Boolesche Operationen auf Wort- und Baum-Automaten mit kurzen Schleifen

Johannes Waldmann

Institut für Informatik, Universität Leipzig
Augustusplatz 10-11, D-04109 Leipzig, Germany
joe@informatik.uni-leipzig.de
http://www.informatik.uni-leipzig.de/~joe/

Zusammenfassung Wir betrachten hier Wort- und Baum-Automaten, in deren Zustands-Graphen nur Schleifen der Länge eins vorkommen. Wir zeigen, daß die so festgelegte Sprachklasse unter booleschen Operationen abgeschlossen ist. Ein Hauptpunkt der vorliegenden Arbeit ist ein Algorithmus, der Komplemente berechnet. Die Sprachen beschreiben wir dabei durch eingeschränkte reguläre Ausdrücke, und geben die Algorithmen als darauf operierende Termersetzungssysteme an.

1 Einleitung

Die Untersuchung von Abschlußeigenschaften von Baumsprachen unter Term-Ersetzung motiviert die Beschäftigung mit interessanten Teilklassen regulärer Sprachen. Reguläre Sprachen von Wörtern oder Bäumen werden durch endliche Automaten beschrieben. Wir betrachten hier Automaten, in deren Graphen nur Schleifen der Länge eins vorkommen, und auch von diesen nicht beliebige.

Die einfachsten endlichen Wort-Automaten sind solche für *endliche* Sprachen. Sie enthalten keine Schleifen: wir können die Zustandsmenge Q eines solchen Automaten A derart ordnen, daß bei jeder Transition $q \stackrel{a}{\to} q'$ gilt: $q > q'$.

Wenn wir in akzeptierenden Zuständen q Schleifen $q \stackrel{\Sigma}{\to} q$ zulassen, erhalten wir *definite* Sprachen. (Siehe Beispiel 1 im folgenden Abschnitt.) Beide Klassen (endliche, definite Sprachen) sind abgeschlossen unter Booleschen Operationen.

Es liegt nahe, Schleifen der Länge eins in jedem Zustand zu erlauben. Die Klasse der durch solche Automaten erzeugten Sprachen ist aber nicht mehr Boolesch abgeschlossen. (Siehe Beispiel 2.) Wir zeigen in Abschnitt 5, daß wir diesen Abschluß zurückgewinnen, wenn Schleifen in inneren Zuständen immer mit nur einem Buchstaben beschriften sein müssen. In den darauf folgenden Abschnitten übertragen wir dieses Resultat von Wörtern auf Bäume. Wir stellen dabei Sprachen nicht durch Automaten, sondern durch eingeschränkte reguläre Ausdrücke dar, wobei die Einschräkung die Kürze der Schleifen ausdrückt.

2 Beispiele

Wörter. Wir fixieren das Alphabet $\Sigma = \{a, b\}$.

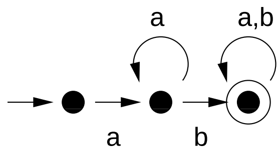
1.) Das Komplement der endlichen (sogar einelementigen) Sprache ab ist

$$\epsilon \cup b\Sigma^* \cup a(\epsilon \cup a\Sigma^* \cup b(a\Sigma^* \cup b\Sigma^*)),$$

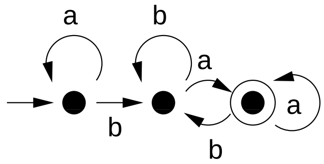
eine definite Sprache.

2.) Die Sprache $X = \Sigma^*(aa \cup bb)\Sigma^* \cup b\Sigma^* \cup \Sigma^* a$ hat offensichtlich einen Automaten mit kurzen Schleifen (wir brauchen je eine für die Σ^*). Das Komplement von X ist $(ab)^*$, mit einer Schleife der Länge 2.

3.) Die Sprache $Y = aa^*b\Sigma^*$ hat einen Automaten mit kurzen Schleifen. Er ist sogar deterministisch.



Hingegen enthält der minimale deterministische Automat für die gespiegelte Sprache $Y_M = \Sigma^* ba^* a$ eine Schleife der Länge 2.



Bäume. Die Signatur hat ein binäres Symbol $\cdot$ und ein Blattsymbol b.

4.) Betrachte die von Z in der folgenden Grammatik erzeugte Sprache:

$$Z \to Z \cdot b,\ Z \to M,\ M \to (b \cdot b) \cdot *,$$

wobei $*$ die Menge aller Bäume erzeugt. Die Regel $Z \to Z \cdot b$ ist offenbar eine kurze Schleife. Besitzt das Komplement von Z eine solche Darstellung?

5.) Das hier benutze M ist eine definite Baumsprache, ihr Komplement ist

$$\overline{M} = b \cup b \cdot * \cup (* \cdot (* \cdot *)) \cdot * \cup ((* \cdot *) \cdot *) \cdot *,$$

und sowohl M als auch $\overline{M}$ sind mit kurzen Schleifen darstellbar. (Wir brauchen ja nur eine Schleife in dem Zustand, der $*$ erzeugt.)

