Automatenbasierte Entscheidungsverfahren für Teilsysteme der Arithmetik

Felix Klaedtke

Departement Informatik, ETH Zürich
felixkl@inf.ethz.ch

Abstract: Vor ungefähr vierzig Jahren wurde entdeckt, dass Automaten ein nützliches mathematisches Werkzeug sind, um die Entscheidbarkeit bestimmter Teilsysteme der Arithmetik zu *verstehen*. Heutzutage werden Automaten auch als Werkzeug eingesetzt, um Entscheidungsverfahren für eben solche logischen Theorien *umzusetzen*. Meine Dissertation behandelt Fragestellungen aus eben diesem Bereich zwischen Automatentheorie und Logik. Ziel dieses Artikels ist es, Einblicke in diesen Themenbereich zu geben und die Beiträge meiner Dissertation zu beschreiben.

1 Einleitung

Es ist allgemein bekannt, dass es kein Verfahren geben kann, um die Wahrheit von arithmetischen Aussagen automatisch zu überprüfen. Selbst Teilprobleme der Arithmetik, wie z.B. das zehnte Hilbertsche Problem sind bekannterweise unentscheidbar. Andererseits kennt man entscheidbare Fragmente der Arithmetik wie z.B. *Presburger-Arithmetik* (die Logik der ersten Stufe über den ganzen Zahlen mit der Ordnungsrelation und der Addition)[1] und *WS1S* (die schwache monadische Logik der zweiten Stufe über den natürlichen Zahlen mit einer Nachfolgerfunktion)[2]. Diese entscheidbaren Logiken besitzen zahlreiche Anwendungen. In der Systemverifikation etwa werden diese Entscheidungsverfahren verwendet, um die Korrektheit eines Systems vollautomatisch zu beweisen oder um Teile eines solchen Korrektheitsbeweises zu automatisieren.

Die Beschäftigung mit monadischen Logiken war anfangs durch Entscheidungsprobleme für Teilsysteme der Arithmetik motiviert und deckte eine enge Verbindung zwischen Automaten und Logiken auf: die Klasse der regulären Sprachen besitzt zusätzlich zur automatentheoretischen Charakterisierung auch eine logische. Durch Übersetzungen in beide Richtungen wurde gezeigt, dass WS1S und Automaten bzgl. der Ausdrucksstärke

[1] Mit Presburger-Arithmetik wird auch häufig die Logik der ersten Stufe über den natürlich Zahlen mit der Addition bezeichnet. Beide Bedeutungen des Begriffs sind äquivalent in dem Sinne, dass sich Aussagen der einen Logik auf einfache Weise in logisch äquivalente Aussagen der anderen Logik übersetzen bzw. interpretieren lassen.

[2] Die Abkürzung WS1S leitet sich aus dem Englischen her: WS1S steht für „weak monadic second-order logic of one successor“.

gleichmächtig sind. Die Entscheidbarkeit von WS1S folgt insbesondere aus der Übersetzung von Formeln in Automaten.

Seit diesem Resultat, welches auf Büchi [Büc60], Elgot [Elg61] und Trakhtenbrot [Tra66] zurückgeht, wurden zahlreiche unterschiedliche Richtungen auf diesem Gebiet zwischen Logik und Automatentheorie verfolgt. So wurde z.B. mit Hilfe von Automatentheorie die Entscheidbarkeit von etlichen logischen Theorien gezeigt (siehe z.B. [Rab77]). Die Frage, welche logische Theorien über automatenbasierte Entscheidungsverfahre verfügen, ist bis heute ein aktives Forschungsgebiet (siehe hierzu [KN95, BG04, Rub04]). Eine andere Richtung erweiterte die Verbindung von Automaten und Logiken zu einem einheitlichem Paradigma für die Spezifikation, Verifikation und Synthese von nicht-terminierenden, nebenläufigen Programmen. So kann z.B. das Modelcheckingproblem für endliche Zustandssysteme als reines automatentheoretisches Problem aufgefasst werden [VW94].

Zahlreiche Algorithmen und Techniken wurden entwickelt, um das Modelcheckingproblem durch Automaten effizienter zu lösen. Viele dieser Algorithmen findet man z.B. in dem Modelchecker SPIN [Hol97] wieder, wie etwa *partial-order reduction* oder *on-the-fly model checking*.

Die Verwendung von Automaten für die algorithmische Umsetzung von Entscheidungsproblemen ist nicht auf das Modelcheckingproblem für endliche Zustandssysteme beschränkt. Vielmehr können Automaten als ein Paradigma dienen, um Entscheidungsverfahren für logische Theorien zu entwerfen und zu mechanisieren. Dieser Ansatz besitzt folgende Vorteile. (1) Eine vielzahl von logischen Theorien lassen sich durch Automaten entscheiden. (2) Automatenbasierte Entscheidungsverfahren sind konzeptionell sauber und einfach. (3) Da Automaten graphenartige Strukturen sind, die sich für eine algorithmisch Behandlung eignen, lassen sich diese Entscheidungsverfahren effektiv umsetzen.

