

Satz von Kuroda und LBA-Probleme

Prof. Dr. David Sabel

LFE Theoretische Informatik



Letzte Änderung der Folien: 20. Juni 2022

Kontextsensitive Sprachen durch LBAs erkennbar

Satz

Jede kontextsensitive Sprache wird von einem LBA erkannt.

Beweis:

- Sprache sei als $G = (V, \Sigma, P, S)$ in Kuroda-Normalform gegeben.
- Konstruiere TM mit Bandalphabet $((\Sigma \cup V) \cup \widehat{\Sigma} \cup \widehat{V} \cup \square) \subseteq \Gamma$
- Zur einfacheren Illustration unterscheiden wir nicht zwischen a und \hat{a} und schreiben a auch für den letzten Buchstaben der Eingabe, aber: Wir gehen davon aus, dass der LBA entsprechend programmiert ist, die notwendigen Ersetzungen zu machen.
- Die TM versucht nichtdeterministisch für $w \in \Sigma^*$ das Startsymbol S der Grammatik rückwärts herzuleiten, durch rückwärts Anwenden der Produktionen $\ell \rightarrow r \in P$: ersetze Vorkommen von r durch ℓ
- ...

Satz von Kuroda

Theorem (Satz von Kuroda)

Kontextsensitive Sprachen werden genau von den LBAs erkannt.

Der Beweis erfolgt in zwei Teilen:

- Kontextsensitive Sprachen sind durch LBAs erkennbar
- LBAs erkennen kontextsensitive Sprachen

Dabei auch: Betrachtung von allgemeinen TMs und Typ 0-Sprachen.

Kontextsensitive Sprachen durch LBAs erkennbar (2)

- Für Produktionen $A \rightarrow a, A \rightarrow B, AB \rightarrow CD$ kann man direkt ersetzen, für den Fall $A \rightarrow BC$ wird BC durch $\square A$ ersetzt und dann alle Zeichen von links um eins nach rechts verschoben.
- Akzeptiere, wenn Startsymbol S alleine auf dem Band steht.
- Nichtdeterminismus: Welche Produktion wird rückwärts angewendet und für welches Vorkommen einer rechten Seite?

Suche nach einer rechter Seite r :

- Beginne links an der Eingabe und laufe diese durch.
- Speichere im aktuellen Zustand: Symbol links vom Schreib-Lesekopf
- Entscheide mit dem aktuellen Symbol, ob es passende Produktion gibt (da rechte Seiten von Produktionen in Kuroda-Normalform aus maximal 2 Zeichen bestehen, genügt dies).

Kontextsensitive Sprachen durch LBAs erkennbar (3)

Ersetzung r durch ℓ :

- Für $A \rightarrow a$ und $A \rightarrow B$ wird das aktuelle Symbol durch A ersetzt, anschließend wird der nächste Schritt gestartet (d.h. es gibt einen Zustand, der den Schreib-Lesekopf nach links fährt).
- Für $AB \rightarrow CD$, wird B geschrieben und der Kopf nach links wechseln, dann A geschrieben und der nächste Schritt gestartet.
- Für $A \rightarrow BC$ schreibe A und wechsele nach links, schreibe \square , fahre ganz nach links und starte Prozedur zum Verschieben der Zeichen nach rechts, solange bis die Lücke geschlossen ist.

Verschieben nach rechts:

- Zustand speichert das linke Symbol
- Laufen nach rechts: aktuelles Symbol mit dem gespeicherten vertauschen
- Vertauschen beenden nachdem \square mit einem anderen Symbol vertauscht wird.

Kontextsensitive Sprachen durch LBAs erkennbar (4)

Die TM ist ein LBA:

- Da für alle Produktionen $\ell \rightarrow r \in P$ gilt: $|\ell| \leq |r|$, werden nur Teilworte r durch gleichlange oder kürzere Teilworte ℓ ersetzt
- TM kommt mit dem Platz der Eingabe aus \square

Bemerkungen und Typ 0-Grammatiken

Bemerkung 1:

- Konstruktion funktioniert auch für Grammatiken nicht in Kuroda-Normalform, ist aber komplizierter:
- Speichere im Zustand $q - 1$ Zeichen, wobei q die Länge der längsten rechten Seite
- Funktioniert immer noch mit endlich vielen Zuständen und als LBA

Bemerkung 2:

- Konstruktion funktioniert auch für Typ 0-Grammatiken: Platz allerdings dann unbeschränkt (kein LBA!)

Satz

Jede Typ i -Sprache (für $i=0,1,2,3$) wird von einer nichtdeterministischen Turing-Maschine akzeptiert.

LBAs erkennen kontextsensitive Sprachen

Satz

Sei M ein LBA. Dann ist $L(M)$ eine kontextsensitive Sprache.

