Grundlagen der Theoretischen Informatik

4. Kellerautomaten und kontextfreie Sprachen (V)

23.06.2015

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

Push-Down-Automat: Beispiel

Die Sprache

$$L = \{w \in \{a, b\}^* \mid \#_a(w) = \#_b(w)\}$$

wird über finalen Zustand akzeptiert von dem PDA

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

mit . . .

Push-Down-Automat: Beispiel

Idee:

- auf dem Stack mitzählen, wieviel A-Überhang oder B-Überhang momentan besteht
- Der Stack enthält zu jedem Zeitpunkt
  - entweder nur $A/\underline{A}$ (A-Überhang)
  - oder nur $B/\underline{B}$ (B-Überhang)
  - oder nur das Symbol Z_0 (Gleichstand).
- Das unterste A bzw. B auf dem Stack ist durch einen Unterstrich gekennzeichnet.

  So weiß $\mathcal{M}$, wenn er dies Stacksymbol löscht, dass dann bis zu diesem Moment gleich viel as wie bs gelesen wurden.

Push-Down-Automat: Beispiel

$$(s_0, a, Z_0)\ \Delta\ (s_1, \underline{A}) \qquad (s_0, b, Z_0)\ \Delta\ (s_1, \underline{B})$$

$$(s_1, a, \underline{A})\ \Delta\ (s_1, A\underline{A}) \qquad (s_1, b, \underline{B})\ \Delta\ (s_1, B\underline{B})$$

$$(s_1, a, A)\ \Delta\ (s_1, AA) \qquad (s_1, b, B)\ \Delta\ (s_1, BB)$$

$$(s_1, a, \underline{B})\ \Delta\ (s_0, Z_0) \qquad (s_1, b, \underline{A})\ \Delta\ (s_0, Z_0)$$

$$(s_1, a, B)\ \Delta\ (s_1, \varepsilon) \qquad (s_1, b, A)\ \Delta\ (s_1, \varepsilon)$$

PDA: Finaler Zustand / leerer Keller

PDA: Finaler Zustand / leerer Keller

Definition (von PDA akzeptierte Sprache)

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

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

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

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

Bemerkung

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

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

ist gefordert.

PDA: Finaler Zustand / leerer Keller

Theorem (finale Zustände $\rightarrow$ leerer Keller)

Zu jedem PDA $\mathcal{M}_1$ existiert ein PDA $\mathcal{M}_2$ mit

$$L_f(\mathcal{M}_1) = L_l(\mathcal{M}_2)$$

PDA: Finaler Zustand / leerer Keller

Theorem (finale Zustände $\rightarrow$ leerer Keller)
Zu jedem PDA $\mathcal{M}_1$ existiert ein PDA $\mathcal{M}_2$ mit

$$L_f(\mathcal{M}_1) = L_l(\mathcal{M}_2)$$

Beweisidee

- Wir simulieren die Maschine $\mathcal{M}_1$, die über finale Zustände akzeptiert, durch die Maschine $\mathcal{M}_2$, die über leeren Keller akzeptiert.
- $\mathcal{M}_2$ arbeitet wie $\mathcal{M}_1$, mit dem Unterschied:

  Wenn ein Zustand erreicht wird, der in $\mathcal{M}_1$ final war, kann $\mathcal{M}_2$ seinen Keller leeren.

PDA: Finaler Zustand / leerer Keller

Theorem (leerer Keller $\rightarrow$ finale Zustände)
Zu jedem PDA $\mathcal{M}_1$ existiert ein PDA $\mathcal{M}_2$ mit

$$L_l(\mathcal{M}_1) = L_f(\mathcal{M}_2)$$

Beweisidee

- Wir simulieren die Maschine $\mathcal{M}_1$, die über leeren Keller akzeptiert, durch die Maschine $\mathcal{M}_2$, die über finale Zustände akzeptiert.
- $\mathcal{M}_2$ arbeitet wie $\mathcal{M}_1$,
  legt aber ein zusätzliches Symbol ganz unten in den Keller.
  Wenn $\mathcal{M}_1$ seinen Keller geleert hätte (also das neue unterste Symbol sichtbar wird),
  kann $\mathcal{M}_2$ in einen finalen Zustand gehen.

Gleichmächtigkeit: PDAs und cf-Grammatiken

Theorem (PDA akzeptieren $\mathbf{L}_2$)

Die Klasse der PDA-akzeptierten Sprachen ist $\mathbf{L}_2$.

Gleichmächtigkeit: PDAs und cf-Grammatiken

Theorem (PDA akzeptieren $\mathbf{L}_2$)
Die Klasse der PDA-akzeptierten Sprachen ist $\mathbf{L}_2$.

Beweis Dazu beweisen wir die folgenden zwei Lemmata, die zusammen die Aussage des Satzes ergeben.

Lemma (cf-Grammatik $\to$ PDA)
Zu jeder kontextfreien Grammatik G gibt es einen PDA $\mathcal{M}$ mit

$$L(\mathcal{M}) = L(G)$$

Lemma (PDA $\to$ cf-Grammatik)
Zu jedem Push-Down-Automaten $\mathcal{M}$ gibt es eine kontextfreie Grammatik G mit

$$L(G) = L(\mathcal{M})$$

Gleichmächtigkeit: PDAs und cf-Grammatiken

Lemma (cf-Grammatik $\rightarrow$ PDA) Zu jeder kontextfreien Grammatik G gibt es einen PDA $\mathcal{M}$ mit

$$L(\mathcal{M}) = L(G)$$

Beweis

O.B.d.A. sei die kontextfreie Grammatik $G = (V, T, R, S)$ in Greibach-Normalform: Alle Grammatikregeln haben die Form

$$A \rightarrow au \qquad \text{mit } A \in V,\ a \in T,\ u \in V^*$$

Wir konstruieren zu G einen PDA $\mathcal{M}$, der $L(G)$ akzeptiert.

