Proseminar SoSe 08
Perlen der Informatik, Prof. Nipkow
Autor: Armin Heller

Ein Entscheidungsverfahren für die Presburger Arithmetik

Einführung

Die Presburger Arithmetik (PrA) ist ein Axiomensystem der Prädikatenlogik erster Stufe, welches aus der bekannteren Peano-Arithmetik für natürliche Zahlen hervorgeht, indem man die Multiplikation weglässt. Aussagen in der Theorie der PrA sind, im Gegensatz zu jenen aus der Peano-Arithmetik, entscheidbar. Begriffe wie derjenige der Primzahl lassen sich in der PrA nicht formulieren, sie ist echte „Teiltheorie“der Peano-Arithmetik.

Der Beweis der Entscheidbarkeit der PrA wird in dieser Arbeit durch Konstruktion eines Entscheidungsverfahrens geführt. Dazu ist es notwendig, den Begriff des endlichen Automaten einzuführen, auf den im ersten Abschnitt eingegangen wird. Im zweiten Abschnitt wird die PrA konkret als Axiomensystem eingeführt. Danach wird das eigentliche Entscheidungsverfahren beschrieben.

Der endliche Automat

Definition: Ein endlicher Automat ist ein 5-Tupel $(S, \Sigma, \Delta, s_0, F)$.
Dabei ist:

- Σ das endliche Eingabealphabet.
- Σ^* die Menge aller Wörter aus Zeichen des Eingabealphabets und abzählbar. Es ist $\varepsilon \in \Sigma^* \setminus \Sigma$ das leere Wort.
- S der endliche Zustandsraum des Automaten.
- $s_0 \in S$ der Startzustand des Automaten.
- $F \subseteq S$ die Menge der akzeptierenden Zustände.
- $\Delta \subseteq (S \times \Sigma) \times S$ die Übergangsrelation.

Der Begriff „endlicher Automat“ wird mit FA (für finite automaton) abgekürzt.
Es gibt einen wichtige Spezialfall der FAs:

- Ist $\Delta : S \times \Sigma \to S$ eine totale Funktion, so wird der FA als deterministischer endlicher Automat bezeichnet, was mit DFA abgekürzt wird.

Die Hülle der Übergangsrelation, $\hat{\Delta} \subseteq (S \times \Sigma^*) \times S$, ist wie folgt rekursiv definiert:

$$((s, \varepsilon), s) \in \hat{\Delta} \quad \text{für alle} \quad s \in S$$
$$\exists r \in S.((s, w), r) \in \hat{\Delta} \wedge ((r, a), t) \in \Delta \Longrightarrow ((s, wa), t) \in \hat{\Delta}$$

Definition: Die von einem FA $A = (S, \Sigma, \Delta, s_0, F)$ akzeptierte Sprache $L(A)$ wird wie folgt definiert:

$$L(A) := \{w \in \Sigma^* | \exists f \in F.((s_0, w), f) \in \hat{\Delta}\}$$

Man sagt, dass der Automat A ein Wort $w \in \Sigma^*$ akzeptiert, wenn $w \in L(A)$ ist.

Zu gegebenen Automaten A und B über dem Alphabet Σ lassen sich jeweils FAs konstruieren, die $L(A) \cap L(B)$, $L(A) \cup L(B)$, $\Sigma^* \setminus L(A)$ akzeptieren. Desweiteren lässt sich zu jedem nichtdeterministischen FA (NFA) A ein DFA $\mathcal{P}(A)$ konstruieren, mit $L(A) = L(\mathcal{P}(A))$. Diese Konstruktionen sind die Themen der nächsten Abschnitte.

