Einzelsträngen, die sich durch Basenpaarung antiparallel und komplementär aneinander gelagert haben. Antiparallel bedeutet hierbei, dass ein Strang in sense- und ein Strang in antisense-Richtung liegt. Die Enden eines DNA-Doppelstrangs können entweder sticky oder blunt sein, wobei der Doppelstrang entweder einen Einzelstrangüberhang am $3'-$ oder $5'-$ Ende hat oder kein Einzelstrangüberhang vorhanden ist. Neben der Primär- und Sekundärstruktur besitzt die DNA auch eine Tertiärstruktur, die an dieser Stelle jedoch nicht weiter untersucht werden soll, da sie für die formalen Modelle in späteren Kapiteln nicht von Bedeutung ist. Eine wichtige Eigenschaft der DNA-Stränge ist deren elektrische Ladung. DNA-Stränge sind alle mit unterschiedlichen negativen

Ladungen versehen, wodurch Analysemethoden, wie die Gel-Elektrophorese, genutzt werden können.

Bis jetzt wurde die DNA in ihren Ausprägungen und in ihrer Struktur betrachtet. Interessant ist nun, welche Berechnungsmöglichkeiten sie uns bietet. Die DNA soll als Datenstruktur dienen, welche Informationen enthält. Um Berechnungen ausführen zu können, müssen jedoch Operationen zur Manipulation der DNA vorhanden sein. Diese sollen im nächsten Abschnitt besprochen werden.

2.1.2 Operationen über DNA-Molekülen

DNA-Moleküle sind in jedem Lebewesen vorhanden, jedoch kann es für Berechnungen notwendig sein, dass DNA-Moleküle erzeugt werden müssen. Deswegen ist es von essentieller Bedeutung, dass eine Operation existiert, mit welcher DNA gewonnen werden kann.

Definition 2.1.1. *Synthesis*

Mit Synthesis wird die Operation bezeichnet, die DNA-Einzelstränge mit frei wählbaren Nukleotidsequenzen in einer praktisch realisierbaren Anzahl erzeugt.

Mit der Synthesis ist es möglich Einzelstränge zu erzeugen, allerdings sollen die Automaten, die später genauer betrachtet und definiert werden, auf DNA-Doppelsträngen lesen. Dementsprechend wird noch eine Operation benötigt, mit der DNA-Einzelstränge beliebig zu DNA-Doppelsträngen aneinander gelagert werden können.

Definition 2.1.2. *Annealing*

Mit Annealing wird das Zusammenlagern von mindestens zwei DNA-Einzelsträngen oder eines Einzelstranges mit sich selber bezeichnet. Dabei werden die DNA-Einzelstränge an ihren antiparallel-komplementären Stellen zu DNA-Doppelsträngen unter Bildung aller Anlagerungsmöglichkeiten zusammengelagert.

Damit sind alle nötigen Grundbegriffe und Operationen definiert, damit beliebige DNA-Doppelstränge erzeugt werden können.

2.2 Mathematische Grundlagen

Dieses Kapitel soll später benötigte Definitionen aus der Mathematik und Theoretischen Informatik bereitstellen. In der Theoretische Informatik werden Informationen durch Wörter über einem Alphabet dargestellt. Da in späteren Kapiteln diverse enzymatische Operationen auf DNA-Molekülen formalisiert und in einem Berechnungsmodell zusammengefasst werden, führt dies unumgänglich zur formalen Sprachtheorie, welche als Mittel zur Formalisierung und Analyse dient. Auf den nachfolgenden Seiten soll eine Einführung in die formale Sprachtheorie gegeben werden.

Definition 2.2.1. *Wörter über einem Alphabet*

Ein Alphabet Σ ist in der formalen Sprachtheorie eine endliche und nichtleere Menge von Symbolen $a \in \Sigma$. Ein Wort ist als endliche Folge von Symbolen $w = w_1 \dots w_n$ mit $w_i \in \Sigma$, $1 \leq i \leq n$ und n der Länge des Wortes w über dem Alphabet definiert.

Definition 2.2.2. *Längenoperator, Häufigkeitsoperator*

Mit $|w| = n$ ist die Länge eines Wortes $w = w_1 \dots w_n$ gegeben. Das leere Wort ε hat die Länge $|\varepsilon| = 0$.

Die Anzahl der Vorkommen eines Symbols a in einem Wort w ist mit $|w|_a$ notiert.

Definition 2.2.3. *Teilwort, Präfix*

Ein Wort w' ist Teilwort eines Wortes w, wenn $w = uw'v$ mit $u \geq 1, v \geq 1$. Dagegen ist ein Wort $w' = Pre_{|w'|}(w)$ Prefix von w, wenn $w = w'v$ mit $v \geq 0$.

Definition 2.2.4. *Konkatenation, Kleene'sche Hülle, positive Hülle, formale Sprache*

Die Konkatenation $\circ$ zweier Wörter u und v ist das Wort $u \circ v = uv$. Die Kleene'sche Hülle Σ^* bezeichnet die Menge aller Wörter über einem Alphabet.

Die positive Hülle $\Sigma^+ = \Sigma^* - \varepsilon$ ist die Menge aller nichtleeren Wörter über dem Alphabet Σ. Mathematisch betrachtet ist Σ^* die Trägermenge des freien Monoids $(\Sigma^*, \circ, \varepsilon)$.

Eine formale Sprache L ist eine Menge von Wörtern $L \subseteq \Sigma^*$. Die Operatoren $L_1 \cap L_2$, $L_1 \cup L_2$, $\overline{L} = \Sigma^* - L$, $L_1 - L_2 = L_1 \cap \overline{L_2}$ sind bereits aus der Mengentheorie bekannt. Sprachen können ebenso wie Wörter konkateniert werden. Die Konkatenation von L_1 und L_2 ist gegeben durch $L_1 \circ L_2 = \{u \circ v \mid (u \in L_1) \wedge (v \in L_2)\}$. Oftmals wird der Konkatenationspunkt $\circ$ aus Gründen der Übersicht weggelassen. Die Kleene'sche Hülle über einer Sprache L ist induktiv definiert:

$$L^0 = \{\varepsilon\}$$
$$L^{n+1} = L^n \circ L, n \in N$$
$$L^* = \bigcup_{n \geq 0} L^n$$
$$L^+ = \bigcup_{n \geq 1} L^n$$

Definition 2.2.5. *Kardinalität einer Sprache*

Für eine Sprache $L \subseteq \Sigma^*$ gibt die Kardinalität die Anzahl der in ihr enthaltenen Wörter an.

$$L = \{a_1, a_2, \ldots, a_n\} \Longleftrightarrow ||L|| = n.$$

Definition 2.2.6. *Potenzmenge*

Die Potenzmenge 2^M einer beliebigen Menge M enthält alle Teilmengen X von M. Sie ist mit

$$2^M = \{X \mid X \subseteq M\}$$

gegeben.

Definition 2.2.7. *Morphismus, Inverser Morphismus, Projektion*

Eine Abbildung

$$h : \Sigma_1^* \longmapsto \Sigma_2^*$$

mit

$h(\varepsilon) = \{\varepsilon\}$,
$h(ab) = h(a)h(b)$, $a, b \in \Sigma^*$

wird Morphismus genannt. Darüber hinaus kann auch eine Sprache abgebildet werden.

Für $L \subseteq \Sigma^*$ gilt $h(L) = \bigcup_{w \in L}\{h(w) | w \in L\}$.

Ebenso existiert zu h der inverse Morphismus

$$h^{-1} : \Sigma_2 \longmapsto 2^{\Sigma_1}$$

mit

$$h^{-1}(w) = \{x \in \Sigma_1^* \mid w = h(x)\}.$$

Eine Projektion

$$pr_{\Sigma_1} : (\Sigma_1 \cup \Sigma_2)^* \longmapsto \Sigma_1^*$$

ist ein Morphismus für den

$$pr_{\Sigma_1}(a) = a \text{ für } a \in \Sigma_1$$
$$pr_{\Sigma_1}(a) = \varepsilon \text{ für } a \notin \Sigma_1$$
$$pr_{\Sigma_1}(w) = pr_{\Sigma_1}(w_1) \dots pr_{\Sigma_1}(w_n) \text{ für } w \in (\Sigma_1 \cup \Sigma_2)^*$$

gilt.

Definition 2.2.8. *Equalityset zweier Morphismen*

Das Equalityset zweier Morphismen $h_1, h_2 : \Sigma_1^* \longmapsto \Sigma_2^*$ enthält alle Wörter $w \in \Sigma_1^*$, die von den Morphismen auf gleiche Wörter abgebildet werden.

$$EQ = \{w \in \Sigma_1^* \mid h_1(w) = h_2(w)\}.$$

Definition 2.2.9. *Relation*

Eine Relation R ist Teilmenge des kartesischen Produktes von n Mengen $M_1, \dots, M_n$.

$$R \subseteq M_1 \times \dots \times M_n \text{ mit } M_1 \times \dots \times M_n = \{(m_1, \dots, m_n) \mid (m_1 \in M_1) \wedge \dots \wedge (m_n \in M_n)\}$$

Definition 2.2.10. *Parikh-Vektor*

Der Parikh-Vektor eines Wortes $w \in \Sigma^*$ bezüglich des Alphabets $\Sigma = \{a_1, \dots, a_n\}$ mit $n = ||\Sigma||$ ist mit $\Psi_\Sigma(w) = (|w|_{a_1}, |w|_{a_2}, \dots, |w|_{a_n})$ gegeben. Für eine Sprache $L \subseteq \Sigma^*$ ist die Menge der Parikh-Vektoren $\Psi_\Sigma(L) = \{\Psi_\Sigma(w) \mid w \in L\}$.

Definition 2.2.11. *Semilinearität und Linearität einer Menge*

Eine Menge $M \subseteq N^n$ heißt linear, falls Vektoren $v_i \in N^n$, $i \leq i \leq m$ existieren und M durch

$$M = \{v_0 + \sum_{i=1}^{m} \alpha_i v_i \mid \alpha_1, \dots, \alpha_m \in N\}$$

erzeugt werden kann. Eine Menge $K \subseteq N^n$ heißt semilinear, falls lineare Mengen $M_1, \dots, M_k$ existieren und

$$M = \bigcup_{1 \leq i \leq k} M_i$$

gilt. Eine Sprache $L \subseteq \Sigma^*$ heißt linear bzw. semilinear, falls die Menge $\Psi_\Sigma(L)$ linear bzw. semilinear ist. Eine Sprache heißt auch linear, falls sie durch eine lineare Grammatik erzeugt werden kann. Entsprechend ist auch der Schnitt zweier semilinearen bzw. linearen Mengen wiederum semilinear bzw. linear.

Definition 2.2.12. *Satz von Parikh*

Für eine kontextsensitive Sprache mit $L \subseteq \Sigma^*$ (s. Definition 2.2) ist $\Psi_\Sigma(L)$ semilinear.

Daraus folgt, dass eine reguläre Sprache $R \subseteq (\Sigma')^*$ mit $R = \Psi_\Sigma(L)$ existiert. Es ist nicht notwendig, dass Σ und Σ' übereinstimmen.

Der Satz von Parikh besagt, dass die kontextfreien Sprachen mit den regulären Sprachen übereinstimmen, falls nur die Häufigkeit der Symbole betrachtet wird.

Definition 2.2.13. *Endlicher Automat*

Ein endlicher Automat A wird durch ein Tupel

$$A = (Q, \Sigma, q_0, \delta, F)$$

mit

Q der Menge der Zustände,

Σ dem Bandalphabet,

q_0 dem Anfangszustand,

δ der Übergangsfunktion und

F der Menge der Endzustände

beschrieben. Es gilt $F \subseteq Q$ und $q_0 \in Q$. Ist $||\delta(q, a)|| \leq 1$ für alle $q \in Q$ und $a \in \Sigma$, dann ist der Automat deterministisch. Ein Übergang im Automaten ist mit der Übergangsfunktion

$$\delta : Q \times \Sigma^* \longmapsto 2^Q$$

gegeben. Dabei bedeutet

$$q' \in \delta(q, a),$$

dass der Automat A von einem Zustand q mit dem Einlesen des Symbols a in den Zustand q' übergeht. Ein Konfigurationsübergang in A ist mit der Relation $\vdash$ für $q, q' \in Q$, $a \in \Sigma$, $w \in \Sigma^*$ definiert:

$$(q, aw) \vdash (q', w) \text{ falls } q' \in \delta(q, a).$$

Mit $\vdash^*$ ist die reflexive und transitive Hülle von $\vdash$ gegeben. Der Automat akzeptiert, falls ein Endzustand erreicht wurde und alle Symbole auf dem Eingabeband eingelesen wurden. Somit ist die von A akzeptierte Sprache

$$L(A) = \{w \in \Sigma^* \mid (q_0, w) \vdash^* (q_f, \varepsilon), q_f \in F\}.$$

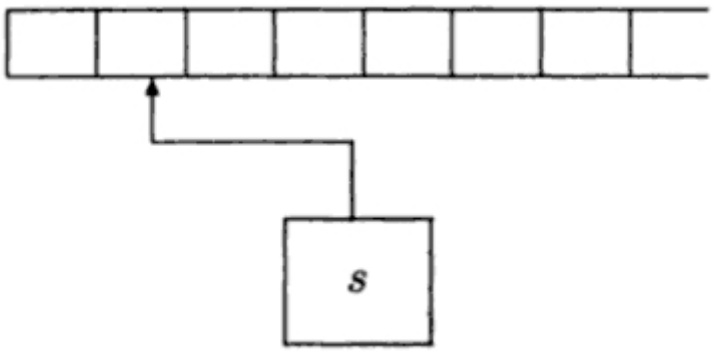

Abbildung 2.2: Schema eines endlichen Automaten. Abbildung aus [PRS98] entnommen.

Definition 2.2.14. *Formale Grammatik*

Eine formale Grammatik ist ein Notationsmittel, um formale Sprachen zu erzeugen und eindeutig zu beschreiben. Beschrieben wird eine formale Grammatik G durch ein Tupel

$$G = (N, \Sigma, P, S).$$

Dabei ist

N die Menge der Nichtterminalsymbole,

Σ die Menge der Terminalsymbole,

P die Menge der Produktionsregeln und

S das Startsymbol.