Gleichmächtigkeit: PDAs und cf-Grammatiken

Beweis (Forts.)

Idee: Der Automat $\mathcal{M}$

- vollzieht die Grammatikregeln nach, die angewendet worden sein könnten, um das aktuelle Eingabewort zu erzeugen und
- merkt sich das aktuelle Wort in der Ableitung bzw. dessen Rest
- merkt sich auf dem Keller alle Variablen, die im gedachten Ableitungswort noch vorkommen und noch ersetzt werden müssen.
- Die linkeste Variable liegt zuoberst: $\mathcal{M}$ arbeitet mit der Linksableitung.

Gleichmächtigkeit: PDAs und cf-Grammatiken

Beweis (Forts.)

Genauer:

- Erzeugung eines Wortes mit G beginnt beim Startsymbol S.

  Deshalb S bei $\mathcal{M}$ in Startkonfiguration oben auf dem Keller.

- Angenommen, G hat 2 Regeln mit S auf der linken Seite:
  $S \rightarrow aA_1A_2$ und $S \rightarrow bB_1B_2$

  Angenommen, der erste Buchstabe des Input-Wortes w ist ein a.

  Wenn w von G erzeugt wurde, hat G die erste der zwei S-Produktionen angewendet.

  Entsprechend: Der Automat $\mathcal{M}$ schiebt A_1A_2 auf den Stack.

Gleichmächtigkeit: PDAs und cf-Grammatiken

Beweis (Forts.)

Genauer:

- Der zweite Buchstabe des Eingabeworts muss durch Anwendung einer Regel $A_1 \to a_1\alpha$ erzeugt worden sein.

  Angenommen, der zweite Buchstabe des Eingabeworts ist a_1. Dann müssen die nächsten Buchstaben des Wortes aus den Variablen in α entstehen.

  Der Automat entfernt A_1 vom Stack und legt α auf den Stack.

- Wenn es zwei Regeln $A_1 \to a_1\alpha_1$ und $A_1 \to a_1\alpha_2$ gibt, dann wählt $\mathcal{M}$ indeterminiert eine der Regeln aus.

- Der PDA hat nur einen einzigen Zustand und akzeptiert über den leeren Keller.

Gleichmächtigkeit: PDAs und cf-Grammatiken

Beweis (Forts.)

Formal:

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

mit

$$\begin{aligned}
K &:= \{s_0\} \\
\Sigma &:= T \\
\Gamma &:= V \\
Z_0 &:= S \\
F &:= \emptyset \\
\Delta &:= \{((s_0, a, A), (s_0, \alpha)) \mid A \to a\alpha \in R\}
\end{aligned}$$

Gleichmächtigkeit: PDAs und cf-Grammatiken

Beweis (Forts.)

Damit gilt (Beweis s. Buch):

$$\text{Es gibt eine Linksableitung } S \Longrightarrow_G^* x\alpha \text{ mit } x \in T^*, \alpha \in V^*$$

$$\underline{\text{gdw}}$$

$$\mathcal{M} \text{ rechnet } (s_0, x, S) \vdash_{\mathcal{M}}^* (s_0, \varepsilon, \alpha)$$

Daraus folgt unmittelbar:

$$L(G) = L_\ell(\mathcal{M})$$

Gleichmächtigkeit: PDAs und cf-Grammatiken

Beispiel. Die Sprache

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

wird generiert von der GNF-Grammatik $G = (\{S, A, B\}, \{a, b\}, R, S)$ mit

$$\begin{aligned} R = \{ \quad S &\rightarrow aSA \mid bSB \mid aA \mid bB \\ A &\rightarrow a \\ B &\rightarrow b \; \} \end{aligned}$$

Daraus kann man einen PDA mit den folgenden Regeln konstruieren:

$(s_0, a, S)\ \Delta\ (s_0, SA)$

$(s_0, a, S)\ \Delta\ (s_0, A)$

$(s_0, b, S)\ \Delta\ (s_0, SB)$

$(s_0, b, S)\ \Delta\ (s_0, B)$

$(s_0, a, A)\ \Delta\ (s_0, \varepsilon)$

$(s_0, b, B)\ \Delta\ (s_0, \varepsilon)$

Gleichmächtigkeit: PDAs und cf-Grammatiken

Lemma (PDA $\rightarrow$ cf-Grammatik) Zu jedem Push-Down-Automaten $\mathcal{M}$ gibt es eine kontextfreie Grammatik G mit

$$L(G) = L(\mathcal{M})$$

Beweis

Sei $\mathcal{M}$ ein PDA, der eine Sprache L **über leeren Keller** akzeptiert.

Wir konstruieren aus dem Regelsatz von $\mathcal{M}$ eine kontextfreie Grammatik, die L erzeugt.

Gleichmächtigkeit: PDAs und cf-Grammatiken

Beweis (Forts.)

Idee:

Die Variablen der Grammatik sind 3-Tupel der Form

$$[q, A, p]$$

Bedeutung:

Grammatik kann Wort x aus Variablen $[q, A, p]$ ableiten

gdw

$\mathcal{M}$ kann vom Zustand q in den Zustand p übergehen,
dabei A vom Keller entfernen (sonst den Keller unveränder lassen) und
das Wort x lesen:

$$\big([q, A, p] \Longrightarrow^* x\big) \quad \underline{\text{gdw}} \quad \big((q, x, A\gamma) \vdash^* (p, \varepsilon, \gamma)\big)$$

Gleichmächtigkeit: PDAs und cf-Grammatiken

Beweis (Forts.)

Formale Konstruktion: nächste Vorlesung