Das Komplement von Z ist aber *nicht* die Sprache

$$Z' \to Z' \cdot b,\ Z' \to \overline{M},$$

denn $(b \cdot b) \cdot b \in (b \cdot b) \cdot * \subseteq Z$, aber auch $(b \cdot b) \cdot b \in (b \cdot *) \cdot b \subseteq \overline{M} \cdot b \subseteq Z'$.

Wir werden einen Algorithmus angeben, der eine Darstellung von $\overline{Z}$ mit kurzen Schleifen berechnen kann.

6.) Wir bemerken, daß Z nicht top-down-deterministisch ist und der minimale deterministische bottom-up-Automat für Z

$$b \to q_b, \quad q_b \cdot q_b \to q_1$$

$$\left.\begin{array}{c} q_1 \cdot q_2 \ \cup\ q_2 \cdot q_2 \ \cup\ q_2 \cdot Z \ \cup\ q_2 \cdot q_b \ \cup\ Z \cdot q_1 \\ \cup\ Z \cdot q_2 \ \cup\ Z \cdot Z \ \cup\ q_b \cdot q_1 \ \cup\ q_b \cdot q_2 \ \cup\ q_b \cdot Z \end{array}\right\} \to q_2,$$

$$q_1 \cdot q_1 \ \cup\ q_1 \cdot q_2 \ \cup\ q_1 \cdot Z \ \cup\ q_1 \cdot q_b \ \cup\ Z \cdot q_b \to Z$$

eine Schleife der Länge 2 enthält (zwischen den Zuständen q_2 und Z).

3 Reguläre Ausdrücke für Wortsprachen

Wir interessieren uns für Wortautomaten, in deren Zustandsgraphen nur kurze Schleifen vorkommen. Wir charakterisieren diese Sprachklasse durch passend eingeschränkte reguläre Ausdrücke.

Definition 1 *Die Menge* SLWX *von Ausdrücken über einem Alphabet* Σ *ist*

1. *(leeres Wort:)* $\epsilon \in \mathsf{SLWX}$, *(leere Menge:)* $\emptyset \in \mathsf{SLWX}$, *(alle Wörter:)* $\Sigma^* \in \mathsf{SLWX}$,
2. *(Vereinigung:) für alle* $M \in \mathsf{SLWX}, N \in \mathsf{SLWX}$: $M \vee N \in \mathsf{SLWX}$
3. *(Verkettung, Stern:)*
   für alle $a \in \Sigma, M \in \mathsf{SLWX}$: $a \cdot M \in \mathsf{SLWX}$ *und* $a^* \cdot M \in \mathsf{SLWX}$

Die letzte Zeile erklärt für jedes Zeichen $a \in \Sigma$ zwei einstellige Funktionssymbole $a\cdot$ und $a^*\cdot$. Den Punkt $(\cdot)$ lassen wir im folgenden weg.

Auf offensichtliche Weise beschreibt jeder Ausdruck $X \in \mathsf{SLWX}$ eine reguläre Wortsprache $\mathsf{Lang}(X)$. Die Menge der so beschreibbaren Sprachen nennen wir SLWL.

Es ist klar, daß die Menge SLWL abgeschlossen ist unter Vereinigung. Sie ist auch abgeschlossen unter Durchschnitt. Zum Beweis dieser Behauptung konstruieren wir aus SLWX die Menge SLWX_1, indem wir zusätzlich das Symbol $\wedge$ zulassen. Dann geben wir ein Term-Ersetzungs-System R_1 an, bezüglich dessen jeder Ausdruck $M \in \mathsf{SLWX}_1$ ein äquivalente Normalform $N \in \mathsf{SLWX}$ besitzt.

Das System R_1 besteht aus

1. $\emptyset \wedge X \to \emptyset$,
2. $\Sigma^* \wedge X \to X$,
3. $(X \vee Y) \wedge Z \to (X \wedge Z) \vee (Y \wedge Z)$,
4. $\epsilon \wedge aX \to \emptyset$,
5. für alle $a \in \Sigma$: $aX \wedge aY \to a(X \wedge Y)$,
6. für alle $a, b \in \Sigma$ mit $a \neq b$: $aX \wedge bY \to \emptyset$,
7. für alle $a \in \Sigma$: $a^*X \wedge aY \to a(a^*X \wedge Y) \vee (X \wedge aY)$,
8. für alle $a, b \in \Sigma$ mit $a \neq b$: $a^*X \wedge bY \to X \wedge bY$,
9. für alle $a \in \Sigma$: $a^*X \wedge a^*Y \to a^*((a^*X \wedge Y) \vee (X \wedge a^*Y))$,
10. für alle $a, b \in \Sigma$ mit $a \neq b$: $a^*X \wedge b^*Y \to (a^*X \wedge Y) \vee (X \wedge b^*Y)$.

sowie den linksseitig (an $\wedge$) gespiegelten Varianten der Regeln 1–4,7,8.

Folgendes ist leicht zu verifizieren:

- die Normalformen von R_1 sind die Terme in SLWX,
- wenn $M \to_{R_1} N$, dann $\mathsf{Lang}(M) = \mathsf{Lang}(N)$,

Daß R_1 terminiert, zeigen wir durch die rekursive Pfadordnung zur Präzedenz $\wedge > \vee > \{a^*\cdot, a\cdot\}$.