Es gilt $N \cap \Sigma = \emptyset$, $S \in N$ und $P \subseteq (N \cup \Sigma)^* N (N \cup \Sigma)^* \times (N \cup \Sigma)^*$. Die Produktionsregeln sind Tupel $(u, v) \in P$, werden jedoch mit $u \longrightarrow v$ notiert. Eine Ableitung eines Wortes $y \in (N \cup \Sigma)^*$ aus $x \in (N \cup \Sigma)^*$ in G ist wie folgt definiert:

$$x \Longrightarrow_G y \text{ gdw. } x = x_1 u x_2, y = x_1 v x_2, \text{ für } x_1, x_2 \in (N \cup \Sigma)^* \text{ und } u \longrightarrow v \in P.$$

Dementsprechend ist die von G beschriebene Sprache mit der reflexiven und transitiven Hülle der Ableitung $\Longrightarrow_G^*$

$$L(G) = \{w \in \Sigma^* \mid S \Longrightarrow_G^* w\}.$$

Definition 2.2.15. *Endliche, reguläre, lineare, kontextfreie, kontextsensitive, unbeschränkte Grammatiken*

Eine Grammatik wird

regulär genannt, falls für jede Regel $u \longrightarrow v \in P$, $u \in N$ und $v \in \Sigma N \cup \Sigma \cup \{\varepsilon\}$ gilt,

linear genannt, falls für jede Regel $u \longrightarrow v \in P$, $u \in N$ und $v \in (\Sigma^* \cup \Sigma^* N \Sigma^*)$ gilt,

kontextfrei genannt, falls für jede Regel $u \longrightarrow v \in P$, $u \in N$ und $v \in (N \cup \Sigma)^*$ gilt,

kontextsensitiv genannt, falls für jede Regel $u \longrightarrow v \in P$ $u = u_1 A u_2$, $v = u_1 x u_2$ für $u_1, u_2 \in (N \cup \Sigma)^*$, $A \in N$ und $x \in (N \cup \Sigma)^+$ gilt,

unbeschränkt genannt, wenn die Regeln keinen Einschränkungen unterliegen.

Definition 2.2.16. *Chomsky Hierarchie*

Ein weiterer wichtiger Aspekt ist die Chomsky-Hierarchie, die formale Sprachen in Klassen bzw. Sprachfamilien unterteilt. Eine Sprachfamilie oder Klasse von Sprachen ist eine Menge von Sprachen. Dabei ist die Zugehörigkeit zu einer dieser Klassen von der Erzeugbarkeit durch reguläre, kontextfreie, kontextsensitive Grammatiken oder endliche Automaten, Kellerautomaten, Turingmaschinen mit linearer Bandbeschränkung, unbeschränkte Turingmaschinen abhängig.

Mit REG, LIN, CF, CS und RE werden jeweils die Sprachfamilien der Sprachen bezeichnet, die durch reguläre, lineare, kontextfreie (context-free), kontextsensitive (context-sensitive) und unbeschränkte Grammatiken erzeugt werden. Die Familie FIN bezeichnet die endlichen Sprachen.

Es gelten folgende Inklusionen:

$$FIN \subset REG \subset LIN \subset CF \subset CS \subset RE.$$

Folgende Tabelle zeigt die Chomsky Hierarchie entsprechend den Inklusionen zwischen den Sprachfamilien:

Typ	Grammatikart	Automatenart
Typ 0	unbeschränkte Grammatik	Turingmaschine
Typ 1	kontextsensitive Grammatik	Turingmaschine mit linearer Bandbeschränkung
Typ 2	kontextfreie Grammatik	Kellerautomat
Typ 3	reguläre Grammatik	endlicher Automat.

Definition 2.2.17. *Kontextfreie Matrix-Grammatik*

Eine kontextfreie Matrix-Grammatik wird durch ein Tupel

$$G = (N, \Sigma, S, M)$$

beschrieben. Dabei sind N, S und Σ von den formalen Grammatiken bekannt. M ist eine endliche Menge von Sequenzen der Form $(A_1 \longrightarrow v_1, \ldots, A_n \longrightarrow v_n)$ mit $n \geq 1$. Dabei sind $A_1 \longrightarrow v_1, \ldots, A_n \longrightarrow v_n$ mit $A_i \in N$, $v_i \in (N \cup \Sigma)^*$ und $1 \leq i \leq n$ kontextfreie Regeln über $N \cup \Sigma$. Für $x \in (N \cup \Sigma)^*$ werden die Produktionsregeln eines Elements $m \in M$ mit $m = (p_1, \ldots, p_n)$ in genau der aufgeführten Reihenfolge ausgeführt. Man schreibt für das Wort y, das man durch Ausführen von m erhält, $x \Longrightarrow_G y$ und sagt, dass es von x direkt abgeleitet wurde. Die durch die kontextfreien Matrix-Grammatiken beschriebene Sprachfamilie wird mit MAT^{ε} abgekürzt. Der Exponent ε bedeutet, dass auch Produktionsregeln $p : u \longrightarrow v$ mit $v = \varepsilon$ erlaubt sind.

3 Watson-Crick-Automaten

In diesem Kapitel werden die verschiedenen Ausprägungen von Watson-Crick-Automaten erklärt. Zuerst wird jedoch ein einfaches abstraktes DNA-Modell erläutert, welches dazu dient, die Elemente für die Bänder, auf dem die Watson-Crick-Automaten lesen, bereitzustellen. Die Definitionen sind [PRS98] entnommen.

3.1 Ein einfaches DNA-Modell

Ausgehend von den molekularbiologischen Grundlagen kann ein formales Modell der DNA definiert werden, die Watson-Crick-Domain. Das Modell beschreibt die Sekundärstruktur der DNA. Dabei wird von DNA-Doppelsträngen und der Komplementarität von Nukleotiden abstrahiert.

Es sei Σ ein Alphabet und $\rho \subseteq \Sigma \times \Sigma$ eine binäre symmetrische Relation, ρ wird auch die Komplementrelation genannt. Die Elemente $(x_1, x_2) \in \Sigma^* \times \Sigma^*$ werden mit $\begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$ und $\Sigma^* \times \Sigma^*$ mit $\begin{pmatrix} \Sigma^* \\ \Sigma^* \end{pmatrix}$ notiert. Dementsprechend ist die Konkatenation zweier Paare mit

$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} x_1 y_1 \\ x_2 y_2 \end{pmatrix}$$

gegeben.

Definition 3.1.1. *Watson-Crick-Domain* $WK_\rho(\Sigma)$

Die Menge

$$WK_\rho(\Sigma) = \left[\begin{matrix} \Sigma \\ \Sigma \end{matrix} \right]_\rho^*$$

mit

$$\left[\begin{matrix} \Sigma \\ \Sigma \end{matrix} \right] = \{ \left[\begin{matrix} a \\ b \end{matrix} \right] \mid a, b \in \Sigma, (a, b) \in \rho \}$$

wird Watson-Crick-Domain genannt.

Die Elemente $\begin{bmatrix} a_1 \\ b_1 \end{bmatrix}, \ldots, \begin{bmatrix} a_1 \\ b_1 \end{bmatrix} \in WK_\rho(\Sigma)$ können durch $\begin{bmatrix} w_1 \\ w_2 \end{bmatrix}$ mit $w_1 = a_1 \ldots a_n$ und $w_2 = b_1 \ldots b_n$ ersetzt werden.

Die Elemente $\begin{bmatrix} w_1 \\ w_2 \end{bmatrix} \in WK_\rho(\Sigma)$ werden Moleküle oder doppelsträngige Sequenzen genannt und die Wörter w_1, w_2 werden jeweils oberer und unterer Strang genannt.

Nach Definition ist das Molekül $\begin{bmatrix} \varepsilon \\ \varepsilon \end{bmatrix}$ in $WK_\rho(\Sigma)$ enthalten.

Für zwei Moleküle $\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}$ ist die Konkatenation definiert durch

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} x_1 y_1 \\ x_2 y_2 \end{bmatrix} \in WK_\rho(\Sigma).$$

An dieser Stelle sei noch der Unterschied zwischen den Notationen $\begin{pmatrix} x \\ y \end{pmatrix}$ und $\begin{bmatrix} x \\ y \end{bmatrix}$ hervorgehoben. Während ersteres nur eine andere Notation für das Paar (x, y) ist, bedeutet zweiteres, dass $(x, y) \in \rho$ gilt und somit ein Molekül ist.

Für alle $\begin{pmatrix} w_1 \\ w_2 \end{pmatrix} \in WK_\rho(\Sigma)$ folgt außerdem aus der Definition, dass $|w_1| = |w_2|$.

In den nächsten Abschnitten werden Watson-Crick-Automaten vorgestellt, die Elemente von $WK_\rho(\Sigma)$ einlesen.

3.2 Endliche Watson-Crick-Automaten

Endliche Watson-Crick-Automaten sind das Pendant zu den endlichen Automaten in der Automatentheorie. Im Gegensatz zu den endlichen Automaten, welche auf einem linearen Eingabeband arbeiten, lesen die Watson-Crick-Automaten Elemente von einem Watson-Crick-Band. Dabei liest ein Kopf des Automaten auf dem oberen und ein Kopf auf dem unteren Strang des Bandes. Namensgebend sind hierbei die Elemente aus der Watson-Crick-Domain $\mathrm{WK}_\rho(\Sigma)$, die das Watson-Crick Band repräsentieren.

Definition 3.2.1. *Endliche Watson-Crick-Automaten*

Ein endlicher Watson-Crick-Automat M wird durch ein Tupel

$$M = (Q, \Sigma, \rho, q_0, \delta, F)$$

mit

Q der Menge der Zustände,

Σ dem Bandalphabet,

ρ der Komplementrelation,

q_0 dem Anfangszustand,

δ der Übergangsfunktion und

F der Menge der Endzustände

beschrieben.

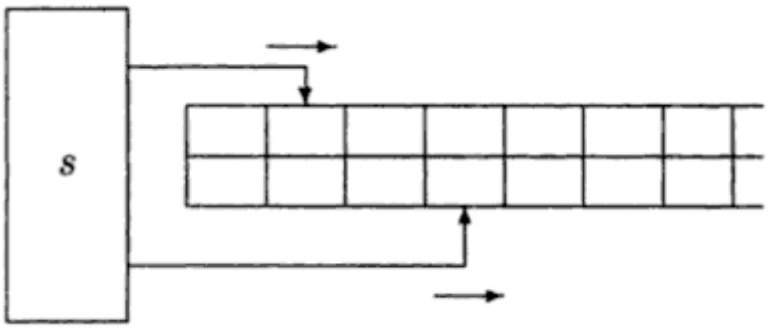

Abbildung 3.1: Schema eines endlichen Watson-Crick-Automaten. Abbildung aus [PRS98] entnommen.

Dabei gilt für die beiden endlichen Mengen Σ und Q: $\Sigma \cap Q = \emptyset$. Die Komplementrelation $\rho \subseteq \Sigma \times \Sigma$ ist eine symmetrische Relation über dem Bandalphabet Σ. Man nennt $(a, b) \in \rho$ mit $a, b \in \Sigma$ komplementär. Hierbei ist jedoch nicht Komplementarität im klassischen Sinne gemeint, denn es können zwei beliebige Elemente aus Σ komplementär sein. Ein Watson-Crick-Automat hat nur einen Anfangszustand $q_0 \in Q$, jedoch mindestens einen Endzustand. Die Übergangsfunktion

$$\delta : Q \times \begin{pmatrix} \Sigma^* \\ \Sigma^* \end{pmatrix} \longmapsto 2^Q$$

bildet Tripel aus $Q \times \Sigma^* \times \Sigma^*$ auf Teilmengen aus der Potenzmenge 2^Q ab und ist nur für endlich viele Tripel $(q, w_1, w_2) \in Q \times \Sigma^* \times \Sigma^*$ definiert:

$$\delta(q, \begin{pmatrix} w_1 \\ w_2 \end{pmatrix}) \neq \emptyset.$$

Ist der Watson-Crick-Automat im Zustand q, der Folgezustand q' und der obere sowie der untere Lesekopf lesen das Wort w_1 bzw. w_2, dann kann dieser Übergang durch

$$q' \in \delta(q, \begin{pmatrix} w_1 \\ w_2 \end{pmatrix}$$

dargestellt werden. Genauso kann der Konfigurationsübergang jedoch auch durch die Produktionsregel

$$q \begin{pmatrix} w_1 \\ w_2 \end{pmatrix} \longrightarrow \begin{pmatrix} w_1 \\ w_2 \end{pmatrix} q'$$

ausgedrückt werden. Hierbei wird der Watson-Crick-Automat nicht mehr durch das Tupel $M = (Q, \Sigma, \rho, q_0, F, \delta)$, sondern durch $M = (Q, \Sigma, \rho, q_0, F, P)$ beschrieben. Dabei ist P die Menge aller Produktionsregeln, welche durch Tupel beschrieben werden:

$$q \begin{pmatrix} w_1 \\ w_2 \end{pmatrix} \longrightarrow \begin{pmatrix} w_1 \\ w_2 \end{pmatrix} q' \Longleftrightarrow (q \begin{pmatrix} w_1 \\ w_2 \end{pmatrix}, \begin{pmatrix} w_1 \\ w_2 \end{pmatrix} q') \in P$$

Für die Zustände $q, q' \in Q$ und die Wortpaare $\begin{pmatrix} x_1 \\ x_2 \end{pmatrix}, \begin{pmatrix} y_1 \\ y_2 \end{pmatrix}, \begin{pmatrix} z_1 \\ z_2 \end{pmatrix} \in \begin{pmatrix} \Sigma^* \\ \Sigma^* \end{pmatrix}$, welche Watson-Crick-komplementär sind, d.h. $\begin{bmatrix} x_1 y_1 z_1 \\ x_2 y_2 z_2 \end{bmatrix} \in WK_\rho(\Sigma)$, ist dementsprechend ein Konfigurationsübergang im Watson-Crick-Automat durch

$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} q \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} \begin{pmatrix} z_1 \\ z_2 \end{pmatrix} \Longrightarrow_M \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} q' \begin{pmatrix} z_1 \\ z_2 \end{pmatrix} \text{ gdw. } q' \in \delta(q, \begin{pmatrix} y_1 \\ y_2 \end{pmatrix})$$

definiert. Ein Übergang ist auch möglich, wenn $\begin{pmatrix} x_1 y_1 \\ x_2 y_2 \end{pmatrix} \notin WK_\rho(\Sigma)$ gilt. Die Wörter auf dem oberen und unteren Strang werden akzeptiert, wenn $\begin{bmatrix} x_1 y_1 z_1 \\ x_2 y_2 z_2 \end{bmatrix} \in WK_\rho(\Sigma)$, alle Wörter auf den beiden Strängen komplett gelesen wurden und gleichzeitig ein Endzustand $q_f \in F$ erreicht ist. Die transitive und reflexive Hülle der Relation $\Longrightarrow_M$ wird durch das Symbol $\Longrightarrow_M^*$ formalisiert. Somit kann die durch den endlichen Watson-Crick-Automaten beschriebene

