Institut für Formale Methoden der Informatik
Universität Stuttgart
Universitätsstraße 38
D-70569 Stuttgart

Studienarbeit Nr. 2425

Deterministische endliche Automaten und Zwei-Variablen-Logik erster Stufe

Sebastian Müller

Studiengang:	Diplom Informatik
Prüfer:	Prof. Dr. V. Diekert
Betreuer:	Dr. M. Kufleitner
Beginn am:	18.4.2013
Beendet am:	18.10.2013
CR-Nummer:	F.4.3

Kurzfassung:

Therien und Wilke zeigten in einer Arbeit von 1998, dass 2-Variablen-Logik erster Stufe (FO^2) einer entscheidbaren Klasse endlicher Monoide entspricht. Damit läßt sich insbesondere für jede reguläre Sprache entscheiden, ob sie in FO^2 definierbar ist. Dieses Entscheidbarkeitsresultat konnte 2012 in einer Arbeit von Weil und Kufleitner auf die Alternierungshierarchie innerhalb von FO^2 ausgedehnt werden.
Im Rahmen dieser Arbeit wird untersucht, wie effizient sich diese Entscheidbarkeitsresultate umsetzen lassen, wenn die reguläre Sprache durch deterministische endliche Automaten gegeben ist.
Als Vorstufe hierzu werden geeignete algebraische Charakterisierungen der Alternierungshierarchie innerhalb von FO^2 recherchiert.
Basierend darauf werden Entscheidungsverfahren auf Basis sogenannter Verbotsmuster entwickelt.

Inhaltsverzeichnis

1 Einleitung

McNaughton und Papert zeigten 1965, dass die sogenannten sternfreien Sprachen in Logik erster Stufe (First-Order Logic, FO) beschrieben werden können [1]. Im selben Jahr bewies Schützenberger, dass die Klasse der sternfreien Sprachen aperiodische syntaktische Monoide hat [2]. Kamp zeigt kurz darauf, dass sich alle FO-definierbaren Sprachen auch in First-Order-Logik mit nur 3 Variablen (FO^3) beschreiben lassen [3].
Der Zusammenhang zwischen formalen (regulären) Sprachen, deren logischer Beschreibungsformen und endlichen Halbgruppen eröffnete die Möglichkeiten zur Anwendung algebraischer Methoden auf dem Feld der Theorie formaler Sprachen.

In einer Arbeit von 1998 zeigten Therien und Wilke, dass 2-Variablen-Logik erster Stufe ($\mathbf{FO^2}$) einer entscheidbaren Klasse endlicher Monoide entspricht ([4]). Damit ließ sich insbesondere für jede reguläre Sprache entscheiden, ob sie in $\mathbf{FO^2}$ definierbar ist. Dieses Entscheidbarkeitsresultat konnte 2012 von Weil und Kufleitner auf die Alternierungshierarchie innerhalb von $\mathbf{FO^2}$ ausgedehnt werden [5].

Im Rahmen dieser Arbeit wird untersucht, wie effizient sich diese Entscheidbarkeitsresultate umsetzen lassen, wenn die reguläre Sprache durch deterministische endliche Automaten gegeben ist.
Als Vorstufe hierzu werden geeignete algebraische Charakterisierungen der Alternierungshierarchie innerhalb von $\mathbf{FO^2}$ recherchiert.
Basierend darauf werden Entscheidungsverfahren auf Basis sogenannter Verbotsmuster entwickelt. Darüber hinaus wird geprüft, wie leicht sich für eine gegebene $\mathbf{FO^2}$-basierte Sprache die minimale Anzahl benötigter Alternierungen berechnen lässt.

2 Grundlagen und Definitionen

In diesem Kapitel werden Grundlagen und Definitionen behandelt, die zum Verständnis des weiteren Inhaltes dieser Arbeit notwendig sind, sowie verwendete Begriffe und Notationen erläutert.
Es geht vor allem um die verschiedenen Formen, reguläre Sprachen bzw. bestimmte Unterklassen davon zu definieren. Weiter geht es um algebraischen Hilfsmittel, hauptsächlich die greenschen Relationen, die ein bedeutendes Werkzeug bei der Arbeit mit Halbgruppen darstellen [6].
Standardlehrbücher, die einen ausführlicheren Überblick geben sind [7] und [8].

2.1 Reguläre Sprachen

Bekannte Formen eine reguläre Sprache zu definieren sind reguläre Ausdrücke, Grammatiken und deterministische endliche Automaten (DFA). Weitere Möglichkeiten zur Charakterisierung sind logische Formeln und algebraische Strukturen. Beides wird in diesem Kapitel vehandelt.
Nicht behandelt, aber an dieser Stelle erwähnt, werden andere Logikarten wie z.B. Unary-Temporal-Logic (UTL) und sogenannte Ranker (siehe z.B. [9] und [10]).

2.1.1 Definition durch endliche deterministische Automaten

Die Notation für die in dieser Arbeit verwendeten endlichen deterministischen Automaten wird die folgende sein: ein DFA ist ein Tupel $\mathcal{M} = \{\Sigma, Z, \delta, z_0, E\}$ mit Alphabet Σ, Zustandsmenge Z, (eindeutigem) Startzustand $z_0 \in Z$, Endzustandsmenge $E \subseteq Z$ und der Zustandsübergangsfunktion $\delta : \Sigma \times Z \to Z$.
Ein DFA akzeptiert ein Wort w, wenn die Zustandsübergangsfunktion δ nach Eingabe von w einen Endzustand ergibt.
Wir schreiben $L(\mathcal{M})$ für die Sprache, die der Automat $\mathcal{M}$ akzeptiert. Dies ist die Menge aller Wörter, die $\mathcal{M}$ akzeptiert.