4 Ableitungen von Wortsprachen

Definition 2 *Die Ableitung einer Sprache L nach einem Wort u ist*

$$\frac{\partial}{\partial u} L = \{w \mid uw \in L\}$$

Es ist bekannt, [Brz64] daß jede Ableitung einer regulären Sprache selbst wieder regulär ist, und daß jede reguläre Sprache L nur endlich viele verschiedene Ableitungen hat. Die endlich vielen Äquivalenzklassen der Relation

$$u \sim v \iff \frac{\partial}{\partial u} L = \frac{\partial}{\partial v} L$$

sind genau die Zustände des minimalen vollständigen deterministischen Automaten für L.

Wir zeigen, daß die Ableitung eines $L \in \mathsf{SLWL}$ sogar in SLWL liegt, und benutzen dazu wieder ein Ersetzungssystem R_2, das die Ableitung einer Sprache nach einem Buchstaben ausrechnet. Es besteht aus den Regeln

$$\frac{\partial}{\partial a}\emptyset \to \emptyset,\ \frac{\partial}{\partial a}\Sigma^* \to \Sigma^*,\ \frac{\partial}{\partial a}X \vee Y \to \frac{\partial}{\partial a}X \vee \frac{\partial}{\partial a}Y,$$
$$\frac{\partial}{\partial a}\epsilon \to \emptyset,\ \frac{\partial}{\partial a}aX \to X,\ \text{für alle } b \in \Sigma \setminus \{a\}\text{: } \frac{\partial}{\partial a}bX \to \emptyset,$$
$$\frac{\partial}{\partial a}a^*X \to a^*X \vee \frac{\partial}{\partial a}X,\ \text{für alle } b \in \Sigma \setminus \{a\}\text{: } \frac{\partial}{\partial a}b^*X \to \frac{\partial}{\partial a}X$$

Auf den rechten Regelseiten ist $\vee$ der einzige Konstruktor, der über einem Ableitungssymbol steht. und als Argumente von $\vee$ und $\frac{\partial}{\partial a}$ kommen nur Teilterme der linken Seiten vor.

Das heißt, daß jede Ableitung einer durch $X \in \mathsf{SLWX}$ gegebenen Sprache dargestellt wird als Vereinigung von Teiltermen von X. Da X nur endlich viele Teilterme hat, liefert das einen „Automaten-freien“ Beweis dafür, daß $\mathsf{Lang}(X)$ nur endlich viele verschiedene Ableitungen hat.

5 Komplemente von Wortsprachen

Das Komplement eines $X \in \mathsf{SLWX}$ berechnen wir, wie vorhin, durch zeitweiliges Erweitern der Signatur von SLWX, diesmal um die Symbole $\neg$ (für Komplement) sowie als Hilfssymbole $\wedge$ (für Durchschnitt) und $\frac{\partial}{\partial a}$ (für Ableitung). Wir erhalten die Menge SLWX_3.

Das Ersetzungssystem R_3, das die Komplemente ausrechnet, besteht aus den Regeln

1. $\neg\emptyset \to \Sigma^*$,
2. $\neg\Sigma^* \to \emptyset$,
3. $\neg(X \vee Y) \to \neg X \wedge \neg Y$,
4. $\neg\epsilon \to a_1\Sigma^* \vee \ldots \vee a_n\Sigma^*$, wobei $\Sigma = \{a_1, \ldots, a_n\}$,
5. $\neg aX \to \epsilon \vee b_1\Sigma^* \vee \ldots \vee b_n\Sigma^* \vee a\neg X$, wobei $\Sigma \setminus \{a\} = \{b_1, \ldots, b_n\}$,

Es fehlt noch die Behandlung von $\neg(a^*X)$. Für Ausdrücke ohne Konstruktoren $a^*\cdot$ können wir uns bereits von Vollständigkeit, Korrektheit und Termination (Präzedenz $\neg > \wedge > \ldots$) überzeugen.

Für den Fall $\neg(a^*X)$ berechnen wir:

$$\overline{a^*X} = \overline{X \cup aa^*X} = \overline{X} \cap \overline{aa^*X}$$

Wir setzen

$$N = \epsilon \cup b_1\Sigma^* \cup \ldots \cup b_n\Sigma^* \qquad \text{mit} \quad \Sigma \setminus \{a\} = \{b_1, \ldots, b_n\} \tag{1}$$

und fahren fort:

$$\begin{aligned} \overline{a^*X} &= \overline{X} \cap (N \cup a\, \overline{a^*X}) = \overline{X} \cap N \cup \overline{X} \cap a\, \overline{a^*X} \\ &= \overline{X} \cap N \cup a \left(\frac{\partial}{\partial a} \overline{X} \cap \overline{a^*X} \right) \end{aligned}$$

Hier nennen wir $M_1 = \dfrac{\partial}{\partial a}\overline{X}$ und bestimmen

$$M_1 \cap \overline{a^*X} = M_1 \cap \overline{X \cup aa^*X} = M_1 \cap \overline{X} \cap \overline{aa^*X}$$