Sprache

$$L_u(M) = \{w_1 \in \Sigma^* \mid q_0 \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} \} \Longrightarrow_M^* \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} q_f\}$$

mit $q_0 \in Q,\ q_f \in F$ und $w_2 \in \Sigma^*, \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} \in WK_\rho(\Sigma)$ angegeben werden. Dabei werden jedoch nur die Wörter auf dem oberen Strang berücksichtigt. (Der Index u steht hierbei für *upperstrand*)

Außerdem gibt es eine weitere wichtige Sprache, die durch Watson-Crick-Automaten beschrieben wird. Für einen Watson-Crick-Automaten $M = (Q, \Sigma, \rho, q_0, F, P)$ kann eine eindeutige Beschriftung $e : P \longmapsto Lab$ der Übergangsregeln angegeben werden, wobei $Lab = \{p \mid p : u \longrightarrow v \in P\}$ die Menge der Beschriftungen ist. Somit existiert für jede mögliche Berechnung $\sigma : q_0 w \Longrightarrow_M^* wq_f$ in dem Watson-Crick-Automaten mit $w \in WK_\rho(\Sigma)$, $q \in Q$ und $q_f \in F$ ein Kontrollwort $e(\sigma) = e(p_1)e(p_2)...e(p_n)$. Dabei sind $p_1, ..., p_n \in Lab$ die Beschriftungen der Produktionsregeln, die zur Akzeptanz des Wortes w führen. Demnach ist das Kontrollwort $e(\sigma)$ eine Sequenz von Beschriftungen für die Produktionsregeln, die zur Akzeptanz eines Wortes w durch den zugehörigen Watson-Crick-Automaten führen. Die zugehörige Sprache für die Menge der Kontrollwörter ist

$$L_{ctr}(M) = \{e(\sigma) \mid \sigma :\ q_0 w \Longrightarrow_M^* wq_f\}$$

mit $q_0, q_f \in Q$, $q_f \in F$ und $w \in WK_\rho(\Sigma)$. Die Sprache der Kontrollwörter kann im DNA-Computing von großem Nutzen sein, da alle Probleme letztendlich über dem reduzierten Alphabet der vier verschiedenen Nukleotide kodiert werden. So bietet die Sprache der Kontrollwörter eine weitere Beschreibungsmöglichkeit.

3.3 Endliche Watson-Crick Transducer

Während bisher nur von Automaten ausgegangen wurde, die auf einem Watson-Crick-Band lesen, ändert das folgende Modell diese Betrachtungsweise und orientiert sich dabei stark an endlichen Transducern.

Endliche Watson-Crick-Transducer bauen auf dem Modell der endlichen Watson-Crick-Automaten auf. Es wird jedem Übergang im endlichen Watson-Crick-Automat eine Ausgabe zugewiesen und auf ein Ausgabeband geschrieben. Das Ausgabeband ist ein einsträngiges Band auf dem Symbole des Ausgabealphabets stehen.

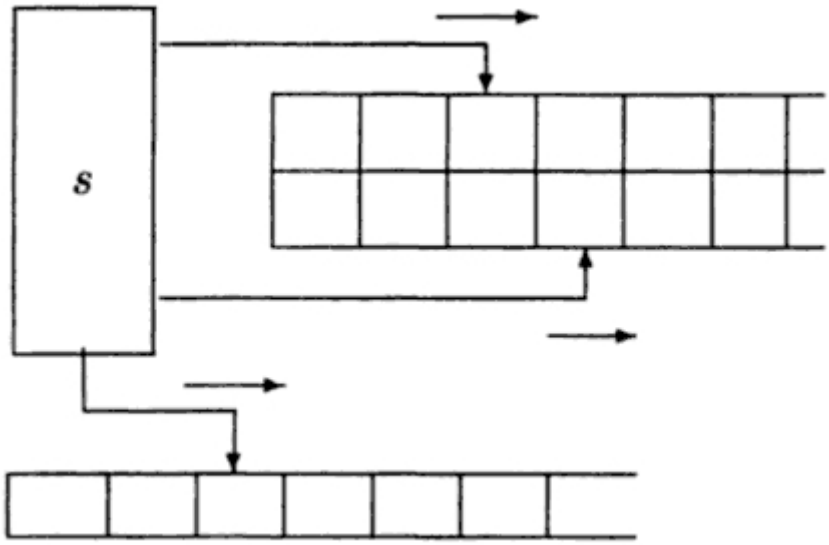

Abbildung 3.2: Schema eines endlichen Watson-Crick-Transducers. Abbildung aus [PRS98] entnommen.

Definition 3.3.1. *Endliche Watson-Crick-Transducer*

Ein endlicher Watson-Crick-Transducer wird durch ein Tupel

$$M = (Q, \Sigma_I, \rho_I, \Sigma_O, q_0, \delta, F)$$

mit

Q der Menge der Zustände,

Σ_I dem Eingabebandalphabet,

ρ_I der Komplementrelation des Eingabebandes,

Σ_O dem Ausgabebandalphabet

q_0 dem Anfangszustand,

δ der Übergangsfunktion und

F der Menge der Endzustände

beschrieben. Für die Menge der Endzustände F, die Komplementrelation ρ_I und den Anfangszustand q_0 gilt $q_0 \in Q$, $\rho_I \subseteq V_I \times V_I$ und $F \subseteq Q$.

Die Übergangsfunktion δ des endlichen Watson-Crick-Transducers ist eine Abbildung von der Menge $Q \times \begin{pmatrix} \Sigma_I^* \\ \Sigma_I^* \end{pmatrix}$ in die Menge $2^{Q \times \Sigma_O^*}$ für die nur endliche viele Wertepaare $(q, u, v) \in Q \times \Sigma_I \times \Sigma_I$ mit $\delta(q, \begin{pmatrix} u \\ v \end{pmatrix}) \neq 0$ existieren. Eine Übergangsregel im Automaten kann nun wie folgt formalisiert werden:

$$(q', x) \in \delta(q, \begin{pmatrix} u \\ v \end{pmatrix})$$

mit $u \in \Sigma_I^*$, $v \in \Sigma_I^*$. Der Watson-Crick-Transducer ist im Zustand q, liest das Wort u auf dem oberen Strang, das Wort v auf dem unteren Strang, gibt x auf dem Ausgabeband aus und geht in den Zustand q' über.

Für $q, q' \in Q$, $u, v, u', v', a, b \in \Sigma_I^*$, $x, z \in \Sigma_O^*$ kann ein Übergang im Automaten wie folgt definiert werden: Der Watson-Crick-Transducer, welcher im Zustand q ist, auf dessen Eingabeband sich auf dem oberen Strang das Wort $u = au'$ und auf dem unteren Strang das Wort $v = bv'$ befindet und bereits z auf das Ausgabeband geschrieben hat, wechselt mit der Eingabe $\begin{pmatrix} a \\ b \end{pmatrix}$ in den Zustand q' über und schreibt das Wort x auf das Ausgabeband.

$$zq \begin{pmatrix} u \\ v \end{pmatrix} \Longrightarrow_M \begin{pmatrix} u \\ v \end{pmatrix} zxq' \begin{pmatrix} u' \\ v' \end{pmatrix} \text{ falls } u = au', v = bv' \text{und ein Übergang } (q', x) \in \delta(q, \begin{pmatrix} a \\ b \end{pmatrix}) \text{ existiert.}$$

Ein endlicher Watson-Crick-Transducer ist deterministisch, falls pro Konfigurationsübergang nur mögliche Konfiguration existiert, in die gewechselt werden kann. Die Sprache für einen Watson-Crick-Transducer mit der Eingabe $w = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix}$ sei die Menge aller Übergänge, die bei vollständig eingelesenem Band zu einem Endzustand führt.

$$L(M) = g(w) = \{x \in \Sigma_O \mid q_0 \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} \Longrightarrow_M xq_f \begin{bmatrix} \varepsilon \\ \varepsilon \end{bmatrix},\ q_f \in F\}$$

3.4 Endliche Reverse Watson-Crick Automata

Aus den biologischen Grundlagen ist bekannt, dass die DNA-Moleküle in 5'-3'-Richtung und 3'-5'-Richtung angeordnet sind. Aus diesem Grund ist es sinnvoll, auch ein Automatenmodell zu betrachten, welches diese natürlich gegebene Restriktion mit einbezieht und dessen Leseköpfe ebenfalls in entgegengesetzte Richtung laufen.

Definition 3.4.1. *Endliche Reverse Watson-Crick-Automaten*

Ein endlicher Reverse-Watson-Crick Automat resultiert aus dem Modell des endlichen Watson-Crick-Automaten. Der Unterschied besteht in der Laufrichtung der Leseköpfe und der Position der Leseköpfe, bei vollständig gelesenem Eingabeband. Der Automat wird durch das Tupel

$$M = (Q, \Sigma, \rho, q_0, \delta, F)$$

beschrieben. Im Vergleich zu den endlichen Automaten entspricht Σ ebenso dem Bandalphabet, ρ der Komplementrelation, Q der Menge der Zustände mit dem Anfangszustand q_0 und δ der Übergangsfunktion. Ein Konfigurationsübergang im Automaten sei wieder durch die bereits bekannte Relation $\Longrightarrow_M$ mit $w_1, w_2, w_1', w_2', u, v \in \Sigma^*,\ q, q' \in Q$ gegeben:

$$\begin{pmatrix} w_1 \\ w_2 v \end{pmatrix} q_0 \begin{pmatrix} u w_1' \\ w_2' \end{pmatrix} \Longrightarrow_M \begin{pmatrix} w_1 u \\ w_2 \end{pmatrix} q' \begin{pmatrix} w_1' \\ v w_2' \end{pmatrix} \text{ falls } \begin{bmatrix} w_1 u w_1' \\ w_2 v w_2' \end{bmatrix} \in WK_\rho(V), q' \in \delta(q, \begin{pmatrix} u \\ v \end{pmatrix}).$$

Die, durch einen endlichen Reverse Watson-Crick-Automaten, beschriebene Sprache

$$L_u(M) = \{w_1 \in \Sigma \mid \begin{pmatrix} \varepsilon \\ w_1 \end{pmatrix} q \begin{pmatrix} w_2 \\ \varepsilon \end{pmatrix} \Longrightarrow_M^* \begin{pmatrix} w_1 \\ \varepsilon \end{pmatrix} q_f \begin{pmatrix} \varepsilon \\ w_2 \end{pmatrix},\ q_f \in F\}$$

Ebenso wie bei den endlichen Watson-Crick-Automaten kann auch hier wieder eine Sprache $L_{ctr}(M)$ angegeben werden.

Dieses Automatenmodell ist von untergeordneter Bedeutung für die Universalitätsbetrachtungen im nächsten Kapitel, jedoch ist es einer laborpraktischen Implementierung näher als die endlichen Watson-Crick-Automaten.

4 Universalitätsbetrachtungen

4.1 Beziehungen zwischen den Watson-Crick-Familien

Betrachtet man die vorangegangene Definition der Watson-Crick-Automaten, so ist es natürlich möglich, die Automaten unter verschiedenen Restriktionen zu betrachten. Mit Hilfe dieser Restriktionen lassen sich unterschiedliche Sprachfamilien angeben, mit welchen sogar Charakterisierungen der rekursiv aufzählbaren Sprachen erreicht werden können. Die Beweise orientieren sich an [PRS98].

Definition 4.1.1. *Varianten endlicher Watson-Crick-Automaten*

Ein endlicher Watson-Crick-Automat $M = (Q, \Sigma, \rho, q_0, F, P)$ ist

(N) stateless, falls $Q = F = \{q_0\}$,

(F) all-final, falls $F = Q$,

(S) simple, falls für alle Übergänge $q \begin{pmatrix} w_1 \\ w_2 \end{pmatrix} \longrightarrow \begin{pmatrix} w_1 \\ w_2 \end{pmatrix} q' \in P$ entweder $w_1 = \varepsilon$ oder $w_2 = \varepsilon$ gilt und

(1) 1-limited, falls für alle Übergänge $q \begin{pmatrix} w_1 \\ w_2 \end{pmatrix} \longrightarrow \begin{pmatrix} w_1 \\ w_2 \end{pmatrix} q' \in P$ $|w_1 w_2| = 1$ gilt.

Definition 4.1.2. *Sprachfamilien der einzelnen Automatenarten.*

Mit

$AWK(\alpha)$, $\alpha \in \{u, ctr\}$ wird die Menge der Sprachen bezeichnet, die mittels der unbeschränkten (arbitrary) endlichen Watson-Crick-Automaten beschrieben wird,

$NWK(\alpha)$, $\alpha \in \{u, ctr\}$ wird die Menge der Sprachen bezeichnet, die mittels der stateless (N für "no state") endlichen Watson-Crick-Automaten beschrieben wird,

$FWK(\alpha)$, $\alpha \in \{u, ctr\}$ wird die Menge der Sprachen bezeichnet, die mittels der all-final (F) endlichen Watson-Crick-Automaten beschrieben wird,

$SWK(\alpha)$, $\alpha \in \{u, ctr\}$ wird die Menge der Sprachen bezeichnet, die mittels der simple (S) endlichen Watson-Crick-Automaten beschrieben wird,

$1WK(\alpha)$, $\alpha \in \{u, ctr\}$ wird die Menge der Sprachen bezeichnet, die mittels der 1-limited (1) endlichen Watson-Crick-Automaten beschrieben wird,

$NSWK(\alpha)$, $\alpha \in \{u, ctr\}$ wird die Menge der Sprachen bezeichnet, die mittels der stateless und simple (NS) endlichen Watson-Crick-Automaten beschrieben wird,

$N1WK(\alpha)$, $\alpha \in \{u, ctr\}$ wird die Menge der Sprachen bezeichnet, die mittels der stateless und 1-limited (N1) endlichen Watson-Crick-Automaten beschrieben wird,

$F1WK(\alpha)$, $\alpha \in \{u, ctr\}$ wird die Menge der Sprachen bezeichnet, die mittels der all-final und 1-limited (F1) endlichen Watson-Crick-Automaten beschrieben wird,

$FSWK(\alpha)$, $\alpha \in \{u, ctr\}$ wird die Menge der Sprachen bezeichnet, die mittels der all-final und simple (FS) endlichen Watson-Crick-Automaten beschrieben wird.