Der Produktautomat

Definition: Es seien

$$A = (P, \Sigma, \Delta, p_0, E)$$
$$B = (Q, \Sigma, \Gamma, q_0, F)$$

endliche Automaten. Dann ist ihr Produktautomat definiert als:

$$A \times B := (P \times Q, \Sigma, \Phi, (p_0, q_0), E \times F)$$

mit

$$\begin{aligned}(((p,q),a),(r,s)) \in \Phi :\Longleftrightarrow \\ ((p,a),r) \in \Delta \\ \wedge((q,a),s) \in \Gamma\end{aligned}$$

Lemma: Es seien A und B FAs mit den Bezeichnungen wie in der Definition des Produktautomaten. Dann gilt für alle $p, r \in P$, $q, s \in Q$ und $w \in \Sigma^*$:

$$(((p,q),w),(r,s)) \in \hat{\Phi} \Leftrightarrow ((p,w),r) \in \hat{\Delta} \wedge ((q,w),s) \in \hat{\Gamma}$$

Beweis:
$\curvearrowright$:
Der Beweis wird mit Induktion nach der Definition von $\hat{\Phi}$ geführt.
Basisfall: Nach Definition von $\hat{\Phi}$ gilt für alle $p \in P, q \in Q$:

$$((p,\varepsilon),p) \in \hat{\Delta} \wedge ((q,\varepsilon),q) \in \hat{\Gamma} \quad \surd$$

Rekursions-Schritt:
Es gelte $(((x,y),a),(r,s)) \in \Phi$ und die Induktionsvorraussetzung:

$$((p,w),x) \in \hat{\Delta} \wedge ((q,w),y) \in \hat{\Gamma}$$

Nach der Definition von Φ gilt dann auch:

$$((x,a),r) \in \Delta \wedge ((y,a),s) \in \Gamma$$

Daraus folgt mit den Definitionen von $\hat{\Gamma}$ und $\hat{\Delta}$:

$$((p,wa),r) \in \hat{\Delta} \wedge ((q,wa),s) \in \hat{\Gamma} \quad \surd$$

Damit ist die Richtung $\curvearrowright$ gezeigt.

$\curvearrowleft$:
Diese Richtung wird mit Induktion nach der Länge $|w|$ des Wortes $w \in \Sigma^*$ gezeigt.
$\underline{|w| = 0}$: Es ist $w = \varepsilon$. Die Voraussetzungen sind:

$$((p, \varepsilon), r) \in \hat{\Delta} \wedge ((q, \varepsilon), s) \in \hat{\Gamma}$$

Die beiden Relationen können nur über die erste Regel entstanden sein, d.h. $p = r$ und $q = s$. Dann folgt aber nach Definition von $\hat{\Phi}$:

$$(((p, q), \varepsilon), (p, q)) \in \hat{\Phi} \qquad \surd$$

$\underline{|w'| = n + 1}$: Dann ist $w' = wa$ für $w \in \Sigma^*$ und $a \in \Sigma$. Die Voraussetzungen sind:

$$((p, wa), r) \in \hat{\Delta} \wedge ((q, wa), s) \in \hat{\Gamma}$$

Diese beiden Relationen müssen durch Anwendung der zweiten Regel der Definition von $\hat{\Delta}$ bzw. $\hat{\Gamma}$ entstanden sein. Es muss also gelten:

$$\begin{aligned} &((x, a), r) \in \Delta \wedge ((y, a), s) \in \Gamma \\ \wedge &((p, w), x) \in \hat{\Delta} \wedge ((q, w), y) \in \hat{\Gamma} \end{aligned}$$

für zwei Zustände $x \in P$, $y \in Q$. Dann folgt nach Induktionsvoraussetzung:

$$(((p, q), w), (x, y)) \in \hat{\Phi}$$

Nach der Definition von Φ hat man auch:

$$(((x, y), a), (r, s)) \in \Phi$$

Aus den letzten beiden Aussagen folgt:

$$(((p, q), wa), (r, s)) \in \hat{\Phi}$$

was zu zeigen war. $\surd$

Damit ist das Lemma bewiesen.