Wie bei der vorigen Rechnung ergibt sich schließlich

$$M_1 \cap \overline{a^*X} = M_1 \cap \overline{X} \cap N \cup a \left(\frac{\partial}{\partial a}(M_1 \cap \overline{X}) \cap \overline{a^*X} \right)$$

Jetzt definieren wir allgemein

$$M_0 = \Sigma^*,\ M_{k+1} = \frac{\partial}{\partial a}(M_k \cap \overline{X}). \tag{2}$$

Das ist konsistent mit obigen Bezeichnungen, und liefert

$$M_k \cap \overline{a^*X} = M_k \cap \overline{X} \cap N \cup a \left(M_{k+1} \cap \overline{a^*X} \right) \tag{3}$$

Wir zeigen jetzt, daß die Folge $(M_0, M_1, \ldots)$ schließlich stationär ist.

Wir hatten $M_0 = \Sigma^*, M_1 = \dfrac{\partial}{\partial a}\overline{X}$ und bestimmen M_2:

$$M_2 = \frac{\partial}{\partial a}\left(M_1 \cap \overline{X}\right) = \frac{\partial}{\partial a}\left(\frac{\partial}{\partial a}\overline{X} \cap \overline{X}\right) = \frac{\partial}{\partial a}\frac{\partial}{\partial a}\overline{X} \cap \frac{\partial}{\partial a}\overline{X} = \frac{\partial}{\partial a^2}\overline{X} \cap \frac{\partial}{\partial a}\overline{X}$$

Das ergibt per Induktion $M_k = \bigcap\limits_{1 \leq i \leq k} \dfrac{\partial}{\partial a^i}\overline{X}$.

Daraus folgt auch $M_k \supseteq M_{k+1}$, also ist die Folge $(M_0, M_1, \ldots)$ monoton fallend. Jedes M_k ist ein Durchschnitt von Ableitungen einer SLWL-Sprache (nämlich $\overline{X}$). Da jede solche Sprache nur endlich viele verschiedene Ableitungen hat, gibt es nur endlich viele verschiedene M_k. Wegen der Monotonie gibt es dann einen Index k mit $M_k = M_{k+1} = \ldots$, und dieser läßt sich auf effektiv finden (durch Vergleich regulärer Sprachen).

Für dieses k gilt

$$M_k \cap \overline{a^*X} = M_k \cap \overline{X} \cap N \cup a\left(M_k \cap \overline{a^*X}\right)$$

und diese Gleichung besitzt die Lösung

$$M_k \cap \overline{a^*X} = a^*(M_k \cap \overline{X} \cap N) \tag{4}$$

Damit, und durch die Gleichungen 1, 2 und 3, haben wir die gewünschte Darstellung von $\neg(a^*X)$ in SLWX.

Beispiel 3 (fortgesetzt). Wir suchen das Komplement von $L = a^*X$ mit $X = ab\Sigma^*$.

Wir bestimmen zunächst $\overline{X} = \epsilon \cup a \cup aa\Sigma^* \cup b\Sigma^*$.

Nach obigem Schema ist $N = \epsilon \cup b\Sigma^*$ und damit auch $\overline{X} \cap N = \epsilon \cup b\Sigma^*$.

Weiter ist $M_1 = \dfrac{\partial}{\partial a}\overline{X} = \epsilon \cup a\Sigma^*$.

Wir erhalten die Gleichung $\overline{a^*X} = (\epsilon \cup b\Sigma^*) \cup a(M_1 \cap \overline{a^*X})$.

Es gilt $\Sigma^* = M_0 \supset M_1$, wir wiederholen das Verfahren.

Es entsteht $M_1 \cap \overline{X} = \epsilon \cup a \cup aa\Sigma^*$, und daraus $M_2 = \dfrac{\partial}{\partial a}(M_1 \cap \overline{X}) = \epsilon \cup a\Sigma^*$.

Also gilt $M_2 = M_1$, und das Verfahren bricht nach diesem Schritt ab.

Weiter ist $M_1 \cap \overline{X} \cap N = \epsilon$, das ergibt $M_1 \cap \overline{a^*X} = \epsilon \cup a(M_1 \cap \overline{a^*X})$ mit der Lösung $M_1 \cap \overline{a^*X} = a^*\epsilon$.

Weiter oben eingesetzt, folgt $\overline{a^*X} = (\epsilon \cup b\Sigma^*) \cup aa^*$.

Kommentar. Wir hätten das gewünschte Komplement auch aus dem minimalen deterministischen Automaten ablesen können. Diese Idee funktioniert aber für Baumsprachen nicht (siehe Beispiel 6). Hingegen werden wir das hier abgegebene Verfahren in den folgenden Abschnitten leicht von Wort- auf Baumsprachen verallgemeinern.

6 Reguläre Ausdrücke für Baumsprachen

Wir möchten nun von Wort- zu Baumsprachen übergehen. Wir suchen eine Sprachklasse, die durch ähnlich eingeschränkte reguläre Ausdrücke beschreibbar ist, und abgeschlossen ist unter booleschen Operationen.