Die stateless endlichen Watson-Crick-Automaten werden nur durch ein Tupel $M = (\Sigma, \rho, \delta)$ bzw. $M = (\Sigma, \rho, P)$ beschrieben.

Mit obiger Definition der einzelnen Sprachfamilien sind bereits diverse Beziehungen gegeben.

So ist zunächst offensichtlich, dass jede der oben genannten Sprachfamilien in der Familie der unbeschränkten endlichen Watson-Crick-Automaten enthalten sein muss.

Lemma 4.1.1. $XWK(\alpha) \subseteq AWK(\alpha)$, $\alpha \in \{u, ctr\}$, $X \in \{N, F, S, 1, NS, N1, FS, F1\}$.

Ebenso besteht eine Inklusion zwischen $NWK(\alpha)$, $NSWK(\alpha)$ bzw. $N1WK(\alpha)$ und $FWK(\alpha)$, $FSWK(\alpha)$ bzw. $F1WK(\alpha)$, da jeder stateless endliche Watson-Crick-Automat auch ein all-final endlicher Watson-Crick-Automat ist.

Lemma 4.1.2. $NWK(\alpha) \subseteq FWK(\alpha), NSWK(\alpha) \subseteq FSWK(\alpha), N1WK(\alpha) \subseteq F1WK(\alpha)$, $\alpha \in \{u, ctr\}$.

Genauso ergeben sich die folgenden Inklusionen, da jeder stateless und all-final simple bzw. jeder stateless und all-final 1-limited endliche Watson-Crick-Automat offensichtlich auch ein stateless bzw. 1-limited endlicher Watson-Crick-Automat ist.

Lemma 4.1.3. $XSWK(\alpha) \subseteq SWK(\alpha), X1WK(\alpha) \subseteq 1WK(\alpha)$, $X \in \{N, F\}$, $\alpha \in \{u, ctr\}$.

Weiterhin ist ersichtlich, dass die regulären Sprachen REG in $1WK(\alpha)$ enthalten sind. Es kann jeder 1-limited endliche Watson-Crick-Automat die Sprache eines beliebigen endlichen Automaten akzeptieren, indem er nur auf dem oberen Strang liest und der untere Lesekopf nicht bewegt wird.

Lemma 4.1.4. *$REG \subseteq 1WK(\alpha), \alpha \in \{u, ctr\}$.*

Beweis. Angenommen es existiert ein endlicher Automat

$$A = (Q, \Sigma, q_0, \delta, F),$$

dann kann ein 1-limited endlicher Watson-Crick-Automat

$$M = (Q, \Sigma, id_\Sigma, q_0, \delta', F)$$

konstruiert werden, für den $L(M) = L(A)$ gilt. Die Übergangsfunktion δ' des Watson-Crick-Automaten M sei wie folgt definiert:

Es ist ein Übergang $q' \in \delta'(q, \begin{pmatrix} a \\ a \end{pmatrix})$ mit $q, q' \in Q$ in M vorhanden, falls der Übergang $q' \in \delta(q, a)$ im endlichen Automaten A existiert.

Somit entspricht die Sprache $L_u(M) = \{w_1 \in \Sigma^* \mid q_0 \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} \Longrightarrow_M^* \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} q_f\}$ der Sprache $L(A)$ des endlichen Automaten.

□

Es ist aus [Ku05] bekannt, dass die endlichen Watson-Crick-Automaten nur Sprachen innerhalb der kontextsensitiven Sprachfamilie akzeptieren. Es wird gezeigt, dass eine linear bandbeschränkte Turingmaschine einen Watson-Crick-Automat simulieren kann.

Lemma 4.1.5. *$AWK(\alpha) \subseteq CS, \alpha \in \{u, ctr\}$*

Außerdem ist ersichtlich, dass eine Sprache in der Familie $XWK(ctr)$ über eine Kodierung in eine Sprache der Familie $XWK(u)$ überführt werden kann. Es ist aus der Definition der endlichen Watson-Crick-Automaten bekannt, dass die Kontrollwörter $e(\sigma)$ einer Berechnung σ im Automaten eine Kodierung der Übergänge ist. Entsprechend kann beispielsweise eine Kodierung konstruiert werden, die jedem Übergang das Wort zuweist, das mit diesem Übergang gelesen wird.

Lemma 4.1.6. *Jede Sprache in $XWK(u)$ ist eine Kodierung mindestens einer Sprache in $XWK(ctr)$, $X \in \{A, N, F, S, 1, NS, N1, FS, F1\}$.*

Mit folgendem Lemma wird die Komplexität der unbeschränkten endlichen Watson-Crick-Automaten etwas aufgelöst, indem auf die 1-limited endlichen Watson-Crick-Automat zurückgegriffen wird und nachfolgend die Gleichheit von $1WK(u)$ und $AWK(u)$ gezeigt.

Lemma 4.1.7. $AWK(u) \subseteq 1WK(u)$.

Beweis. Angenommen es existiert ein unbeschränkter endlicher Watson-Crick-Automat

$$M = (Q, \Sigma, \rho, q_0, P, F).$$

Für M kann ein 1-limited endlicher Watson-Crick-Automat

$$M' = (Q', \Sigma, \rho, q_0, P', F)$$

konstruiert werden, für den $L(M') = L(M)$ gilt. Die Übergangsregeln P' seien wie folgt definiert:

Existiert eine Übergangsregel

$$p \in P : q \begin{pmatrix} a_1 a_2 \dots a_n \\ b_1 b_2 \dots b_m \end{pmatrix} \longrightarrow \begin{pmatrix} a_1 a_2 \dots a_n \\ b_1 b_2 \dots b_m \end{pmatrix} q'$$

mit $n \geq 0, m \geq 0$ und $n + m \geq 2$ im Automaten M, dann werden zu P' folgende Regeln hinzugefügt:

In den Zustandsübergängen von q zu $q_{p,1}$ und $q_{p,i}$ zu $q_{p,i+1}$ mit $1 \leq i \leq n-1$ liest der Automat jeweils das Symbol a_i auf dem oberen Strang ein.

Es wird das erste Symbol auf dem oberen Strang gelesen:

$$p_{p,1} : q \begin{pmatrix} a_1 \\ \varepsilon \end{pmatrix} \longrightarrow \begin{pmatrix} a_1 \\ \varepsilon \end{pmatrix} q_{p,1}.$$

Alle weiteren Symbole auf dem oberen Strang werden eingelesen:

$$p_{p,i+1} : q_i \begin{pmatrix} a_{i+1} \\ \varepsilon \end{pmatrix} \longrightarrow \begin{pmatrix} a_{i+1} \\ \varepsilon \end{pmatrix} q_{p,i+1}, 1 \leq i \leq n-1.$$

Folglich wird in den Zustandsübergängen von $q_{p,n}$ zu $q_{p,1}$, $q'_{p,i}$ zu $q'_{p,i+1}$ mit $1 \leq i \leq m-1$ jeweils das Symbol b_i auf dem unteren Band gelesen.

$$p'_{p,1} : q_{p,n} \begin{pmatrix} \varepsilon \\ b_1 \end{pmatrix} \longrightarrow \begin{pmatrix} \varepsilon \\ b_1 \end{pmatrix} q'_{p,1}.$$

$$p'_{p,i+1} : q'_{p,i} \begin{pmatrix} \varepsilon \\ b_i \end{pmatrix} \longrightarrow \begin{pmatrix} \varepsilon \\ b_i \end{pmatrix} q'_{p,i+1}, 1 \leq i \leq m-2.$$

$$p'_{p,m} : q'_{p,m-1} \begin{pmatrix} \varepsilon \\ b_m \end{pmatrix} \longrightarrow \begin{pmatrix} \varepsilon \\ b_m \end{pmatrix} q'$$

Zusammenfassend wird jede Transition aus dem unbeschränkten endlichen Watson-Crick-Automat M durch $m+n$ Transitionen in M' simuliert, die jeweils nur ein Symbol lesen. Es gilt $L(M') = L(M)$. □

Aus der Transitivität der Teilmengenrelation $\subseteq$, insbesondere aus den Inklusionen $1WK(u) \subseteq SWK(u)$, $SWK(u) \subseteq AWK(u)$ und Lemma 4.1.7 folgt $1WK(u) = AWK(u)$. Weiterhin lässt sich analog aus $1WK(u) \subseteq SWK(u)$, $SWK(u) \subseteq AWK(u)$ und Lemma 4.1.7 die Gleichheit von $AWK(u)$ und $SWK(u)$ folgern.

Korollar 4.1.1. *$AWK(u) = 1WK(u) = SWK(u)$.*

Das Korollar 4.1.1 kann nicht auf die Sprachfamilien der Kontrollwörter $AWK(ctr)$, $SWK(ctr)$, $1WK(ctr)$ übertragen werden, da bisher noch nicht bekannt ist, ob die Inklusion Lemma 4.1.7 gleichermaßen für $AWK(ctr)$ und $1WK(ctr)$ gilt. Der Grund dafür ist, dass mit dem Hinzufügen neuer Übergangsregeln in M' die Menge P und somit auch die Menge Lab (Def. siehe Abschnitt 3.2) und gleichzeitig auch σ (Def. siehe Abschnitt 3.2) für das akzeptierte Wort geändert werden. Somit gilt $L_{ctr}(M) \neq L_{ctr}(M')$.

Am Anfang des Abschnittes 3.2 wurde bereits erwähnt, dass die endlichen Watson-Crick-Automaten das Pendant zu den endlichen Automaten sind und in Lemma 4.1.4 wurde gezeigt, dass die regulären Sprachen in der Sprachfamilie der 1-limited endlichen Watson-Crick-Automaten enthalten sind. Nun arbeiten die endlichen Automaten, im Gegensatz zu den endlichen Watson-Crick-Automaten, nur mit einem Lesekopf, jedoch existieren gleichermaßen endliche Automaten mit zwei Leseköpfen, die auf einem Eingabeband simultan lesen können.

Definition 4.1.3. *Endliche Two-Head Automaten (TH-Automaten)*

Ein endlicher Two-Head Automat B wird durch ein Tupel

$$B = (Q, \Sigma, q_0, \delta, F)$$

mit

Q der Menge der Zustände,

Σ dem Bandalphabet,

q_0 dem Anfangszustand,

δ der Übergangsfunktion und

F der Menge der Endzustände

beschrieben.

Die Übergangsfunktion ist eine Abbildung

$$\delta : Q \times (\Sigma \cup \{\varepsilon\}) \times (\Sigma \cup \{\varepsilon\}) \longmapsto 2^Q$$

Mit Hilfe der Konfigurationsübergangsrelation $\Longrightarrow_B$ und ihrer reflexiven und transitiven Hülle $\Longrightarrow_B^*$

$$(w_1, w_2)q(aw_1', bw_2') \Longrightarrow_B (w_1a, w_2b)q'(w_1', w_2') \text{ falls } q' \in (q, a, b)$$

sei ein Übergang mit $w_1, w_2, w_1', w_2' \in \Sigma^*$, $a, b \in V \cup \{\varepsilon\}$ zwischen den Konfigurationen gegeben. Somit kann eine Sprache für den endlichen TH-Automat B mit

$$L(B) = \{w \in \Sigma^* \mid q_0(w, w) \Longrightarrow_B^* (w, w)q_f,\ q_f \in F\}$$

angegeben werden.

TH ist die Bezeichnung für die Sprachfamilie der endlichen TH-Automaten.

Folglich eröffnet sich die Frage, wie sich die Sprachfamilie TH in die Hierarchie der Watson-Crick-Sprachfamilien einordnet.

Lemma 4.1.8. $TH = 1WK$.

Es ist offensichtlich, dass die Sprachklasse TH der Sprachklasse $1WK(u)$ entspricht, da jeder endliche TH-Automat durch einen 1-limited endlichen Watson-Crick-Automaten mit $\rho = id_\Sigma$ (id_Σ ist die Identitätsrelation über dem Alphabet Σ) beschrieben werden kann. Im Gegensatz dazu kann jeder 1-limited endliche Watson-Crick-Automat mit beliebiger Komplementrelation durch einen endlichen TH-Automaten beschrieben werden. Dabei rät der TH-Automat ein Symbol b komplementär zum aktuellen Symbol a auf dem Band und geht nur dann zum nächsten Zustand über, wenn der jeweilige Lesekopf des 1-limited endlichen Watson-Crick-Automaten ebenso das Symbol b lesen würde.

Lemma 4.1.9. $NSWK(u) \subseteq REG$.

Beweis. Angenommen, es existiert ein simple, stateless endlicher Watson-Crick-Automat

$$M = (\Sigma, \rho, P)$$

mit $p_1, \ldots, p_n \in P$. Es gilt

$$p_1 : \begin{pmatrix} u_1 \\ v_1 \end{pmatrix} \text{ mit } u_1 = \varepsilon \text{ oder } v_1 = \varepsilon,$$

$$\vdots$$

$$p_n : \begin{pmatrix} u_n \\ v_n \end{pmatrix} \text{ mit } u_n = \varepsilon \text{ oder } v_n = \varepsilon.$$

Die Funktion des unteren Lesekopfes ist, den unteren Strang einzulesen und die Komplementarität zum oberen Strang zu überprüfen. Die Transitionen, mit Hilfe derer auf dem unteren Strang gelesen wird, stellen nur eine zusätzliche Möglichkeit dar, Wörter auszusortieren.

Somit kann die durch M akzeptierte Sprache $L(M)$ nur von der Form $L(M) = (u_1^* u_2^* \ldots u_n^*)^*$ sein. Dementsprechend gilt $L(M) \subseteq REG$. □

Bisher ist aus Lemma 4.1.4 bekannt, dass $REG \subseteq 1WK(\alpha)$, $\alpha \in \{u, ctr\}$. Diese Inklusion kann noch für $F1WK(u)$ verfeinert werden.

Lemma 4.1.10. $REG \subseteq F1WK(u)$.