Beweis:

- Sei $M = (Z, \Sigma \cup \hat{\Sigma}, \Gamma, \delta, z_0, \square, E)$
- Wir konstruieren eine Typ 1-Grammatik G mit $L(G) = L(M)$
- Idee für die Grammatik:
 - 1 Erzeuge beliebiges $w \in \Sigma^*$ und Startkonfiguration von M für w
 - 2 Simuliere LBA zum Prüfen, ob $w \in L(M)$
 - 3 Wenn LBA akzeptiert erzeuge w endgültig
- Variablen der Grammatik:
 - Neue Variablen S und A
 - Variablen der Form $\begin{pmatrix} u \\ v \end{pmatrix}$ wobei $u \in \Sigma$ und $v \in \Gamma \cup \{Z\}$Obere Komponenten ergeben Wort w , untere Komponenten ergeben TM-Konfiguration.

LBA erkennen kontextsensitive Sprachen (2)

- Regeln zur Erzeugung von $w \in \Sigma^*$ Startkonfiguration zw :

$$P_1 := \left\{ S \rightarrow A \left\langle \frac{a}{\hat{a}} \right\rangle \mid a \in \Sigma \right\} \cup \left\{ A \rightarrow A \left\langle \frac{a}{a} \right\rangle \mid a \in \Sigma \right\} \cup \left\{ A \rightarrow \left\langle \frac{a}{z_0 a} \right\rangle \mid a \in \Sigma \right\}$$

- Worte $w \in L(M)$ mit $|w| < 2$ können dadurch nicht erzeugt werden, daher direkt alle Worte aus $L(M)$ der Länge < 2 erzeugen:

$$P_0 = \{ S \rightarrow w \mid |w| < 2, w \in L(M) \}$$

- Für $a_1 \cdots a_n \in \Sigma^*$ mit $n > 1$ gilt: $S \Rightarrow_{P_1} A \left\langle \frac{a_n}{\hat{a}_n} \right\rangle \Rightarrow_{P_1}^* \left\langle \frac{a_1}{z_0 a_1} \right\rangle \left\langle \frac{a_2}{a_2} \right\rangle \cdots \left\langle \frac{a_n}{\hat{a}_n} \right\rangle$
- Regelsatz P_2 simuliert M auf **den unteren** Komponenten. Wir bilden:

$$P_2 := \left\{ \left\langle \frac{a}{u} \right\rangle \left\langle \frac{b}{v} \right\rangle \rightarrow \left\langle \frac{a}{u'} \right\rangle \left\langle \frac{b}{v'} \right\rangle \mid a, b \in \Sigma \text{ und } uv \rightarrow u'v' \in P_2^{\text{unten}} \right\} \\ \cup \left\{ \left\langle \frac{a}{u} \right\rangle \rightarrow \left\langle \frac{a}{u'} \right\rangle \mid a \in \Sigma \text{ und } u \rightarrow u' \in P_2^{\text{unten}} (\text{mit } u, u' \in \Gamma \cup (ZT)) \right\}$$

wobei wir P_2^{unten} noch definieren.

LBA erkennen kontextsensitive Sprachen (3)

$$P_2^{\text{unten}} := \{ cza \rightarrow z'cb \mid \text{für alle } c \in \Gamma \text{ und } (z', b, L) \in \delta(z, a) \} \\ \cup \{ zac \rightarrow bz'c \mid \text{für alle } c \in \Gamma \text{ und } (z', b, R) \in \delta(z, a) \} \\ \cup \{ za \rightarrow zb \mid \text{für alle } c \in \Gamma \text{ und } (z', b, N) \in \delta(z, a) \}$$

Es gilt:

- $wzw' \vdash_M^* uz'u'$ g.d.w. $wzw' \Rightarrow_{P_2^{\text{unten}}}^* uz'u'$
- Dabei: Darstellung von z, z' in der Ableitung immer verbunden mit einem Zeichen aus Γ

P_3 : Nach Akzeptieren des LBA, erstelle aus Tupelfolgen das Wort $a_1 \cdots a_n$

$$P_3 := \left\{ \left\langle \frac{b}{za} \right\rangle \rightarrow b \mid z \in E, a \in \Gamma, b \in \Sigma \right\} \cup \left\{ \left\langle \frac{b}{a} \right\rangle \rightarrow b \mid a \in \Gamma, b \in \Sigma \right\}$$

Es gilt $\left\langle \frac{a_1}{b_1} \right\rangle \cdots \left\langle \frac{a_m}{b_m} \right\rangle \left\langle \frac{a_{m+1}}{zb_{m+1}} \right\rangle \left\langle \frac{a_{m+2}}{b_{m+2}} \right\rangle \cdots \left\langle \frac{a_n}{b_n} \right\rangle \Rightarrow_{P_3}^* a_1 \cdots a_n$.

LBA erkennen kontextsensitive Sprachen (4)

- Sei $G = \left(\{S, A\} \cup \left\{ \left\langle \frac{u}{v} \right\rangle \mid u \in \Sigma, v \in \Gamma \cup (ZT) \right\}, \Sigma, P_0 \cup P_1 \cup P_2 \cup P_3, S \right)$.