Der Zustandübergangsgraph $G(\mathcal{M})$ von $\mathcal{M}$ ist ein Graph mit Knotenmenge N (die Zustände Z von $\mathcal{M}$) und einer mit $a \in \Sigma$ beschrifteten Kante zwischen zwei Knoten z_1 und z_2 genau dann, wenn $\delta(a, z_1) \to z_2$. Beide Ausdrucksformen (Tupel und Graph) werden im Folgenden nicht jedesmal explizit angegeben. Die jeweils verwendete Ausdrucksform sollte stets aus dem Kontext ersichtlich sein.

Der erweiterte Zustandübergangsgraph G' entspricht von den Zuständen her dem Graphen G, aber wir erweitern die Übergangsfunktion δ um Wörter beliebiger Länge, so dass auch Kanten für Wörter der Länge $l = 2, 3, 4, ...$ existieren. Der Graph G' ist ebenfalls endlich und deterministisch. Man kann dann mit einem Wort w von einem Zustand z_1 aus direkt zu z_2 gehen, wenn $\delta(w, z_1) \to z_2$. Dies wird uns die Arbeit in Abschnitt 2.4 erleichtern.

Ein Automat $\mathcal{M}$ wird minimal genannt, wenn alle Zustände unterscheidbar sind. Zwei Zustände sind unterscheidbar genau dann, wenn es ein Wort w gibt, für das der eine Zustand akzeptiert d.h. in einen Endzustand führt und der andere nicht. Es wird davon ausgegangen, dass alle in dieser Arbeit auftretenden Automaten minimal sind, solange nichts anderes angegeben wird.

Wir reden von einer w-Schleife für ein Wort w und einen Zustand z, wenn $\delta(w, z) \to z$. Da ein DFA für jedes Wort w von jedem beliebigen Zustand aus für ausreichend viele Wiederholungen von w in eine solche w-Schleife läuft ($\delta(www...w, z_a) \to z_w$), sagen wir auch wir laufen in eine w-Schleife bei einem Zustand z_w. Der Zustand, an dem wir mit der Eingabe der w's beginnen ist dabei unbedeutend.

2.1.2 Definition durch logische Formeln

Reguläre Sprachen lassen sich durch logische Formeln definieren. Dabei enthält die Sprache L sämtliche Wörter, die ein Model für die definierende Formel bilden. Für ein Alphabet Σ und eine Formel ϕ ist

$$L(\phi) = \forall w \in \Sigma^* : w \models \phi$$

Bekannte Ergebnisse sind, dass reguläre Sprachen in Monadic-Second-Order Logik (MSO) definiert werden können ([11]), und dass die sogenannten sternfreien Sprachen, die durch einen regulären Ausdruck ohne Verwendung des Kleene-Sterns gebildet werden („sternfreier Ausdruck“), den in First-Order-Logik (FO) definierbaren Sprachen entsprechen [2].
Durch Einschränkungen z.B. eine Beschränkung der Variablenanzahl oder der maximalen Anzahl alternierenden Quantorenblöcken können weitere Sprachklassen unterhalb von FO definiert werden.

Kamp [3] zeigte, dass sich alle FO-definierbaren Sprachen auch in First-Order-Logik mit nur 3 Variablen (FO^3) beschreiben lassen, solange die Variablennamen wiederverwendet werden dürfen. Beide Sprachklassen sind somit identisch.

$FO^2[<]$ enthält alle Sprachen, welche in First-Order-Logik mit nur zwei Variablen definierbar sind. Variablennamen dürfen hierbei wiederverwendet werden (z.B. $\exists x : \exists y : \exists x$, wiederverwendetes x). Darüber hinaus wird nur die Labelfunktion λ, Quantoren ($\exists$, $\forall$) und die Relation $<$ bzw. $>$ zur Ordnung der Variablenpostionen benutzt. Wir verzichten deshalb im weiteren auf die explizite Angabe $FO^2[<]$.

Mit FO^2_m sind alle FO^2-Formeln gemeint, die maximal m alternierende Blöcke gleicher Quantoren haben. Zum Beispiel sind $(\exists x(\exists y(\forall x(\exists x...))))$ drei Blöcke Quantoren: zwei Blöck $\exists$-Quantoren, getrennt durch einen Block $\forall$-Quantoren. Dies wird auch die Alternierungshierarchie innerhalb von FO^2 genannt [12]. Es ist eine strikte Hierarchie d.h. jede Stufe $m+1$ ist Ausdruckstärker als die vorherige Stufe m.
Mit $FO^2_{m,n}$ beschränkt man zusätzlich die maximale Quantorentiefe auf n.

In dieser Arbeit wird es vor allem um First-Order-Logik mit nur zwei Variablen (FO^2) bzw. die damit verbundene Varietät **DA** gehen (siehe Kapitel 3).

Beispiel. Es lässt sich leicht nachvollziehen, dass für ein Alphabet $\Sigma = \{a, b, c\}$ die Formel

$$\phi = \exists x : (\lambda(x) = a \wedge \forall y : y >= x)$$

die Sprache definiert, bei der alle Wörter mit einem a beginnen. Da sich diese Sprache in First-Order-Logik ausdrücken lässt ist sofort klar, dass sie sich auch mit einem sternfreien Ausdruck darstellen lässt, denn beide Ausdrucksformen sind „gleichmächtig“. Außerdem enthält die Formel nur zwei Variablen, womit die definierte Sprache auch in FO^2 ist.

2.1.3 Definition durch algebraische Strukturen

Im folgenden werden der Vollständigkeit wegen zuerst einmal ein paar grundlegende algebraische Definitionen wiederholt.

Definition 2.1 (Halbgruppe, Monoid und Idempotente)**.** *Sei eine Menge S und ein Operator $\otimes : S \times S \rightarrow S$ auf S gegeben, so dass S unter $\otimes$ abgeschlossen ist. S wird eine Halbgruppe genannte, wenn der $\otimes$-Operator assoziativ ist, also $(a \otimes b) \otimes c = a \otimes (b \otimes c)$.*
Ein Element $1 \in S$ für das für alle $s \in S : 1 \otimes s = s\dot{1} = s$ wird neutrales Element von S genannt. Eine Halbgruppe mit neutralem Element nennt man Monoid.
Enthält eine Halbgruppe S (ein Monoid S) ein Element e für das gilt $e^2 = e$, so nennt man dieses Element ein Idempotentes von S. Wir definieren dazu $E(S)$ als die Menge aller Idempotente in der Halbgruppe (dem Monoid) S.