Beweis. Ausgehend von einem endlichen Automaten $A = (Q, \Sigma, q_0, \delta, F)$ kann ein all-final, 1-limited endlicher Watson-Crick-Automat

$$M = (Q', \Sigma, \rho, q_0, \delta', F)$$

mit

$$\rho = \{(a, a) \mid a \in \Sigma\},$$
$$Q' = Q \cup \{q_f\} \text{ für } q_f \notin Q,$$
$$\delta'(q, \begin{pmatrix} a \\ \varepsilon \end{pmatrix}) = \delta(q, a) \cup Final(q, a), q \in Q, a \in \Sigma,$$

$$\text{wobei } Final(q, a) = \begin{cases} \{q_f\}, & \text{falls } \delta(q, a) \cap F \neq \emptyset, \\ \emptyset, & \text{sonst,} \end{cases}$$

$$\delta'(q_f, \begin{pmatrix} \varepsilon \\ a \end{pmatrix}) = \{q_f\} \text{ mit } a \in \Sigma,$$

$$\delta'(q, \begin{pmatrix} u \\ v \end{pmatrix}) = \emptyset \text{ mit } u, v \in \Sigma \text{ sonst,}$$

konstruiert werden.

Der Automat M nutzt die Eigenschaft von endlichen Watson-Crick-Automaten aus, dass nur vollständige Moleküle akzeptiert werden und arbeitet sonst wie der endliche Automat A.

Mit $\delta'(q, \begin{pmatrix} a \\ \varepsilon \end{pmatrix}) = \delta(q,a) \cup Final(q,a), q \in Q, a \in \Sigma$ und $Final(q,a) = \begin{cases} \{q_f\}, & \text{falls } \delta(q,a) \cap F \neq \emptyset \\ \emptyset, & \text{sonst,} \end{cases}$
wird der gesamte obere Strang eingelesen und zu q_f gewechselt, falls der Zustand q in A ein Endzustand ist.

Entsprechend wird mittels $\delta'(q_f, \begin{pmatrix} \varepsilon \\ a \end{pmatrix}) = \{q_f\}$ mit $a \in \Sigma$ der gesamte untere Strang eingelesen und das vollständige Molekül akzeptiert. Es gilt $L_u(M) = L(A)$ □

Es kann gezeigt werden, dass alle Sprachen in $NWK(\alpha), \alpha \in \{u, ctr\}$ und $N1WK(u)$ gewissen Restriktionen unterliegen, die Aufschluss über die unterschiedlichen Sprachen geben, die in diesen beiden Sprachfamilien enthalten sind.

Lemma 4.1.11. *Falls $L \in NWK(\alpha), \alpha \in \{u, ctr\}$, dann ist L von der Form $L = L^+$.*

Beweis. Angenommen es existiert ein stateless endlicher Watson-Crick-Automat

$$M = (\Sigma, \rho, P)$$

und $u = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}, v = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} \in WK_\rho(\Sigma)$ mit

$$q_0 \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \Longrightarrow_M \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} q_0 \text{ und } p_u : \begin{pmatrix} u_1 \\ u_2 \end{pmatrix} \in P$$

$$q_0 \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} \Longrightarrow_M \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} q_0 \text{ und } p_v : \begin{pmatrix} v_1 \\ v_2 \end{pmatrix} \in P$$

kann von M gelesen werden. Dementsprechend kann ebenso $uv = \begin{bmatrix} u_1 v_1 \\ u_2 v_2 \end{bmatrix} \in WK_\rho(\Sigma)$ mit

$$q_0 \begin{bmatrix} u_1 v_1 \\ u_2 v_2 \end{bmatrix} \Longrightarrow_M \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} q_0 \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} \Longrightarrow_M \begin{bmatrix} u_1 v_1 \\ u_2 v_2 \end{bmatrix} q_0 \text{ und } p_u : \begin{pmatrix} u_1 \\ u_2 \end{pmatrix}, p_v : \begin{pmatrix} v_1 \\ v_2 \end{pmatrix} \in P$$

von M gelesen werden. Es kann folglich jede Transition beliebig wiederholt und verkettet werden und demnach auch jedes akzeptierte Wort. Demzufolge gilt $L_\alpha(M)^+ \subseteq L_\alpha(M), \alpha \in \{u, ctr\}$. Nach Definition gilt $L_\alpha(M) \subseteq L(M)^+, \alpha \in \{u, ctr\}$ und somit $L_\alpha(M) = L_\alpha(M)^+, \alpha \in \{u, ctr\}$. □

Lemma 4.1.12. *Für alle Sprachen $L \in N1WK(u)$ gilt $L = \Sigma^+$ für ein Alphabet Σ.*

Beweis. Angenommen es existiert ein stateless, 1-limited endlicher Watson-Crick-Automat

$$M = (\Sigma, \rho, P).$$

Dann wird pro Übergang jeweils nur ein Symbol auf dem oberen oder unteren Strang gelesen und somit kann analog zu Lemma 4.1.11 jede Konkatenation der Symbole in Σ akzeptiert werden. Damit gilt $\Sigma^+ \subseteq L_u(M)$, $L_u(M) \subseteq \Sigma^+$ und somit $L_u(M) = \Sigma^+$. □

Korollar 4.1.2. $REG - XWK(\alpha) \neq \emptyset, \alpha \in \{u, ctr\}, X \in \{N, NS\}$.

Korollar 4.1.3. $F1WK(u) - NWK(u) \neq \emptyset$.

Beweis. Nach Lemma 4.1.10 ist bekannt, dass die Sprache $ab^* \in REG$ in $F1WK(u)$ liegt. Jedoch unterliegt sie nicht der Eigenschaft aus Lemma 4.1.11. Somit gilt $ab^* \notin NWK(u), NSWK(u)$. □

Mit diesen beiden Korollaren ist gezeigt, dass $FWK(u)$ und $F1WK(u)$ Sprachen enthalten, die nicht in $NWK(u)$ bzw. $NSWK(u)$ liegen.

Korollar 4.1.4. *Die Inklusionen $NWK(u) \subset FWK(u)$ und $NSWK(u) \subset F1WK(u)$ sind strikte Inklusionen.*

Wichtig ist noch zu untersuchen, ob die Inklusion in Lemma 4.1.5 echt oder strikt ist.

Dazu wird folgendes Lemma bewiesen.

Lemma 4.1.13. *Jede unäre Sprache in $AWK(u)$ ist regulär.*

Beweis. Es wird von einem unbeschränkten endlichen Watson-Crick-Automaten

$$M = (Q, \Sigma, \rho, q_0, \delta, F)$$

ausgegangen. Übergangsregeln von der Form $q' \in \delta(\begin{pmatrix} a^i \\ w \end{pmatrix})$, $i \in N$, wobei w an beliebiger Stelle ein nicht zu a komplementäres Symbol w_j, $(a, w_j) \notin \rho$, $1 \leq j \leq |w|$ enthält, können

ignoriert werden. Diese Übergangsregeln lesen Wörter ein, die nie akzeptiert werden könnten, da auf dem oberen Band ausschließlich Wörter a^k, $k \in N$ gelesen werden. Dementsprechend wird angenommen, dass $(a, w_j) \in \rho$ gilt. Für M kann eine lineare Grammatik

$$G = (Q, \{a, b\}, P, S)$$

mit

$$P = \{q \longrightarrow a^i q' b^j \mid q' \in \delta(\begin{pmatrix} a^i \\ w \end{pmatrix}), q, q' \in Q, i \geq 0, j = |w|\}$$
$$\cup\ \{q \longrightarrow a^i b^j \mid \delta(\begin{pmatrix} a^i \\ w \end{pmatrix}) \cap F \neq \emptyset, q \in Q, i \geq 0, j = |w|\}$$

konstruiert werden.

Aus der Linearität der Grammatik G folgt, dass die beschriebene Sprache $L(G)$ linear und zugleich semilinear ist, da aus der Linearität die Semilinearität einer Sprache folgt. Dementsprechend muss $\Psi_{\{a,b\}}(L(G))$ ebenso linear sein. Betrachtet man nun

$$L = \{a^n b^n \mid n \geq 1\},$$

so ist einerseits aus der Sprachtheorie bekannt, dass L eine kontextfreie Sprache ist und andererseits infolge des Satzes von Parikh, dass L semilinear ist. Demgemäß ist der Schnitt der semilinearen Sprachen L und $L(G)$

$$L(G) \cap L = \{a^n b^m \mid (n, m) \in \Psi_{\{a,b\}}(L(G)) \cap \{(p, p) \mid p \in N\}\}$$

ebenso semilinear. Da der endliche Watson-Crick-Automat M nur akzeptiert, wenn das Wort auf dem oberen Strang die gleiche Länge wie das Wort auf dem unteren Strang hat, gilt

$$L(G) \cap L = \{a^n b^n \mid a^n \in L_u(M)\}.$$

Dementsprechend ist

$$L_u(M) = h(L(G) \cap L)$$

mit dem Morphismus

$$h : \{a, b\}^* \longmapsto \{\varepsilon, a\}$$

$h(a) = a$,
$h(b) = \varepsilon$,
$h(w) = h(w_1) \ldots h(w_{|w|})$ mit $w \in \{a, b\}^*$.

Der anfänglich definierte endliche Watson-Crick-Automat M akzeptierte nur Wörter über

einem unären Alphabet, es gilt $L_u(M) \subseteq a^*$. Dementsprechend folgt aus der Semilinearität der Sprache $(L(G) \cap L)$ bzw. der Menge $\Psi_\Sigma(L(G) \cap L)$ die Semilinearität der Parikh-Menge $\Psi_\Sigma(L_u(M)) = \{n \mid a^n \in L_u(M)\}$. Weiterhin gilt nach dem Satz von Parikh, dass eine reguläre Grammatik R mit $L(R) = \Psi_\Sigma(L_u(M)) = L_u(M)$ existiert. Folglich ist $L_u(M)$ regulär. □

Mit Lemma 4.1.13 kann nun gezeigt werden, dass die Inklusion von AWK in CS strikt ist und die unbeschränkten Watson-Crick-Automaten nicht alle kontextsensitiven Sprachen akzeptieren können.

Korollar 4.1.5. $AWK(u) \subset CS$

Beweis. Der Beweis folgt direkt aus Lemma 4.1.13. Es ist bekannt, dass die Sprache $L = \{a^{2^n} \mid 2^n\}$ in der Sprachfamilie der kontextsensitiven Sprachen CS liegt und nicht regulär ist. Die Zugehörigkeit von L kann durch eine kontextsensitive Grammatik G gezeigt werden.

$$G = (\{S, A, E, N, B\}, \{a\}, P, S)$$

mit

$$\begin{aligned} P = \{ \quad & (1)\ S \longrightarrow AE, && (5)\ A \longrightarrow a, \\ & (2)\ E \longrightarrow BNE, && (6)\ E \longrightarrow a, \\ & (3)\ NB \longrightarrow BNN, && (7)\ N \longrightarrow a\} \\ & (4)\ AB \longrightarrow AN, \end{aligned}$$

Betrachtet man den Ableitungsbaum der Grammatik G, so ist ersichtlich, dass mit der Ableitung $S \Longrightarrow_G AE \Longrightarrow_G^2 aa$ das Wort a^2 erzeugt wird. Mit der Regel (1) werden jeweils nur Wörter abgeleitet, die mit dem Markersymbol A anfangen und E enden. Mit der Regel (2) können neue Symbole N zusammen mit dem Markersymbol B an das Ende des bisher generierten Wortes eingefügt werden. Die Regel (3) dient dazu, dass das Markersymbol B über das bisher generierte Wort zum Anfangsmarker A laufen und alle Symbole N verdoppeln kann und dann mit Regel (4) zu N wird.
Die Anzahl der Symbole N, die mit der Regel (3) eingefügt werden entspricht genau dem Exponenten $n-1$, wenn das Wort a^{2^n} berechnet werden soll. Mit Regel (3) werden alle Symbole N, bis auf das letzte N, verdoppelt. Durch Regel (4) wird noch ein N hinzugefügt. Wurden $n-1$ Symbole N hinzugefügt, so wird dann mit den Regeln (5), (6) und (7) das Wort a^{2^n} abgeleitet. Es gilt $L(G) = L$.

Mit Hilfe des Pumping-Lemmas kann gezeigt werden, dass L nicht regulär ist.

Angenommen L ist regulär und $w \in L$. Demnach existiert eine Pumpingkonstante p mit $w = a^{2^p} \in L$ und eine Zerlegung $w = xyz$, so dass $xy^iz \in L$ mit $i \in N$ gilt. Weiterhin

gilt $|xy| \leq p$ nach der Definition des Pumping-Lemmas. Es kann gezeigt werden, dass für jede Zerlegung $w = xy^2z$, das Wort w mit $i = 2$ nicht in der Sprache enthalten ist. Es gilt $|y|^2 \leq 2p$, folglich ist $|xy^2z|$ kleiner als die Zweierpotenz 2^{p+1}.

$$w = a^{2^p} < xy^2z = a^{2^p+k} < a^{2^{p+1}} \text{ mit } k = |y|, 1 \leq k \leq p.$$

Dementsprechend ist $xy^2z \notin L$ und L nicht regulär.

Mit dem Beweis, dass L nicht regulär ist, folgt auch mit Lemma 4.1.13, dass $L \notin AWK(u)$. Somit gilt $AWK \subset CS$. □

Weiterhin konnte man an den vorangegangen Lemmata sehen, dass die Sprachklassen recht komplexe Sprachen enthalten, dies soll noch explizit für die Sprachfamilie $NWK(u)$ gezeigt werden. Die hohe Komplexität der Sprachen steht natürlich im direkten Zusammenhang mit der Komplementrelation und den zweisträngigen Sequenzen, die akzeptiert werden können.

Lemma 4.1.14. $NWK(u) - MAT^{\varepsilon} \neq \emptyset$

Beweis. Es wird von einem stateless endlichen Watson-Crick-Automaten

$$M = (\{a, b, c, d, e, f\}, \rho, P)$$

mit den Zuständen a, b, c, d, e und f, der Komplementrelation

$$\rho = \{(a, a), (b, c), (c, b), (a, d), (d, a), (e, f), (f, e)\}$$

und den Übergangsregeln

$$P = \{\begin{pmatrix} c \\ \varepsilon \end{pmatrix}, \begin{pmatrix} d \\ \varepsilon \end{pmatrix}, \begin{pmatrix} a \\ a \end{pmatrix}, \begin{pmatrix} b \\ b \end{pmatrix}, \begin{pmatrix} b \\ c \end{pmatrix}, \begin{pmatrix} e \\ c \end{pmatrix}, \begin{pmatrix} e \\ d \end{pmatrix}, \begin{pmatrix} \varepsilon \\ f \end{pmatrix}\}$$

ausgegangen. Außerdem wird die reguläre Sprache

$$R = c(dd^+b)(aa^+b)^+a^+e^+$$

benötigt, um die Struktur der von M akzeptierten Wörter einzuschränken. Gleichzeitig wird noch der Morphismus

$$h : \{a, b, c, d, e, f\}^* \longmapsto \{\varepsilon, a\}$$

$$\begin{aligned} &h(a) = a, && \\ &h(b) = \varepsilon && \text{mit } b \in \{b, c, d, e, f\}, \\ &h(w) = h(w_1) \dots h(w_{|w|}) && \text{mit } w \in \{a, b, c, d, e, f\}^*. \end{aligned}$$

definiert.