Pionierarbeit zur Umsetzung von automatenbasierten Entscheidungsverfahren für Teilsysteme der Arithmetik sind die Arbeiten [HJJ+96, KMS02]. Hier wurden Datenstrukturen und Algorithmen für Automaten entwickelt, um ein leistungsfähiges Entscheidungsverfahren für die Logik WS1S zu bauen. Motiviert durch die Ergebnisse aus diesen Arbeiten, wurden zahlreiche Methoden und Werkzeuge zur automatischen Systemverifikation von unendlichen Zustandssystemen entwickelt, die auf automatenbasierte Entscheidungsverfahren für logische Theorien zurückgreifen.

Trotz der Vorteile, Erfolge und der hohe praktische Nutze von Automaten in diesem Gebiet, steckt die Umsetzung eben solcher Entscheidungsverfahren noch in den Kinderschuhen und viele grundlegenden Forschungsfragen sind noch unbeantwortet oder nur teilweise beantwortet. Meine Dissertation [Kla04] entwickelt und untersucht Verfahren, die auf Automaten basieren und Teilsysteme der Arithmetik entscheiden und trägt somit direkt und indirekt zur Weiterentwicklung von Werkzeugen zur Systemverifikation bei. Die Dissertation lässt sich in zwei Teile gliedern, die grob durch die Schlagwörter „automatenbasierte Entscheidungsverfahren für Presburger-Arithmetik“ und „monadische Logiken erweitert um Kardinalitätsvergleichen“ umrissen werden können.

Im Folgenden soll nun ausführlicher auf diese beiden Teile der Dissertation eingegangen werden: Kapitel 2 beschäftigt sich mit dem automatenbasierten Ansatz für Presburger-Arithmetik und Kapitel 3 mit einer Erweiterung von monadischen Logiken mit Kardi-

nalitätsvergleichen. Kapitel 4 enthält abschließende Bemerkungen und Ausblicke. Es sei bemerkt, dass wir bei der Darstellung, die grundlegenden Begriffe und Sachverhalte der Automatentheorie und der Logik voraussetzen. Details hierzu findet man z.B. in den Lehrbüchern [HU79] und [End72].

2 Automaten und Presburger-Arithmetik

Presburger-Arithmetik (PA) ist die Logik der ersten Stufe über der Struktur $\mathfrak{Z} = (\mathbb{Z}, +, <)$. Einfache Beispiele von PA-Formeln sind $\exists z.\forall x.x + z = x$ und $\exists y.y + y + y = x$. Die erste Formel ist die wahre Aussage, die besagt, dass ein neutrales Element bzgl. der Addition existiert; die zweite Formel ist genau für die Belegungen der freien Variable x wahr, die durch drei teilbar sind.

Die Entscheidbarkeit von PA wurde um 1930 unabhängig von Mojżesz Presburger [Pre30] und Thoralf Skolem [Sko31] gezeigt. Sowohl Presburger als auch Skolem zeigten die Entscheidbarkeit mittels Quantorenelimination. Die Komplexität des Erfüllbarkeitproblems für PA und die Komplexität von verschiedenen Entscheidungsverfahren ist gut untersucht. Das Erfüllbarkeitsproblem ist vollständig in der Komplexitätsklasse $LATIME\big(2^{2^{\mathrm{O}(n)}}\big)$.[3] Dieses Ergebnis wurde von Berman [Ber80] gezeigt, dessen Beweis auf Erkenntnisse in [FR74] und [FR79] zurückgreift. Es war Oppen, der als erster zeigte, dass PA in nichtelementarer Zeit entscheidbar ist. In seinem Artikel [Opp78] zeigte er, dass die Formel, die durch Coopers Quantoreneliminationsverfahren [Coo72] produziert wird, dreifach exponentiell in der Länge der Eingabeformel beschränkt ist. Hieraus leitet man leicht ein Entscheidungsverfahren für PA ab, welches im schlimmsten Fall eine dreifach exponentielle deterministische Laufzeit hat.

Statt Quantoreneliminationsverfahren können auch Automaten benutzt werden, um PA zu entscheiden. Die Idee des auf Automaten basierenden Ansatzes, ist schnell erklärt: Zahlen werden als Wörter dargestellt. Die Sprache eines Automaten beschreibt nun die erfüllenden Belegungen einer Formel. Abbildung 1 zeigt solch einen DFA für eine PA-Formel. Für eine gegebene Formel wird der Automat rekursiv über den Formelaufbau konstruiert. Z.B. wird das Negationsymbol durch die Komplementierung des Automaten behandelt und die Quantifikation entspricht der Projektion. In Arbeiten wie z.B. [BC96, WB00, KMS02, GBD02, BB03] wurden spezielle Automatenkonstruktionen und Datenstrukturen für automatenbasierte Entscheidungsverfahren für PA entwickelt.

