Fixpunkte von Morphismen und Normalformen von Ersetzungssystemem

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 können Bäume, die sich als Normalformen von Ersetzungssystemen ergeben, in einigen Fällen auch durch iterierte Morphismen beschreiben, und unendliche Normalformen von top–terminierenden Systemen durch Fixpunkte von Morphismen.

1 Morphismen und HD0L

Das bekannte Fibonacci–Wort $f = 01001010010010\ldots$ (siehe [Lot97b]) ist der Fixpunkt des Morphismus ϕ:

$$f = \lim f_n,\ f_n = \phi^n(0),\ \phi : 0 \mapsto 01, 1 \mapsto 0$$

Das folgende Haskell–Programm berechnet die Liste der Buchstaben von f:

```
phi :: [ Char ] -> [ Char ]
phi ('0' : rest) = '0' : '1' : phi rest
phi ('1' : rest) = '0'       : phi rest

f :: [ Char ]
f = '0' : t where t = '1' : phi t
```

Das können wir ganz leicht als eine Wort–Ersetzungs–System R auffassen mit der Regelmenge

$$R : T \to 1\Phi T, \Phi 0 \to 01\Phi, \Phi 1 \to 0\Phi$$

Dieses System terminiert nicht, aber bei fairer Reduktion (kein Redex lebt unendlich lange) wandern die Redexe nach rechts, und wir erhalten im Limes das Fibonacci–Wort $f = R^\infty(0T)$.

Die Konstruktion war generisch; offensichtlich können wir so jeden Fixpunkt eines monotonen Morphismus als Links–Normalform eines Ersetzungssystems darstellen.

Umgekehrt finden wir in einigen Fällen zu einem gegebenem Wort einen passenden Morphismus. Betrachten wir

$$q = 1\ 1\ 0\ 1\ 00\ 1\ 000\ 1 \dots$$

aus immer längeren Blöcken von 0.
Dieses können wir durch einen iterierten Morphismus f erzeugen,

$$f : S \mapsto S1, 1 \mapsto 10, 0 \mapsto 0$$

von dem wir anschließend ein homomorphes Bild nehmen (um das „Startsymbol“ S zu löschen)

$$g : S \mapsto \epsilon, 0 \mapsto 0, 1 \mapsto 1; \quad q = \lim g(f^n(S))$$

Das Wachstum dieses Morphismus ist polynomiell (quadratisch) beschränkt. Das Tupel (g, f, S) ist ein HD0L–System. (Für Definitionen fehlt hier der Platz, siehe jedoch [KRS97] .)

Ein weiteres Beispiel (mit exponentiellem Wachstum) benötigen wir gleich:

$$N_k = h^k(R) \textbf{ wobei } h : R \mapsto RL, L \mapsto RR$$

Beispiel: $N_2 = RLRR, N_3 = RLRRRLRL$. Offensichtlich ist $|N_k| = 2^k$.
Wir bezeichnen mit w' das Wort w ohne seinen letzten Buchstaben. Dann zeigt man leicht durch Induktion

$$N'_{k+1} = \textbf{wenn } 2 \mid k \textbf{ dann } N'_k R N'_k \textbf{ sonst } N'_k L N'_k$$

Im Beispiel ist das $N'_3 = RLRRRLR = RLR\ R\ RLR = N'_2 R N'_2$.

2 Normalformen für den Kombinator J

Wir betrachten Terme in CL(J), Kombinatorischer Logik mit Kombinator J. Dort haben wir genau zwei Funktionssymbole, ein zweistelliges, das wir als Operator @ schreiben, sowie ein nullstelliges, nämlich J.
Dann ist CL(J) ein Term–Ersetzungs–System mit der einzigen Regel

$$(((\mathrm{J}@a)@b)@c)@d \to (a@b)@((a@d)@c).$$

Nach Vereinbarung lassen wir @ und einige Klammern weg und schreiben kürzer

$$\mathrm{J}abcd \to ab(adc).$$

Bekannterweise ist $\{\mathrm{I}, \mathrm{J}\}$ eine Basis. Umso größer war die Überraschung, als Probst und Studer kürzlich zeigten [PS00], daß CL(J) terminiert. Daraufhin habe ich einige J–Terme untersucht und folgende interessante Familie gefunden:

$$X_0 = \mathrm{J}, X_{k+1} = L[X_k] \textbf{ wobei } L[x] = \mathrm{J}x\mathrm{J}, R[x] = \mathrm{JJ}x$$