Dementsprechend werden die folgenden Moleküle von M akzeptiert und haben, entsprechend R, die Struktur:

$$\begin{bmatrix} cd^{n_1} & ba^{n_2} & b \ \ldots & ba^{n_{m-1}} & ba^{n_m} & e^{n_{m+1}-1} \\ bx_1 & cx_2 & c \ \ldots & cx_{m-1} & cx_m & f^{n_{m+1}-1} \end{bmatrix} \in WK_\rho(\{a,b,c,d,e,f\})$$

mit $m \geq 3$, $n_i \geq 2$, $1 \leq i \leq m$, $n_{m+1} \geq 1$.

Aus der Komplementarität, die aus der Menge der Paare in ρ resultiert, ist bekannt, dass $x_i = a^{n_i}$, $|x_i| = n_i$ für $2 \leq i \leq m-1$. Weiterhin folgt aus den Übergangsregeln $\begin{pmatrix} b \\ b \end{pmatrix}, \begin{pmatrix} b \\ bc \end{pmatrix}$, dass das Vorkommen von b auf dem oberen Strang in einem Vorkommen von c oder b auf dem unteren Strang resultiert. Wird nun jedem Vorkommen von jedem Symbol, das durch diese Regeln gelesen wurde, ein Index zugewiesen, so haben die Moleküle folgende Struktur:

$$\begin{bmatrix} c_0d^{n_1} & b_1a^{n_2} & b_2 \ \ldots & b_{m-2}a^{n_{m-1}} & b_{m-1}a^{n_m} & e_me^{n_{m+1}-1} \\ b_1x_1 & c_2x_2 & c_3 \ \ldots & c_{m-1}x_{m-1} & c_mx_m & ff^{n_{m+1}-1} \end{bmatrix} \in WK_\rho(\{a,b,c,d,e,f\})$$

Daraus geht hervor, dass mit $\begin{pmatrix} e \\ c \end{pmatrix}$ das letzte Vorkommen von c auf dem unteren Strang, d.h. c_m, mit dem ersten Vorkommen von e auf dem oberen Strang, d.h. e_m, zusammenhängt. Ebenso kann ein Symbol a nur mit $\begin{pmatrix} a \\ a \end{pmatrix}$ gelesen werden. Ein Vorkommen von a auf dem unteren Strang setzt demnach ein Vorkommen von a auf dem oberen Strang voraus. Gleiches gilt für e auf dem oberen Strang und d auf dem unteren Strang. Demnach gilt

$$x_i \in a^+, 2 \leq i \leq m-1,$$
$$x_m = d^{n_{m+1}-1}$$
$$|x_i| = n_{i+1}, 1 \leq i \leq m-1.$$

Aus der Komplementarität der Symbole ging bereits $|x_i| = n_i$ hervor. Zusammen mit $|x_i| = n_{i+1}$ gilt $n_i = n_{i+1}$, $1 \leq i \leq m-1$. Entsprechend haben die akzeptierten Moleküle folgende Form:

$$\begin{bmatrix} cd^n & ba^n & b \ \ldots & ba^n & ba^n & e^{n+1} \\ ba^n & ca^n & c \ \ldots & ca^n & cd^n & f^{n+1} \end{bmatrix} \in WK_\rho(\{a,b,c,d,e,f\}),$$

wobei ba^n mit $n \geq 2$ mindestens zweimal vorkommen muss. Folglich erhält man die Sprache

$$\begin{aligned} h(L_u(M) \cap R) &= \{a^{nm} \mid n, m \geq 2\} \\ &= \{a^p \mid p \text{ ist eine zusammengesetzte Zahl}\}. \end{aligned}$$

In [HJ94] wurde mit Theorem 5.3 gezeigt, dass alle in MAT^{ε} enthaltenen unären Sprachen regulär sind. Die Sprache $h(L_u(M) \cap R)$ ist nicht semilinear und somit nach dem Satz von Parikh auch nicht regulär. Dementsprechend ist $h(L_u(M) \cap R)$ auch nicht in MAT^{ε} enthalten. Aus der Abgeschlossenheit von MAT^{ε} bezüglich des Schnitts mit regulären Sprachen und Morphismen geht hervor, dass $L_u(M) \notin MAT^{\varepsilon}$. □

In diesem Abschnitt wurden die einzelnen Beziehungen zwischen den Sprachfamilien der endlichen Watson-Crick-Automaten untersucht und eine Hierarchie entwickelt, die äußerst hilfreich bei Beweisen mit endlichen Watson-Crick-Automaten ist. So können beispielweise komplexe Automaten stark vereinfacht werden, aber diese Betrachtungen sind auch für den folgenden Abschnitt wichtig. Es sollen nun nicht mehr die Beziehungen zwischen den endlichen Watson-Crick-Automaten betrachtet werden, sondern mögliche Charakterisierungen der rekursiv aufzählbaren Sprachen mit Hilfe der Watson-Crick-Automaten untersucht werden.

4.2 Universalitätsbeweis

In diesem Abschnitt soll betrachtet werden, inwiefern rekursiv aufzählbare Sprachen durch Watson-Crick-Automaten beschrieben werden können. Die führende Frage ist, ob eine Darstellung von RE gefunden werden kann, die mit Watson-Crick-Automaten realisierbar ist und aus Operationen mit primitiven Sprachen, wie z.B. den regulären Sprachen, resultiert. Mit Lemma 4.2.1 wurde eine solche Charakterisierung gefunden, indem $L \in RE$ durch eine Komposition von Morphismen und Schnitt mit einer regulären Sprache erzeugt wird.

Lemma 4.2.1. *Für jede rekursiv aufzählbare Sprache $L \subseteq \Sigma^*$ existieren zwei ε-freie Morphismen h_1, h_2, eine reguläre Sprache R und eine Projektion pr_Σ, so dass $L = pr_\Sigma(h_1(EQ(h_1, h_2)) \cap R)$.*

Beweis. Gegeben sei eine Typ-0-Grammatik $G = (N, \Sigma, P, S)$ für eine beliebige rekursiv aufzählbare Sprache $L \subseteq \Sigma^*$. Für jede Produktionsregel wird vorausgesetzt, dass sie durch genau ein Element aus $Lab = \{p \mid p : u \longrightarrow v \in P\}$ beschriftet ist. Außerdem wird ohne Beschränkung der Allgemeinheit angenommen, dass die Grammatik G ε-frei ist und dass

die Mengen $\Sigma' = \{a' \mid a \in \Sigma\}$, $\Sigma'' = \{a'' \mid a \in \Sigma\}$, $Lab' = \{p' \mid p \in Lab\}$ gegeben sind. Aus Gründen der Übersichtlichkeit führen wir einen Morphismus $d : (N \cup \Sigma)^* \longmapsto (N \cup \Sigma')^*$ mit $d(A) = A$ für $A \in N$, $d(a) = a'$ für $a \in \Sigma$ und $d(w) = d(w_1)...d(w_n)$ mit $w \in (N \cup \Sigma)^*$, $w_i \in (N \cup \Sigma)$ $i \in \{1, ..., n\}$ und $n = |w|$ ein. Die Morphismen $h_1, h_2 : \Sigma_2^* \longmapsto \Sigma_1^*$ sind wie folgt definiert:

$\Sigma_1 = N \cup \Sigma \cup \Sigma' \cup \{B, F, c\}$
$\Sigma_2 = N \cup \Sigma \cup \Sigma' \cup \Sigma'' \cup Lab \cup Lab' \cup \{B, F, c, c'\}$

$h_1(B) = BSc$	$h_2(B) = B$	
$h_1(c) = c$	$h_2(c) = c$	
$h_1(p) = d(v)$	$h_2(p) = d(u)$	für $p : u \longrightarrow v \in P$
$h_1(p') = v$	$h_2(p') = d(u)$	für $p : u \longrightarrow v \in P$
$h_1(A) = A$	$h_2(A) = A$	für $A \in N$
$h_1(a') = a'$	$h_2(a') = a''$	für $a' \in \Sigma'$
$h_1(a'') = a$	$h_2(a'') = a'$	für $a'' \in \Sigma''$
$h_1(a) = F$	$h_2(a) = a$	für $a \in \Sigma$
$h_1(c') = F$	$h_2(c') = c$	
$h_1(F) = F$	$h_2(F) = FF$	
$h_1(w) = h_1(w_1) \dots h_1(w_n)$	$h_2(w) = h_2(w_1) \dots h_1(w_n)$	für $w = w_1 \dots w_n$ $n = \|w\|$

$$R = \{BS\}(\{c\}(N \cup \Sigma')^*)^*\{c\}\Sigma^*\{F\}^+.$$

Zuerst muss gezeigt werden, dass die Teilmengenbeziehung $L(G) \subseteq pr_\Sigma(h_1(EQ(h_1, h_2)) \cap R)$ gilt und somit jedes $w \in L(G)$ auch in $L = pr_\Sigma(h_1(EQ(h_1, h_2)) \cap R)$ enthalten ist.

Die Idee des Beweises ist, dass jedes Wort $w \in L(G)$ der letzte Ableitungsschritt einer Ableitung D von G ist. Demnach enthält jede Kodierung $c(D)$ der Ableitungen D des Wortes $w \in L(G)$ das Wort w in kodierter Form. Es existieren Kodierungen, die die Gleichung $h_1(c(D)) = h_2(c(D))$ erfüllen. Betrachtet man alle Präfixe einer solchen Kodierung $c(D)$ so ist ersichtlich, dass

$$h_1(Pre_i(c(D))) \neq h_2(Pre_i(c(D)))$$

gilt und die Gleichung

$$h_1(Pre_j(c(D))) = h_2(Pre_i(c(D)))$$

nur für bestimmte i, j mit $1 \leq i < j < |c(D)|$ zutrifft. Erst für $Pre_{|c(D)|}(c(D)) = c(D)$ ist die Gleichung

$$h_1(Pre_{|c(D)|}(c(D))) = h_2(Pre_{|c(D)|}(c(D)))$$

und somit auch die Bedingung für das Equalityset $EQ(h_1, h_2)$ erfüllt. Die konkrete Form einer solchen Kodierung sei im Beweis für die Gegenrichtung gegeben. Jedes $w \in L(G)$ ist somit als

Teilwort in der Menge $EQ(h_1, h_2)$ enthalten. Da w Teilwort eines Wortes $w' \in EQ(h_1, h_2)$ ist, $h_2(w) = w$ und $h_2(c(D)) = h_1(c(D))$ gilt, muss jedes w als Teilwort in $h_1(EQ(h_1, h_2))$ und für eine entsprechende Kodierung auch in $(EQ(h_1, h_2)) \cap R$ enthalten sein. Infolgedessen ist jedes $w \in L(G)$ auch in $w \in pr_\Sigma(h_1(EQ(h_1, h_2)) \cap R)$ enthalten.

Im folgenden wird gezeigt, dass jedes $w \in pr_\Sigma(h_1(EQ(h_1, h_2)) \cap R)$ ebenso in $L(G)$ enthalten ist. Falls die Sprache das leere Wort ε beinhaltet, dann wird ein zusätzliches Symbol d eingeführt und den Morphismen $h_1(d) = h_2(d) = BScF$ hinzugefügt.

Ausgehend von einem Wort

$$w \in pr_\Sigma(h_1(EQ(h_1, h_2)) \cap R)$$

erhält man $w = pr_\Sigma(x)$ durch die Projektion pr_Σ angewendet auf ein beliebiges Wort

$$x \in h_1(EQ(h_1, h_2)) \cap R.$$

Entsprechend R muss x von der Form

$$BScx_1cx_2c \dots cx_nF^l$$

sein. Dabei gilt für $x_1, \dots, x_{n-1} \in (N \cup \Sigma')^*$, $x_n \in \Sigma^*$ und $l > 0$. Ebenso erhält man $x = h_1(y)$ aus der Anwendung des Morphismus h_1 auf ein $y \in EQ(h_1, h_2)$. Das Wort y hat demnach folgende Struktur:

$$By_1cy_2c \dots cy_nc'y_{n+1}F^m$$

mit $h_2(y_1) = S$, $h_1(y_i) = h_2(y_{i+1}) = x_i$ für $1 \leq i \leq n$, $m = 2l$ und $h_1(y_{n+1}) = F^{m-1}$. Es kann jedoch nie auftreten, dass $y_j = y_{j+1}$ für $1 \leq j \leq n$.

Aus diesen Vorbetrachtungen ergeben sich verschiedene Bedingungen für y. Es ist klar, dass $y_1 = p$ oder $y_1 = p'$ (falls $n = 1$) für $p : S \to z \in P$. Außerdem geht aus der Struktur von R und aus $h_1(y_i) = h_2(y_{i+1}) = x_i$ hervor, dass $y_i \in (N \cup \Sigma' \cup Lab)^* Lab (N \cup \Sigma' \cup Lab)^*$ für $2 \leq i \leq n$, $y_t \in (N \cup \Sigma'' \cup Lab')^*$, $y_{t+1} \in \Sigma^*$ und $h_1(y_i) = x_i$, $h_2(y_i) = x_{i-1}$ für $1 \leq i \leq n$. Aus diesen Bedingungen folgt, dass

$$d^{-1}(x_{i-1}) \Longrightarrow_G^+ d^{-1}(x_i)$$

für $2 \leq i \leq n-1$. Für zwei beliebige Ableitungsschritte $w_1u_1w_2 \Longrightarrow_G w_1v_1w_2 = w_1'u_2w_2' \Longrightarrow_G w_1'v_2w_2'$ mit $p_1 : u_1 \longrightarrow v_1 \in P$, $p_2 : u_2 \longrightarrow v_2 \in P$, $w_1, w_2, w_1', w_2' \in (N \cup \Sigma)^*$ und $S \Longrightarrow_G^* w_1u_1w_2$ kann eine Kodierung $cw_1p_1w_2cw_1'p_2'w_2'$ angegeben werden. Daraus folgt, dass $S \Longrightarrow_G d^{-1}(x_1)$, $d^{-1}(x_{n-1}) \Longrightarrow_G^+ x_n$ und somit auch $S \Longrightarrow_G^+ x_n$. Da $w = pr_\Sigma(x) = x_n \in \Sigma^*$ ist, muss $w \in L$ sein. □

Beispiel 4.2.1. *Eine Kodierung für die Grammatik* $G = (\{S, A, B\}, \{a, b\}, P, S)$ *mit* $P = \{ S \longrightarrow AB,\ A \longrightarrow a,\ B \longrightarrow a\}$.