Obwohl der auf Automaten basierende Ansatz schon mindestens seit dem Artikel [Büc60] von Büchi bekannt ist, erfreut sich dieser Ansatz neuerdings einer erhöhten Aufmerksamkeit, da automatenbasierte Entscheidungsverfahren für PA in der Praxis oft gute Ergebnisse bzgl. der Laufzeit liefern [SKR98, BB03, GBD02]. Eine schlüssige Begründung, wieso dieser Ansatz überhaupt praktikabel ist, konnte in all den bisherigen Arbeiten nicht geklärt werden. Eine hiermit verwandte und oft gestellte Frage ist, die Größe der Automaten in Abhängigkeit der Eingabeformel genauer zu bestimmen. Eine naive Komplexitätsanalyse

[3] In $LATIME\big(2^{2^{\mathrm{O}(n)}}\big)$ sind genau die Probleme enthalten, die durch alternierende Turingmaschinen mit maximal linear vielen Alternierungen entschieden werden können und dabei höchstens doppelt exponentiell viel Zeit benötigen.

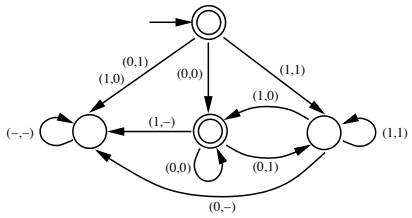

Abbildung 1: Minimaler DFA, der genau die erfüllenden Belegungen in 2'er Komplementdarstellung der Formel $2x - y = 0$ akzeptiert. Die Darstellungen für die Belegungen der einzelnen Variablen werden „synchron" gelesen. So entspricht z.B. das Wort $\binom{0}{1}\binom{1}{1}\binom{1}{0}\binom{0}{1}$ der Belegung, in der x mit 6 und y mit -3 belegt wird. Das Zeichen „–" steht für ein Don't-care, also für eines der beiden Zeichen 0 oder 1. Man beachte, dass Zahlen in der big-endian Darstellung gelesen werden.

des automatenbasierten Ansatzes liefert nur eine nicht-elementare obere Schranke, da jede Quantorenalternierung zu einer potentiellen exponentiellen Vergrößerung des Automaten führen kann.

2.1 Größe der Automaten

Die Autoren des Artikels [BC96] behaupten, dass der minimale DFA für eine PA-Formel höchstens dreifach exponentiell viele Zustände in Abhängigkeit der Formellänge haben kann. Deren Beweis ist allerdings falsch [WB00]. Zwar argumentieren Wolper und Boigelot in ihrem Artikel [WB00], dass es eine elementare obere Schranke für die Automaten für PA-Formeln geben muss, allerdings ist ihre Argumentation vage und weit von einem Beweis entfernt. Meine Dissertation gibt eine Antwort auf die Frage nach der Größe der Automaten für PA-Formeln.

Zum einen wird bewiesen, dass der minimale DFA für eine PA-Formel der Länge n höchstens $2^{2^{2^{O(n)}}}$ viele Zustände hat. Zum anderen wird gezeigt, dass für eine Familie von Formeln die Automaten mindestens dreifach exponentiell viele Zustände haben müssen. Hierbei ist es unerheblich, ob DFAs oder NFAs benutzt werden, um die erfüllenden Belegungen dieser Formeln durch Automaten darzustellen. Dieses worst-case Beispiel zeigt, dass die dreifach exponentielle obere Schranke auch scharf ist. Die obere Schranke bzgl. der Automatengröße für PA ist nicht offensichtlich. Erweitert man PA z.B. um die Funktion $V_2 : \mathbb{N} \setminus \{0\} \to \mathbb{N}$ mit $V_2(x) = \max\{2^k \,:\, 2^k \text{ teilt } x \text{ für ein } k \in \mathbb{N}\}$, so können die Automaten in Abhängigkeit der Formellänge nicht-elementar groß werden.

Alle vorherigen Versuche eine elementare obere Schranke der Automatengröße für PA zu zeigen, sind hauptsächlich deswegen gescheitert, da versucht wurde, direkt über die Struktur und die Konstruktion der Automaten zu argumentieren. Es ist zwar nicht undenkbar, dass dieser Weg zum Ziel führt, allerdings scheint er sehr schwierig zu sein. Hauptschwierigkeit ist die Behandlung von Quantoren. Der Beweis in der Dissertation umgeht diese Hauptschwierigkeit, indem Quantoren zuerst durch ein Quantoreneliminationsverfahren entfernt werden. Der Beweis der oberen Schranke setzt sich nun aus den zwei Teilergebnissen zusammen: (1) eine obere Schranke der Automaten für *quantorenfreie* Formeln und

(2) eine Analyse der Formeln, die entstehen, wenn man Quantoren mittels einem Quantoreneliminationsverfahren entfernt.

