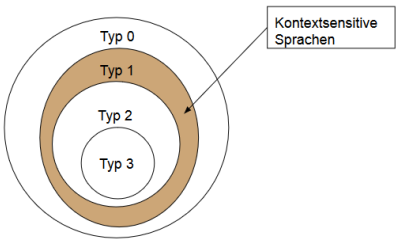
Proseminar Theoretische Informatik 17.11.2015

Kontextsensitive Sprachen

Christopher Mühl und Hanna Lachnitt *Wolfgang Mulzer*

1 Kontextsensitive Sprachen

Übersicht:
Die Kontextsensitven Sprachen werden durch Typ-1-Grammatiken gebildet. Sie entsprechen also genau den Typ-1-Sprachen. Die monotonen Grammatiken sind den kontextsensitiven äquivalent.



Definition:
Sei G eine Grammatik G = (N,T,P,S), wobei N die Nichtterminalsymbole, T die Terminalsymbole, S das Startsymbol aus N und P die Menge der Produktionsregeln bezeichnet.
G ist eine Typ-1-Grammatik (Kontextsensitive Grammatik) wenn alle Produktionsregeln die Form :

1. $S \to \epsilon$

oder

2. $\alpha \to \beta$ wobei $|\alpha| \leq |\beta|$ mit α , $\beta \subseteq (T \cup N)^*$, α enthält mindestens ein Nichtterminalsymbol.

haben.

Beispiel:
$L(G) = \{a^n b^n c^n | n \geq 1\}$ ist eine kontextsensitve Sprache.
G = (N,T,P,S), T = {a,b,c}, N = {S,A,B}
P = { S → aSA, S → aB, BA → bBc, cA → Ac, B → bc }

2 Kuroda Normalform

Definition:
Eine Typ-1-Grammatik ist in Kuroda Normalform, falls alle Regeln eine der vier Formen haben:

1. $A \to a$
2. $A \to B$
3. $A \to BC$
4. $AB \to CD$

Dabei sind A, B, C, D Nichtterminale und a ist ein Terminal.

Satz:
Für jede Typ-1-Grammatik G mit $\varepsilon \notin L(G)$ gibt es eine Grammatik G' in Kuroda-Normalform mit $L(G) = L(G')$.

3 Turingmaschine

Die Turingmaschine ist ein Rechnermodell, das 1936 vom britischen Mathematiker Alan M. Turing als Lösung des sogenannten *Entscheidungsproblems* entwickelt wurde.

Definition:
M sei TM mit $M = (Z,\ \Sigma,\ \Gamma,\ \delta,\ z_0,\ \square,\ E)$, wobei
Z die endliche Zustandsmenge,
Σ das Eingabealphabet,
$\Gamma \supset \Sigma$ das Arbeitsalphabet,
$\delta : Z \times \Gamma \to\ Z \times \Gamma \times \{\text{L, R, N}\}$ die Überführungsfunktion
(bzw. $\delta : Z \times \Gamma \to \mathcal{P}(Z \times \Gamma \times \{\text{L, R, N}\})$ im nichtdeterministischen Fall),
$z_0 \in Z$ der Startzustand,
$\square \in \Gamma \setminus \Sigma$ das Blank und
$E \subseteq Z$ die Menge der Endzustände ist.

Konfiguration:
Eine Konfiguration einer Turingmaschine ist ein Wort $k \in \Gamma^* Z \Gamma^*$.
Dabei wird $k = \alpha z \beta$ folgendermaßen interpretiert:
Die TM befindet sich aktuell im Zustand z und der Schreib-Lese-Kopf liest das erste Zeichen von β. Also ist β der nicht-leere Teil des Bandes rechts vom Schreib-Lese-Kopf und α der nicht-leere Teil des Bandes links vom Schreib-Lese-Kopf.

