Adrian Kramkowski, Abdelhadi Fares, Yousef Al Sahli und Abdelraoof Sahli

Modul: Compilerbau

Aufgabe A2.4 – Kontextfreie Grammatik & PDA

Gegeben: Sprache

```
L = { a^i b^j c^k | i = j ∨ j = k }
```

Gesucht:

- Eine kontextfreie Grammatik (CFG) für L
- Nachweis der Mehrdeutigkeit
- Ein Pushdown-Automat (PDA), der dieselbe Sprache akzeptiert

1. Kontextfreie Grammatik

Die Sprache besteht aus Wörtern, bei denen entweder die Anzahl der `a` und `b` gleich ist (`i=j`), oder die Anzahl der `b` und `c` gleich ist (`j=k`).

```
G = ( {S, S1, S2}, {a, b, c}, P, S )

P = {
  S  → S1 | S2
  S1 → a S1 b | ε
  S2 → b S2 c | ε
}
```

Erklärung:

- `S1` erzeugt Wörter der Form `aⁿbⁿ` (gleiche Anzahl a und b, beliebig viele c's können später folgen).
- `S2` erzeugt Wörter der Form `bⁿcⁿ` (gleiche Anzahl b und c, beliebig viele a's können davor stehen).

- `S → S1 | S2` wählt zwischen beiden Bedingungen („oder“ in der Definition).

2. Beweis der Mehrdeutigkeit

Ja, die Grammatik ist mehrdeutig.

Ein Wort wie `abc` kann auf zwei verschiedene Arten erzeugt werden:

Ableitung 1 über S1:
S → S1
→ a S1 b
→ a ε b
→ a b
→ a b c (weitere c durch nachfolgende c's erlaubt)

Ableitung 2 über S2:
S → S2
→ b S2 c
→ b ε c
→ b c
→ a b c (vorangestelltes a erlaubt)

Da das Wort `abc` sowohl über `S1` als auch über `S2` ableitbar ist, gibt es mindestens zwei unterschiedliche Ableitungsbäume → **mehrdeutig.**

> **Erinnerung:** Eine Grammatik ist mehrdeutig, wenn ein Wort mehrere gültige Ableitungsbäume besitzt (siehe Vorlesung, Folie 14).

3. Pushdown-Automat (PDA)

Da die Sprache `L` die Vereinigung zweier kontextfreier Sprachen ist ($L_1 = \{a^n b^n c^*\}$ und $L_2 = \{a^* b^n c^n\}$), genügt ein nichtdeterministischer PDA, der zwischen beiden Teilsprachen wählen kann.

P = (Q, Σ, Γ, δ, q0, ⊥, F)

Q = { q0, q1, q2, q3, qf }

$\Sigma = \{ a, b, c \}$
$\Gamma = \{ A, B, \perp \}$
$F = \{ qf \}$

Übergänge δ:

1. q0 liest 'a' → pusht A → bleibt in q0 (Zählen von a)
   $\delta(q0, a, \perp) = (q0, A\perp)$
   $\delta(q0, a, A) = (q0, AA)$

2. q0 liest 'b' → wechselt zu q1 (Abbau für a=b)
   $\delta(q0, b, A) = (q1, \varepsilon)$

3. q1 liest 'b' → bleibt in q1, Stack bleibt gleich
   $\delta(q1, b, A) = (q1, \varepsilon)$

4. q1 liest 'c' → wechselt zu q2 (b=c-Teil)
   $\delta(q1, c, \varepsilon) = (q2, \varepsilon)$

5. q2 liest 'c' → bleibt in q2
   $\delta(q2, c, \varepsilon) = (q2, \varepsilon)$

6. q2 ε, Stack leer → akzeptiert
   $\delta(q2, \varepsilon, \perp) = (qf, \varepsilon)$

Der Automat wählt zu Beginn, ob er den `aⁿbⁿ` - oder den `bⁿcⁿ` -Teil akzeptieren möchte und verarbeitet entsprechend das Eingabewort. Dies entspricht der Unterscheidung zwischen `S1` und `S2` in der Grammatik.

4. Fazit

- Die Grammatik erzeugt alle Wörter mit `i=j` oder `j=k` .
- Durch die Regel `S → S1 | S2` entsteht Mehrdeutigkeit (ein Wort kann über beide Wege abgeleitet werden).
- Ein PDA kann die Sprache akzeptieren, indem er nichtdeterministisch entscheidet, ob er den linken oder rechten Teil der Grammatik verarbeitet.

Ergebnis: Die Grammatik ist kontextfrei und mehrdeutig, die Sprache ist durch einen nichtdeterministischen PDA akzeptierbar.