Definition 2.2 (Homomorphismus, Isomorphismus)**.** *Seien S und S' Halbgruppen (bzw. Monoide) und η eine Abbildung $\eta : S \rightarrow S'$ so dass $\eta(x \otimes y) = \eta(x) \otimes \eta(y)$, so nennt man die Abbildung η einen Homomorphismus.*
Sind S und S' Monoide, so muss zusätzlich erfüllt sein, dass $\eta(1_S) = 1_{S'}$.
Ein bijektiver Homomorphismus wird Isomorphismus genannt. S und S' nennt man isomorph, falls ein Isomorphismus zwischen ihnen existiert.

Da wir uns ausschließlich mit endlichen Halbgruppen bzw. Monoiden beschäftigen werden sind im folgenden einige bekannte Eigenschaften davon gezeigt.

Lemma 1. *Sei S eine endliche Halbgruppe. Jedes Element $x \in S$ besitzt eine Potenz π_x so dass x^{π_x} idempotent ist.*

Beweis. *Jedes Element einer endlichen Halbgruppe besitzt einen Index i und eine Periode p, so dass $x^{i+p} = x^i$. Sei für ein Element x der Index i und die Periode p. Für jedes $j > i$ ist $x^j = x^{j+p}$. Ist j ein Vielfaches von p so gilt (für jedes $a > 1$)*

$$(x^j)^2 = x^{2j} = x^{j+ap} = x^j$$

Damit ist x^j ein Idempotentes.

Daraus lässt sich direkt folgern, dass es ein auch ein Π gibt, so dass für alle $x \in S$ gilt, dass x^Π idempotent ist. Man nehme dazu für Π einfach das kleinste gemeinsame Vielfache aller Potenzen der Elemente aus S. Außerdem lässt sich folgern, dass jede nicht-leere endliche Halbgruppe (mindestens) ein Idempotentes enthält.

Lemma 2. *Sei S eine endliche Halbgruppe. Es gibt ein Π für das für alle Elemente $x \in S$ gilt x^Π ist idempotent.*

Lemma 3. *Sei S eine endliche, nicht-leere Halbgruppe. Es existiert mindestens ein Idempotentes in S.*

Lemma 4. *Sei S eine endliche Halbgruppe und $n = |S|$. Dann existiert für jede Sequenz $s_1 s_2 ... s_n$ von mindestens n Elementen aus S ein Index i und ein Idempotentes e, so dass gilt $s_1 s_2 ... s_i = s_1 s_2 ... s_i e$. Man sagt das Element e stabilisiert die Sequenz $s_1 s_2 ... s_i$.*

Beweis. *Gegeben sei eine Folge $s_1 s_2 ... s_n$ von Elementen aus S, $n \geq |S|$. Dann enthält die Folge für den Fall, dass alle Elemente s_i verschieden sind , mindestens ein Idempotentes (siehe 3). Enthält die Folge $s_1 s_2 ... s_n$ nicht alle Elemente aus S, so kommt mindestens ein Element zweimal in der Folge vor z.B. s_j und s_k $(j < k)$. Dann ist*

$$s_1 s_2 ... s_j = s_j (s_{j+1} ... s_k) = s_1 s_2 ... s_j (s_{j+1} ... s_k)^\pi$$

wobei π die Potenz (aus Lemma 2) für S ist.

Damit hat man nun eine weitere Möglichkeit reguläre Sprachen zu charakterisieren, die sogenannten syntaktischen Halbgruppen.

Definition 2.3 (Syntaktische Halbgruppe). *Für eine Sprache $L \in \Sigma^*$ sei die Relation $\sim_L$ definiert, so dass $v \sim_L w$ genau dann, wenn*

$$pvq \in L \leftrightarrow pwq \in L.$$

für alle $p, q \in \Sigma^*$. Die Relation $\sim_L$ wird die syntaktische Äquivalenz auf L genannt.
Die syntaktische Halbgruppe ist die Menge der Äquivalenzklassen von $\sim_L$ mit der Konkatenation als Operator.
Man erhält sie auch als den Quotient von S bezüglich der syntaktischen Äquivalenz $\sim_L$ ($Synt(L) = \Sigma^* / \sim_L$, siehe 2.3 für Quotientenbildung bei Halbgruppen).

Um mit der syntaktischen Halbgruppe $Synt(L)$ die Sprache L erkennen zu können, muss man hierfür noch den Erkennbarkeitsbegriff definieren. Es gilt, dass eine Halbgruppe S die Sprache $L \in \Sigma^*$ erkennt, falls ein erkennender Homomorphismus von Σ^* nach S existiert.

Definition 2.4 (Erkennender Homomorphismus). *Für eine Sprache $L \in \Sigma^*$ und eine Halbgruppe S sei ein Homomorphismus $h : \Sigma^* \rightarrow S$ definiert, so dass $h^{-1}(h(L)) = L$. Man nennt diesen Homomorphismus einen erkennenden Homomorphismus für L. Existiert ein erkenndener Homomorphismus von Σ^* nach S sagt man auch, dass die Halbgruppe S die Sprache L erkennt.*

Bei der syntaktischen Halbgruppe handelt es sich um die minimale Halgruppe, für die ein erkennender Homomorphismus existiert.

Durch den erkennenden Homomorphismus und die Definiton der syntaktischen Äquivalenz wird klar, dass entweder alle, oder aber keines der Wörter einer Äquivalenzklasse von $\sim_L$ in L liegen. Man könnte in diesem Zusammenhang analog zu Automaten von „Endzuständen“ und „Nicht-Endzuständen“ sprechen, in die sich die Elemente der Halbgruppe aufteilen.