Konfigurationsübergang:
Der Konfigurationsübergang einer Turingmaschine ist eine zweistellige Relation $\vdash$, welche als Schritt einer Turingmaschine interpretiert werden kann und folgendermaßen definiert wird:

$$a_1...a_m z b_1...b_2 \ \vdash \begin{cases} a_1...a_m z' c b_2...b_n, & \delta(z,\ b_1) = (z',\ c,\ N), \quad m \geq 0,\ n \geq 1 \\ a_1...a_m c z' b_2...b_n, & \delta(z,\ b_1) = (z',\ c,\ R), \quad m \geq 0,\ n \geq 2 \\ a_1...a_{m-1} z' a_m c b_2...b_n, & \delta(z,\ b_1) = (z',\ c,\ L), \quad m \geq 1,\ n \geq 1 \end{cases}$$

Sonderfälle:

$$\begin{array}{lcll} a_1...a_m z b_1 & \vdash & a_1...a_m c z' \square, & \delta(z,\ b_1) = (z',\ c,\ R), \quad m \geq 0,\ n = 1 \\ z b_1...b_n & \vdash & z' \square c b_2...b_n, & \delta(z,\ b_1) = (z',\ c,\ L), \quad m = 0,\ n \geq 1 \end{array}$$

akzeptierte Sprache:
Die von einer Turingmaschine akzeptierte Sprache ist die Menge aller Wörter, die nach beliebig vielen Schritten der TM zu einem Endzustand führen. Formal bedeutet dies:
$T(M) = \{x \in \Sigma^* \mid z_0 x \vdash^* \alpha z \beta;\ \alpha, \beta \in \Gamma^*;\ z \in E\}$

Linear beschränkte Turingmaschinen
Eine linear beschränkte Turingmaschine verlässt nie die Felder des Bandes auf denen die Eingabe stand. Dabei sind das letzte und auch das erste Zeichen der Eingabe speziell markiert. Folgend ist ein Beispiel für solch eine Maschine gegeben.
Gesucht ist eine TM die die letzte Stelle einer Binärzahl löscht. Dafür geht sie zum letzten Zeichen und überschreibt es mit dem Blanksymbol. In diesem Beispiel ist es nur notwendig, dass das letzte Zeichen des Eingabewortes markiert ist. Die Markierung folgt durch das ˆ Symbol. Dabei ist:

- $Z = \{z_0, z_e\}$ und z_0 ist das Startsymbol, $E = \{z_e\}$
- $\Gamma = \{0, 1, \square\}$ und $\Sigma = \{0, 1, \hat{0}, \hat{1}\}$
-

δ	0	1	$\hat{0}$	$\hat{1}$
z_0	(z_0, 0, R)	(z_0, 1, R)	(z_e,$\square$,N)	(z_e,$\square$,N)

Nichtdeterministische Turingmaschinen
Eine nichtdeterministisch arbeitende Turingmaschine hat eine veränderte Übergangsfunktion. Befindet sich die TM in einem bestimmten aktuellen Zustand und liest das aktuelle Zeichen, kann es, anders als bei einer deterministischen TM mehrere Möglichkeiten für Übergänge geben. An unserem Beispiel kann man dies genauer erkennen. Es verändert sich im Vergleich zum letzten Abschnitt:

- die Zustandsmenge $Z = \{z_0, z_1, z_e\}$ und das Eingabealphabet $\Sigma = \{0, 1\}$
- sowie die Übergangsfunktion

δ	0	1	$\square$
z_0	{(z_0,0,R), (z_1,$\square$,R)}	{(z_0,1,R), (z_1,$\square$,R)}	
z_1			{(z_e,$\square$,N)}

4 Satz von Kuroda

Die von nichtdeterministisch linear beschränkten Turingmaschinen akzeptierten Sprachen sind genau die kontextsensitiven Sprachen.
Der Beweis dieses Satzes erfolgt in zwei Richtungen. Für jede der Richtungen wird ein Konstruktionsbeweis genutzt. Interessierte können den Beweis bei Schöning [1] oder Erk und Priese [3] nachlesen.

