Technische Universität Braunschweig

Theoretische Informatik 2

Arne Schmidt

Fragen von Kuroda

2. Sind die NLBA-Sprachen unter Komplement abgeschlossen?

Kapitel 2 – Satz von Immerman & Szelepcsényi

Nächstes Kapitel

Theorem 2.1 (Immerman & Szelepcsényi, 1988 &1987)
Wenn eine Sprache $\mathcal{L} \subseteq \Sigma^*$ kontextsensitiv ist, dann ist auch ihr Komplement $\overline{\mathcal{L}}$ kontextsensitiv.

Beweisidee:
Sei $\mathcal{L} = \mathcal{L}(G)$ für eine Typ-1-Grammatik $G = (N, \Sigma, P, S)$.

Konstruiere einen NLBA, welcher ein Eingabewort $w \in \Sigma^*$ akzeptiert, falls es **keine Ableitung** $S \Rightarrow_G^* w$ in der Grammatik G gibt.

Wir wollen also prüfen, ob vom Wurzelknoten des **Ableitungsgraphen** das Wort w erreicht werden kann.

Ableitungsgraph

Definition 2.2
Der **Ableitungsgraph** $\text{Graph}_{|w|}$ zu einer Grammatik $G = (N, \Sigma, P, S)$ und einem Wort $w \in \Sigma^*$ hat

- Als Knotenmenge die Menge der Satzformen der Länge $\leq |w|$ und
- Die Kanten sind durch die Ableitungsrelation $\Rightarrow_G$ gegeben.

Formal:

$$\text{Graph}_{|w|} = \big(\, (\Sigma \cup N)^{\leq |w|}, \{(a, b) |\ a \Rightarrow_G b\}\big)$$

Damit können wir folgende Aussage direkt festhalten:
Es gibt genau dann **keine** Ableitung $S \Rightarrow_G^* w$, wenn es keinen Pfad von S zu w in $\text{Graph}_{|w|}$ gibt.

Was ist zu tun?

Entwirf einen nicht-deterministischen Algorithmus (bzw. NTM)

- Gegeben: Ein Graph G (bei uns $\text{Graph}_{|w|}$) und zwei Knoten s und t (bei uns Startsymbol S und w)
- Entscheide, ob es keinen Pfad von s nach t gibt.

Benötige dabei nur logarithmisch viel Platz in der Größe von G.

- Dann ist die NTM auch ein NLBA, denn
- $O(\log G) = O(|w|)$, also linear in der Länge von w.

Idee 1:

$$\text{Graph}_{|w|} = \left((\Sigma \cup N)^{\leq |w|}, \{(a,b) \mid a \Rightarrow_G b\} \right)$$

Wir wollen einen NLBA konstruieren, der Graph ist aber exponentiell groß in $|w|$...

Aber: Jeder Pfad hat maximal Länge $|w|$.

Können wir einen Pfad raten und prüfen, ob dieser nicht gültig ist?

Das Problem: **Jede** Auswahl von geratenen Pfaden darf nicht gültig sein!

Idee 2:

$$\text{Graph}_{|w|} = \left((\Sigma \cup N)^{\leq |w|}, \{(a,b) \mid a \Rightarrow_G b\} \right)$$

Berechne alle von S aus erreichbaren Knoten, ...

Prüfe, ob t darunter ist.

Geht nicht mit einem NLBA!

Reicht das Zählen von erreichbaren Knoten?

UNREACH(G,s,t)

Wir nehmen zunächst an, wir kennen die Anzahl N an erreichbaren Knoten.

Wie viel Platz benötigt dieser Algorithmus?

- Zahl N: $\log N \in O(|w|)$
- Pfad raten? Knotenweise +Länge des Pfades: $\sim O(\log |w|)$

```
count := 0
for Knoten v do
    Rate, ob v von s aus erreichbar ist
    if Ja then
        Rate einen Pfad von s nach v der Länge ≤ n
        if Falls das geratene kein gültiger Pfad nach v ist then
            return false            // Erreichbarkeit oder Pfad falsch geraten
        end if
        if v = t then
            return false            // t ist erreichbar
        end if
        count++                     // Erreichbarer Knoten gefunden
    end if
end for
if count ≠ N then
    return false                    // für mindestens einen Knoten falsch geraten
else
    return true                     // immer richtig geraten und t wirklich unerreichbar
end if
```

UNREACH(G,s,t)

Lemma 2.4
Sei N initialisiert mit der Anzahl der von s aus erreichbaren Knoten. Es gibt eine Berechnung zum nicht-deterministischen Algorithmus, die $\textit{UNREACH}(G, s, t)$ true zurück gibt, genau dann wenn es keinen Pfad von s nach t gibt.

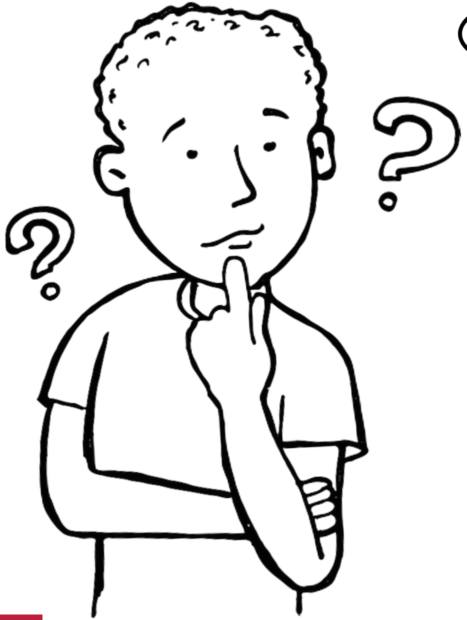
Beweis: Algorithmus gibt nur dann true zurück, wenn wir genau die erreichbaren Knoten als erreichbar raten.

- Knoten unerreichbar: Pfad Verifikation schlägt immer fehl. (Zeile 4/7)
- Zu wenig Knoten als erreichbar geraten: Überprüfung der Anzahl scheitert (Zeile 15)

Ist t nicht erreichbar, ergibt genau diese Berechnung true.

Gibt es eine Berechnung, die true zurückgibt, kann t nicht erreichbar sein:
Alle erreichbaren Knoten sind identifiziert und t war nicht darunter.
Sonst hätten wir false zurückgegeben (Zeile 9/10).

Zählen von erreichbaren Knoten



Wie können wir die Zahl N bestimmen?

Können wir die Idee von unreach nutzen und die Zahl induktiv bestimmen?

Schauen wir uns das an der Tafel an!

Nächstes Mal