Lokale Sprachen

Vortrag Proseminar Automatentheorie WS 15/16 von Jonas Werner

Einleitung

Worum geht es?

- *lokale Sprachen*
  - → Klasse *formaler Sprachen*
  - → Teilklasse der *regulären Sprachen*
  - → werden jeweils von *lokalen Automaten* erkannt

- Zusammenhang *lokaler Sprachen* und *lineare Ausdrücke*
  - → in Bezug auf den *Berry-Sethi-Algorithmus*

Inhalt

1. **Lokale Sprachen** - Was sind *lokale Sprachen*?

2. **Lokale Automaten** - Wie sehen *lokale Automaten* aus?

3. **Berry-Sethi-Algorithmus** - Wie hängen *lokale Sprachen* mit dem *Berry-Sethi-Algorithmus* zusammen?

1. Lokale Sprachen

1.1 Definition

Sei L eine Sprache, Σ ein Alphabet, man definiert folgende Mengen:

1. *Präfixmenge:*

$$P(L) = \{ a \in \Sigma \mid a\Sigma^* \cap L \neq \emptyset \}$$

2. *Suffixmenge:*

$$S(L) = \{ a \in \Sigma \mid \Sigma^* a \cap L \neq \emptyset \}$$

3. Menge der nicht erlaubte Subwörter der Länge 2:

$$N(L) = \Sigma^2 \setminus \{ x \in \Sigma^2 \mid \Sigma^* x \Sigma^* \cap L \neq \emptyset \}$$

Wenn L nun *lokal* sein soll, gilt:

$$L \setminus \{\varepsilon\} = (P\Sigma^* \cap \Sigma^* S) \setminus \Sigma^* N \Sigma^*$$

1. Lokale Sprachen

1.2 Beispiel 1

Sei $\Sigma = \{a, b, c\}$, die Sprache L

$L = (abc)^* = \{\varepsilon\} \cup (\{a\}\Sigma^* \cap \Sigma^*\{c\}) \setminus \Sigma^*\{aa, ac, ba, bb, cb, cc\}\Sigma^*$ ist *lokal*

Mit:

$P(L) = \{a\}$

$S(L) = \{c\}$

$N(L) = \{aa, ac, ba, bb, cb, cc\}$

1. Lokale Sprachen

1.2 Beispiel 2

Sei $\Sigma=\{a,b,c\}$, die Sprache *L(e)*, die vom regulären Ausdruck

$e=(a+bb^*a)(a+cc^*bb^*a)^*c^*$ erzeugt wird, ist *lokal*

Mit:

$$P(e)=\{a,b\}$$
$$S(e)=\{a,c\}$$
$$N(e)=\{ab,bc,ca\}$$

$$\Rightarrow L(e)=(\{a,b\}\Sigma^*\cap\Sigma^*\{a,c\})\setminus\Sigma^*\{ab,bc,ca\}\Sigma^*$$

1. Lokale Sprachen

1.3 Wörter

Um zu überprüfen, ob ein Wort *w* in *L* liegt, muss man kontrollieren, ob …

… das *erste* Zeichen in *P(L)* liegt

… das *letzte* Zeichen in *S(L)* liegt

… es *kein* Subwort der Länge zwei gibt, das in *N(L)* liegt

→ Man geht mit einem Fenster der Länge zwei durch das Wort

→ Hat man die Mengen gegeben, kann man einen DEA konstruieren

2. Lokale Automaten

2.1 Definition

Ein *lokaler Automat A* ist ein deterministischer endlicher Automat

$A = (Q, \Sigma, \delta, q_0, F)$

→ *Nicht* zwangsläufig *vollständig*

→ Für jedes $a \in \Sigma$ enthält die Menge $\{\delta(q, a) | q \in Q\}$ *höchstens* ein Element

→ Es gibt also einen *eindeutigen* Nachfolgezustand

2.2 Von lokaler Sprache zu lokalem Automaten

Sei $L = (P\Sigma^* \cap \Sigma^* S) \setminus \Sigma^* N \Sigma^*$ eine *lokale Sprache*

Für den lokalen Automaten *A*, der *L* erkennt, gilt:

- $Q = \Sigma \cup \{\varepsilon\}$
- $q_0 = \varepsilon$
- $F = S(L)$
- Für die *Übergangsfunktion* δ gilt:
  - $\delta(\varepsilon, a) = a \Leftrightarrow a \in P(L)$
  - $\delta(a, b) = b \Leftrightarrow ab \notin N(L)$

Satz 2.3:

Zu jeder lokalen Sprache gibt es einen lokalen Automaten.

