Deterministische Kellerautomaten (DPDAs)

Intuitiv nennen wir einen Kellerautomaten deterministisch, wenn er über Endzustände (statt mit leerem Keller) akzeptiert und wenn er in jeder Konfiguration maximal eine mögliche Transition (inklusive der ε–Transitionen) zur Verfügung hat.

Formal: Ein Kellerautomat M heißt deterministisch oder kurz DPDA **gdw** gilt:

- Neben den üblichen Komponenten eines Kellerautomaten ist noch eine Menge $E \subseteq Z$ von Endzuständen spezifiziert.
- Für alle $z \in Z$, $a \in \Sigma$ und $A \in \Gamma$ gilt

$$|\delta(z, a, A)| + |\delta(z, \varepsilon, A)| \leq 1 \ .$$

Deterministisch kontextfreie Sprachen

Die von einem DPDA $M = (Z, \Sigma, \Gamma, \delta, z_0, \#, E)$ erkannte Sprache ist dann gegeben durch

$$\{w \in \Sigma^* | \ \exists z \in E, \exists \alpha \in \Gamma^* : (z_0, w, \#) \vdash^* (z, \varepsilon, \alpha)\} \ .$$

Eine Sprache L heißt deterministisch kontexftfrei **gdw** ein DPDA existiert, der L erkennt.

Bemerkung (s. Vorlesung über Compilerbau):

DPDAs $\leftrightarrow$ LR(k)–Grammatiken

Deterministisch kontextfreie Sprachen (fortgesetzt)

Satz: Jede reguläre Sprache ist deterministisch kontexfrei.

denn: Jeder DFA kann als DPDA aufgefasst werden, der weder von seinem Keller noch von ε–Transitionen Gebrauch macht.

Satz (ohne Beweis): Jeder PDA, der mit leerem Keller akzeptiert, kann in einen äquivalenten PDA transformiert werden, der mit Endzuständen akzeptiert (und umgekehrt).

Folgerung: Jede deterministisch kontextfreie Sprache ist kontextfrei.

denn: Jeder DPDA kann als Spezialfall eines mit Endzuständen akzeptierenden PDA angesehen werden.

Beispiele

Die Sprache

$$L_1 = \{a^m b^m c^n \mid m, n \geq 1\}$$

kann von einem DPDA mit $E = \{z_2\}$ und den folgenden Transitionen erkannt werden:

$z_0 a\# \rightarrow z_0 a\#$	,	$z_0 aa \rightarrow z_0 aa$	Push–Phase
$z_0 ba \rightarrow z_1 \varepsilon$	,	$z_1 ba \rightarrow z_1 \varepsilon$	Umschalten + Pop–Phase
$z_1 c\# \rightarrow z_2\#$	,	$z_2 c\# \rightarrow z_2\#$	Umschalten + Endroutine

(Diskussion an der Tafel)

Beispiele (fortgesetzt)

Die folgenden Sprachen sind deterministisch kontextfrei:

$$\begin{aligned} L_1 &= \{a^m b^m c^n \mid m, n \geq 1\} \\ L_2 &= \{a^n b^m c^m \mid m, n \geq 1\} \\ L_3 &= \{a_1 \cdots a_n \$ a_n \cdots a_1 \mid n \geq 1, a_i \in \{a, b\}\} \end{aligned}$$

- DPDA für L_2 analog zum DPDA für L_1.
- L_3 erkennbar mit einer DPDA–Variante des PDA, der früher schon mal für diese Sprache angegeben wurde.

Die Sprache

$$L_1 \cap L_2 = \{a^m b^m c^m \mid m \geq 1\}$$

ist (wie wir bereits wissen) nicht kontextfrei.

Folgerung: Weder deterministisch kontexfreie Sprachen noch kontextfreie Sprachen sind abgeschlossen unter „$\cap$“.

