Grundlagen der Theoretischen Informatik

4. Kellerautomaten und kontextfreie Sprachen (VII)

1.07.2015

Viorica Sofronie-Stokkermans

e-mail: sofronie@uni-koblenz.de

Übersicht

1. Motivation
2. Terminologie
3. Endliche Automaten und reguläre Sprachen
4. Kellerautomaten und kontextfreie Sprachen
5. Turingmaschinen und rekursiv aufzählbare Sprachen
6. Berechenbarkeit, (Un-)Entscheidbarkeit
7. Komplexitätsklassen P und NP

Bis jetzt

- PDA (Definition)
- Ein PDA $\mathcal{M}$ kann auf zwei verschiedene Arten eine Sprache akzeptieren:
  - über **finale Zustände**
  - über **leeren Keller**

  $L_f(\mathcal{M}), L_l(\mathcal{M})$
  - Zu jedem PDA $\mathcal{M}_1$ existiert ein PDA $\mathcal{M}_2$ mit $L_f(\mathcal{M}_1) = L_l(\mathcal{M}_2)$
  - Zu jedem PDA $\mathcal{M}_1$ existiert ein PDA $\mathcal{M}_2$ mit $L_l(\mathcal{M}_1) = L_f(\mathcal{M}_2)$
- Gleichmächtigkeit: PDAs und cf-Grammatiken
- Abschlusseigenschaften:
  - $\mathbf{L}_2$ ist abgeschlossen gegen: $\cup$, $\circ$, *
  - $\mathbf{L}_2$ ist nicht abgeschlossen gegen: $\cap$, $\neg$
- Wortprobleme

Wortprobleme

Problem

Gegeben: eine cf-Grammatik G, so dass $L(G)$ eine Sprache ist über Σ, und ein Wort $w \in \Sigma^*$

Frage: Ist $w \in L(G)$?

Cocke-Younger-Kasami-Algorithmus (CYK-Algorithmus)
Auch: Chart-Parsing

Chart-Parsing

Gegeben: Ein Wort

$$w = a_1 \dots a_n$$

Idee

- Prinzip der dynamischen Programmierung
- 1.: Ermittle woraus sich die einstelligen Teilworte ableiten lassen
  2.: Ermittle woraus sich die zweistelligen Teilworte ableiten lassen
  ...
  n.: Ermittle woraus sich die n-stelligen Teilworte (w selbst) ableiten lassen

Chart-Parsing

Darstellung als Array: Für eine Kante, die den i. bis j. Buchstaben überspannt und mit A markiert ist, steht im $[i, j]$-Element des Arrays die Eintragung A.

Sei $L = L(G)$ kontextfrei, $G = (V, T, R, S)$ in Chomsky-Normalform, $M, N \subseteq V$.

Definition ($M * N$): $M * N := \{A \in V \mid \exists B \in M, \exists C \in N : \; A \to BC \in R\}$

Definition ($\mathbf{w_{i,j}}, \mathbf{V_{i,j}}$): Sei $w = a_1 \ldots a_n$ mit $a_i \in \Sigma$.

- $w_{i,j} := a_i \ldots a_j$ ist das Fragment von w vom i-ten bis zum j-ten Buchstaben
- $V_{i,j} := \{A \in V \mid A \Longrightarrow_G^* w_{i,j}\}$

Lemma. Sei $w = a_1 \ldots a_n$ mit $a_i \in \Sigma$. Dann gilt:

1. $V_{i,i} = \{A \in V \mid A \to a_i \in R\}$
2. $V_{i,k} = \bigcup_{j=i}^{k-1} V_{i,j} * V_{j+1,k}$ für $1 \leq i < k \leq n$

Beachte: Die Grammatik muss in Chomsky-Normalform sein!

Chart-Parsing

Beweis.

1. $V_{i,i} = \{A \in V \mid A \Longrightarrow_G^* a_i\} = \{A \in V \mid A \to a_i \in R\}$, da G in CNF ist.

$A \in V_{i,k}$ mit $1 \leq i < k \leq n$

<u>gdw</u> $A \Longrightarrow_G^* a_i \dots a_k$

<u>gdw</u> $\exists j,\ i \leq j < k:\ \exists B, C \in V : A \Longrightarrow BC$, und
$B \Longrightarrow_G^* w_{i,j} \neq \varepsilon$
und $C \Longrightarrow_G^* w_{j+1,k} \neq \varepsilon$ (da G in CNF ist)

<u>gdw</u> $\exists j,\ i \leq j < k:\ \exists B, C \in V : A \Longrightarrow BC$
und $B \in V_{i,j}$ und $C \in V_{j+1,k}$

2. <u>gdw</u> $\exists j,\ i \leq j < k : A \in V_{i,j} * V_{j+1,k}$

CYK-Algorithmus (Cocke-Younger-Kasami)

Algorithmus

Input sei eine Grammatik G in CNF und ein Wort $w = a_1 \ldots a_n \in \Sigma^*$.

(i) **for** $i := 1$ *to* n **do** `/* Regeln A → a eintragen */`

$$V_{i,i} := \{A \in V \mid A \to a_i \in R\}$$

(ii) **for** $h := 1$ *to* $n - 1$ **do**

for $i := 1$ *to* $n - h$ **do**

$$V_{i,i+h} = \bigcup_{j=i}^{i+h-1} V_{i,j} * V_{j+1,i+h}$$

(iii) **if** $S \in V_{1,n}$ **then return** `Ausgabe` $w \in L(G)$

else return `Ausgabe` $w \notin L(G)$

CYK-Algorithmus (Cocke-Younger-Kasami)

Beispiel:

$G = (\{S, A, B\}, \{a, b\}, R, S)$ mit

$$
\begin{aligned}
R: \quad & S \to AB \mid BC \\
& A \to BA \mid a \\
& B \to CC \mid b \\
& C \to AB \mid a
\end{aligned}
$$

An der Tafel.

CYK-Algorithmus (Cocke-Younger-Kasami)

Eigenschaften

Für Wörter der Länge $|w| = n$ entscheidet der CYK-Algorithmus in der Größenordnung von n^3 Schritten, ob $w \in L(G)$ ist.

CYK-Algorithmus (Cocke-Younger-Kasami)

Beispiel.

Eine Grammatik in CNF, die dieselbe Sprache wie oben erzeugt:
$G = (\{S, S_a, S_b, A, B\}, \{a, b\}, R, S\}$

$$
\begin{aligned}
R = \{ \quad & S \to AS_a \mid BS_b \mid AA \mid BB \\
& S_a \to SA \\
& S_b \to SB \\
& A \to a \\
& B \to b\}
\end{aligned}
$$

Die Sprache ist: $L(G) = \{vv^R \mid v \in \{a, b\}^+\}$

An der Tafel.

Übersicht

1. Motivation
2. Terminologie
3. Endliche Automaten und reguläre Sprachen
4. Kellerautomaten und kontextfreie Sprachen
5. Turingmaschinen und rekursiv aufzählbare Sprachen
6. Berechenbarkeit, (Un-)Entscheidbarkeit
7. Komplexitätsklassen P und NP

Übersicht

1. Motivation
2. Terminologie
3. Endliche Automaten und reguläre Sprachen
4. Kellerautomaten und kontextfreie Sprachen
5. Turingmaschinen und rekursiv aufzählbare Sprachen
6. Berechenbarkeit, (Un-)Entscheidbarkeit
7. Komplexitätsklassen P und NP