Analog hierzu lassen sich syntaktische Monoide definieren. Man ersetze dazu die Halbgruppe S durch den Monoid S^1, mit $S^1 = S \cup \{1\}$. Falls S bereits ein neutrales Element besitzt gilt $S = S^1$. Alle anderen Definition gelten weiterhin.

Der Vorteil des algebraischen Erkennbarkeitsbegriffs und der Verwendung von Halbgruppen (bzw. Monoiden) ist, dass dieses Gebiet der Algebra schon lange und ausführlich erforscht wird, und sich dadurch eine Vielzahl an bereits existierenden Werkzeugen und Hilfsmitteln anbieten.

Diese Arbeit befasst sich mit Sprachen unterhalb der sternfreien Sprachen, also sind alle auftretenden Halbgruppen (bzw. Monoide), solange nicht anders angegeben, stets endlich.

2.2 Greensche Relationen

Die Greenschen Relationen ([13]) stellen ein wichtiges Hilfmittel für die Arbeit mit Halbgruppen dar. Mit ihnen werden alle Elemente einer Halbgruppe in Äquivalenzklassen eingeteilt. Elemente der selben Klasse verhalten sich dabei ähnlich, vor allem im Zusammenhang mit Idempotenten.

Es gilt für eine Halbgruppe S und Elemente $x, y \in S$, dass $x \preceq_{\mathcal{R}} y \leftrightarrow xS \subseteq yS$. Gilt dies in beide Richtungen so dass $(x \preceq_{\mathcal{R}} y \wedge y \preceq_{\mathcal{R}} x)$ liegen x und y in der selben $\mathcal{R}$-Klasse. Man schreibt dann $x\mathcal{R}y$. Anders ausgedrückt bedeutet $x\mathcal{R}y$, dass es $a, b \in S$ gibt, so dass $x = ya$ und $y = xb$. Es existiert also mindestens jeweils ein Element in S, das, wenn man es von rechts an x (bzw. y) multipliziert, y (bzw. x) ergibt.

Definition 2.5 (Greensche Relationen)**.** *Für $x, y \in S$ sind die Relationen $\mathcal{R}$, $\mathcal{L}$ und $\mathcal{J}$ wie folgt definiert:*

$$x\mathcal{R}y \Leftrightarrow xS = yS \tag{1}$$

$$x\mathcal{L}y \Leftrightarrow Sx = Sy \tag{2}$$

$$x\mathcal{J}y \Leftrightarrow SxS = SyS \tag{3}$$

Außerdem existieren noch die von $\mathcal{R}$ und $\mathcal{L}$ abgeleiteten Relationen $\mathcal{H}$ und $\mathcal{D}$:

$$x\mathcal{H}y \leftrightarrow (x \preceq_R y \wedge x \preceq_L y) \tag{4}$$

$$\begin{aligned} x\mathcal{D}y \leftrightarrow & (\exists z \in S : x\mathcal{R}z \wedge z\mathcal{L}y) \\ & \textit{oder} \\ & (\exists z \in S : x\mathcal{L}z \wedge z\mathcal{R}y) \end{aligned} \tag{5}$$

Ist eine Halbgruppe endlich so folgt aus dieser Tatsache direkt, dass auch die Anzahl der $\mathcal{R}$- und $\mathcal{L}$-Klassen endlich ist. Sie formen jeweils eine Partition von S.

Hat jedes Element aus S seine eigene $\mathcal{R}$-Klasse ($\mathcal{L}$-Klasse, $\mathcal{J}$-Klasse), so spricht man bei S von einer $\mathcal{R}$-trivialen ($\mathcal{L}$-trivialen, $\mathcal{J}$-trivialen) Halbgruppe. Die Relation $\mathcal{R}$ ($\mathcal{L}$, $\mathcal{J}$) ist in diesem Fall die Identität z.B. für eine $\mathcal{R}$-triviale Halbgruppe S und $x, y \in S$ ist $x\mathcal{R}y \leftrightarrow x = y$.

2.3 Varietäten

Sei S eine Halbgruppe (bzw. ein Monoid) und $R \subset S \times S$ eine Relation auf der Halbgruppe (bzw. dem Monoid) S. Anstatt $(x, y) \in R$ schreibt man auch $x \sim_R y$ bzw. einfach $x \sim y$ wenn die verwendete Relation aus dem Kontext hervorgeht.
Erfüllt R die folgenden drei Eigenschaften so nennt man R eine Äquivalenzrelation:

1. $(x, x) \in R$ (reflexiv)
2. $(x, y) \in R \leftrightarrow (y, x) \in R$ (symmetrisch)
3. $(x, y) \in R \wedge (y, z) \in R \rightarrow (x, z) \in R$ (transitiv)

Äquivalenzrelationen, für die gilt

$$x \sim y \wedge a \sim b \rightarrow (x \otimes a) \sim (y \otimes b), \forall a, b, x, y \in S$$

nennt man kompatibel mit der Operation $\otimes$ der Halbgruppe (bzw. des Monoiden). Man spricht dann von einer Kongruenz auf der Halbgruppe (bzw. dem Monoiden).
Damit lässt sich der Quotient Q von S bezüglich der Kongruenz $\sim_R$, geschrieben $Q = S/ \sim_R$, definieren. Es handelt sich dabei um die Menge der

Äquivalenzklassen von R mit dem (kompatiblen) Operator $\otimes$ der Halbgruppe (bzw. des Monoiden). Man kann nun von Teilbarkeit sprechen, wenn eine Halbgruppe (bzw. ein Monoid) S' isomorph zu einer Unterhalbgruppe (bzw. einem Untermonoid) von S ist. Es ist klar, dass insbesondere alle Unterhalbgruppen (bzw. Untermonoide) von S ein Quotient von S sind.