Abschlusseigenschaften

Satz: Kontextfreie Sprachen sind abgeschlossen unter den Operationen „$\cup, \cdot, *$“.

denn: Gehe aus von den Grammatiken

$$G_1 = (V_1, \Sigma, P_1, S_1)\ ,\ G_2 = (V_2, \Sigma, P_2, S_2)\ ,\ V_1 \cap V_2 = \emptyset$$

für kontextfreie Sprachen L_1, L_2. Übernahme der Regeln aus P_1 und P_2 und Hinzufügen der Regeln

$$S \rightarrow S_1 \mid S_2 \text{ bzw. } S \rightarrow S_1 S_2 \text{ bzw. } S \rightarrow \varepsilon \mid S_1 \mid S_1 S$$

mit neuem Startsymbol S führt zu einer kontextfreien Grammatik für $L_1 \cup L_2$ bzw. für $L_1 \cdot L_2$ bzw. $(L_1)^*$.

Abschlusseigenschaften (fortgesetzt)

Kontextfreie Sprachen sind nicht abgeschlossen unter den Operationen „$\cap, \neg$“.

denn:

- Nicht–Abgeschlossenheit unter „$\cap$“ wurde bereits gezeigt.
- Mit deMorgan folgt die Nicht–Abgeschlossenheit unter „$\neg$“.
  Genauer: Abschluss unter „$\neg$“ hätte wegen

$$L_1 \cap L_2 = \overline{\overline{L_1 \cap L_2}} = \overline{\bar{L}_1 \cup \bar{L}_2}$$

  den Abschluss unter „$\cap$“ zur Folge (und ist daher unmöglich).

Abschlusseigenschaften (fortgesetzt)

Satz: Die deterministisch kontextfreien Sprachen sind abgeschlossen unter „$\neg$“, aber nicht abgeschlossen unter „$\cap, \cup, \cdot, *$“.

- Abschluss unter „$\neg$“ ist nicht ganz einfach zu beweisen.
  (Hier: ohne Beweis; an der Tafel: kurze Diskussion)
- Nicht–Abgeschlossenheit unter „$\cap$“ wurde bereits gezeigt.
- Nicht–Abgeschlossenheit unter „$\cup$“ folgt dann direkt mit deMorgan.
- Nicht–Abgeschlossenheit unter „$\cdot, *$“ ist einfacher zu beweisen, wenn das Konzept des „Homomorphismus“ $\varphi : \Sigma^* \rightarrow \Sigma^*$ zur Verfügung steht.
  (Hier: ohne Beweis)

Abschlusseigenschaften (fortgesetzt)

Satz: Der Durchschnitt einer (deterministisch) kontextfreien Sprache mit einer regulären Sprache liefert wieder eine (deterministisch) kontextfreie Sprache.

Zu einem gegeben PDA $M_1 = (Z_1, \Sigma, \Gamma_1, \delta_1, z_{01}, E_1, \#)$ — Akzeptieren mit Endzuständen ! — und einem gegebenen DFA $M_2 = (Z_2, \Sigma, \delta_2, z_{02}, E_2)$ betrachte den „Produktautomaten“

$$M = (Z_1 \times Z_2, \Sigma, \Gamma_1, \delta, (z_{01}, z_{02}), \#, E_1 \times E_2)$$

mit $((z_1', z_2'), B_1 \cdots B_k) \in \delta((z_1, z_2), a, A)$ **gdw**

$$(z_1', B_1 \cdots B_k) \in \delta_1(z_1, a, A) \text{ und } z_2' = \delta_2(z_2, a) \ .$$

Intuition: M führt die „Rechnungen“ von M_1 und M_2 parallel aus.

Dann gilt:

- M ist ein PDA und erkennt die Sprache $N(M_1) \cap T(M_2)$.
- Wenn M_1 ein DPDA ist, dann ist auch M ein DPDA.

Das Wortproblem

Spezielles Wortproblem für eine kontexfreie Sprache $L \subseteq \Sigma^*$

Eingabe: $x \in \Sigma^*$

Frage: $x \in L$?