- Dann gilt für alle $w \in \Sigma^*$: $S \Rightarrow_G^* w$ genau dann, wenn $w \in L(M)$.
- Des weiteren gilt, dass G eine kontextsensitive Grammatik ist, da es keine verkürzenden Regeln gibt. \square

Typ 0-Sprachen

Die Konstruktion der Typ 1-Grammatik aus einem LBA kann für beliebige NTMs angepasst werden:

- Zusätzliche Tupel $\left\langle \frac{\$}{c} \right\rangle$ für $c \in \Gamma \cup ZT$ und $\$$ ein neues Symbol.
- Darstellung von Konfiguration, die länger als das Eingabewort sind:

$$\left\langle \frac{a_1}{c_1} \right\rangle \cdots \left\langle \frac{a_n}{c_n} \right\rangle \left\langle \frac{\$}{c_{n+1}} \right\rangle \cdots \left\langle \frac{\$}{z_i c_m} \right\rangle \left\langle \frac{\$}{c_r} \right\rangle$$

- Regelsatz P_3 enthält Regeln $\left\langle \frac{\$}{c_i} \right\rangle \rightarrow \varepsilon$ (nicht kontextsensitiv!)

Satz

Die durch (allgemeine) nichtdeterministischen Turingmaschinen akzeptierten Sprachen sind genau die Typ 0-Sprachen.

LBA-Probleme

1. LBA-Problem

Erkennen deterministische LBAs die selben Sprachen wie nichtdeterministische LBAs?

Bis heute ungeklärt!

2. LBA-Problem

Sind die kontextsensitiven Sprachen abgeschlossen unter Komplementbildung?

Formuliert 1964 von Kuroda, 1987 gelöst von Neil Immerman als auch Róbert Szelepcsényi

Überraschenderweise positiv:

Theorem (Satz von Immerman und Szelepcsényi)

Die kontextsensitiven Sprachen sind abgeschlossen unter Komplementbildung.

Satz von Immerman und Szelepcsényi

Beweisskizze:

- Sei $G = (V, \Sigma, P, S)$ eine Typ 1-Grammatik mit $L(G) = L$.
- Konstruiere LBA M für $\bar{L} = \Sigma^* \setminus L$
- Sei $w \in \Sigma^*$. M berechnet die exakte Anzahl $A \in \mathbb{N}$ der von S aus erzeugbaren Satzformen der Länge $n \leq |w|$
- $A \leq (|V| + |\Sigma| + 1)^n$ und kann daher in $(k + 1) \cdot n$ -Bits dargestellt werden: Die passen auf das Band von M , wenn man Symbole für je $k + 1$ -Bitblöcke hat
- Anschließend: Zähle alle Satzformen u der Länge $\leq |n|$ aus $(V \cup \Sigma^*)$ auf (außer w selbst) und prüfe ob $S \Rightarrow_G^* u$ gilt
- Dabei wird ein Zähler mitgeführt, der hochgezählt wird, wenn Ableitung möglich ist
- Wenn der Zähler die Zahl A erreicht, dann akzeptiert M :
Es wurden alle ableitbaren Worte der Länge $\leq n$ aufgezählt, w war nicht dabei.
Also $w \notin L$ und damit $w \in \bar{L}$.

Satz von Immerman und Szelepcsényi (2)

Berechnung der Zahl A :

- Sei $A(m, n)$ die Zahl der Satzformen, die in höchstens m Schritten aus S erzeugbar sind und deren Länge n nicht überschreitet:
 $A(m, n) = |\{w \in (V \cup \Sigma)^* \mid |w| \leq n, S \Rightarrow^{\leq m} w\}|$.
- Wenn wir $A(i, n)$ für $i = 0, 1, 2, \dots$ berechnen, muss irgendwann $A(i, n) = A(i + 1, n)$ gelten, dann haben wir A gefunden.

Berechnung von $A(m, n)$

- Starte mit $A(0, n) = |\{S\}| = 1$
- Berechne $result = A(m + 1, n)$ durch Eingabe von $A(m, n)$
- Initial: $result = 0$.
- Äußere Schleife zählt alle Satzformen u bis zur Länge n auf
- Innere Schleife zählt nochmal alle Satzformen v bis zur Länge n auf.
- Vor Beginn der inneren Schleife: $count = 0$
- In der inneren Schleife: prüfe nichtdeterministisch, ob $S \Rightarrow^{\leq m} v$
Wenn ja $count = count + 1$;
Wenn $v = u$ oder $v \Rightarrow u$ gilt, $result = result + 1$
- Nach Ablauf der inneren Schleife, prüfe ob $count = A(m, n)$ gilt.
Wenn nein, dann verwirf diese nichtdeterministische Berechnung
Wenn ja, dann war dies die richtige nichtdeterministische Berechnung und es wurde für alle in $\leq m$ -Schritten aus S herleitbaren Satzformen v geprüft, ob durch Verlängern mit $=$ oder \Rightarrow eine der Satzformen u herleitbar ist d. h. $result$ enthält den Wert $A(m + 1, n)$.