Varietäten definieren Klassen von Halbgruppen (bzw. Monoiden). Sie sind unter Teilbarkeit und endlichem direkten Produkt abgeschlossen. Ein Mittel zur Beschreibung von Varietäten sind Omega-Terme.

Omega-Terme (im folgenden auch ω-Terme genannt) sind ein verbreitetes Mittel, um Varietäten zu definieren. Sie werden induktiv über eine (endliche) Variablenmenge Var definiert:

1. Jedes $u \in Var$ ist ein ω-Term.
2. Sind u und v ω-Terme, so ist uv ein ω-Term.
3. Ist u ein ω-Term, so ist u^ω ein ω-Term.

Eine Varietät enthält nun alle Halbgruppen (bzw. Monoide), die eine oder mehrere durch ω-Terme gegebene Identität(en) für jeden Homomorphismus $h : Var \to S^*$ erfüllen. Wir schreiben für zwei ω-Terme u und v dann

$$[[u = v]]$$

um die Varietät zu bezeichnen, in welcher alle Halbgruppen (bzw. Monoide) die Identität $u = v$ erfüllen.
Jeder Homomorphismus h von Σ^* nach S bildet eine natürliche Erweiterung für ω-Terme indem $h(u^\omega) = h(u)^\omega$ (das Idempotente $h(u)^\omega$, das von $h(u)$ erzeugt wird).
Dadurch wird es ermöglicht, dass man mit ganzen Familien von Halbgruppen (bzw. Monoiden) und ihren Eigenschaften arbeitet, ohne eine bestimmte auszuwählen.
Sind alle Halbgruppen (bzw. Monoide) endlich, so spricht man in der Regel von einer Pseudovarietät. Im folgenden wird auf die Unterscheidung beider Begriffe der Einfachkeit halber verzichtet, da es sich bei allen vorkommenden Halbgruppen (bzw. Monoiden) wie in 2.1.3 erwähnt stets um endliche handelt.

Beispiel. Die Varietät $\boldsymbol{AP}$ beschreibt die Klasse der aperiodischen Monoide. Ihre Identität lautet

$$[[x^\omega = x^\omega x]]$$

Sie bildet die algebraische Beschreibung der Klasse der sternfreien Sprachen ([2]).

In dieser Arbeit liegt der Fokus auf der Varietät **DA**. Sie wird in Kapitel 3 vorgestellt, zusammen mit einer in ihr enthaltenen Hierarchie an Untervarietäten.

2.4 Verbotsmuster

Ein Verbotsmuster P ist ein (nicht notwendigerweise zusammenhängender) Graph, dessen Kanten mit Variablen für Wörter $w \in \Sigma^*$ beschriftet sind. Es wird in der Regel grafisch ausgedückt (siehe z.B. Abbildung 1).
Oft werden zusätzliche Bedingungen gestellt, z.B. dass zwei Knoten innerhalb des Verbotsmusters verschieden sein müssen. Um zu zeigen, dass zwei zustände verschieden (bzw. unterscheidbar) sein sollen, wird in der Regel jeweils ein Pfeil zu $+/-$ bzw. $-/+$ eingezeichnet, was ausdrücken soll, dass es Wörter gibt, für die der eine Zustand akzeptiert $(+)$ und der andere nicht $(-)$.
Man spricht davon, dass der Graph G ein Verbotsmuster P vermeidet, wenn es keine Auswahl der Knoten von G für die Knoten von P gibt und keine Belegung für die Variablen existiert, so dass alle Bedingungen erfüllt sind.

Verbotsmuster sind dazu geeignet, die definierenden Identitäten einer Varietät (gegeben in ω-Termen) nachzubilden und so die Zugehörigkeit einer durch einen DFA gegebenen Sprache direkt anhand der Struktur des Automaten zu testen.

Beispiel. Das Verbotsmuster für die Varietät der aperiodischen Monoide (**AP**) ist in Abbildung 1 gegeben.

Es ergibt sich aus der Identität $x^\omega = x^\omega x$ für **AP**.
Man kann leicht nachvollziehen, dass der syntaktische Monoid M einer Sprache L, gegeben durch den Automat A, der das gegebene Muster vermeidet, aperiodisch sein muss. Vermeidet der Automat das Verbotsmuster, bekommt man für jedes Wort $w \in \Sigma^*$ als Belegung für x und Anwendung des erkennenden Homomorphismus

$$h(w^\omega) = h(w)^\omega = h(w)^\omega h(w) = h(w^\omega w)$$

Dies bestätigt die Identität von **AP**.

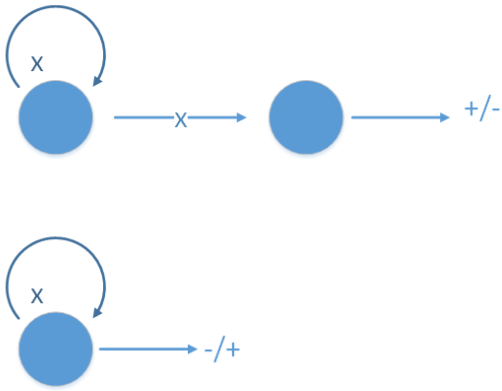

Abbildung 1: Verbotsmuster P_{AP}

Ist der aperiodische Monoid M für L gegeben, und ein Automat A der L erkennt, so muss A das Verbotsmuster vermeiden. Man nehme an, A vemeide das Verbotsmuster nicht. Dann existiert kein erkennender Homomorphismus vo Σ^* auf die Elemente des Monoiden, denn die Wörter $pw^\omega q$ und $pw^\omega wq$ müssten auf verschiedene Element von M abgebildet werden (da sie in verschiedenen Äquivalenzklassen von $\sim_L$ liegen). Dies ist nicht möglich wenn der Automat das gegebene Verbotsmuster enthält.