Korollar: Es seien A und B FAs. Dann gilt:

$$L(A \times B) = L(A) \cap L(B)$$

Beweis:
Die Bezeichnungen seien so gewählt, wie in der Definition des Produktautomaten. Es ist zu zeigen:

$$w \in L(A \times B) \Leftrightarrow w \in L(A) \wedge w \in L(B)$$

Es sei also $(((p_0, q_0), w), (e, f)) \in \hat{\Phi}$ für $(e, f) \in E \times F$. Dann ist nach dem Lemma $((p_0, w), e) \in \hat{\Delta}$ und $((q_0, w), f) \in \hat{\Gamma}$. Außerdem ist $e \in E$ und $f \in F$, also $w \in L(A) \wedge w \in L(B)$.

Es seien umgekehrt $((p_0, w), e) \in \hat{\Delta}$ und $((q_0, w), f) \in \hat{\Gamma}$, mit $e \in E$ und $f \in F$. Dann ist nach dem Lemma $(((p_0, q_0), w), (e, f)) \in \hat{\Phi}$ und offensichtlich $(e, f) \in E \times F$. Damit ist $w \in L(A \times B)$. $\surd$

Die Vereinigung

Definition: Es seien

$$A = (P, \Sigma, \Delta, p_0, E)$$
$$B = (Q, \Sigma, \Gamma, q_0, F)$$

FAs. Dann ist ihr Vereinigungsautomat definiert als:

$$A \cup B := (P \times Q, \Sigma, \Phi, (p_0, q_0), (E \times Q) \cup (P \times F))$$

mit

$$\begin{aligned}(((p,q),a),(r,s)) \in \Phi :\Longleftrightarrow \\ ((p,a),r) \in \Delta \\ \wedge((q,a),s) \in \Gamma\end{aligned}$$

Lemma: Es gilt:

$$L(A \cup B) = L(A) \cup L(B)$$

Beweis: Analog zum Produktautomaten. $\surd$

Es ist insbesondere anzumerken, dass $A \cup B$ deterministisch ist, wenn A und B deterministisch sind, wie beim Produktautomaten.

Die Determinierung

Definition: Es sei $A = (Q, \Sigma, \Delta, q_0, F)$ ein FA.
Dann ist $\mathcal{P}(A) := (\mathcal{P}(Q), \Sigma, \Delta', \{q_0\}, \{f \subseteq Q | f \cap F \neq \varnothing\})$, wobei

$$((p,c),q) \in \Delta' :\Longleftrightarrow q = \{y \in Q | \exists x \in p.((x,c),y) \in \Delta\}$$

Lemma Es sei A ein FA. Dann ist $\mathcal{P}(A)$ ein DFA mit

$$L(A) = L(\mathcal{P}(A))$$

Bei einer konkreten Implementierung, die die Determinierung $\mathcal{P}(A)$ von A berechnet, müssen nicht alle Kanten aus $\mathcal{P}(A)$ berücksichtigt werden. Geht man bei der Berechnung von $\{q_0\}$ aus und berechnet konsequent nur die erreichbaren Zustände und entsprechende Übergänge, so erhält man einen DFA $A' = (Q', \Sigma, \delta, \{q_0\}, F')$, mit $Q' \subseteq \mathcal{P}(Q)$, $\delta \subseteq \Delta'$ und $F' \subseteq \{f \subseteq Q | f \cap F \neq \varnothing\}$ und $L(A') = L(A)$, der unter Umständen sehr viel kleiner ausfallen kann.

Das Komplement

Definition: Es sei $A = (Q, \Sigma, \delta, q_0, F)$ ein DFA. Dann ist

$$\overline{A} := (Q, \Sigma, \delta, q_0, Q \setminus F)$$

der zu A komplementäre FA.

Lemma Es sei A ein DFA. Dann ist $\overline{A}$ ein DFA mit

$$L(\overline{A}) = \Sigma^* \setminus L(A)$$

Es ist insbesondere zu beachten, dass A hier ein DFA sein muss. Würde man in der Definition allgemeinere FAs zulassen, so ließe sich leicht ein Gegenbeispiel für das obige Lemma finden.

Die Projektion

Für die PrA ist die speziellere Automaten-Konstruktion der Projektion von Bedeutung. Für sie werden schon gewisse Bedingungen an das zugrunde liegende Alphabet gestellt. Zunächst einmal eine Definition, die den Umgang mit diesen Alphabeten erleichtert.