Eine *Signatur* Σ ist eine endliche Menge von Funktionssymbolen mit jeweils bestimmter Argumentzahl. Wir schreiben $\Sigma = \Sigma_0 \cup \Sigma_1 \cup \ldots$, dabei enthält Σ_i die Symbole der Stelligkeit i. Ein Baum (Term) ist ein Element der von Σ erzeugten freien Algebra. Sie heißt $\mathsf{Tree}(\Sigma)$ oder auch Σ^*.

Definition 3 *Die Menge* SLTX *von Ausdrücken über einer Signatur* Σ *ist*

1. *(leere Menge:)* $\emptyset \in \mathsf{SLTX}$, *(alle Bäume):* $\Sigma^* \in \mathsf{SLTX}$,
2. *(Vereinigung:) für alle* $M \in \mathsf{SLTX}, N \in \mathsf{SLTX} : M \vee N \in \mathsf{SLTX}$,
3. *(nullstellige Symbole:) für alle* $f \in \Sigma_0 : f \in \mathsf{SLTX}$,
4. *(andere Symbole:) für alle* $k > 0, f \in \Sigma_k$
   und $X_1, \ldots, X_k \in \mathsf{SLTX}$*:* $f(X_1, \ldots, X_k) \in \mathsf{SLTX}$.
5. *(Stern:) für alle* $k > 0, f \in \Sigma_k, 1 \leq i \leq k$
   und $X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_k \in \mathsf{Tree}(\Sigma), Y \in \mathsf{SLTX}$*:*
   $f(X_1, \ldots, X_{i-1}, \bullet, X_{i+1}, \ldots, X_k)^* Y \in \mathsf{SLTX}$.

Jeder Ausdruck $X \in \mathsf{SLTX}$ beschreibt auf offensichtliche Weise eine reguläre Baumsprache. Insbesondere bezeichnet die letzte Zeile der Definition die Sprache, die durch die folgende Grammatik erzeugt wird:

$$H \to Y,\ H \to f(X_1, \ldots, X_{i-1}, H, X_{i+1}, \ldots, X_k)$$

Ausdrücke der Form $C = f(X_1, \ldots, X_{i-1}, \bullet, X_{i+1}, \ldots, X_k)$ nenne wir *Kontexte*. Wir haben eben C^*Y definiert, zur Vereinheitlichung der Notation legen wir noch $CY = f(X_1, \ldots, X_{i-1}, Y, X_{i+1}, \ldots, X_k)$ fest.

Beachte, daß die X_i nicht aus SLTX, sondern aus $\mathsf{Tree}(\Sigma)$ sind, also jeweils genau einen Baum bezeichnen, und nicht eine Menge von Bäumen. Diese Einschränkung ist notwendig für den Komplementabschluß. Ansonsten könnten wir nämlich $(a \cup b)^*(aa \cup bb)(a \cup b)^*$ simulieren (Beispiel 2).

Die Menge der durch SLTX beschreibbaren Sprachen heißt SLTL.

Die Klasse SLTL ist abgeschlossen unter Vereinigung. Abschluß gegen Durchschnitt beweisen wir wieder durch ein Ersetzungssystem:

1. $\emptyset \wedge X \to \emptyset$,
2. $\Sigma^* \wedge X \to X$,
3. $(X \vee Y) \wedge Z \to (X \wedge Z) \vee (Y \wedge Z)$,
4. für alle $k \geq 0, f \in \Sigma_k$:
   $f(X_1, \ldots, X_k) \wedge f(Y_1, \ldots, Y_k) \to f(X_1 \wedge Y_1, \ldots, X_k \wedge Y_k)$,
5. für alle $f, g \in \Sigma$ mit $f \neq g$: $f(X_1, \ldots) \wedge g(Y_1, \ldots) \to \emptyset$,
6. für alle $k \geq 1, f \in \Sigma_k, 1 \leq i \leq k$:
   wenn $H = f(X_1, \ldots, X_{i-1}, \bullet, X_{i+1}, \ldots, X_k)^* Z$,
   dann $H \wedge f(Y_1, \ldots, Y_k) \to$
   $(Z \wedge f(Y_1, \ldots, Y_k)) \vee$
   $f(X_1 \wedge Y_1, \ldots, X_{i-1} \wedge Y_{i-1}, (H \wedge Y_i), X_{i+1} \wedge Y_{i+1}, \ldots, X_k \wedge Y_k)$,
7. für alle $k \geq 1, f \in \Sigma_k, 1 \leq i \leq k, g \in \Sigma, f \neq g$:
   $f(X_1, \ldots, X_{i-1}, \bullet, X_{i+1}, \ldots, X_k)^* Z \wedge g(Y_1, \ldots, Y_j)$
   $\to (Z \wedge f(Y_1, \ldots, Y_j))$
8. für alle $k \geq 1, f \in \Sigma_k, 1 \leq i \leq k, l \geq 1, g \in \Sigma_l, g \neq f, 1 \leq j \leq l$:
   wenn $M = f(X_1, \ldots, X_{i-1}, \bullet, X_{i+1}, \ldots, X_k)^* P$
   und $N = g(Y_1, \ldots, Y_{j-1}, \bullet, Y_{j+1}, \ldots, Y_l)^* Q$,
   dann $M \wedge N \to (M \wedge Q) \vee (P \wedge N)$,
9. für alle $k \geq 1, f \in \Sigma_k, 1 \leq i \leq k, 1 \leq j \leq k, i \neq j$:
   wenn $M = f(X_1, \ldots, X_{i-1}, \bullet, X_{i+1}, \ldots, X_k)^* P$
   und $N = f(Y_1, \ldots, Y_{j-1}, \bullet, Y_{j+1}, \ldots, Y_k)^* Q$,
   dann $M \wedge N \to \begin{cases} \quad (P \wedge N) \vee (M \wedge Q) \\ \vee\, f(X_1 \wedge Y_1, \ldots, M \wedge Y_i, \ldots, X_j \wedge N, \ldots, X_k \wedge Y_k), \end{cases}$
10. für alle $k \geq 1, f \in \Sigma_k, 1 \leq i \leq k$:
    wenn $M = f(X_1, \ldots, X_{i-1}, \bullet, X_{i+1}, \ldots, X_k)^* P$
    und $N = f(Y_1, \ldots, Y_{i-1}, \bullet, Y_{i+1}, \ldots, Y_k)^* Q$:
    falls für alle $1 \leq j \leq k, j \neq i : X_i = Y_i$,
    dann $M \wedge N \to f(X_1, \ldots, X_{i-1}, \bullet, X_{i+1}, \ldots, X_k)^* \left((P \wedge N) \vee (M \wedge Q)\right)$,
    sonst $M \wedge N \to (P \wedge N) \vee (M \wedge Q)$.