3 Die Varietät DA

Nachdem in Kapitel 2 die greenschen Relationen und Varietäten vorgestellt wurden, wird hier die Varietät **DA** behandelt. Diese Varietät hat zahlreiche Charakterisierungen (siehe [14]). Über ω-Terme definiert entspricht sie der Varietät mit Identität

$$[[(xy)^\omega = (xy)^\omega x (xy)^\omega]]$$

Alternativ kann dies auch in zwei Identitäten ausgedrückt werden:

$$[[x = x(xy)^\omega]]$$
$$[[x = (xy)^\omega x]]$$

Das folgende Lemma zeigt eine wichtige Eigenschaft der Varietät **DA**:

Lemma 5. *Gegeben ein Monoid $M \in$ **DA** und $x, y, a \in M$. Es gilt, dass wenn $x\mathcal{R}y$ und $y\mathcal{R}ya$ dann auch $x\mathcal{R}xa$. Symmetrisch für $\mathcal{L}$ gilt, dass wenn $x\mathcal{L}y\mathcal{L}by$ dann auch $x\mathcal{L}bx$.*

%

Dies bedeutet, dass die Tatsache, ob $x\mathcal{R}xa$ nur vom Element a und der $\mathcal{R}$-Klasse von x abhängt. Gilt $x\mathcal{R}xa$, so gilt dies auch für alle anderen Elemente aus der $\mathcal{R}$-Klasse von x.

Alle Sprachen, deren syntaktische Monoide in der Varietät **DA** liegen, entsprechen der Klasse der in FO^2 definierbaren Sprachen [12].

3.1 Die Trotter-Weil-Hierarchie

Innerhalb von **DA** existiert die sogenannte Trotter-Weil-Hierarchie [15]. Sie besteht aus einem „Gitter" von (Pseudo-)varietäten mit unendlich vielen „Stufen". Die Vereinigung aller Stufen ist gleich **DA**. Eine Skizze ist in Abbildung 2 gegeben.

Wir verwenden vor allem die folgenden ω-Terme für die Stufen der Trotter-Weil-Hierarchie:

$$\begin{aligned} U_1 &= (sx_1)^\omega s (y_1 t)^\omega \\ V_1 &= (sx_1)^\omega t (y_1 t)^\omega \\ U_m &= (U_{m-1} x_m)^\omega U_{m-1} (y_m U_{m-1})^\omega \\ V_m &= (U_{m-1} x_m)^\omega V_{m-1} (y_m U_{m-1})^\omega \end{aligned} \tag{6}$$

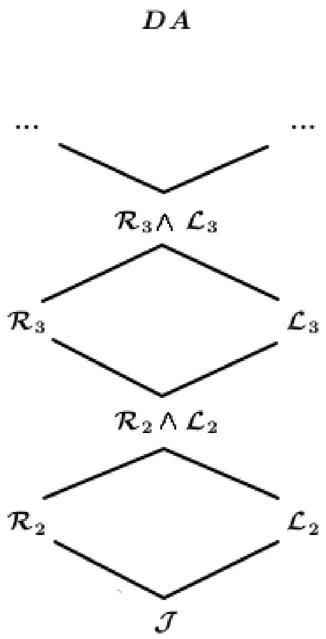

Abbildung 2: Struktur der Trotter-Weil-Hierarchie

Die Stufe W_1 entspricht der Varietät der $\mathcal{J}$-trivialen Monoide auf der untersten Stufe der Hierarchie.
Die m-te Stufe dieses Gitters, der Vereinigung von R_{m+1} und L_{m+1}, bildet eine Varietät W_m mit Identität

$$[[U_m = V_m]]$$

Man sieht leicht, dass alle Stufen von W_m in **DA** liegen. Dazu muss man lediglich für U_1 und V_1 $s = x_1 = x$ und $t = y_1 = y$ setzen, für alle anderen Stufen U_m und V_m seien die Variablen $x_m = y_m = 1$. Dies ergibt direkt die Identität von **DA**.
Wir haben die folgenden Zugehörigkeiten zu den „Ecken“ des Gitters []:

$$\begin{aligned} \mathbf{R_1} = \mathbf{L_1} &= [[U_1 = V_1]] \\ \mathbf{R_m} \cap \mathbf{L_m} &= [[U_{m-1} = V_{m-1}]] \\ \mathbf{R_m} &= [[(U_{m-1}x_m)^\omega U_{m-1} = (U_{m-1}x_m)^\omega V_{m-1}]] \\ \mathbf{L_m} &= [[U_{m-1}(U_{m-1}x_m)^\omega = V_{m-1}(U_{m-1}x_m)^\omega]] \end{aligned} \tag{7}$$

Die Stufen W_m der Trotter-Weil-Hierarchie stimmen mit der Alternierungshierarchie innerhalb von FO^2_m überein. Ein ausführlicher Beweis hierfür findt sich in [16]. Das heißt, dass der syntaktische Monoid einer Sprache, die durch eine FO^2_m-Formel definiert ist (und nicht in FO^2_{m-1} definiert werden kann), in der Varietät W_m liegt (aber nicht in W_{m-1} enthalten ist).

Ein Problem mit dem sich Kapitel 4 beschäftigen wird ist, wie groß der Aufwand ist, um mittels Verbotsmustern festzustellen, auf welcher Stufe der Trotter-Weil-Hierarchie bzw. der Alternierungshierarchie innerhalb von FO^2 sich eine durch einen DFA gegebene Sprache befindet.

4 Verbotsmuster für DA

Es existieren bereits Entscheidungsverfahren für die Varietät **DA**, basierend auf dem syntaktischen Monoid einer Sprache [9] [5]. Da Sprachen jedoch meist in einer einfachen Beschreibungsform wie einem DFA vorliegen, ist es in der Regel nötig, den syntaktischen Monoiden zu berechnen, bevor man ein Entscheidungsverfahren aufgrund des syntaktischen Monoids durchführen kann. Dies kann sehr aufwändig sein. Der syntaktische Monoid eines Automaten mit n Zuständen hat $n!$ Elemente haben. Die Multiplikationstabelle des Monoiden ist quadratisch hierzu.
Ein Entscheidungsverfahren unter Zuhilfenahme von Verbotsmustern kann direkt mit DFAs arbeiten. Es bietet somit Vorteile für alle Fälle, in denen nur der DFA gegeben ist, oder eine Grammatik bzw. ein regulärer Ausdruck, aus denen sich leicht ein DFA erzeugen lässt.

