Vorlesung

Grundlagen der Theoretischen Informatik / Einführung in die Theoretische Informatik I

Bernhard Beckert

Institut für Informatik

UNIVERSITÄT KOBLENZ · LANDAU

Sommersemester 2007

Dank

Diese Vorlesungsmaterialien basieren ganz wesentlich auf den Folien zu den Vorlesungen von

Katrin Erk (gehalten an der Universität Koblenz-Landau)

Jürgen Dix (gehalten an der TU Clausthal)

Ihnen beiden gilt mein herzlicher Dank.

– Bernhard Beckert, April 2007

Teil IV

Erzeugende Grammatiken – akzeptierende Automaten

Erinnerung: Reguläre Sprachen

- werden erzeugt von rechtslinearen Grammatiken
- werden akzeptiert von endlichen Automaten

Jetzt: Kontextfreie Sprachen

- werden erzeugt von kontextfreien Grammatiken
- werden akzeptiert von **Pushdown-Automaten**

Idee des Push-Down-Automaten

Beispiel 22.1

Die „prototypische“ cf-Sprache

$$\{a^n b^n \mid n \in \mathbb{N}_0\}$$

- Endliche Automaten reichen nicht aus.
- Sie können sich nicht merken, wie oft sie einen Zustand durchlaufen haben.
- Für $a^n b^n$ muss man aber **mitzählen**.

Idee des Push-Down-Automaten

Idee: Wie kann man diese Sprache akzeptieren?

- Weitere Informationen auf dem Stack sichern
- Später zurückholen
- Ähnlich einem „Prozeduraufruf“
- Grammatikregel wie $S \longrightarrow aAb$ entspricht Aufruf einer Prozedur für das A.

Stack, Stapel, Keller

- Last in, first out
- Zuletzt gespeicherte Information liegt immer „obenauf“
- Beliebig viel Information kann gespeichert werden
  (Aber kein beliebiger Zugriff!)

Push-Down-Automat

Push-Down-Automat (PDA): Informell

- Wie endlicher Automat, aber **zusätzlicher** einen Stack
- Übergangsrelation bezieht das oberste Stacksymbol in den Übergang ein
- Bei Zustandsübergang: lesen und schreiben auf Stack

Push-Down-Automat

Definition 22.2 (Push-Down-Automat)

Ein **Push-Down-Automat (PDA)** ist ein Tupel

$$\mathcal{M} = (K, \Sigma, \Gamma, \Delta, s_0, Z_0, F)$$

Dabei ist

K	eine endliche Menge von Zuständen
Σ	das Eingabealphabet
Γ	das Stack- oder Kelleralphabet
$s_0 \in K$	der Startzustand
$Z_0 \in \Gamma$	das Anfangssymbol im Keller
$F \subseteq K$	eine Menge von finalen Zuständen
Δ	die Zustandsübergangsrelation, eine endliche Relation:

$$\Delta \subset \quad (K \times (\Sigma \cup \{\varepsilon\}) \times \Gamma) \times (K \times \Gamma^*)$$

Push-Down-Automat

Arbeitsschritt eines PDA

In Abhängigkeit

- vom aktuellen Zustand
- vom nächsten Eingabezeichen (oder auch unabhängig davon)
- vom obersten Kellersymbol

geschieht folgendes

- nächstes **Eingabezeichen** wird **gelesen oder nicht** (bei ε),
- das oberste **Kellersymbol** wird **entfernt**,
- der **Zustand** wird **geändert**,
- es werden null oder mehr **Zeichen auf den Keller** geschoben
  Bei neuen Keller-Wort $\gamma = A_1 \ldots A_n$ wird A_n zuerst auf den Keller geschoben usw., so daß am Schluss A_1 obenauf liegt.

Push-Down-Automat

Notation

- a, b, c für Buchstaben aus Σ
- u, v, w für Wörter aus Σ^*
- A, B für Stacksymbole aus Γ
- γ, η für Stackinhalte aus Γ^*

Push-Down-Automat: Konfiguration

Konfiguration eines PDA: Informell

- Konfiguration beschreibt die aktuelle Situation des PDA **komplett**
- Bestandteile:
  - **aktueller Zustand**
  - **noch zu lesendes Restwort**
  - **kompletter Stackinhalt**
- Für Konfigurationen C_1, C_2 bedeutet

$$C_1 \vdash C_2$$

daß der PDA in einem Schritt von C_1 nach C_2 gelangen kann.

Push-Down-Automat: Konfiguration

Definition 22.3 (Konfiguration eines PDA, $\vdash$)

Eine **Konfiguration** C eines PDA $\mathcal{M} = (K, \Sigma, \Gamma, \Delta, s_0, Z_0, F)$ ist ein Tripel

$$(q, w, \gamma) \in K \times \Sigma^* \times \Gamma^*.$$

- q der aktuelle Zustand
- w der noch zu lesendes Restwort
- γ der komplette Stackinhalt

Definition 22.4 (Startkonfiguration)

Bei Eingabewort w ist die **Startkonfiguration**:

$$(s_0, w, Z_0)$$

Push-Down-Automat: Konfiguration

Definition 22.5 (Nachfolgekonfiguration)

C_2 heißt **Nachfolgekonfiguration** von C_1,

$$C_1 \vdash C_2$$

falls

$$\exists a \in \Sigma \; \exists A \in \Gamma \; \exists w \in \Sigma^* \; \exists \gamma, \eta \in \Gamma^*$$

so dass

entweder $C_1 = (q_1, aw, A\gamma)$, $C_2 = (q_2, w, \eta\gamma)$, und $(q_1, a, A) \, \Delta \, (q_2, \eta)$,

oder $C_1 = (q_1, w, A\gamma)$, $C_2 = (q_2, w, \eta\gamma)$, und $(q_1, \varepsilon, A) \, \Delta \, (q_2, \eta)$,