für das sich Vollständigkeit, Korrektheit und Termination wiederum leicht verifizieren lassen.

Wir fassen dabei jeden Kontext als ein eigenes (einstelliges) Funktionssymbol auf. Das vereinfacht die Beweise und ist gestattet, weil sich die Menge der Kontexte durch die Regelanwendungen nicht vergrößert.

7 Ableitungen von Baumsprachen

Definition 4 *Die Ableitung von* $L \subseteq \mathsf{Tree}(\Sigma)$ *nach einem Kontext* C *ist*

$$\frac{\partial}{\partial C} L = \{t \mid t \in \mathsf{Tree}(\Sigma), Ct \in L\}$$

Wir haben jedoch keinen einfachen Zusammenhang mit dem minimalen deterministischen Automaten für L, da die Relation $x \sim y \iff \frac{\partial}{\partial x} L = \frac{\partial}{\partial y} L$ auf Kontexten (und nicht auf Bäumen selbst) erklärt ist. Bei Wörtern hatten wir diesen Unterschied nicht bemerkt, da ein gespiegelter Kontext wie ein Wort aussieht.

Wir berechnen Ableitungen wieder durch ein Ersetzungssystem. Wir fixieren dazu den Kontext $C = f(X_1, \ldots, X_{i-1}, \bullet, X_{i+1}, \ldots, X_k)$.

1. $\frac{\partial}{\partial C} \emptyset \to \emptyset$
2. $\frac{\partial}{\partial C} \Sigma^* \to \Sigma^*$
3. $\frac{\partial}{\partial C}(Y \vee Z) \to \frac{\partial}{\partial C} Y \vee \frac{\partial}{\partial C} Z$
4. wenn für alle $1 \le j \le k, j \ne i$: $X_j \in \mathsf{Lang}(Y_j)$, dann $\frac{\partial}{\partial C} f(Y_1, \ldots, Y_k) \to Y_i$, sonst $\frac{\partial}{\partial C} f(Y_1, \ldots, Y_k) \to \emptyset$,
5. für alle $g \in \Sigma \setminus \{f\}$: $\frac{\partial}{\partial C} g(Y_1, \ldots, Y_l) \to \emptyset$,
6. $\frac{\partial}{\partial C}(C^* Z) \to (C^* Z) \vee \frac{\partial}{\partial C} Z$
7. wenn nicht für alle $1 \le j \le k, j \ne i$: $X_j = Y_j$, dann $\frac{\partial}{\partial C} f(Y_1, \ldots, Y_{i-1}, \bullet, Y_{i+1}, \ldots, Y_k)^* Z \to \frac{\partial}{\partial C} Z$,
8. für alle $1 \le l \le k, l \ne i$: Sei $D = f(Y_1, \ldots, Y_{l-1}, \bullet, Y_{l+1}, \ldots, Y_k)$. Wenn für alle $1 \le j \le k, j \ne l, j \ne i$: $X_j = Y_j$ und $X_l \in \mathsf{Lang}(D^* Z)$, dann $\frac{\partial}{\partial C}(D^* Z) \to Y_i \vee \frac{\partial}{\partial C} Z$, sonst $\frac{\partial}{\partial C}(D^* Z) \to \frac{\partial}{\partial C} Z$,
9. für alle $g \in \Sigma \setminus \{f\}$: $\frac{\partial}{\partial C} g(Y_1, \ldots, Y_{j-1}, \bullet, Y_{j+1}, \ldots, Y_l)^* Z \to \frac{\partial}{\partial C} Z$

Das ist ein konditionelles Ersetzungssystem. Den Terminationsbeweis führen wir für das System, das entsteht, wenn wir sowohl Ja- als auch Nein-Zweige als unkonditionelle Regeln betrachten.