2. Lokale Automaten

2.4 Beweis

⇒: **Zu zeigen: Wort, das von A akzeptiert wird, liegt in L**

Sei $w = a_1 \dots a_n$ ein Wort, das von A akzeptiert wird

→ Es gibt einen Pfad:

$$\varepsilon \xrightarrow{a_1} a_1 \xrightarrow{a_2} a_2 \dots a_{n-1} \xrightarrow{a_n} a_n$$

→ Es gibt einen Übergang von ε nach a_1

→ $a_1 \in P(L)$

→ a_n ist ein akzeptierender Zustand

→ $a_n \in S(L)$

→ Für jedes i mit $1 \leqslant i \leqslant n-1$ gibt es einen Übergang $a_i \xrightarrow{a_{i+1}} a_{i+1}$

→ $a_i a_{i+1} \notin N(L)$

⇒ $w \in L$

2. Lokale Automaten

2.4 Beweis

⇐: **Zu zeigen: Wort, das in L liegt, wird von A akzeptiert**

Sei $w = a_1 \dots a_n \in L$

→ $a_1 \in P(L)$

→ $a_n \in S(L)$

→ Für alle i mit $1 \leqslant i \leqslant n-1$ gilt: $a_i a_{i+1} \notin N(L)$

⇒ $\varepsilon \xrightarrow{a_1} a_1 \xrightarrow{a_2} a_2 \dots a_{n-1} \xrightarrow{a_n} a_n$ ist ein akzeptierender Pfad von *A*

→ *A akzeptiert w*

2.4 Beweis

Sonderfall:

$$w = \varepsilon$$

Dazu setze man

$$F = S(L) \cup \{\varepsilon\}$$

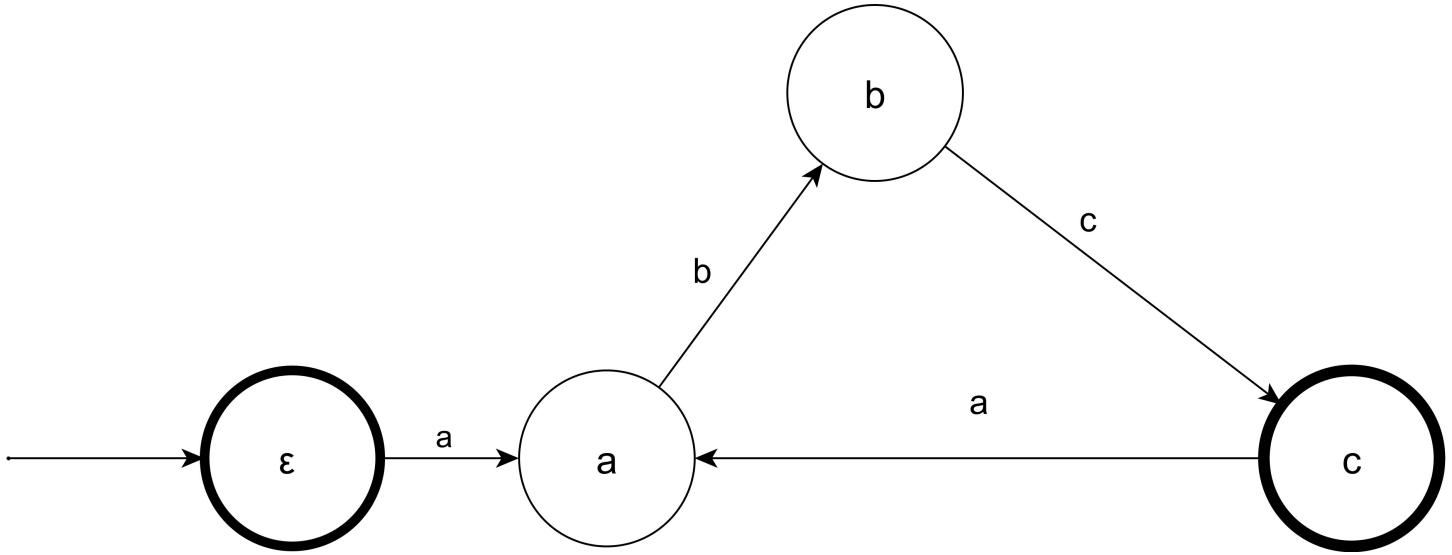
□

2.5 Beispiel 1

Betrachte wieder die *lokale Sprache*

$$L = (abc)^* = \{\varepsilon\} \cup (\{a\}\Sigma^* \cap \Sigma^*\{c\}) \setminus \Sigma^*\{aa, ac, ba, bb, cb, cc\}\Sigma^*$$

Dafür kann folgender lokaler Automat konstruiert werden:



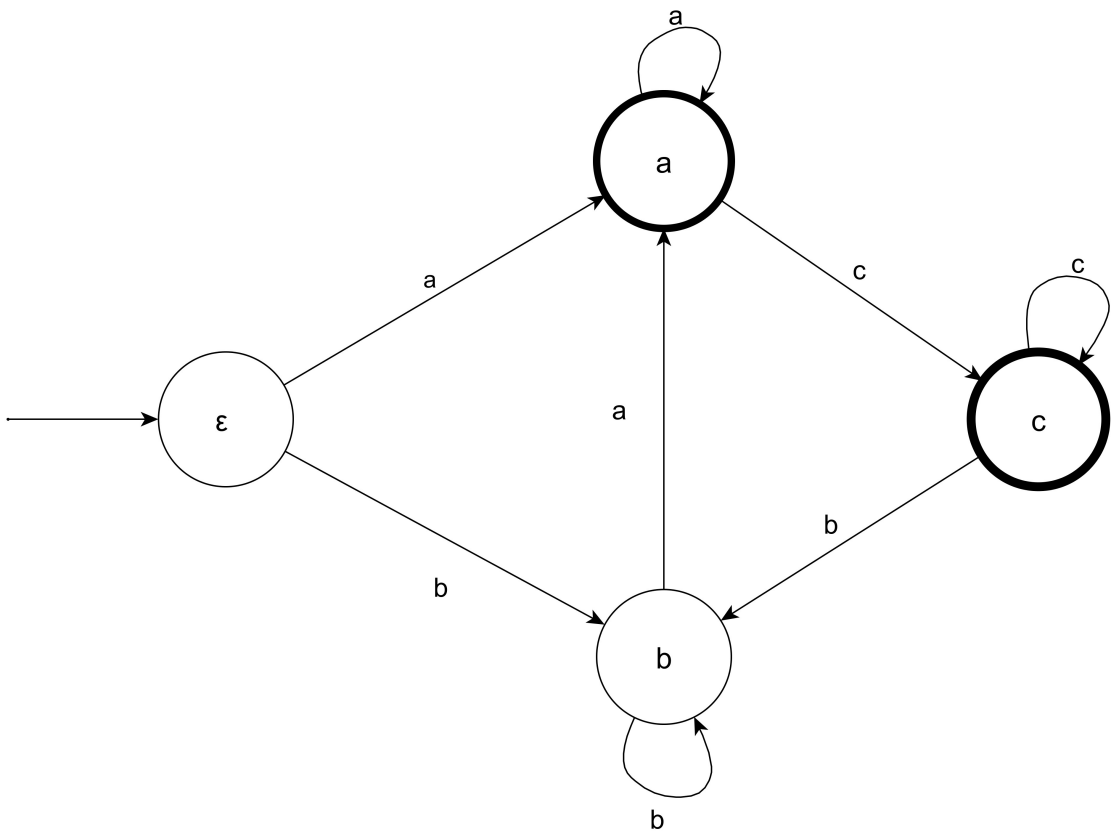
2. Lokale Automaten

2.5 Beispiel 2

Betrachte wieder die *lokale Sprache* zu $e = (a + bb^* a)(a + cc^* bb^* a)^* c^*$

$L(e) = (\{a, b\}\Sigma^* \cap \Sigma^*\{a, c\}) \setminus \Sigma^*\{ab, bc, ca\}\Sigma^*$