5 Folgerung für das Wortproblem

Das Wortproblem einer Sprache ist ein Entscheidungsproblem. Zu einem gegebenen Wort soll festgestellt werden, ob dieses Wort in der Sprache liegt oder nicht. Das heißt, existiert ein Algorithmus der in endlicher Zeit herausfinden kann, ob das Wort in der Sprache ist oder nicht, ist das Wortproblem entscheidbar.
Das Wortproblem für Typ-1 Sprachen ist entscheidbar. Die Idee eines Algorithmus ist es, Ableitungswege, die w erzeugen nach einem bestimmten Schema zu erstellen.
Sei G = (V,Σ,P,S) und w ∈ Σ* das gegebene Wort.
Sei für alle natürlichen m,n: $T_m^n = \{w \in (V \cup \Sigma)^* \mid |w| \leq n$ und w lässt sich aus S in höchstens m

Schritten ableiten }
Es gilt: $T_0^n = S$, sowie $T_{m+1}^n = T_m^n \cup \{w \in (V \cup \Sigma)^* \mid |w| \leq n \text{ und } w' \Rightarrow w \text{ für ein } w \in T_m^n\}$
Es kann aber immer nur eine endliche Anzahl von Wörtern in $(V \cup \Sigma)^*$ geben. Also ist irgendwann $T_m^n = T_{m+1}^n$. Dann bricht die Berechnung der Mengen ab. Nun überprüfen wir, ob w in einem der $T_m^{|w|}$ liegt. Wenn ja, ist w Element von L(G) sonst nicht.
Der Zeitbedarf für das Wortproblem ist höchstens exponentiell, die Platzkomplexität ist linear.

6 LBA-Probleme

1. LBA-Problem:
Existiert zu jedem LBA M ein DLBA M' mit $T(M) = T(M')$?
Diese Frage konnte bis heute nicht beantwortet werden.

2. LBA-Problem:
Existiert zu jedem LBA M ein LBA M' mit $T(M') = \Sigma^* \setminus T(M)$?
Diese Frage konnte unabhängig voneinander von einem amerikanischen Wissenschaftler und einem slowakischen Studenten zeitgleich durch folgenden Satz beantwortet werden.

Satz von Immerman und Szelepcsényi:
Die Klasse der kontextsensitiven (also Typ-1-) Sprachen ist unter Komplementbildung abgeschlossen.

7 Abschlusseigenschaften

Die kontextsensitiven Sprachen sind unter anderem abgeschlossen unter Vereinigung, Durchschnitt, Konkatenation, Komplementbildung und Kleene- Stern.

8 Anwendungen

Aufgrund ihrer Mächtigkeit gäbe es viele interessante praktische Anwendungen für kontextsensitive Sprachen. Zum Beispiel sind die natürlichen Sprachen nicht kontextfrei, aber es existiert die These, dass sie zumindestens kontextsensitiv sind. Allerdings haben die kontextsensitiven Sprachen exponentiellen Zeitbedarf und sind durch ihre nichtdeterministische linear beschränkte Arbeitsweise zumeist nicht effizient nutzbar.

9 Literatur und Quellen

[1] U. Schöning, Theoretische Informatik - kurz gefasst, Spektrum Akademischer Verlag, 5. Auflage, 2006
[2] A. Salomaa, Formal languages and power series, Herausgegeben von Jan van Leeuwen in Formal Models and semantics, Elevesier, 1. Auflage, 1990
[3] K. Erk, L. Priese, Theoretische Informatik, Springer, 1. Auflage, 2000
[4] R. Winter, Theoretische Informatik, Oldenbourgh, 1. Auflage, 2002
[5] M. Muth, http://theo.cs.uni-magdeburg.de/lehre04w/ti_lab/skript/script4.pdf, abgerufen am 15.11.2015