Die Beweistechnik Quantoreneliminationsverfahren zu benutzen, um Rückschlüsse auf die Automatengröße zu ziehen, lässt sich auf andere Logiken anwenden. Beispiele sind die Logiken der ersten Stufe über $(\mathbb{R}, +, <)$ und über Queues. Des Weiteren trägt das Aufzeigen der Verbindung von Quantoreneliminationsverfahren und dem automatenbasierten Verfahren zum Verständnis bei, wann automatenbasierte Entscheidungsverfahren besser als Quantoreneliminationsverfahren sind.

2.2 Mechanisierung

Weitere Beiträge der Dissertation, die hier kurz erwähnt werden sollen, betreffen die Umsetzung der Entscheidungsverfahren, die auf Automaten basieren.

Verbesserte Automatenkonstruktion. Die Automatenkonstruktionen aus [WB00, GBD02] für atomare Formeln werden verbessert. Es wird bewiesen, dass diese verbesserten Automatenkonstruktionen optimal sind. Hierbei heißt optimal, dass die erzeugten DFAs minimal sind. Des Weiteren wird die Automatenkonstruktion zur Behandlung von den Junktoren $\vee$ und $\wedge$ optimiert. Experimentelle Ergebnisse zeigen, dass diese verbesserten Automatenkonstruktionen zu einer deutlichen Effizienzsteigerung führen.

Vergleich von Datenstrukturen. Es werden die Datenstrukturen zur Darstellung von Automaten aus [KMS02] und [WB00] verglichen. Es wird argumentiert, dass der auf BDDs beruhende Ansatz aus [KMS02] Vorteile hat. Die Argumente werden durch experimentelle Ergebnisse gestützt.

3 Automaten und monadische Logiken mit Kardinalitätsvergleichen

WS1S ist die monadische Logik der zweiten Stufe über der Struktur $\mathfrak{N} = (\mathbb{N}, +1)$, wobei die Quantifikation über monadische zweite Stufe Variablen auf endliche Teilmengen von $\mathbb{N}$ eingeschränkt ist. Ein einfaches Beispiel einer WS1S-Formel ist die Formel $\exists y.X(y)$, die genau für die Belegungen von X erfüllt wird, in denen X mit einer nicht leeren Menge belegt wird. Ein etwas komplizierteres Beispiel ist die Formel $\forall Z.Z(y) \wedge \big(\forall z.Z(z+1) \rightarrow Z(z)\big) \rightarrow Z(x)$, die die Relation $\leq$ definiert.

Trotz der nicht-elementaren worst-case Komplexität [Mey75, Sto74] des Entscheidungsproblems für WS1S wurde eben diese Logik in zahlreichen Bereichen der Systemverifikation erfolgreich eingesetzt, um Verifikationsaufgaben durch MONA[4] [HJJ+96, KMS02] automatisch zu lösen. Beispiele auf dem Gebiet der Schaltkreisverifikation sind z.B. in [ABF99, BK98] beschrieben und in den Arbeiten [KNS96, SK00] wird MONA zur automatischen Protokollverifikation eingesetzt. Das Entscheidungsverfahren wurde ebenfalls in die Theorembeweiser PVS und Isabelle integriert [BF00, OR00]. Nichtsdestotrotz lassen sich viele Verifikationsprobleme nicht in WS1S ausdrücken und entziehen sich somit

[4] MONA ist eine Implementierung des auf Automaten basierende Entscheidungsverfahren von WS1S.

unentscheidbar	$[\forall_{\text{MSO}}\exists^*_{\text{MSO}}\mathsf{FO}]$	$[\exists_{\text{MSO}}\forall_{\text{FO}}\exists^5_{\text{MSO}}\mathsf{FO}]$
entscheidbar	$[\mathsf{FO}(\exists^*_{\text{MSO}} \cup \forall^*_{\text{MSO}}); \mathsf{R}_{\text{WS1S}}]$	

Abbildung 2: Zusammenfassung der (Un-)Entscheidbarkeitsresultate für WS1S$^{\text{card}}$, wobei in den eckigen Klammern der erlaubte Quantorenpräfix angegeben ist. FO steht hier für eine beliebige Anzahl und Reihenfolge für Quantoren der ersten Stufe.

