DETERMINISTISCHE INDIZIERTE GRAMMATIKEN

Karl Weiß
Institut für Informatik I
Universität Karlsruhe
75 Karlsruhe

Eine wesentliche gemeinsame Eigenschaft von indizierten Grammatiken und kontextfreien Grammatiken ist die Invarianz ihrer Erzeugungskraft unter verschiedenartigen Ableitungsrelationen wie Links- oder Rechtsableitung oder Ersetzung an einer beliebigen Stelle. Es liegt deshalb nahe, mit Hilfe spezieller Ableitungen gewonnene Begriffe aus der Theorie kontextfreier Grammatiken auf indizierte Grammatiken zu übertragen. Die in Definition 1 eingeführten ILL(k)-Grammatiken sind eng mit den kontextfreien LL(k)-Grammatiken verwandt:

Definition 1:
Sei $G = (N,T,F,R,S)$ eine indizierte Grammatik, $k \in \mathbb{N}$, $r: A \to \alpha \in f$ und $r' : B \to \beta \in R$. Sei weiter

$$p_k(r) := \{u;\ u \in T^*,\ l(u) \leq k : \exists\, \delta,\gamma,\gamma' \in (N \cup F \cup T)^* :$$
$$\text{(a) } l(u) < k \curvearrowright \gamma' = \varepsilon$$
$$\text{(b) } Af\delta \Rightarrow_r \gamma \overset{*}{\Rightarrow} u\,\gamma'\}$$
$$p_k(r') := \{u;\ u \in T^*,\ l(u) \leq k : \exists\, \delta,\gamma,\gamma' \in (N \cup F \cup T)^* :$$
$$\text{(a) } l(u) < k \curvearrowright \gamma' = \varepsilon$$
$$\text{(b) } A\delta \Rightarrow_r \gamma \Rightarrow u\,\gamma'\}$$

Dabei soll $\Rightarrow$ die Relation "Linksableitung" bezeichnen. G heißt ILL(k)-Grammatik, wenn für alle $f \in F$ aus $r, s \in f \cup R$ mit lhs (r) = lhs (s) folgt:

$$p_k(r) \cap p_k(s) = \emptyset .$$

Für eine ILL(k)-Grammatik ist zu jeder Satzform u und jedem bis zu k Zeichen langen "look-ahead" $w \in T^*$ diejenige Regel eindeutig bestimmt, die man als nächste auf u anwenden muß, um schließlich eine mit w beginnende Satzform zu erhalten. Diese Tatsache impliziert ein Top-Down-Syntaxanalyseverfahren für jede von einer ILL(k)-Grammatik erzeugte Sprache: Man beginnt mit der Startvariablen S und wendet jeweils die durch die nächsten k Zeichen des zu analysierenden Wortes und die vorliegende

Satzform bestimmte Regel an. Auf diese Weise erhält man für ε-freie ILL(k)-Grammatiken Verfahren, die mit einem zur Länge des analysierenden Wortes quadratischen Zeitaufwand arbeiten.

Für RIR-Grammatiken (indizierte Grammatiken mit Regeln $A \to a$, $A \to aB\alpha$, $a \in T \cup \{\varepsilon\}$, $\alpha \in F^*$ und ebensolchen Indexregeln) werden in Definition 1 δ durch $\bar{\delta} \in F^*$ und γ, γ' durch $\bar{\gamma}, \bar{\gamma}' \in NF^* \cup \{\varepsilon\}$ ersetzt; die so erhaltene Eigenschaft wird als RIRLL(k)-Eigenschaft bezeichnet. RIRLL(k)-Grammatiken erzeugen für beliebiges $k \in \mathbb{N}$ dieselbe Sprachklasse wie RIRLL(1)-Grammatiken; die entsprechenden Syntaxanalyseverfahren arbeiten mit linearem Zeitaufwand.

Eigenschaften von ILL(k)-Grammatiken

Satz 1
Zu jeder ILL(k)-Grammatik G gibt es einen deterministischen nested-stack-automaton A mit

$$L(G) = T(A).$$

Der Beweis verläuft im wesentlichen wie der Beweis des nichtdeterministischen Falles in /2/; A wird dadurch deterministisch gehalten, daß man jeweils die nächsten k Eingabezeichen im Zustand kodiert.

Satz 2
Jede von einer ILL(k)-Grammatik G erzeugte Sprache L (G) ist präfixfrei.

Der Beweis beruht auf der leicht einzusehenden Tatsache, daß für eine ILL(k)-Grammatik aus $w \in p_k(r)$, $l(w) = n < k$ stets $\{wv;\ v \in T^*,\ l(v) \leq k - n\} \subset p_k(r)$ folgt. Es sei hier schon angemerkt, daß diese Aussage für RIRLL(1)-Grammatiken nicht gilt und es nicht-präfixfreie Sprachen gibt, die von RIRLL(1)-Grammatiken erzeugt werden.