Die Ableitung $S \Longrightarrow_G AB \Longrightarrow_G aB \Longrightarrow_G aa$ kann kodiert werden als $B1c2Bca'3ca''a''c'aaFFF$ mit

$$\begin{aligned} h_1(B1c2Bca'3ca''a''c'aaFFF) &= h_2(B1c2Bca'3ca''a''c'aaFFF) \\ &= BScABca'Bca'a'caaFFFFFF \\ pr_\Sigma(h_1(B1c2Bca'3ca''a''caaFFF) \cap R) &= aa \in L(G) \end{aligned}$$

Es konnte nun mit obigem Beweis gezeigt werden, dass jede rekursiv aufzählbaren Sprache durch eine Komposition von Morphismen und Mengenoperationen beschrieben werden kann. Anhand dieser Komposition kann ein Watson-Crick-Automat konstruiert werden, der die rekursiv aufzählbaren Sprachen akzeptiert. Zuerst soll das folgende Theorem zeigen, dass zu jeder rekursiv aufzählbaren Sprache $L \in RE$ ein endlicher Watson-Crick-Automat konstruiert werden kann, der, zusammen mit einer Projektion, L vollständig charakterisiert.

Theorem 4.2.1. *Jede rekursiv aufzählbare Sprache* $L = pr_\Sigma(L') \in RE$ *ist das Bild einer Projektion* pr_Σ *angewendet auf eine Sprache* $L' \in AWK(u)$*, die durch einen endlichen Watson-Crick-Automaten beschrieben werden kann.*

Beweis. Anknüpfend an das Lemma 4.2.1 existiert ein endlicher Watson-Crick-Automat M für die Sprache $L' = L_u(M) = (h_1(EQ(h_1, h_2)) \cap R)$, wobei $h_1, h_2 : \Sigma_2^* \longmapsto \Sigma_1^*$ ε-freie Morphismen sind und $R \subseteq \Sigma_1^*$ ist.

Auf der Grundlage des endlichen Automaten $A = (Q, \Sigma_1, q_0, \delta, F)$ für die reguläre Sprache $R = L(A)$ kann ein endlicher Watson-Crick-Automat $M = (Q, \Sigma_1, \rho, q_0, \delta', F)$ mit

$\rho = \{(a, a) \mid a \in \Sigma_1\} = id_{\Sigma_1}$ mit der Identitätsrelation id_{Σ_1},

$\delta'(q, \begin{pmatrix} h_1(a) \\ h_2(a) \end{pmatrix}) = \{q'\}$ mit $q, q' \in Q$, $a \in \Sigma_2$ und dem Übergang im endlichen Automaten $(q, h_1(a)) \vdash_A (q', \varepsilon)$ und

$\delta'(q, \begin{pmatrix} h_1(a) \\ h_2(a) \end{pmatrix}) = \emptyset$ sonst

konstruiert werden.

Der endliche Watson-Crick-Automat M übernimmt die Zustandsmenge, den Anfangszustand und die Endzustände des endlichen Automaten A. Ebenso existiert genau dann ein Übergang

im endlichen Watson-Crick-Automaten M, wenn auch ein Übergang im endlichen Automaten A vorhanden ist. Während A auf dem Band Symbole $h_1(a) \in \Sigma_1$ mit $a \in \Sigma_2$, welche die Bilder des Morphismus h_1 sind, liest, verarbeitet der endliche Watson-Crick-Automat M hingegen zusätzlich auf dem zweiten Strang Symbole $h_2(a) \in \Sigma_1$ mit $a \in \Sigma_2$, welche wiederum die Bilder des Morphismus h_2 sind. Aus Lemma 4.2.1 ist bekannt, dass die reguläre Sprache R von der Form

$$R = \{BS\}(\{c\}(N \cup \Sigma')^*)^*\{c\}\Sigma^*\{F\}^+$$

ist. Indem der endliche Watson-Crick-Automat M die Zustände, den Anfangszustand, die Endzustände von A übernimmt und die Existenz eines Übergangs in A einen Übergang in M zur Folge hat, wird die durch M beschriebene Sprache nie Elemente außerhalb von R enthalten. Gleichzeitig muss jedoch auch die Komplementeigenschaft des Watson-Crick-Bandes im Zusammenhang mit der Komplementrelation beachtet werden. Bei dem endlichen Watson-Crick-Automaten M entspricht die Komplementrelation der Identitätsrelation id_{Σ_2}, wodurch auf dem oberen und unteren Strang nur Wörter $h_1(w) = h_2(w)$ mit $w \in \Sigma_2$ akzeptiert werden. Der endliche Watson-Crick-Automat beschreibt somit die Sprache

$$L_u(M) = L' = (h_1(EQ(h_1, h_2)) \cap R).$$

□

Theorem 4.2.2. *Jede rekursiv aufzählbare Sprache $L \in RE$ kann durch einen Watson-Crick-Transducer M' beschrieben werden.*

Beweis. Analog zu Theorem 4.2.1 wird auf die Charakterisierung einer beliebigen rekursiv aufzählbaren Sprache L in Lemma 4.2.1 Bezug genommen. Da $L = pr_\Sigma(L')$ aus der Projektion bzgl. Σ angewendet auf L' resultiert, ist es noch notwendig, die Sprache $L' = L_u(M)$ auf Symbole des Alphabets Σ zu reduzieren. Es ist offensichtlich, dass in der Sprachfamilie der endlichen Watson-Crick-Transducer die Klasse aller Morphismen enthalten ist und somit die Watson-Crick-Transducer jeden Morphismus beschreiben. Es kann ein Watson-Crick-Transducer M', der als Eingabe die Sprache $L_u(M)$ erhält und die Funktion der Projektion pr_Σ übernimmt, für die Sprache L angegeben werden.

$$M' = (\Sigma_1, \rho, \Sigma, Q, q_0, F, \delta'')$$

mit

$\rho = \{(a, a) \mid a \in \Sigma_1\}$,

$\delta''(q, \begin{pmatrix} h_1(a) \\ h_2(a) \end{pmatrix}) = \{h_1(a), q'\}$ falls $h_1(a) \in \Sigma$ mit $q, q' \in Q$, $a \in \Sigma_2$ und dem Übergang im endlichen Watson-Crick-Automaten $(q, h_1(a)) \Longrightarrow_M (q', \varepsilon)$ und

$$\delta''(q, \begin{pmatrix} h_1(a) \\ h_2(a) \end{pmatrix}) = \emptyset \text{ sonst.}$$

Der Transducer M' akzeptiert die Sprache L' als Eingabe, gibt jedoch nur Symbole aus Σ aus. Dementsprechend beschreibt M' die Sprache $L(M') = L = pr_\Sigma(L')$. □

Es wurde nun gezeigt, dass zu jeder rekursiv aufzählbaren Sprache L ein Watson-Crick-Transducer konstruiert werden kann, der genau diese Sprache beschreibt. Damit ist das Berechnungsmodell der Watson-Crick-Transducer universell. Da bewiesen wurde, dass die Watson-Crick-Transducer universell sind, stellt sich natürlich die Frage, ob es eventuell auch einen endlichen Watson-Crick-Transducer gibt, der in der Lage ist, ähnlich wie die universelle Turingmaschine Turingmaschinen simulieren kann, endliche Watson-Crick-Transducer zu simulieren.

Die Klasse der zu simulierenden Watson-Crick-Transducer wurde reduziert, es kann nun ein universeller Watson-Crick Transducer definiert werden.

4.3 Ein universeller Watson-Crick-Transducer

Ziel in diesem Abschnitt ist es, einen Watson-Crick-Transducer mit geringer Komplexität zu konstruieren, der universell für alle Watson-Crick-Transducer ist. Es wird infolgedessen die Menge der endlichen Watson-Crick-Transducer reduziert, die simuliert werden müssen, indem mit Lemma 4.3.1 und Lemma 4.3.2 die Anzahl der zu lesenden Symbole auf dem Eingabeband je Übergang und der auszugebenden Symbole je Übergang auf maximal ein Symbol reduziert wird. Schließlich wird gezeigt, dass diese äquivalent zu unbeschränkten endlichen Watson-Crick-Transducern sind. Auf diese Weise wird die Größe und Komplexität der Eingabekodierung der zu simulierenden endlichen Watson-Crick-Transducer verringert. Mit Theorem 4.3.1 wird ein universeller endlicher Watson-Crick-Transducer konstruiert, welcher universell für die eben beschriebene Klasse von endlichen Watson-Crick-Transducern ist.

Lemma 4.3.1. *Für jeden beliebigen Watson-Crick-Transducer M kann ein Watson-Crick-Transducer M' konstruiert werden, der bei jedem Übergang nur maximal ein Symbol auf dem oberen und maximal ein Symbol auf dem unteren Strang liest.*

Beweis. Angenommen es existiert ein Watson-Crick-Transducer

$$M = (\Sigma_I, \rho_I, \Sigma_O, Q, q_0, F, \delta)$$

mit einem Übergang

$$(q', x) \in \delta(q, \begin{pmatrix} u \\ v \end{pmatrix}).$$

Dann kann zu diesem Watson-Crick-Transducer M ein Watson-Crick-Transducer M' mit

$$M' = (\Sigma_I, \rho_I, \Sigma_O, Q', q_0, F, \delta')$$

konstruiert werden, welcher nur noch Übergänge enthält, die ein Symbol pro Strang einlesen. Für jeden Übergang der Form $(q', x) \in \delta(q, \begin{pmatrix} u \\ v \end{pmatrix})$ mit $u = u_1 \dots u_{|u|}$, $v = v_1 \dots v_{|v|}$,$|v| > 1$, $|u| > 1$ in δ werden die Übergänge

$$(q_1, \varepsilon) \in \delta(q, \begin{pmatrix} u_1 \\ v_1 \end{pmatrix}), \dots, (q_{|u|}, \varepsilon) \in \delta(q_{|u|-1}, \begin{pmatrix} u_{|u|} \\ v_{|u|} \end{pmatrix}), \dots, \dots, (q', x) \in \delta(q_{|v|-1}, \begin{pmatrix} \varepsilon \\ v_{|v|} \end{pmatrix})$$

hinzugefügt, falls |u| < |v| und die Übergänge

$$(q_1, \varepsilon) \in \delta(q, \begin{pmatrix} u_1 \\ v_1 \end{pmatrix}), \dots, (q_{|v|}, \varepsilon) \in \delta(q_{|v|-1}, \begin{pmatrix} u_{|v|} \\ v_{|v|} \end{pmatrix}), \dots, \dots, (q', x) \in \delta(q_{|u|-1}, \begin{pmatrix} u_{|u|} \\ \varepsilon \end{pmatrix})$$

hinzugefügt, falls |v| < |u|. Alle anderen Übergänge mit $|u| \leq 1$ und $|v| \leq 1$ werden in δ' übernommen. Dementsprechend hat der Watson-Crick-Transducer M' nur noch Übergänge, die auf dem oberen und unteren Strang maximal ein Symbol lesen. Es gilt $L(M) = L(M')$. □

Lemma 4.3.2. *Für jeden beliebigen Watson-Crick-Transducer M kann ein Watson-Crick-Transducer M' konstruiert werden, der bei jedem Übergang nur maximal ein Symbol ausgibt.*

Beweis. Gegeben sei ein Watson-Crick-Transducer

$$M = (\Sigma_I, \rho_I, \Sigma_O, Q, q_0, F, \delta)$$

mit einem Übergang

$$(q', x) \in \delta(q, \begin{pmatrix} u \\ v \end{pmatrix})$$

mit $|x| > 1$ und $x = x_1 \dots x_{|x|}$. Der Übergang mit $|x| > 1$ kann nun auf $|x|$ Übergänge mit der Ausgabe $|x_i|$ $1 \leq i \leq |x|$ aufgeteilt werden:

$$(q_1, x_1) \in \delta(q, \begin{pmatrix} u \\ v \end{pmatrix}), (q_2, x_2) \in \delta(q_1, \begin{pmatrix} \varepsilon \\ \varepsilon \end{pmatrix}), \dots, (q_{|x|}, x_{|x|}) \in \delta(q_{|x|-1}, \begin{pmatrix} \varepsilon \\ \varepsilon \end{pmatrix}).$$

Alle anderen Übergänge werden in δ' übernommen. Somit enthält M' nur noch Übergänge, die maximal ein Symbol ausgeben. Es gilt $L(M) = L(M')$. □

Theorem 4.3.1. *Es existiert ein universeller Watson-Crick-Transducer UWKT, mit dem jeder beliebige Watson-Crick-Transducer M für eine Eingabe w simuliert werden kann.*

Beweis. Die Konstruktion geht von einem universellen Watson-Crick-Transducer

$$UWKT = (\Sigma_{I,U}, \rho_{I,U}, \Sigma_{O,U}, Q_U, q_0, F_U, \delta_U)$$

aus.

Zuerst ist es notwendig, für einen beliebigen Watson-Crick-Transducer mit einer beliebigen Eingabe eine Kodierung zu konstruieren, die dem universellen Watson-Crick-Transducer wiederum als Eingabe dient.