der direkten Automatisierung durch MONA. Ein häufiger Grund wieso WS1S nicht ausreicht, ist, dass man Aussagen hat wie z.B. „die Anzahl der Elemente x mit der Eigenschaft $P(x)$ ist gleich der Anzahl der Elemente y mit der Eigenschaft $Q(y)$“. Selbst wenn $P(x)$ und $Q(y)$ Eigenschaften sind, die sich in WS1S formulieren lassen, ist WS1S im Allgemeinen zu schwach, um auszudrücken, dass die Mengen $\{x : P(x)\}$ und $\{y : Q(y)\}$ gleich mächtig sind. Um das Einsatzgebiet von monadischen Logiken zu vergrößern, haben wir WS1S erweitert, um diese Art von Aussagen ausdrücken zu können. Wir wollen nun im Folgenden diese Erweiterung und die in der Dissertation hierzu enthaltenen Ergebnisse genauer beschreiben.

3.1 WS1S erweitert mit Kardinalitätsvergleichen

WS1S wird um atomare Formeln der Form $|X_1| + \cdots + |X_r| < |Y_1| + \cdots + |Y_s|$ erweitert, wobei die X_i und Y_i monadische Variablen der zweiten Stufe sind. Solch eine Formel ist genau dann erfüllt, wenn die Summe der Kardinalitäten der Belegungen der Variablen X_i kleiner als die Summe der Kardinalitäten der Belegungen der Variablen Y_i ist. Diese Logik wird im Folgenden mit WS1S$^{\text{card}}$ bezeichnet. Z.B. lässt sich die Aussage „Y enthält mindestens doppelt soviele Elemente wie x“ in WS1S$^{\text{card}}$ durch die Formel $\exists X.|X| + |X| \leq |Y| \wedge \forall z.X(z) \leftrightarrow z < x$ ausdrücken.

Das Erfüllbarkeitsproblem für WS1S$^{\text{card}}$ ist im Allgemeinen unentscheidbar. Ein interessanteres Problem ist es, die Grenze zwischen der Entscheidbarkeit und Unentschscheidbarkeit in WS1S$^{\text{card}}$ genauer aufzuzeigen und ggf. Entscheidungsverfahren für praktisch relevante Fragmente von WS1S$^{\text{card}}$ anzugeben. Dies wurde getan, indem drei Fragmente untersucht wurden. Die Fragmente wurden dabei anhand des Quantorenpräfixes definiert. Abbildung 2 gibt einen Überblick über die (Un-)Entscheidbarkeitsresultate, wobei folgende Notation verwendet wird:

- $[\forall_{\text{MSO}}\exists^*_{\text{MSO}}\mathsf{FO}]$ besteht aus den Sätzen der Form $\forall X \exists Y_1 \ldots \exists Y_r Q_1 x_1 \ldots Q_s x_s \varphi$, wobei φ quantorenfrei ist und $Q_1, \ldots, Q_s \in \{\exists, \forall\}$.
- $[\exists_{\text{MSO}}\forall_{\text{FO}}\exists^5_{\text{MSO}}\mathsf{FO}]$ ist analog zu dem Fragment $[\forall_{\text{MSO}}\exists^*_{\text{MSO}}\mathsf{FO}]$ definiert.
- $[\mathsf{FO}(\exists^*_{\text{MSO}} \cup \forall^*_{\text{MSO}}); \mathsf{R}_{\text{WS1S}}]$ besteht aus den Sätzen der Form $Q_1 x_1 \ldots Q_s x_s Q Y_1 \ldots Q Y_r \varphi$, wobei $Q_1, \ldots, Q_s, Q \in \{\exists, \forall\}$ und φ ist eine WS1S$^{\text{card}}$-Formel, in der die Variablen in Kardinalitätsvergleichen in $\{Y_1, \ldots, Y_s\}$ enthalten sind.

Hierbei stehen Großbuchstaben $X, Y, \ldots$ für monadische Variablen der zweiten Stufe und Kleinbuchstaben $x, y, \ldots$ für Variablen der ersten Stufe.

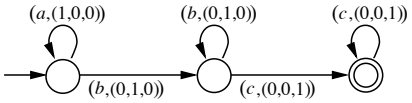

Abbildung 3: Automat des Parikh-Automaten, der die Sprache $L = \{a^n b^n c^n \,:\, n \geq 1\}$ akzeptiert.

Die Unentscheidbarkeit von $[\exists_{\text{MSO}} \forall_{\text{FO}} \exists^5_{\text{MSO}} \mathsf{FO}]$ wird z.B. durch eine Reduktion auf das Halteproblem von Registermaschinen gezeigt. Das Entscheidungsverfahren für $[\mathsf{FO}(\exists^*_{\text{MSO}} \cup \forall^*_{\text{MSO}}); \mathsf{R}_{\text{WS1S}}]$ basiert auf einer Übersetzung von Formeln in *Parikh-Automaten* und einer Übersetzung in PA. Im folgenden Abschnitt wird auf Parikh-Automaten und die Relation zwischen $\text{WS1S}^{\text{card}}$ und Parikh-Automaten etwas genauer eingegangen. Vorher sei aber bemerkt, dass dieses entscheidbare Fragment mächtig genug ist, um Verifikationsaufgaben für parametrisierte sequentielle Schaltkreise automatisch zu lösen, die sich bisher nur für fest gewählte Werte der Parameter mit Hilfe von WS1S und MONA automatisch lösen liesen [BK98].