Definition: Es sei $n \in \mathbb{N}\backslash\{0\}$ und $\Sigma = \zeta^n$ endlich. Für $1 \leq i \leq n$ und $c = (c_1, ..., c_{i-1}, c_i, c_{i+1}, ..., c_n) \in \Sigma$ ist die Projektion der $i$ten Komponente von c definiert als

$$c_{-i} := (c_1, ..., c_{i-1}, c_{i+1}, ..., c_n)$$

Definition: Es sei $n \in \mathbb{N} \setminus \{0\}$ und $\Sigma = \zeta^n$ endlich. Dann ist für den FA $A = (Q, \Sigma, \Delta, q_0, F)$ und $1 \leq i \leq n$ der FA $A_{-i} = (Q, \zeta^{n-1}, \Delta_{-i}, q_0, F)$ definiert mit

$$((p, d), q) \in \Delta_{-i} :\Longleftrightarrow \exists c \in \Sigma.((p, c), q) \in \Delta \wedge d = c_{-i}$$

Einfach gesagt ergibt sich die Projektion der $i$ten Komponente aus A indem man bei jedem Übergang die $i$te Komponente des zugehörigen Zeichens streicht. Es ist dabei zu beachten, dass das Ergebnis der Projektion kein DFA sein muss, wenn der ursprüngliche Automat ein DFA war, da durch das Streichen der $i$ten Komponente verschiedene Zeichen gleich werden können.

Abschließend Bemerkungen zu den Automatenkonstruktionen:
Im schlimmsten Fall lässt sich bei der Potenzmengenkonstruktion nicht vermeiden, dass exponentiell viele neue Zustände entstehen, genau wie sich bei der Projektion nicht vermeiden lässt, dass der Ergebnis-Automat nicht-deterministisch wird. Für die Komplementbildung ist aber die Determinierung unerlässlich. Verschachtelt man also Potenzmengen-Konstruktion, Komplementbildung und Projektion, so können sehr ineffiziente Berechnungsverfahren entstehen. In speziellen Fällen lässt sich allerdings der Aufwand verringern, indem man die auftretenden Automaten minimiert. Die Konstruktion des Minimalautomaten wird hier nicht behandelt, sie sollte jedoch in einer vernünftigen Implementierung dieses Entscheidungsverfahrens der PrA nicht fehlen.

Die Axiome der Presburger Arithmetik

Die PrA wird durch folgendes Axiomensystem charakterisiert:

$$\forall x.\, Sx \neq 0 \tag{1}$$
$$\forall x.\, x = 0 \vee \exists y.\, x = Sy \tag{2}$$
$$\forall x\, y.\, Sx = Sy \rightarrow x = y \tag{3}$$
$$\forall x.\, x + 0 = x \tag{4}$$
$$\forall x\, y.\, x + Sy = S(x + y) \tag{5}$$

Dabei ist S ein einstelliges Funktionszeichen, 0 ist ein nullstelliges Funktionszeichen (= Konstante). + ist ein zweistelliges Funktionszeichen, das infix geschrieben wird (Operator).

Dazu kommt das Axiomenschema der vollständigen Induktion:

$$\Psi[0/x] \wedge (\forall y.\Psi[y/x] \rightarrow \Psi[Sy/x]) \rightarrow \forall x.\Psi \quad (Ind)$$

Dabei ist Ψ eine beliebige Formel. Selbstverständlich sollen die Ersetzungen für gebundene Variablen x nicht durchgeführt werden.

Die zugrunde liegende Logik ist die Prädikatenlogik erster Stufe mit Identität. In dieser Logik kann z.B. die Umkehrung von (3) ohne die obigen Axiome bewiesen werden. (Man darf „Gleiches durch Gleiches ersetzen“.) Zu diesen Axiomen dürfen *keine* rekursiven Definition hinzugenommen werden! Definitionen, die lediglich ersetzenden Charakter haben, sind jedoch erlaubt, da jede Formel, die so definierte Zeichen enthält, in eine äquivalente Formel ohne diese Zeichen (durch Rückersetzung) transformiert werden kann.