Beispielsweise ist $X_3 = \mathrm{J}X_2\mathrm{J} = \ldots = \mathrm{J(J(JJJ)J)J}$. Dann gilt der

Satz 1 *Die Normalform von $X_k\ c\ d$ ist*

$$\textbf{wenn } 2 \mid k \textbf{ dann } N'_k[\mathrm{J}\ c\ d] \textbf{ sonst } N'_k[\mathrm{J}\ d\ c]$$

Hierbei sind $N'_k \in \{L, R\}^*$ die Wörter aus dem vorigen Abschnitt, diesmal aufgefaßt als ineinandergesetzte Kontexte.
Beispiel: $N'_2 = RLR, N'_2[\mathrm{J}\ c\ d] = R[L[R[\mathrm{J}\ c\ d]]]$.
Der Beweis des Satzes erfolgt leicht per Induktion; im Fall $2 \mid k$ so:

$$\begin{aligned} &X_{k+1}\ c\ d = \mathrm{J}X_k\mathrm{J}\ c\ d \to X_k\mathrm{J}(X_k\ d\ c) \to^* X_k\mathrm{J}\ N'_k[\mathrm{J}\ d\ c] \\ &\to^* N'_k[\mathrm{JJ}\ N'_k[\mathrm{J}\ d\ c]] = N'_k[R[N'_k[\mathrm{J}\ d\ c]]] = N'_{k+1}[\mathrm{J}\ d\ c] \end{aligned}$$

Wir haben hier die Normalformen bezüglich eines Ersetzungsssystems durch iterierte Morphismen beschrieben. Bei beiden Verfahren (Terme ersetzen und Morphismen iterieren) werden Änderungen wiederholt ausgeführt. Der enge Zusammenhang ist jedoch nicht selbstverständlich: Termersetzen ist lokal (jeweils nur an einer Stelle), Morphismus anwenden ist global (jeweils überall). Wir sehen im nächsten Abschnitt ein weiteres Term–Ersetzungs–System, bei dem das (anscheinend) funktioniert.
Ich kann bis jetzt nur darüber spekulieren, woran das "wirklich" liegt. Es wird sicher damit zusammenhängen, daß man CL–Terme als Funktionen auffassen sollte, das heißt, eine Betrachtung von J als $\lambda ab.\lambda cd.ab(adc)$ könnte weiterhelfen. In diesem Moment beginnen aber die bekannten Schwierigkeiten der Termersetzung höherer Ordnung.

3 Unendliche Normalformen für CL(S)

Wir betrachten jetzt den Kombinator S mit der Reduktionsregel

$$\mathrm{S}\ x\ y\ z \to x\ z\ (y\ z)$$

Dieses System ist nicht terminierend, aber top–terminierend [Wal99] (und sowieso konfluent). Das bedeutet, daß für Terme X ohne Normalform alle

unendlichen (und fairen) Reduktionsketten, die bei X starten, dem gleichen Limes zustreben. Das ist ein unendlicher Baum, den wir die (unendliche) Normalform von X nennen.
Beispiel: Für die Familie $X_0 = \mathrm{S(SS)(SS)}, X_{k+1} = \mathrm{SS}X_k$ gilt

Satz 2 *Kein Term $X_i X_k$ besitzt eine Normalform.*

Beweis: es gilt $X_0 X_k \to X_{k+1} X_{k+1}$ und $X_{i+1} X_k \to SX_k(X_i X_k)$.
Hier sehen wir auch die unendliche Normalform von $X_0 X_0$:

$$X_0X_0 \to X_1X_1 \to \mathrm{S}X_1(X_0X_1) \to \mathrm{S}X_1(X_2X_2)$$
$$\to \mathrm{S}X_1(\mathrm{S}X_2(X_1X_2)) \to \mathrm{S}X_1(\mathrm{S}X_2(\mathrm{S}X_2(X_0X_2))) \to \ldots$$