Dafür kann folgender lokaler Automat konstruiert werden:



2.6 Abschlusseigenschaften lokaler Sprachen

Seien Σ_1 und Σ_2 disjunkte Teilmengen des *Alphabetes* Σ,
seien $L_1 \subseteq \Sigma_1^*$, $L_2 \subseteq \Sigma_2^*$ *lokale Sprachen*

Es gilt:

→ 1: $L_1 \cup L_2$ ist *lokal*

→ 2: $L_1 L_2$ ist *lokal*

→ 3: L_1^* ist *lokal*

Kurze Wiederholung

lineare reguläre Ausdrücke

Lineare reguläre Ausdrücke sind reguläre Ausdrücke in denen jedes Zeichen *maximal* einmal vorkommt

→ Man kann aus *jedem* regulären Ausdruck einen nicht äquivalenten linearen bilden

Beispiel:

$$e = b + (a\,b\,c)^* + c \quad \overset{\text{linear}}{\rightarrow} \quad e' = a_1 + (a_2\,a_3\,a_4)^* + a_5$$

Satz 3.1:

Für jeden linearen regulären Ausdruck e,
ist die Sprache L(e) lokal

3. Berry-Sethi-Algorithmus

3.2 Beweis

Beweis über induktive Definition regulärer Ausdrücke:

Es gilt:

L(e) ist *lokal*, wenn:

→ $e = \emptyset$

→ $e = \epsilon$

→ $e = a \mid a \in \Sigma$

3. Berry-Sethi-Algorithmus

3.2 Beweis

Seien e_1 und e_2 *lineare Ausdrücke* und nehme man an $(e_1 + e_2)$ sei *linear*

- Seien Σ_1, Σ_2 die *Menge der Zeichen* die jeweils in e_1 *bzw* e_2 vorkommen
- Da $(e_1 + e_2)$ *linear* $\rightarrow$ Σ_1, Σ_2 sind *disjunkt*
- Da $L(e_1) \subseteq \Sigma_1^*$ und $L(e_2) \subseteq \Sigma_2^*$ $\overset{I.V.}{\rightarrow}$ $L(e_1 \cup e_2)$ und $L(e_1 e_2)$ sind *lokal*
- $L(e_1)^*$ nach *Abschlusseigenschaften* lokal

3. Berry-Sethi-Algorithmus

3.2 Beweis

***Achtung*:**

Jeder lineare Ausdruck erzeugt eine lokale Sprache

→ Umkehrung gilt nicht!

Gegenbeispiel:

$(ab)^* a$ erzeugt eine *lokale Sprache*, ist aber nicht linear

3. Berry-Sethi-Algorithmus

3.3 Wofür ist das nützlich?

Man kann aus jedem *regulären Ausdruck* einen *linearen* bilden

→ Dieser erzeugt eine *lokale Sprache*

→ Welche von einem *lokalen Automaten* erkannt wird

Man hat für jeden *regulären Ausdruck* einen deterministischen endlichen Automaten

→ Was das Ziel des *Berry-Sethi-Algorithmus* ist

Zusammenfassung

Lokale Sprachen sind eine Klasse formaler Sprachen

→ Wörter einer *lokalen Sprache* charakterisieren sich dadurch, dass…

… das erste und letzte Zeichen in der definierten *Präfix- bzw Suffixmenge* liegt

… alle Subwörter der Länge zwei nicht in *N(L)* liegen

Lokale Automaten sind deterministische endliche Automaten, die *lokale Sprachen akzeptieren*

→ Für *jede* lokale Sprache existiert ein lokaler Automat

→ Können leicht unter Verwendung der *Präfix-*, *Suffix- und N(L) konstruiert* werden

Zusammenfassung

Jeder *lineare reguläre Ausdruck* erzeugt eine *lokale Sprache*

→ Diese wird wiederum von einem *lokalen Automaten* erkannt

→ Da man jeden *regulären Ausdruck* in einen *linearen* umformen kann, hat man also die Möglichkeit aus *jedem regulären Ausdruck* einen Automaten zu konstruieren

→ Was das *Kernziel* des *Berry-Sethi-Algorithmus* ist

Quellen

„Local languages and the Berry-Sethi algorithm“ by Jean Berstel and Jean-Éric Pin