3.2 Parikh-Automaten

In der Dissertation wird ein Konzept eingeführt, das es erlaubt, Sprachklassen mit so genannten „blinden“ Zählern auszustatten. Dies sind Zähler, die während eines Laufes nicht getestet werden können. Dieses allgemeine Konzept soll hier nur für Automaten vorgestellt werden.

Ein *Parikh-Automat* ist ein Tupel $(\mathcal{A}, C)$, wobei $\mathcal{A}$ ein Automat über einem Alphabet Γ der Form $\Sigma \times D$ mit $D \subseteq \mathbb{N}^n$ ist und $C \subseteq \mathbb{N}^n$ eine PA-definierbare Menge. Die Sprache von $(\mathcal{A}, C)$ ist $L(\mathcal{A}, C) = \{\Psi(w) \,:\, w \in L(\mathcal{A}) \text{ und } \Phi(w) \in C\}$, wobei $L(\mathcal{A})$ die Sprache von $\mathcal{A}$ ist und $\Psi : \Gamma^* \to \Sigma^*$, $\Phi : \Gamma^* \to \mathbb{N}^n$ die Monoid-Homomorphismen, die durch

$$\begin{array}{lll} \Psi(b,d) = d & \Phi(b,d) = d & \text{für } (b,d) \in \Gamma \\ \Psi(uv) = \Psi(u)\Psi(v) & \Phi(uv) = \Phi(u) + \Phi(v) & \text{für } u, v \in \Gamma^+ \end{array}$$

definiert sind. $\Phi(w)$ wird auch *erweitertes* Parikh-Bild von $w \in \Gamma^*$ genannt, da es den Begriff des kommutativen Bildes [Par66] eines Wortes verallgemeinert. Die nicht kontextfreie Sprache $\{a^n b^n c^n \,:\, n \geq 1\}$ wird z.B. durch den Parikh-Automaten $(\mathcal{A}, C)$ erkannt, wobei $\mathcal{A}$ der Automat aus Abbildung 3 ist und $C = \{(z_1, z_2, z_3) \in \mathbb{N}^3 \,:\, z_1 = z_2 = z_3\}$.

Es werden Abschlußeigenschaften dieses neuen Automatenmodell gezeigt und es wird mit anderen Automatenmodellen aus der Literatur verglichen. Des Weiteren wird bewiesen, dass dieses Automatenmodell bzgl. der Ausdrucksmächtigkeit dem existentiellen Fragment von $\text{WS1S}^{\text{card}}$ entspricht. Dies ist eine Übertragung des Theorems von Elgot, Büchi und Trakhtenbrot, welches besagt, dass WS1S und die regulären Sprachen die gleiche Ausdrucksmächtigkeit besitzen. Dieses Resultat zeigt, dass es sich um eine natürliche Sprachklasse handelt, da sie sowohl über eine automatentheoretische als auch eine logische Charakterisierung verfügt.

Die Ergebnisse über die Erweiterung von WS1S werden auf die monadische Logik WS2S übertragen. Insbesondere werden hierzu Parikh-Baumautomaten eingeführt, die eine Ver-

allgemeinerung von Parikh-Automaten darstellen. Ein Anwendungsgebiet für Baumautomaten und damit auch Parikh-Baumautomaten sind z.B. XML-Anfragesprachen, da XML-Dokumente als beschriftete Bäume aufgefasst werden können [KSS03, SSM03].

4 Konklusion und Ausblicke

Meine Dissertation [Kla04] beschäftigt sich mit dem automatenbasierten Ansatz, um praktisch relevante Teilsysteme der Arithmetik zu entscheiden. Es werden wichtige offene Fragen bzgl. der Automatengröße beantwortet und neue Konzepte und Techniken entwickelt, die den Einsatz von Automaten auf diesem Gebiet erweitern und effizienter machen.

Als zukünftige Arbeit soll die Größe von Automaten von weiteren logische Theorien untersucht werden. Hierfür soll die verwendete Methode, die für Presburger-Arithmetik angewendet wurde, auf andere logische Theorien übertragen werden. Naheliegende und praktische relevante Beispiele sind die Logiken der ersten Stufe über $(\mathbb{R}, +, <)$ und über Queues. Des Weiteren sollen die automatenbasierten Entscheidungsverfahren für logische Theorien effizienter gemacht werden, indem Automatenkonstruktionen verbessert werden und neue Datenstrukturen zur Darstellung von Automaten einwickelt werden.