Komplexität: lösbar in „kubischer“ Zeit ($O(n^3)$ Rechenschritte).

Methode: Verwende den CYK–Algorithmus: n^2 Tabelleneinträge; jeder Eintrag kann in Linearzeit ($O(n)$ Rechenschritte) bestimmt werden.

Bemerkung: Für eine deterministisch kontextfreie Sprachen kann das Wortproblem (unter Verwendung einer LR(k)–Grammatik) in Linearzeit gelöst werden.

Nützliche und nutzlose Variable

Eine Variable $A \in V$ einer kontextfreien Grammatik $G = (V, \Sigma, P, S)$ heißt

- generierend **gdw** $\exists w \in \Sigma^* : A \Rightarrow_G^* w$
  (d.h., aus A kann ein Terminalstring abgeleitet werden),
- erreichbar **gdw** $\exists \alpha, \beta \in (V \cup \Sigma)^* : S \rightarrow \alpha A \beta$,
- nützlich **gdw** A generierend und erreichbar ist,
- nutzlos **gdw** A nicht nützlich ist.

Säubern einer kontextfreien Grammatik

Eine kontextfreie Grammatik G heißt gesäubert **gdw** G keine nutzlosen Variablen enthält.

Säubern von G (ohne die Sprache abzuändern):

1. Berechne (s. Übung) die Menge aller generierenden Variablen.
2. Entferne aus G die nicht–generierenden Variablen und die Regeln, welche nicht-generierende Variablen verwenden.
3. **Kommentar:** Jetzt sind alle Variablen generierend.
4. Berechne (s. Übung) die Menge der erreichbaren Variablen.
5. Entferne aus G die nicht–erreichbaren Variablen und die Regeln, welche nicht–erreichbare Variablen enthalten.
6. **Kommentar:** Jetzt sind alle Variablen generierend und erreichbar und somit nützlich.

Bei geschickter Implementierung erfolgt Säubern in Linearzeit !

Das „Leerheitsproblem“

Eingabe: kontexfreie Grammatik G.

Frage: $L(G) = \emptyset$?

Komplexität: Linearzeit

Methode: Nutze aus:

$$L(G) \neq \emptyset \Leftrightarrow \ S \text{ ist generierend}$$

Das „Endlichkeitsproblem“

Eingabe: Kontextfreie Grammatik G

Frage: $|L(G)| < \infty$?

Komplexität: Linearzeit

Methode:

- Säubere G.
- Berechne einen Digraphen D mit Knotenmenge V, der eine Kante (A, B) enthält **gwd** es gibt eine Regel mit A auf der linken und B auf der rechten Seite.
- Nutze aus:

$$|L(G)| = \infty \Leftrightarrow D \text{ enthält einen Zyklus}$$

Ausblick: Nicht–entscheidbare Probleme

Im Kapitel über Berechenbarkeitstheorie
werden wir sehen:
Es gibt algorithmisch nicht entscheidbare Probleme.

Beispiele für nicht–entscheidbare Probleme

Für keines der folgenden Probleme gibt es einen Entscheidungsalgorithmus:

Schnittproblem: Gegeben sind zwei kontextfreie Grammatiken (oder zwei PDAs).

Frage: Existiert ein Terminalstring, der von beiden Grammatiken erzeugt (von beiden PDAs akzeptiert) werden kann ?

Kommentar: Sogar das Schnittproblem für deterministisch kontextfreie Sprachen (etwa gegeben durch DPDAs) ist unentscheidbar.

Äquivalenzproblem Erzeugen zwei gegebene kontextfreie Grammatiken dieselbe Sprache?

Das Äquivalenzproblem für deterministisch kontextfreie Sprachen

— Erkennen zwei gegebene DPDAs dieselbe Sprache ? —

ist hingegen entscheidbar.

Kommentar: 1997 fand Senizergues einen Entscheidungsalgorithmus.