Push-Down-Automat: Rechnung

Definition 22.6 (Rechnung eines PDA)

Sei $\mathcal{A}$ ein Push-Down-Automat.

$$C \vdash_A^* C'$$

gdw es eine Reihe von Konfigurationen

$$C_0, C_1, \ldots, C_n \qquad (n \geq 0)$$

so daß

- $C = C_0$,
- $C' = C_n$,
- $C_i \vdash_A C_{i+1}$ für alle $0 \leq i < n$

Dann heißt $C_0, C_1, \ldots, C_n$ eine **Rechnung** von A

Push-Down-Automat: Akzeptierte Sprache

Definition 22.7 (von PDA akzeptierte Sprache)

Ein PDA $\mathcal{M}$ kann auf zwei verschiedene Arten eine Sprache akzeptieren:

- über **finale Zustände**
- über **leeren Keller**

$$L_f(\mathcal{M}) = \{w \in \Sigma^* \mid \exists q \in F \; \exists \gamma \in \Gamma^* \; \big((s_0, w, Z_0) \vdash^*_{\mathcal{M}} (q, \varepsilon, \gamma)\big)\}$$

$$L_l(\mathcal{M}) = \{w \in \Sigma^* \mid \exists q \in K \; \big((s_0, w, Z_0) \vdash^*_{\mathcal{M}} (q, \varepsilon, \varepsilon)\big)\}$$

Bemerkung

Das zu akzeptierende Wort w muss von $\mathcal{M}$ ganz gelesen werden:

$$(s_0, w, Z_0) \vdash^* (q, \varepsilon, \cdot)$$

ist gefordert.

Bemerkung

- Das unterste Symbol im Keller kann gelöscht werden.
- Dann aber **hängt** der PDA
- Er kann nicht mehr weiter rechnen
- **Es gibt keineNachfolgekonfiguration**

Push-Down-Automat: Beispiel

Beispiel 22.8

Sprache der Palindrome über $\{a, b\}$:

$$L = \{w \in \{a, b\}^* \mid w = w^R\}$$

L wird über leeren Keller akzeptiert von dem PDA

$$\mathcal{M} := (\{s_0, s_1\}, \{a, b\}, \{Z_0, A, B\}, \Delta, s_0, Z_0, \emptyset)$$

mit ...

Push-Down-Automat: Beispiel

Beispiel (Forts.)

Idee:

- Ein Palindrom $w = w^R$ hat die Form

  $$vv^R \quad \text{oder} \quad vav^R \quad \text{oder} \quad vbv^R$$

  für ein $v \in \{a, b\}^*$
- Der Automat $\mathcal{M}$ liest v und merkt sich jeden Buchstaben.
- **Er rät indeterminiert die Wortmitte.**
  Falls das Wort eine ungerade Anzahl von Buchstaben hat,
  also $w = vav^R$ oder $w = vbv^R$,
  dann muss dabei ein Buchstabe überlesen werden.
- Der Stack enthält nun v^R.
  $\mathcal{M}$ muss jetzt nur noch jeden weiteren gelesenen Buchstaben mit dem jeweils obersten Kellersymbol vergleichen.

Push-Down-Automat: Beispiel

Beispiel (Forts.)

$$(s_0, \varepsilon, Z_0)\ \Delta\ (s_1, \varepsilon) \quad \big\}\ \varepsilon \text{ akzeptieren}$$

$$\left.\begin{array}{ll}
(s_0, a, Z_0) & \Delta\ (s_0, A) \\
(s_0, a, A) & \Delta\ (s_0, AA) \\
(s_0, a, B) & \Delta\ (s_0, AB) \\
(s_0, b, Z_0) & \Delta\ (s_0, B) \\
(s_0, b, A) & \Delta\ (s_0, BA) \\
(s_0, b, B) & \Delta\ (s_0, BB)
\end{array}\right\} \text{Stack aufbauen}$$

Push-Down-Automat: Beispiel

Beispiel (Forts.)

$$\left.\begin{array}{ll}(s_0,\varepsilon,A) & \Delta\,(s_1,\varepsilon) \\ (s_0,\varepsilon,B) & \Delta\,(s_1,\varepsilon)\end{array}\right\}\ \begin{array}{l}\text{Richtungswechsel für Palindrome} \\ \text{mit ungerader Buchstabenanzahl}\end{array}$$

$$\left.\begin{array}{ll}(s_0,a,A) & \Delta\,(s_1,\varepsilon) \\ (s_0,b,B) & \Delta\,(s_1,\varepsilon)\end{array}\right\}\ \begin{array}{l}\text{Richtungswechsel für Palindrome} \\ \text{mit gerader Buchstabenanzahl}\end{array}$$

$$\left.\begin{array}{ll}(s_1,a,A) & \Delta\,(s_1,\varepsilon) \\ (s_1,b,B) & \Delta\,(s_1,\varepsilon)\end{array}\right\}\ \text{Stack abbauen}$$

Push-Down-Automat: Beispiel

Beispiel (Forts.)

Für das Eingabewort *abbabba* rechnet $\mathcal{M}$ so:

$$\begin{array}{ccccc} (s_0, abbabba, Z_0) & \vdash & (s_0, bbabba, A) & \vdash & (s_0, babba, BA) \vdash \\ (s_0, abba, BBA) & \vdash & (s_0, bba, ABBA) & \vdash & (s_1, bba, BBA) \vdash \\ (s_1, ba, BA) & \vdash & (s_1, a, A) & \vdash & (s_1, \varepsilon, \varepsilon) \end{array}$$