Für die ω-Terme der Identität von **DA** erhält man folgendes Verbotsmuster:

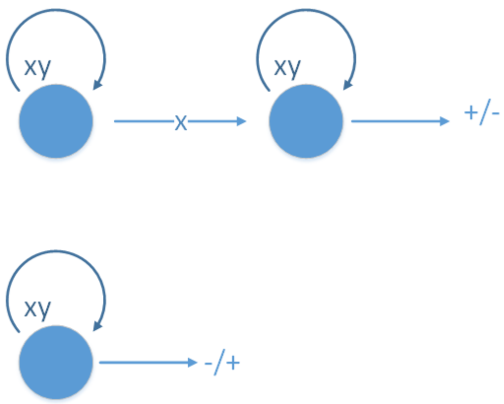

Abbildung 3: Verbotsmuster P_{DA}

Sei der Monoid $M \in \mathbf{DA}$ für eine Sprache L gegeben. Der akzeptierende DFA für L sei A. Man nehme an, A enthält das Verbotsmuster. Es gibt also Worte $p(xy)^\omega x(xy)^\omega q$ und $p(xy)^\omega q$, mit gleichen Prefix p bzw. Suffix q, die einmal in einen Endzustand führen und einmal nicht (denn A ist minimal und die Zustände 1 und 2 verschieden, es gibt also ein Wort w, das sie unterscheidet; wir den Suffix $q = w$).
Damit müssen diese Wörter auf unterschiedliche Elemente in M abgebildet werden. Da Prefix und Suffix gleich sind (und M assoziativ ist), kann es nur im mittleren Teil mit den ω-Termen dazu kommen. Diese ergeben aber nach

der Annahme $M \in \mathbf{DA}$ für die Wörter $(xy)^\omega x(xy)^\omega$ und $(xy)^\omega$ das gleiche Element aus M. Es existiert also kein erkennender Homomorphismus, falls A das Verbotsmuster nicht vermeidet.

Sei nun A der DFA einer Sprache L, und A vermeidet das Verbotsmuster. Der syntaktische Monoid von L sei M. Für alle Wörter $p(xy)^\omega x(xy)^\omega q$ und $p(xy)^\omega q$, mit beliebigen Wörtern als Belegung für Variablen $x, y, p, q \in \Sigma^*$ gilt also

$$p(xy)^\omega x(xy)^\omega q \in L \leftrightarrow p(xy)^\omega q \in L$$

denn der Automat vermeidet das Verbotsmuster. Damit liegen beide Wörter immer in der selben Äquivalenzklasse von $\sim_L$. Da man den syntaktischen Monoiden auch als $Synt(L) = \Sigma^*/\sim_L$ erhält (siehe Kapitel 2.1.3), müssen beide Wörter durch den erkennenden Homomorphimus h auf das selbe Element von M abgebildet werden.

4.1 Verbotsmuster für Stufen der Trotter-Weil-Hierarchie

Nachdem entschieden ist, ob eine Sprache in **DA** liegt, kann es von Interesse sein, auf welcher Ebene der Trotter-Weil-Hierarchie (und damit auch der Alternierungshierarchie in FO^2_m) sie liegt. Dazu braucht man Verbotsmuster für jede Stufe der Varietäten W_m, die iterativ aufgebaut sind. Aus dem ω-Term der ersten Stufe der Trotter-Weil-Hierarchie erhalten wir das folgende Verbotsmuster:

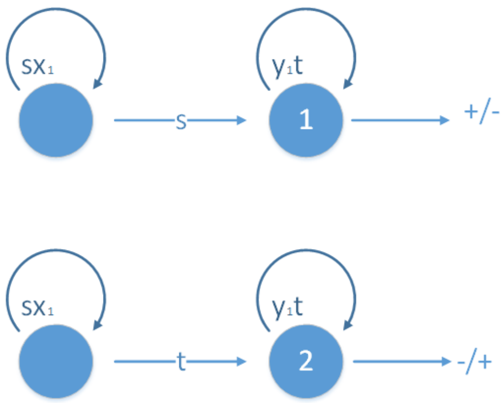

Abbildung 4: Verbotsmuster P_1

Um iterative Verbotsmuster P_i zu definieren müssen wir zuerst den erweiterten Zustandübergangsgraph dahingehend erweitern, dass Kanten mit

einem Muster „beschriftet“ sein können bzw. diese Kanten markieren, denn der erweiterten Zustandübergangsgraph enthält bereits Kanten mit Wörtern beliebiger Länge. Wir verbinden also zwei Zustände, zwischen denen ein bestimmtes Muster auftritt. Dies ist im Grunde identisch mit der Suche danach, ob der Automat ein Verbotsmuster vermeidet.
Da die Verbotsmuster für die Stufen von W_m, die wir verwenden, für jede Seite des ω-Terms einen Teilgraph enthalten, suchen wir für das Muster beider Teilgraphen jeweils seperat. Jeder der beiden Teilgraphen besteht immer aus zwei Teilen $(U_{i-1}x_i)$ und (y_iU_{i-1}), die jeweils durch einen Mittelteil $(U_{i-1}$ bzw. $(V_{i-1}$ ohne ω verbunden sind. Wir fügen eine Kante vom gemeinsamen „Startzustand“ des Verbotsmusters (zu dem uns ein beliebiger Prefix p und $(U_{i-1}x_i)^\omega$ in eine $(U_{i-1}x_i)$-Schleife führt) zum „Endzustand“ d.h. zu dem Zustand, an dem man nach Eingabe des Mittelteils $(U_{i-1}$ bzw. $(V_{i-1}$ des Verbotsmusters durch lesen von $(y_iU_{i-1})^\omega$ in eine (y_iU_{i-1})-Schleife läuft.