Danksagung: Ich möchte mich an dieser Stelle bei meinem Doktorvater David Basin für seine Unterstützung und Wegleitung bedanken. Ich danke auch der Fakultät für Angewandte Wissenschaften an der Albert-Ludwigs-Universität Freiburg für die hervorragenden Arbeitsbedingungen und das freundliche Arbeitsklima.

Literatur

[ABF99] A. Ayari, D. Basin und S. Friedrich. Structural and Behavioral Modeling with Monadic Logics. In *Proc. of the 29th IEEE Int. Symp. on Multiple-Valued Logic (ISMVL'99)*, Seiten 142–151. IEEE Computer Society Press, 1999.

[BB03] C. Bartzis und T. Bultan. Efficient Symbolic Representations for Arithmetic Constraints in Verification. *Int. J. Found. Comput. Sci.*, 14(4):605–624, 2003.

[BC96] A. Boudet und H. Comon. Diophantine Equations, Presburger Arithmetic and Finite Automata. In *Proc. of the 21st Int. Colloq. on Trees in Algebra and Programming (CAAP'96)*, vol. 1059 of *LNCS*, Seiten 30–43, 1996.

[Ber80] L. Berman. The Complexity of Logical Theories. *Theor. Comput. Sci.*, 11:71–77, 1980.

[BF00] D. Basin und S. Friedrich. Combining WS1S and HOL. In *Proc. of the 2nd Int. Workshop on Frontiers of Combining Systems (FroCoS'98)*, Studies in Logic and Computation, Seiten 39–56. Research Studies Press/Wiley, 2000.

[BG04] A. Blumensath und E. Grädel. Finite Presentations of Infinite Structures: Automata and Interpretations. *Theory Comput. Syst.*, 37:641–674, 2004.

[BK98] D. Basin und N. Klarlund. Automata Based Symbolic Reasoning in Hardware Verification. *Formal Methods in Systems Design*, 13(3):255–288, 1998.

[Büc60] R. Büchi. Weak Second-Order Arithmetic and Finite Automata. *Zeitschrift der mathematischen Logik und Grundlagen der Mathematik*, 6:66–92, 1960.

[Coo72] D. Cooper. Theorem Proving in Arithmetic without Multiplication. *Machine Intelligence*, 7:91–99, 1972.

[Elg61] C. Elgot. Decision Problems of Finite Automata Design and Related Arithmetics. *Transactions of the AMS*, 98:21–51, 1961. Errata 103, 558–559 (1962).

[End72] H. Enderton. *A Mathematical Introduction to Mathematical Logic*. Academic Press, 1972.

[FR74] M. Fischer und M. Rabin. Super-exponential complexity of Presburger arithmetic. In *Symp. on Applied Mathematics*, vol. VII of *SIAM-AMS Proceedings*, Seiten 27–41, 1974.

[FR79] J. Ferrante und C. Rackoff. *The Computational Complexity of Logical Theories*, vol. 718 of *LNM*. Springer-Verlag, 1979.

[GBD02] V. Ganesh, S. Berezin und D. Dill. Deciding Presburger Arithmetic by Model Checking and Comparisons with Other Methods. In *Proc. of the 4th Int. Conf. on Formal Methods in Computer-Aided Design (FMCAD'02)*, vol. 2517 of *LNCS*, Seiten 171–186, 2002.

[HJJ+96] J. Henriksen, J. Jensen, M. Jørgensen, N. Klarlund, R. Paige, T. Rauhe und A. Sandholm. Mona: Monadic second-order logic in practice. In *Proc. of the 1st Int. Workshop on Tools and Algorithms for Construction and Analysis of Systems (TACAS'95)*, vol. 1019 of *LNCS*, Seiten 89–110, 1996.

[Hol97] G. Holzmann. The Model Checker SPIN. *IEEE Trans. Software Eng.*, 23(5):279–295, 1997.

[HU79] J. Hopcroft und J. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley, 1979.

[Kla04] F. Klaedtke. *Automata-based Decision Procedures for Weak Arithmetics*. Dissertation, Albert-Ludwigs-Universität Freiburg, Fakultät für Angewandte Wissenschaften, Institut für Informatik, Freiburg i. Br., 2004.

[KMS02] N. Klarlund, A. Møller und M. Schwartzbach. MONA Implementation Secrets. *Int. J. Found. Comput. Sci.*, 13(4):571–586, 2002.

[KN95] B. Khoussainov und A. Nerode. Automatic presentations of structures. In *Proc. of the Int. Workshop on Logical and Computational Complexity (LCC'94)*, vol. 960 of *LNCS*, Seiten 367–392, 1995.