Lineare diophantische Gleichungen

Obwohl sich die Multiplikation von Zahlen mit der Presburger Arithmetik nicht beschreiben lässt, so kann man doch für die Summe fixer Größe von lauter gleichen Summanden eine Abkürzung der folgenden Form einführen:

$$\underbrace{x + \cdots + x}_{\text{n Summanden}} = n \cdot x$$

Der Haken an dieser Notation ist, dass n immer eine konstante Zahl sein muss, über n kann beispielsweise nicht quantifiziert werden. Diese Notation führt allerdings zu einer kompakten Schreibweise einer bestimmten Sorte von Gleichungen:

$$a_1 \cdot x_1 + \cdots + a_n \cdot x_n = c$$

wobei $a_1, ..., a_n, c$ Konstanten sind, $x_1, ..., x_n$ Variablen. Diese Gleichungen sind die sogenannten linearen diophantischen Gleichungen. Ihre Lösungen können von endlichen Automaten erkannt werden. Hierbei kann man sogar negative a_i zulassen, da $\phi - |a_i|x_i = c$ bedeutungsgleich ist mit $\phi = |a_i|x_i + c$, was sich wieder mit der Presburger Arithmetik ausdrücken lässt.

Lösungen der linearen diophantischen Gleichungen durch FAs

Um eine lineare diophantische Gleichung in einen endlichen Automaten zu verwandeln, überlegt man sich zuerst, dass die Gleichung

$$a_1 \cdot x_1 + \cdots + a_n \cdot x_n = c$$

insbesondere modulo 2 erfüllt sein muss. Man kann also die Menge aller Lösungen dieser Gleichung einschränken, indem man das am weitesten rechts stehende Bit der Zahlen x_i bestimmt. Da es nur endlich viele Möglichkeiten gibt, wie diese Bits gesetzt werden können, kann man die Menge aller dieser Bit-Tupel, die die Gleichung modulo 2 erfüllen, in endlich vielen Schritten berechnen.

Das Verfahren kann dann fortgesetzt werden für das zweite Bit, indem man die Summe der am weitesten rechts stehenden Bits für jede Lösung der Gleichung modulo 2 auf die rechte

Seite bringt, und beide Seiten durch 2 teilt. Sind $b_1, ..., b_n$ die Bits, die die Gleichung modulo 2 erfüllen, und $x_1, ..., x_n$ die Reste der Lösungen, so ergibt sich:

$$a_1x_1 + ... + a_nx_n = \frac{c - a_1b_1 - ... - a_nb_n}{2}$$

Diese Idee lässt sich formaler durch einen Algorithmus beschreiben:
Die Zustandsmenge des gesuchten Automaten ist Q, eine Menge von linearen diophantischen Gleichungen. Das Alphabet ist $\{0,1\}^n$.

- "$a_1x_1 + ... + a_nx_n = c$" $\in Q$, d.h. die Ausgangsgleichung ist in Q.
- Es sei "$a_1x_1 + ... + a_nx_n = k$" $\in Q$ und $b = (b_1, ..., b_n) \in \{0,1\}^n$. Es sind zwei Fälle zu unterscheiden:
  - $k - a_1b_2 - ... - a_nb_n \equiv 0 \mod 2$:
    dann gehört der Übergang "$a_1x_1 + ... + a_nx_n = k$" $\rightarrow^b$ "$a_1x_1 + ... + a_nx_n = \frac{c-a_1b_1-...-a_nb_n}{2}$" zum Automaten.
  - $k - a_1b_2 - ... - a_nb_n \equiv 1 \mod 2$:
    dann gehört der Übergang "$a_1x_1 + ... + a_nx_n = k$" $\rightarrow^b \perp$ zum Automaten, wobei $\perp \in Q$ ist, sodass $\perp$ ein Fangzustand ist, d.h. alle Übergänge von $\perp$ führen wieder nach $\perp$ und $\perp$ ist nicht akzeptierend.