Satz 3 befasst sich mit ε-freien indizierten Grammatiken. Um bei ihnen die am weitesten links stehende Variable zum Verschwinden zu bringen, muß mindestens ein Terminalzeichen erzeugt werden. Damit gibt es zu jeder solchen Grammatik G eine RIR-Grammatik G' mit isomorpher Regelmenge und $p_1(r) = p_1(r')$, wenn r' die r entsprechende Regel von G' ist. Unter Vorwegnahme des Resultats von Satz 4 gilt dann

Satz 3
Ist G eine ε-freie indizierte Grammatik, so ist es entscheidbar, ob G die ILL(1)-Eigenschaft besitzt oder nicht.

Eigenschaften von RIRLL(1)-Grammatiken

Satz 4
Ist G eine RIR-Grammatik, so ist es entscheidbar, ob G die RIRLL(1)-Eigenschaft besitzt oder nicht.

Die Schwierigkeit liegt hier in der Bestimmung der Mengen $p_1(r)$. Diese wird dadurch vorgenommen, daß man zu jeder Regel $r : A \to a\alpha \in f$ (oder $\in R$) mit $a \in T$ die Menge M aller Satzformen bestimmt, von denen aus man ohne Erzeugung eines Terminalzeichens zu einer mit Af (oder A) beginnenden Regel gelangen kann. Die Menge aller $\beta \in NF^*$ einer festen Länge, für die es ein $\gamma \in F^*$ gibt, so daß $\beta\gamma$ in M liegt, ist effektiv konstruierbar und für alle Regeln $r : B \to \beta \in h$ (oder R), für die es eine Satzform $\beta\gamma \in M$ gibt, gilt $a \in p_1(r)$.

In /1/ wird gezeigt, daß RIR-Grammatiken und Kellerautomaten dieselbe Sprachklasse definieren (Kellerautomaten sollen immer unter Leeren des Kellers akzeptieren, im Gegensatz zu E-Kellerautomaten, die unter Übergang in einen Endzustand akzeptieren). Der Beweis beruht auf der strukturellen Äquivalenz der Satzformen bzw. Konfigurationen; einer δ-Regel

$$(q', \beta) \in \delta(q,a,f)$$

eines Kellerautomaten entspricht eine grammatikalische Regel

$$q \to a\, q'\beta \in f.$$

Diese Äquivalenz wird ausgenützt zum Beweis von

Satz 5
RIRLL(1)-Grammatiken erzeugen genau die deterministischen kontextfreien Sprachen (d. h. die Menge der von deterministischen E-Kellerautomaten akzeptierten Sprachen).

Durch Kodieren des nächsten zu erzeugenden (akzeptierenden) Zeichens in der Variablen (im Zustand) bleibt der Determinismus der Konzepte in beiden Beweisrichtungen erhalten.

Satz 6
RIR-Grammatiken mit der ILL(1)-Eigenschaft erzeugen genau die präfixfreien deterministischen Sprachen (d. h. die Menge der von deterministischen Kellerautomaten akzeptierten Sprachen).

Schließlich sei noch erwähnt, daß man die deterministischen "one counter languages" ebenfalls durch spezielle RIR-Grammatiken erzeugen kann. Damit stellen RIR-Grammatiken ein wichtiges Hilfsmittel zur Simulation von Automaten im kontextfreien Bereich dar.

Analysegrammatiken

Man kann die Erzeugungskraft deterministischer indizierter Grammatiken dadurch vergrößern, daß man sie in Analysegrammatiken einbaut. Eine Analysegrammatik G ist eine binäre Grammatik, deren erste Komponente G_1 - isoliert betrachtet - eine ILL(k)- oder RIRLL(1)-Grammatik bildet und deren zweite Komponente die Ableitungen bezüglich G_1 kontrolliert. Da sich diese Kontrolle darauf beschränkt, bestimmte bezüglich G_1 erlaubte Ableitungen auszuschließen, benötigt eine Analysegrammatik zur Analyse eines Wortes w höchstens denselben Zeitaufwand, den ihre erste Komponente benötigen würde, um w zu analysieren.

Definition 2:

Eine (RIRLL(1),RIR)-Analysegrammatik (RAG) ist eine binäre Grammatik (siehe /3/), deren Regeln die Gestalt

$$r:(A \to \alpha,\ B \to \beta)\ ,\ \alpha \in TNF^* \cup T \cup \{\varepsilon\},\ \beta \in NF^* \cup \{\varepsilon\}$$

besitzen und deren erste Komponente (die durch die ersten Komponenten der Regeln bestimmte Grammatik) die RIRLL(1)-Eigenschaft besitzt. Eine Ableitungsrelation $\Rightarrow$ auf den Satzformen $(\gamma,\delta) \in T^* NF^* \times NF^*$ erhält man durch komponentenweise Übertragung der Ableitungsrelation für RIR-Grammatiken. Die von einer RAG G erzeugte Sprache ist dann

$$L(G) := \{w;\ (S,S) \overset{*}{\Rightarrow} (w,\varepsilon)\}.$$

Die Sprachklasse

$$L(RAG) = \{L;\ \exists \text{ RAG } G:\ L = L(G)\}$$

besitzt folgende Eigenschaften:

Satz 7

(a) Die Menge aller deterministischen kontextfreien Sprachen ist eine echte Teilmenge von L(RAG).