[KNS96] N. Klarlund, M. Nielsen und K. Sunesen. Automated logical verification based on trace abstraction. In *Proc. of the 15th Annual ACM Symp. on Principles of Distributed Computing (PODC'96)*, Seiten 101–110. ACM Press, 1996.

[KSS03] N. Klarlund, T. Schwentick und D. Suciu. XML: Model, Schemas, Types, Logics and Queries. In Jan Chomicki, Ron van der Meyden und Gunter Saake, Hrsg., *Logics for Emerging Applications of Databases*, Kapitel 1, Seiten 1–41. Springer-Verlag, 2003.

[Mey75] A. Meyer. Weak monadic second-order theory of successor is not elementary-recursive. In *Logic Colloq.*, vol. 453 of *LNM*, Seiten 132–154, 1975.

[Opp78] D. Oppen. A $2^{2^{2^{pn}}}$ Upper Bound on the Complexity of Presburger Arithmetic. *J. Comput. Syst. Sci.*, 16:323–332, 1978.

[OR00] S. Owre und H. Rueß. Integrating WS1S with PVS. In *Proc. of the 12th Int. Conf. on Computer Aided Verification (CAV'00)*, vol. 1855 of *LNCS*, Seiten 548–551, 2000.

[Par66] R. Parikh. On Context-Free Languages. *J. ACM*, 13(4):570–581, 1966.

[Pre30] M. Presburger. Über die Vollständigkeit eines gewissen Systems der Arithmetik ganzer Zahlen, in welchem die Addition als einzige Operation hervortritt. In *Sprawozdanie z I Kongresu metematyków slowiańskich, Warszawa 1929*, Seiten 92–101, 395, 1930.

[Rab77] M. Rabin. Decidable Theories. In J. Barwise, Hrsg., *Handbook of Mathematical Logic*, vol. 70 of *Studies in Logic*, Seiten 595–629. North-Holland, 1977.

[Rub04] S. Rubin. *Automatic Structures*. Dissertation, University of Auckland, Auckland, New Zealand, 2004.

[SK00] M. Smith und N. Klarlund. Verification of a Sliding Window Protocol Using IOA and MONA. In *Proc. of the Joint Int. Conf. on Formal Description Techniques for Distributed Systems and Communication Protocols, and Protocol Specification, Testing and Verification (FORTE/PSTV'00)*, vol. 183 of *IFIP Conference Proceedings*, Seiten 19–34. Kluwer Academic Publishers, 2000.

[Sko31] T. Skolem. Über einige Satzfunktionen in der Arithmetik. In *Skrifter utgitt av Det Norske Videnskaps-Akademi i Oslo, I. Matematisk naturvidenskapelig klasse*, vol. 7, Seiten 1–28, Oslo, 1931.

[SKR98] T. Shiple, J. Kukula und R. Ranjan. A comparison of Presburger engines for EFSM reachability. In *Proc. of the 10th Int. Conf. on Computer Aided Verification (CAV'98)*, vol. 1427 of *LNCS*, Seiten 280–292, 1998.

[SSM03] H. Seidl, T. Schwentick und A. Muscholl. Numerical document queries. In *Proc. of the 22nd ACM SIGACT-SIGMOD-SIGART Symp. on Principles of Database Systems (PODS'03)*, Seiten 155–166. ACM Press, 2003.

[Sto74] L. Stockmeyer. *The complexity of decision problems in automata theory and logic*. Dissertation, Department of Electrical Engineering, MIT, Boston, MA, USA, 1974.

[Tra66] B. Trakhtenbrot. Finite automata and the logic of one-place predicates. *AMS, Transl., II. Ser.*, 59:23–55, 1966.

[VW94] M. Vardi und P. Wolper. Reasoning about infinite computations. *Inf. Comput.*, 115(1):1–37, 1994.

[WB00] P. Wolper und B. Boigelot. On the Construction of Automata from Linear Arithmetic Constraints. In *Proc. of the 6th Int. Conf. on Tools and Algorithms for Construction and Analysis of Systems (TACAS'00)*, vol. 1785 of *LNCS*, Seiten 1–19, 2000.

Felix Klaedtke, geboren 1974, studierte Informatik von 1994 bis 2000 mit Nebenfach Mathematik an der Universität Stuttgart und der Albert-Ludwigs-Universität Freiburg. Er begann 2000 seine Dissertation bei Prof. Dr. David Basin, die er im Juli 2004 an der Albert-Ludwigs-Universität Freiburg verteidigte. Während der Promotion war er als wissenschaftlicher Mitarbeiter am Lehrstuhl für Softwaretechnik angestellt und zweimal als International Fellow für je ein halbes Jahr am SRI in Menlo Park, USA. Seit September 2004 forscht er am Departement Informatik an der ETH Zürich.