Wie bei SLWL sehen wir, daß Ableitungen als Vereinigungen von Teiltermen dargestellt werden, und folgern daraus, daß jedes $L \in \mathsf{SLTL}$ nur endlich viele verschiedene Ableitungen hat (bezüglich jedes fixierten Kontexts, genauer gesagt).

8 Komplemente von Baumsprachen

Wir können das Verfahren für Komplemente von Wortsprachen fast wörtlich übertragen. Wir benutzen die Regeln

1. $\neg\emptyset \to \Sigma^*$,
2. $\neg\Sigma^* \to \emptyset$,
3. $\neg(X \vee Y) \to \neg X \wedge \neg Y$,
4. $\neg f(X_1, \ldots, X_k) \to \begin{cases} \quad g_1(\Sigma^*, \ldots, \Sigma^*) \vee \ldots \vee g_n(\Sigma^*, \ldots, \Sigma^*) \\ \vee f(\neg X_1, \Sigma^*, \ldots, \Sigma^*) \vee \ldots \vee f(\Sigma^*, \ldots, \Sigma^*, \neg X_k) \end{cases}$
   wobei $\Sigma \setminus \{f\} = \{g_1, g_2, \ldots, g_n\}$

und für einen Kontext $A = f(A_1, \ldots, A_{i-1}, \bullet, A_{i+1}, \ldots, A_k)$ berechnen wir $\neg(A^* X)$ durch das Gleichungssystem

$$\text{für } l \geq 0\text{:} \quad M_l \cap \overline{A^* X} = (M_l \cap \overline{X} \cap N) \cup A(M_{l+1} \cap \overline{A^* X}) \tag{5}$$

$$\text{mit} \quad M_0 = \Sigma^*, \quad M_{l+1} = \frac{\partial}{\partial A}(M_l \cap \overline{X}) \tag{6}$$

$$\text{und} \quad N = \begin{cases} \quad g_1(\Sigma^*, \ldots) \cup \ldots \cup g_n(\Sigma^*, \ldots) \\ \cup f(\overline{A_1}, \Sigma^*, \ldots) \cup \ldots \cup f(\ldots, \Sigma^*, \overline{A_{i-1}}, \Sigma^*, \ldots) \\ \cup f(\ldots, \Sigma^*, \overline{A_{i+1}}, \Sigma^*, \ldots) \cup \ldots \cup f(\ldots, \Sigma^*, \overline{A_k}) \end{cases} \tag{7}$$

$$\text{wobei} \quad \Sigma \setminus \{f\} = \{g_1, \ldots, g_n\}$$

Wie früher ist die Folge $(M_0, M_1, \ldots)$ schließlich stationär:
es gibt ein l mit $M_l = M_{l+1} = \ldots$. Damit gilt

$$M_l \cap \overline{A^* X} = A^*(M_l \cap \overline{X} \cap N). \tag{8}$$

Beispiel 4 (fortgesetzt). Wir haben $A = \bullet \cdot b, X = (b \cdot b) \cdot *$.
Wir finden $N = b \cup * \cdot (* \cdot *)$ sowie $\overline{X} = b \cup b \cdot * \cup (* \cdot (* \cdot *)) \cdot * \cup ((* \cdot *) \cdot *) \cdot *$.
Es gilt $M_0 = \Sigma^*$, $M_1 = \dfrac{\partial}{\partial A}(M_0 \cap \overline{X}) = \dfrac{\partial}{\partial A}\overline{X} = b \cup (* \cdot *) \cdot * \cup * \cdot (* \cdot *)$.
Wir haben $M_0 \supset M_1$ und setzen fort: $M_2 = \dfrac{\partial}{\partial A}(M_1 \cap \overline{X}) = (* \cdot *) \cdot * \cup * \cdot (* \cdot *)$.
Wegen $M_1 \supset M_2$ bestimmen wir noch M_3 und stellen $M_2 = M_3$ fest.
Also müssen wir Gleichung 5 für $l = 0, 1$ und Gleichung 8 für $l = 2$ anschreiben.
Dazu brauchen wir noch

$$\begin{aligned} M_0 \cap \overline{X} \cap N &= M_1 \cap \overline{X} \cap N \\ &= b \cup b \cdot (* \cdot *) \cup ((* \cdot *) \cdot *) \cdot (* \cdot *) \cup (* \cdot (* \cdot *)) \cdot (* \cdot *) \\ M_2 \cap \overline{X} \cap N &= \phantom{b \cup{}} b \cdot (* \cdot *) \cup ((* \cdot *) \cdot *) \cdot (* \cdot *) \cup (* \cdot (* \cdot *)) \cdot (* \cdot *) \end{aligned}$$

9 Ausblick

Verwandte Arbeiten. Wenn $L \in \mathsf{SLWL}$, dann ist L sternfrei. Sternfreie Sprachen können wir nach der Straubing-Hierarchie [Pin97] einteilen, die allerdings auf unseren Fall nicht so schön paßt, weil die Straubing-Klassen immer unter Konkatenation abgeschlossen sind, die kurzen-Schleifen-Automaten jedoch nie.