(b) L(RAG) enthält nicht-kontextfreie Sprachen wie etwa
$L = \{a^n c a^n b^{n-1} c a^{n-1} b^{n-2} c \ldots c a^2 b c a d;\ n \geq 2\} \cup \{ad\}$.

(c) Ist M eine deterministische Turing-Maschine, deren mit einem Lese-Schreib-Kopf versehenes Speicherband die Eingabe enthält und ist $\bar{w}$ die Folge der δ-Regeln, die M mit der Eingabe w durchläuft, so liegt die Sprache $\{w\bar{w};w \in T(M)\}$ in L(RAG).

(d) Zu jedem $L \in L(RAG)$ gibt es ein Verfahren, das für jede Eingabe w anhält und dabei in $O(l(w))$-Schritten entscheidet, ob w in L(G) liegt oder nicht. Dieses Verfahren läßt sich auf einem deterministischen Automaten mit zwei Kellern als Speicherbändern realisieren.

(e) Jede Sprache $L \in L(RAG)$ ist kontextsensitiv.

Für Analyse-Grammatiken, deren erste Komponenten jeweils eine ε-freie ILL(1)-Grammatik bilden (IAGen), gilt (b) ebenfalls und die Eigenschaften (d) und (e) lassen sich ersetzen durch:

(d') ersetze in (d) RAG durch IAG und $O(l(w))$ durch $O(l(w)^2)$.

(e') Jede Sprache $L \in L(IAG)$ ist entscheidbar.

Die Punkte (a) und (c) lassen sich wegen der Präfixfreiheit der von IAGen erzeugten Sprachen nicht übertragen.

Die Beweise der einzelnen Punkte von Satz 7 lassen sich folgendermaßen skizzieren:

(a) Ist L eine deterministische kontextfreie Sprache, so gibt es nach Satz 5 eine L erzeugende RIRLL(1)-Grammatik G. Die Regelmenge

$$R = \{(A \to \alpha,\ A \to \bar{\alpha});\ A \to \alpha \text{ ist eine Regel bez. } G \text{ und } \bar{\alpha} \text{ wurde aus } \alpha \text{ durch Streichen aller Terminalzeichen gewonnen.}\}$$

charakterisiert eine RAG, die L erzeugt.

(b) Gibt man zu jeder Komponente einer Regel an, ob sie in R oder einem Index f enthalten ist, so wird jede RAG schon durch ihre Regelmenge zusammen mit dem Startvariablenpaar eindeutig bestimmt (Indizes, Variable oder Terminalzeichen, die in keiner Regel vorkommen, sollen nicht auftreten). Die RAG mit dem Startvariablenpaar (S_o, S_o) und den Regeln

$(S_o \to SZ \in R,\ S_o \to SZ \in R)$ $\quad$ $(A \to BZ \in Z,\ S \to S \in f)$

$(S \to aSf \in R,\ S \to S \in R)$ $\quad$ $(B \to bBf \in R, S \to S \in f)$

$(S \to cA \in R,\ S \to S \in R)$ $\quad$ $(B \to cA \in R,\ S \to SZ \in Z)$

$(A \to aA \in f,\ S \to Sf \in R)$ $\quad$ $(B \to d \in Z,\ S \to \varepsilon \in Z)$

erzeugt gerade die Sprache L.

(c) M ist deterministisch, also gibt es zu jeder Konfiguration $K = xgay$ (siehe /4/) genau eine auf K anwendbare δ-Regel. Durch Simulation der δ-Regeln von M erhält man die Regelmenge einer RAG, die die Behauptung erfüllt. Die K entsprechende Satzform besitzt die Gestalt

$$(w\alpha gx^r,\ SayZ).$$

Dabei ist α eine Vorsilbe von $\bar{w}$.

(d), (d') Das durch eine RAG bzw. IAG induzierte Syntaxanalyseverfahren bricht dann ab, wenn ein look-ahead-string auftritt, der von

der vorliegenden Satzform aus nicht ableitbar ist, oder wenn die zweite Komponente die Ableitung beendet. Der Zeitbedarf berechnet sich nach ähnlichen Methoden wie der Zeitbedarf deterministischer Kellerautomaten.

(e), (e') folgt aus (d), (d').

Literatur:

/1/ AHO, A. "Indexed Grammars - An Extension of Context-Free Grammars", JACM 15,4 (1968), 647-671

/2/ AHO, A. "Nested Stack Automata", JACM 16,3 (1969), 383-406

/3/ CULIK, K. II, MOREY, C. "Formal Schemes for Language Translation", Intern. J. Comp. Math. Sciences A, Vol. 3 (1971), 17-48

/4/ MAURER, H. "Theoretische Grundlagen der Programmiersprachen", BI-Hochschultaschenbücher, Band 404/404 a*, (1969)