Das ist einfach, weil jeder Term genau einen Redex enthält.
Wir erkennen auf dem rechten Rückgrat die Struktur des quadratischen Wortes aus einem vorigen Beispiel: Es stehen dort ein $\mathrm{S}X_1$, dann zwei $\mathrm{S}X_2$, dann drei $\mathrm{S}X_3$ usw. Es ist deswegen klar, daß dieser unendliche Baum als (homomorphes Bild eines) Fixpunktes eines Morphismus auf Bäumen entsteht.
Ich vermute, daß solch eine Darstellung für unendliche Normalformen beliebiger S–Terme möglich ist. Bisher kann ich das aber nur experimentell bestätigen. Für den Term $X_0 X_0$ scheint das alles sehr einfach zu sein, aber es gibt andere Startterme mit höchst komplizierten Entwicklungen.

4 Vergleichbarkeit von Fixpunkten

Angenommen, es läßt sich beweisen, daß unendliche Normalformen in CL(S) immer durch iterierte Morphismen beschrieben werden können. Dann hätten wir einen Ansatz, um das Wortproblem in CL(S) [Wal00] zu bearbeiten: wegen Konfluenz sind zwei Terme konvertierbar, wenn sie die gleiche (unendliche) Normalform haben. Man bestimmt zu jeder den Morphismus, und muß dann „nur noch" entscheiden, ob beide Morphismen den gleichen unendlichen Baum erzeugen.
Das ist im Wesentlichen die Frage nach der Äquivalenz von HD0L–Sprachen. Für Wörter ist das lösbar [KRS97], benutzt aber Ehrenfeuchts Vermutung und Makanins Algorithmus. Wie ist das für HD0L–Baumsprachen? Dort besteht wohl im Allgemeinen wenig Aussicht auf Erfolg, zum Beispiel ist die Lösbarkeit von Baumgleichungen bis heute offen [MR98] (Makanins Algorithmus [Die97] löst Wortgleichungen, und ist schon höchst kompliziert).

Hoffentlich zeigt sich zwischendurch, daß bei CL(S) nicht beliebige HD0L–Sprachen entstehen, sondern vielleicht nur polynomiell beschränkte. Für diese könnte das Entscheidungsverfahren leichter zu finden sein.

Literatur

[Die97] Volker Diekert. *Makanin's Algorithm*, pages 344–391. In [Lot97a], 1997.

[KRS97] Lila Kari, Grzegorz Rozenberg, and Arto Salomaa. *L Systems*, pages 253–328. Volume 1 of Rozenberg and Salomaa [RS97], 1997.

[Lot97a] M. Lothaire. *Algebraic Combinatorics on Words.* `http://www-igm.univ-mlv.fr/~berstel/Lothaire/`, 1997.

[Lot97b] M. Lothaire. *Combinatorics on Words.* Cambridge Mathematical Library. Cambridge University Press, 1997.

[MR98] Sabrina Mantaci and Antonio Restivo. Codes and equations for trees. Technical Report 194, Turku Centre for Computer Science, August 1998. `http://altair.math.unipa.it/CSG/sabrina/publications.html`.

[PS00] Dieter Probst and Thomas Studer. How To Normalize the Jay. *Theoretical Computer Science*, (to appear), 2000. `http://www.iam.unibe.ch/~tstuder/papers/j.pdf`.

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

[Wal99] Johannes Waldmann. (Head normalization and) Top termination in CL(S). Talk given at 4th International Workshop on Termination, Schloß Dagstuhl, May 1999, `http://www.informatik.uni-leipzig.de/~joe/talk/wst99-slides.ps`.

[Wal00] Johannes Waldmann. Problem 97: Is the Word Problem for the S Combinator Decidable? RTA list of open problems, `http://www.lri.fr/~rtaloop/97.html`, 2000.