Für Baumsprachen ist gar nicht klar, wie eine Straubing-Hierarchie auszusehen hätte. Schon der Begriff *sternfrei* ist mit Vorsicht zu genießen: sternfreie Baumsprachen enthalten echt diejenigen mit aperiodischem syntaktischen Monoid, und diese echt die first-order-definierbaren. (Für Wortsprachen fallen diese Klassen zusammen.) [Tho97] In Signaturen ohne unäre Symbole ist sogar jede reguläre Sprache sternfrei. [Pot94]

Die Klasse SLWL enthält die definiten Sprachen. Erweiterungen davon sind revers definite Sprachen, verallgemeinert definite Sprachen, und lokale Sprachen. Alle diese Erweiterungen benötigen Konstruktionen der Form $\Sigma^* w$, und das läßt sich mit kurzen Schleifen in unserem Sinn nicht machen (die benötigte Schleife ist kurz, muß aber mit mehr als einem Buchstaben beschriftet sein).

Ebenso gibt es revers definite, verallgemeinert definite [Heu88] und lokale Baumsprachen, die aber auch kompliziertere Automaten haben und deswegen nicht mit unserer Klasse vergleichbar sind.

Man kann Schleifen in Wort-Automaten völlig verbieten, aber Alternierung erlauben. (Das heißt in diesem Fall: Akzeptanz in einem Zustand ergibt sich aus beliebiger Boolescher Verknüpfung der Resultate in den Folgezuständen.) Nach [SY98] erhält man dadurch bereits alle sternfreien Wortsprachen. Eine Übertragung dieser Untersuchung auf Baumsprachen scheint schwer zu sein, denn alternierende Baum-Automaten haben einige nicht offensichtliche Eigenschaften [Sal89] .

Offene Fragen. Wir sind nicht darauf eingegangen, ob und wie entschieden werden kann, ob eine (durch einen Automaten) gegebene reguläre Sprache L zur Sprachklasse SLWL bzw. SLTL gehört.

Für Wortsprachen ist das möglich, weil der minimale deterministische Automat von L höchstens kurze Schleifen enthält. Für Baumsprachen läßt sich dieser Zusammenhang nicht herstellen, weil sich top-down-Automaten überhaupt nicht deterministisch machen lassen. Wir suchen derzeit nach einer Konstruktion über unzweideutige reguläre Ausdrücke, die ähnliches leistet.

Anwendungen. Wir hoffen, daß SLTL-Sprachen mit gewissen Term-Ersetzungs-Systemen gut zusammenarbeiten. Diese Hoffnung stützt sich auf nachstehende Betrachtung.

Term-Ersetzung ist ein *lokaler* Prozeß: jede einzelne Regelanwendung liest und ändert nur einen kleinen Teil des Baumes. Eine reguläre Baumsprache erscheint demgegenüber *global*: erst an der Wurzel, also dann, wenn der Baum vollständig besucht wurde, sagt der Automat, was er von der Eingabe hält. Eine Automat mit kleinen Schleifen ist nun wieder ein Schritt in Richtung Lokalität, denn es herrscht Ordnung in der Zustandsmenge, und nicht jeder kann auf jeden folgen. Von daher vermuten wir, daß sich Zusammenhänge zwischen SLTL-Sprachen und Term-Ersetzungs-Systemen herstellen lassen.

Bis jetzt ist das in einem nichttrivialen Fall gelungen, der natürlich der Ausgangspunkt auch für vorliegende Arbeit ist: Die Menge der kopfnormalisierenden $\mathbf{CL}(S)$-Terme ist in SLTL. [Wal99] Der entsprechende reguläre Ausdruck hat etwa die Größe 1000.

Literatur

[Brz64] J. A. Brzozowski. Derivatives of regular expressions. *Journal of the ACM*, 11:481–494, 1964.

[Csi89] János Csirik, editor. *Fundamentals of Computtation Theory*, number 380 in LNCS. Springer, 1989.

[Heu88] U. Heuter. Generalized definite tree languages. Aachener Informatik-Berichte 88-24, RWTH Aachen, 1988.

[Pin97] Jean-Eric Pin. *Syntactic Semigroups*, pages 679–748. Volume 1 of Rozenberg and Salomaa [RS97], 1997.

[Pot94] Andreas Potthoff. *Logische Klassifizierung regulärer Baumsprachen*. PhD thesis, Christian-Albrechts-Universität Kiel, 1994. Bericht Nr. 9410.

[RS97] G. Rozenberg and A. Salomaa, editors. *Handbook of Formal Languages*. Springer, 1997.

[Sal89] Kai Salomaa. Representation of recursively enumerable langauges using alternating finite tree recognizers. In Csirik [Csi89], pages 372–383.

[SY98] Kai Salomaa and Sheng Yu. Alternating finite automata and star-free languages. *Theoretical Computer Science*, page (submitted), 1998.

[Tho97] Wolfgang Thomas. *Languages, Automata and Logic*, pages 389–456. Volume 3 of Rozenberg and Salomaa [RS97], 1997.

[Wal99] Johannes Waldmann. (Head normalization and) Top termination in CL(S). talk given at 4th International Workshop on Termination, Schloß Dagstuhl, May 1999.