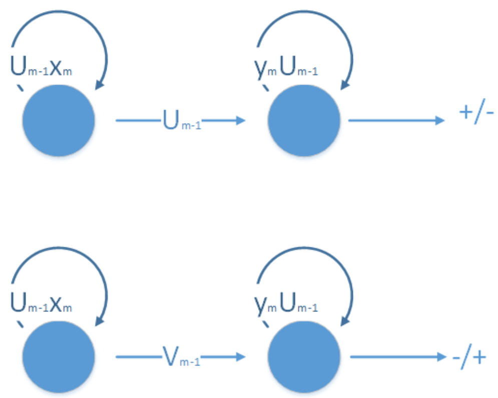

Abbildung 5: Verbotsmuster P_m

Mit den Beschriftungen für die Muster U_i bzw. V_i der vorherigen Stufe von W_m kann man diese nun wie normale Wörter behandeln bzw. es ist bekannt welche Zustände von ihnen verbunden werden.

Ein Algorithmus zur Suche nach den Verbotsmustern könnte wie folgend aussehen (Skizze):

1. Rate Buchstaben aus Σ für die Wörter, die die Variablen des Verbotsmusters belegen sollen. Die Wörter müssen endlich sein, also auch das Wortende wird geraten.

2. Von jedem Zustand des erweiterten Zustandübergangsgraphen aus kann man gleichzeitig die Eingabe lesen (alle Zustände sind duch einen Prefix

p erreichbar). Man geht also für den ersten ω-Teil des Verbotsmusters von jedem Zustand aus solange, bis man in eine Schleife an einem Zustand z_1 läuft. (Man merkt sich diesen Zustand)

3. Von Zustand z_1 aus liest man den Mittelteil des Verbotsmusters, gefolgt von der Eingabe des zweiten ω-Teil des Verbotsmusters. Man läuft wieder in eine Schleife an einem Zustand z_2.

4. Vergleiche z_1 und z_2. Sind sie verschieden wird das Verbotsmuster nicht vermieden und die durch den DFA gegebene Sprache liegt nicht auf Stufe m.
   ((Sind sie nicht verschieden, markiert man sich die Kante von z_1 nach z_2 mit dem Muster, damit man es im Iteratiionschritt verwenden kann.))

5 Zusammenfassung

Es wurde eine Einführung in die grundleggenden Zusammenhänge zwischen regulären Sprachen bzw. deterministischen endlichen Automaten, ihrer Definition mihilfe logischer Formeln und deren algebraischen Beschreibungsformen gegeben. Basierend darauf wurde der Zusammenhang zwischen der Alternierungshierarchie in FO^2 und der Trotter-Weil-Hierarchie innerhalb der Varietät **DA** hergestellt und für die Varietät **DA** bzw. für die Stufen W_m der Trotter-Weil-Hierarchie, Verbotsmuster aufgestellt, und diskutiert, wie effizient diese arbeiten.

Literatur

[1] R. McNaughton and S. Papert. *Counter-Free Automata.* MIT Press, 1971.

[2] M.-P. Schützenberger. On finite monoids having only trivial subgroups. *Information and Control*, 8:190–194, 1965.

[3] J. A. W. Kamp. *Tense Logic and the Theory of Linear Order.* PhD thesis, University of California, 1968.

[4] D. Therien and T. Wilke. Over words, two variables are as powerful as one quantifier alternation, 1998.

[5] M. Kufleitner and P. Weil. The fo2 alternation hierarchy is decidable. *CoRR*, abs/1203.6152, 2012.

[6] J. M. Howie. Semigroups, past, present and future. *Proceedings of the International Conference on Algebra and its Applications*, 2002.

[7] S. Eilenberg. *Automata, Languages and Machines. Pure and Applied Mathmatics.* Academic Press, 1974.

[8] J.E. Pin. Syntactic semigroups, 1997.

[9] M. Kufleitner and A. Lauser. Quantifier alternation in two-variable first-order logic with successor is decidable. *CoRR*, abs/1212.6500, 2012.

[10] M. Kufleitner and P. Weil. On logical hierarchies within fo2-definable languages. *Logical Methods in Computer Science*, 8:1–30, 2012.

[11] J. R. Büchi. Weak second-order arithmetic and finite automata. In *Mathematical Logic Quarterly*, volume 6, page 66–92, 1960.

[12] P. Weis and N. Immerman. Structure theorem and strict alternation hierarchy for fo2 on words. *Logical Methods in Computer Science*, 5(3), 2009.

[13] J. A. Green. On the structure of semigroups. In *Annals of Mathematics*, volume 54, pages 163–172, 1951.

[14] P. Tesson and D. Therien. Diamonds are forever: The variety da. In *Semigroups, Algorithms, Automata and Languages, Coimbra (Portugal) 2001*, pages 475–500. World Scientific, 2002.

[15] P. Trotter and P. Weil. The lattice of pseudovarieties of idempotent semigroups and a non-regular analogue. In *Algebra Univers*, volume 37, pages 491–526, 1997.

[16] Howard Straubing. Algebraic characterization of the alternation hierarchy in fo2[<] on finite words. In Marc Bezem, editor, *Computer Science Logic (CSL'11)*, volume 12 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 525–537. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2011.

Erklärung:

Ich versichere, diese Arbeit selbstständig verfasst zu haben. Ich habe keine anderen als die angegebenen Quellen benutzt und alle wörtlich oder sinngemäß aus anderen Werken übernommene Aussagen als solche gekennzeichnet. Weder diese Arbeit noch wesentliche Teile daraus waren bisher Gegenstand eines anderen Prüfungsverfahrens. Ich habe diese Arbeit bisher weder teilweise noch vollständig veröffentlicht. Das elektronische Exemplar stimmt mit allen eingereichten Exemplaren überein.

(Stuttgart, 18.10.2013, Sebastian Müller)