Angenommen es existiert ein beliebiger Watson-Crick-Transducer

$$M = (\Sigma_I, \rho_I, \Sigma_O, Q, q_0, F, \delta).$$

Ausgehend von M kann eine Kodierung

$$code(M) = [\overline{s}_0(\overline{\alpha}_0, \overline{\beta}_0, \overline{\gamma}_0)\overline{s}'_0][\overline{s}_1(\overline{\alpha}_1, \overline{\beta}_1, \overline{\gamma}_1)\overline{s}'_1] \dots [\overline{s}_n(\overline{\alpha}_n, \overline{\beta}_n, \overline{\gamma}_n)\overline{s}'_n][\overline{s}_{f1}] \dots [\overline{s}_{fm}]$$

mit $s_0, \dots, s_n, s_{f1}, \dots, s_{fm} \in Q$, $s_{f1}, \dots, s_{fm} \in F$, $\overline{s}_0, \dots, \overline{s}_n, \overline{s}_{f1}, \dots, \overline{s}_{fm} \in \{0,1\}^*$,
$|\overline{s}_0| = |\overline{s}_1| = \dots = |\overline{s}_n|$,

$$code(s_0) = \overline{s}_0 \neq code(s_1) = \overline{s}_1 \neq \dots \neq code(s_n) = \overline{s}_n$$

und

$$s'_0 \in \delta(s_0, \begin{pmatrix} \alpha_1 \\ \beta_1 \end{pmatrix}), \dots, s'_n \in \delta(s_n, \begin{pmatrix} \alpha_n \\ \beta_n \end{pmatrix})$$

der Übergangsrelation und der Endzustände angegeben werden.

Dementsprechend ist es möglich, eine Eingabe w_0 für den universellen Watson-Crick-Transducer mit der Eingabe $w = \begin{bmatrix} a_1 a_2 \dots a_r \\ b_1 b_2 \dots b_r \end{bmatrix} \in WK_{\rho_I}$ mit $a_1, \dots, a_r, b_1, \dots, b_r \in \Sigma_I$ des zu simulierenden Watson-Crick-Transducers, zu formulieren:

$$w_0 = \begin{bmatrix} code(M)\#\overline{a}_1, \overline{b}_1 code(M)\#\overline{a}_3, \overline{b}_3 \dots code(M)\#\overline{a}_{r-1}, \overline{b}_{r-1} code(M)c \\ code(M)\#\overline{a}_2, \overline{b}_2 code(M)\#\overline{a}_4, \overline{b}_4 \dots code(M)\#\overline{a}_r \quad , \overline{b}_r \quad code(M)c \end{bmatrix}$$

mit $\overline{a}_1, \ldots, \overline{a}_r, \overline{b}_1, \ldots, \overline{b}_r \in \{0,1\}^*$.

Den einzelnen Zuständen des Watson-Crick-Transducers kann eine Semantik zugeordnet werden.

So liest der obere Lesekopf im Zustand q_0 alle Symbole bis zum ersten Eingabesymbol a_1 und der untere Lesekopf in q_0 alle Symbole bis zum Anfangszustand s_0 von M. Der Wechsel von q_0 zu q_1 erfolgt nichtdeterministisch. In q_1 wird überprüft, ob der Zustand, welcher gelesen wird, wirklich der Anfangszustand ist. Falls dem so ist, dann wird wiederum nichtdeterministisch entschieden, ob der nächste Übergang durch die Eingabesymbole $\alpha_i = a_i$ und $\beta_i = b_i$ mit $i \in N$ ausgelöst wird oder ein ε-Übergang ist. Somit wird entweder zu q_2 oder q_2' gewechselt. Es wird in q_2 $\alpha_i = a_i$ bzw. $\beta_i = b_i$ überprüft und in q_2', ob α_i und β_i mit $\overline{\varepsilon}$ übereinstimmen. Entsprechend liest in Zustand q_4 der obere Lesekopf alle Symbole bis Zustand s_j mit $j \in N$. Es wird ebenso nichtdeterministisch entschieden, wann der Übergang mit Ausgangszustand s_j und wann ein Endzustand s_{fi} erreicht wurde. Entsprechend wird in q_5 oder q_5' überprüft, ob der Folgezustand s_i' mit dem Ausgangszustand s_j des folgenden Übergangs oder mit einem Endzustand s_{fi} übereinstimmt. Weiterführend liest der untere Lesekopf in q_6 alle Symbole bis a_{i+1} und der Automat geht mit # in den Zustand q_7 oder q_7' über. Der Automat arbeitet in q_7 und q_7' analog zu q_2 bzw. q_2' und geht dann zu q_8 über. In q_8 wird γ_i, das sind alle Symbole auf dem oberen Strang, ausgegeben und auf dem unteren Strang wird zu s_k' gelaufen. Der Transducer wechselt zu q_9, womit der untere Lesekopf bis s_l läuft und der Automat zu q_{10} oder q_{10}' wechselt. Im Zustand q_{10} bzw. q_{10}' arbeitet der Automat analog zu q_5 und q_5' und geht zu q_{11} bzw. q_{12} über. Angekommen in q_{11} wird auf dem oberen Strang zu a_{i+2} gelaufen und zu q_2 gewechselt. In q_{12} wird über $code(M)$ gelaufen, bis auf beiden Strängen c erreicht ist und in einen finalen Zustand q_f übergegangen wird.

Es kann die Übergangsfunktion δ angegeben werden:

$$q_0 \in \delta(q_0, \begin{pmatrix} \alpha \\ \varepsilon \end{pmatrix}) \text{ mit } \alpha \in \{[,], (,), 0, 1, ', ,\}$$

$$q_0 \in \delta(q_0, \begin{pmatrix} \varepsilon \\ \alpha \end{pmatrix}) \text{ mit } \alpha \in \{[,], (,), 0, 1, ', ,\}$$

$$q_1 \in \delta(q_0, \begin{pmatrix} \# \\ \varepsilon \end{pmatrix})$$

$$q_2 \in \delta(q_1, \begin{pmatrix} \varepsilon \\ \overline{s}_0(\end{pmatrix}) \text{ mit } \overline{s}_0 \in \{0,1\}^* \text{(Kodierung des Anfangszustandes)}$$

$$q_2' \in \delta(q_1, \begin{pmatrix} \varepsilon \\ \overline{s}_0(\end{pmatrix}) \text{ mit } \overline{s}_0 \in \{0,1\}^* \text{(Kodierung des Anfangszustandes)}$$

$q_2 \in \delta(q_2, \begin{pmatrix} \alpha \\ \alpha \end{pmatrix})$ mit $\alpha \in \{0, 1\}$

$q_2 \in \delta(q_2, \begin{pmatrix} , \\ , \end{pmatrix})$

$q_3 \in \delta(q_2, \begin{pmatrix} \varepsilon \\ , \end{pmatrix})$

$q_3 \in \delta(q_2', \begin{pmatrix} \varepsilon \\ (\overline{\varepsilon}, \overline{\varepsilon}, \end{pmatrix})$ mit $\overline{\varepsilon} \in \{0, 1\}^*$ (Kodierung von ε)

$q_3 \in \delta(q_3, \begin{pmatrix} \varepsilon \\ \alpha \end{pmatrix})$ mit $\alpha \in \{0, 1\}$

$q_4 \in \delta(q_3, \begin{pmatrix} \varepsilon \\ ) \end{pmatrix})$

$q_4 \in \delta(q_4, \begin{pmatrix} \alpha \\ \varepsilon \end{pmatrix})$ mit $\alpha \in \{[,], (,), 0, 1, , \}$

$q_5 \in \delta(q_4, \begin{pmatrix} [\\ \varepsilon \end{pmatrix})$

$q_5' \in \delta(q_4, \begin{pmatrix} [\\ \varepsilon \end{pmatrix})$

$q_5 \in \delta(q_5, \begin{pmatrix} \alpha \\ \alpha \end{pmatrix})$ mit $\alpha \in \{0, 1\}$

$q_6 \in \delta(q_5, \begin{pmatrix} (\\ ] \end{pmatrix})$

$q_5' \in \delta(q_5', \begin{pmatrix} \alpha \\ \alpha \end{pmatrix})$ mit $\alpha \in \{0, 1\}$

$q_{12} \in \delta(q_5', \begin{pmatrix}] \\ ] \end{pmatrix})$

$q_6 \in \delta(q_6, \begin{pmatrix} \varepsilon \\ \alpha \end{pmatrix})$ mit $\alpha \in \{[,], (,), 0, 1, , \}$

$q_7 \in \delta(q_6, \begin{pmatrix} \varepsilon \\ \# \end{pmatrix})$

$q'_7 \in \delta(q_6, \begin{pmatrix} \varepsilon \\ \# \end{pmatrix})$

$q_7 \in \delta(q_7, \begin{pmatrix} \alpha \\ \alpha \end{pmatrix})$ mit $\alpha \in \{0, 1\}$

$q_7 \in \delta(q_7, \begin{pmatrix} , \\ , \end{pmatrix})$

$q_8 \in \delta(q_7, \begin{pmatrix} , \\ \varepsilon \end{pmatrix})$

$q_8 \in \delta(q'_7, \begin{pmatrix} (\overline{\varepsilon}, \overline{\varepsilon}, \\ \varepsilon \end{pmatrix})$ mit $\overline{\varepsilon} \in \{0, 1\}^*$ (Kodierung von ε)

$q_8 \in \delta(q_8, \begin{pmatrix} \alpha \\ \varepsilon \end{pmatrix})$ mit $\alpha \in \{0, 1\}$

$q_9 \in \delta(q_8, \begin{pmatrix}) \\ \varepsilon \end{pmatrix})$

$q_9 \in \delta(q_9, \begin{pmatrix} \varepsilon \\ \alpha \end{pmatrix})$ mit $\alpha \in \{[,], (,), 0, 1, , , \}$

$q_{10} \in \delta(q_9, \begin{pmatrix} \varepsilon \\ [\end{pmatrix})$

$q'_{10} \in \delta(q_9, \begin{pmatrix} \varepsilon \\ [\end{pmatrix})$

$q_{10} \in \delta(q_{10}, \begin{pmatrix} \alpha \\ \alpha \end{pmatrix})$ mit $\alpha \in \{0, 1\}$

$q_{11} \in \delta(q_{10}, \begin{pmatrix}] \\ (\end{pmatrix})$

$q_{10} \in \delta(q_{10}, \begin{pmatrix} \alpha \\ \alpha \end{pmatrix})$ mit $\alpha \in \{0, 1\}$

$q_{12} \in \delta(q_{10}, \begin{pmatrix}] \\ ] \end{pmatrix})$

$q_{11} \in \delta(q_{11}, \begin{pmatrix} \alpha \\ \varepsilon \end{pmatrix})$ mit $\alpha \in \{[,], (,), 0, 1, , , \}$

$$q_2 \in \delta(q_{11}, \begin{pmatrix} \# \\ \varepsilon \end{pmatrix})$$

$$q_2' \in \delta(q_{11}, \begin{pmatrix} \# \\ \varepsilon \end{pmatrix})$$

$$q_{12} \in \delta(q_{12}, \begin{pmatrix} \alpha \\ \varepsilon \end{pmatrix}) \text{ mit } \alpha \in \{[,], (,), 0, 1, ', \}$$

$$q_{12} \in \delta(q_{12}, \begin{pmatrix} \varepsilon \\ \alpha \end{pmatrix}) \text{ mit } \alpha \in \{[,], (,), 0, 1, ', \}$$

$$q_f \in \delta(q_{12}, \begin{pmatrix} c \\ c \end{pmatrix})$$

Es gilt $L(UWKT) = L(M) = f(w_0) = g(w)$, wenn $UWKT$ die Eingabe w_0 und M die Eingabe w erhält.

Wie schon oft erwähnt arbeitet der konstruierte universelle endliche Watson-Crick-Transducer nichtdeterministisch. Wichtig ist natürlich noch zu untersuchen, ob eventuell ein deterministischer endlicher Watson-Crick-Transducer existiert, der zu $UWKT$ äquivalent ist. In dieser Eigenschaft unterscheiden sich die Watson-Crick-Transducern nicht von den endlichen Transducern. Es können nicht alle Watson-Crick-Transducer determinisiert werden.

□

5 Zusammenfassung

In dieser Arbeit wurde das formale Berechnungsmodell der Watson-Crick-Automaten vorgestellt. Angefangen bei den molekularbiologischen und mathematischen Grundlagen wurde zu der Definition der Watson-Crick-Automaten übergegangen. Es wurde danach gezeigt, dass Watson-Crick-Automaten, speziell die Watson-Crick-Transducer, ein universelles Berechnungsmodell sind. Außerdem konnte ein endlicher Watson-Crick-Transducer konstruiert werden, welcher universell für alle endlichen Watson-Crick-Transducer ist.

Weiterführende Fragen wären:

- Kann der vorgestellte universelle Watson-Crick-Transducer minimiert werden?
- Existieren weitere und einfachere Charakterisierungen von rekursiv aufzählbaren Sprachen, die durch einen Watson-Crick-Automaten akzeptiert werden können?
- Existieren andere Watson-Crick-Automaten, die universell für ihre Klasse von Automaten und deterministisch sind?
- Können die Beziehungen zwischen den Sprachfamalien der endlichen Watson-Crick-Automaten verfeinert werden?

Literaturverzeichnis

[Ba10] Franz Baader. Formale Systeme (Skript zur Vorlesung). TU Dresden, 2010.

[HS04] Thomas Hinze and Monika Sturm. Rechnen mit DNA - Eine Einführung in Theorie und Praxis. Oldenbourg Verlag München, 2004.

[PRS98] Gheorghe Paun, Grzegorz Rozenberg, and Arto Salomaa. DNA-Computing - New Computing Paradigms. Springer Verlag Berlin, Heidelberg, 1998.

[Ku05] Dietrich Kuske. Formale Aspekte des DNA-Computing. (Skript zur Vorlesung). Universität Wien. 2005.

[Na10] Benedek Nagy. $5' \rightarrow 3'$ Sensing Watson-Crick Finite Automata. Springer Berlin Heidelberg, Garzon, Max H. und Yan, Hao. DNA-Computing. 2008.

[OSHS] Satoshi Okawa, Sadaki Hirose. The Relations among Watson-Crick-Automata and Their Relations with Contxt-Free Languages. 2006. IEICE - Trans. Inf. Syst., 2006, 10, S. 2591 - 2599.

[HJ94] Dirk Hauschildt, Matthias Jantzen. Petri Net Algorithms in the Theory of Matrix Grammars. Acta Informatica, 1994, 31, S. 719-728.

[St12] Monika Sturm. Molekulares Rechnen (Skript zur Vorlesung). TU Dresden, 2012.

Erklärung

Hiermit versichere ich, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe, dass alle Stellen der Arbeit, die wörtlich oder sinngemäß aus anderen Quellen übernommen wurden, als solche kenntlich gemacht und dass die Arbeit in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegt wurde.

Ort, Datum Unterschrift