Es ist noch zu zeigen, dass bei dieser Beschreibung des Automaten Q endlich ist. Angenommen $|k| > \sum_{i=1}^{n} |a_i|$. Dann ist für alle Folgezustände die rechte Seite der Gleichung betragsmäßig kleiner als $|k|$, da sie maximal um $|k| - 2$ vergrößert, und dann halbiert wird. Also ist der Betrag der rechten Seite aller Zustandsgleichungen durch $\max\{c, \sum_{i=1}^{n} |a_i|\}$ beschränkt, d.h. es gibt nur endlich viele Gleichungen in Q.

Umwandlung allgemeiner Formeln

Da alle atomaren Formeln in der Presburger Arithmetik Gleichungen sein müssen, und diese nur aus $0, S, +$ und Variablen bestehen können, kann man die atomaren Formeln in äquivalente lineare diophantische Gleichungen transformieren, indem man die Konstanten zusammenfasst, die Variablen mit der Anzahl ihres Auftretens als einen Summanden schreibt, und dann die Konstanten von den Variablen trennt. Für diese Gleichungen kann man dann die endlichen Automaten erstellen, die die entsprechenden Lösungen akzeptieren.
Ausgehend von den atomaren Formeln berechnet man rekursiv den Gesamtautomaten. Dafür seien ϕ und ψ Formeln und $A(\phi)$ bzw. $A(\psi)$ die entsprechenden Automaten. Dann erhält man daraus als weitere Zusammenfassung:

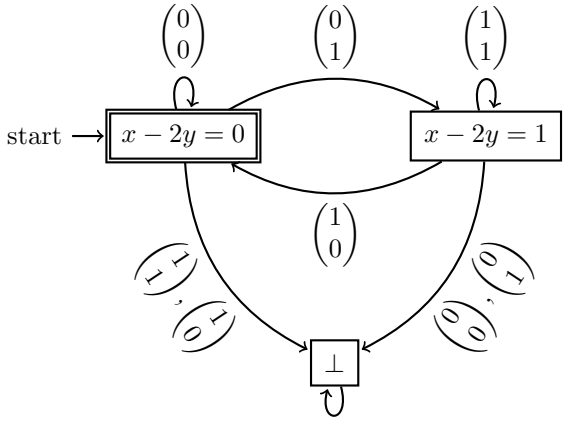
- $A(\phi \wedge \psi) = A(\phi) \times A(\psi)$
- $A(\phi \vee \psi) = A(\phi) \cup A(\psi)$
- $A(\neg\phi) = \overline{\mathcal{P}(A(\phi))}$
- $A(\exists x.\phi) = Repair(A(\phi)_{-i})$, wobei die $i$te Komponente x entspricht. Die Konstruktion *Repair* wird unten erläutert.
- $A(\forall x.\phi) = A(\neg\exists x.\neg\phi)$

Der resultierende Automat akzeptiert dann genau diejenigen Belegungen der freien Variablen, die die Formel wahr machen. Enthält die Formel keine freien Variablen, so muss man im Automaten nur nach einem akzeptierenden Endzustand suchen, der erreichbar ist. Findet man so einen, ist die Formel wahr, ansonsten ist sie falsch.

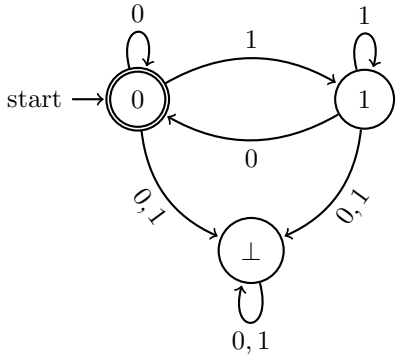
Führende Nullen

In der Formel $\exists x.\phi$ interessiert man sich nicht mehr für den Wert der Variablen x, es reicht, wenn es einen Wert von x gibt, der ϕ erfüllt. Insofern ist es sehr intuitiv in den Tupeln der Lösungszeichen einfach die zu x gehörende Komponente zu streichen, da die konkreten Werte nicht mehr von Interesse sind. Dabei kann es aber passieren, dass der resultierende Automat Lösungen nur dann akzeptiert, wenn sie mindestens eine bestimmte Länge haben, es müssen führende 0en in der Lösung auftauchen. Dazu folgendes Beispiel für die Formel $\exists x.x - 2y = 0$:

Als erstes der Automat, der Lösungen von $x - 2y = 0$ erkennt:



Nach Entfernen, der ersten Komponente (diese entspricht gerade x), erhält man folgenden Automaten:



Eigentlich sollte der Automat unabhängig von der Wahl von y immer akzeptieren, er erzwingt jedoch eine führende 0. Dieses Problem kann umgangen werden, wenn man eine Nachbereitung *Repair* auf dem Automaten ausführt.

Definition: Es sei A ein FA über $\{0,1\}^n$. Dann ist $Repair(A)$ der Automat, der aus A hervorgeht, dadurch, dass man zu jedem Endzustand f alle Zustände, die über 0-Pfade in f führen, zu Endzuständen macht. Dabei sind 0-Pfade Pfade im Graphen des Automaten, die nur über das Zeichen $(0, ..., 0)$ führen.

Offensichtlich ist $Repair(A(a_1x_1 + ... + a_nx_n = c)) = A(a_1x_1 + ... + a_nx_n = c)$. Nach Projektionen kann *Repair* aber durchaus von der Identität verschieden sein. Im obigen Beispiel wird der Zustand 1 von *Repair* auf „akzeptierend“ gesetzt, womit das Problem mit der führenden 0 gelöst ist.

Komplexität des Verfahrens

Die einzige Automaten-Konstruktion, die exponentielle Zeit benötigt, ist die Teilmengenkonstruktion, d.h. die Determinierung. Da die Projektion aber im allgemeinen nicht-deterministische Automaten produziert, und die Negation Determinierung erzwingt, lässt sich folgender worst-case konstruieren:

$$\psi = \neg\exists x_1.\neg\exists x_2....\neg\exists x_h.\phi$$

Angenommen, $A(\phi)$ hat n Zustände. Dann hat $A(\psi)$

$$2^{\left(2^{\left(\cdot^{\cdot^{\cdot^{(2^n)}}}\right)}\right)}$$

Zustände, wobei h die Höhe des Potenzturmes ist. Das Verfahren braucht dafür entsprechend lange. Führt man jedoch nach jeder Determinierung eine Minimierung des Automaten durch, so lässt sich beweisen, dass das Verfahren in der Laufzeit durch

$$O\left(2^{2^{2^{|\psi|}}}\right)$$

beschränkt wird. Dabei ist $|\psi|$ ein Maß für die „Komplexität“ der Formel ψ. (nachzulesen in [1])

Literatur

[1] Alexandre Boudet and Hubert Comon. Diophantine equations, presburger arithmetic and finite automata. In Kirchner [2], pages 30–43.

[2] Hélène Kirchner, editor. *Trees in Algebra and Programming - CAAP'96, 21st International Colloquium, Linköping, Sweden, April, 22-24, 1996, Proceedings*, volume 1059 of *Lecture Notes in Computer Science*. Springer, 1996.

[3] Nils Klarlund. Mona & fido: The logic-automaton connection in practice. In Nielsen and Thomas [4], pages 311–326.

[4] Mogens Nielsen and Wolfgang Thomas, editors. *Computer Science Logic, 11th International Workshop, CSL '97, Annual Conference of the EACSL, Aarhus, Denmark, August 23-29, 1997, Selected Papers*, volume 1414 of *Lecture Notes in Computer Science*. Springer, 1998.

[5] Christoph Wagner. Automatenbasierte Entscheidungsverfahren für Presburger Arithmetik. Diplomarbeit, Technische Universität Kaiserslautern, 2006.