Kellerautomat

Grober Ablauf:

1. Erstes c lesen.
2. Alle a lesen und merken oder zweites c lesen.
3. Alle b lesen und mit Anzahl a vergleichen.
4. Zweites c lesen

$$
\begin{aligned}
\Sigma &= \{a, b, c\} \\
\Gamma &= \{\#, A, C\} \\
Z &= \{z_0, z_1\}
\end{aligned}
$$

$$
\begin{array}{lll}
1) & \delta(z_0, c, \#) = & \{(z_0, C)\} \\
2) & \delta(z_0, c, C) = & \{(z_0, \varepsilon)\} \\
3) & \delta(z_0, a, C) = & \{(z_0, AC)\} \\
4) & \delta(z_0, a, A) = & \{(z_0, AA)\} \\
5) & \delta(z_0, b, A) = & \{(z_1, \varepsilon)\} \\
6) & \delta(z_1, b, A) = & \{(z_1, \varepsilon)\} \\
7) & \delta(z_1, c, C) = & \{(z_1, \varepsilon)\}
\end{array}
$$

Die Zustände z_0 und z_1 eliminieren.

$$
\begin{array}{llcl}
1) & \delta(\cdot, c, (z_0, S, z_0)) & = & \{(\cdot, (z_0, C, z_0))\} \\
 & \delta(\cdot, c, (z_0, S, z_1)) & = & \{(\cdot, (z_0, C, z_1))\} \\
 \\
2) & \delta(\cdot, c, (z_0, C, z_0)) & = & \{(\cdot, \varepsilon)\} \\
 \\
3) & \delta(\cdot, a, (z_0, C, z_0)) & = & \{(\cdot, (z_0, A, z_0)(z_0, C, z_0)), (\cdot, (z_0, A, z_1)(z_1, C, z_0))\} \\
 & \delta(\cdot, a, (z_0, C, z_1)) & = & \{(\cdot, (z_0, A, z_0)(z_0, C, z_1)), (\cdot, (z_0, A, z_1)(z_1, C, z_1))\} \\
 \\
4) & \delta(\cdot, a, (z_0, A, z_0)) & = & \{(\cdot, (z_0, A, z_0)(z_0, A, z_0)), (\cdot, (z_0, A, z_1)(z_1, A, z_0))\} \\
 & \delta(\cdot, a, (z_0, A, z_1)) & = & \{(\cdot, (z_0, A, z_0)(z_0, A, z_1)), (\cdot, (z_0, A, z_1)(z_1, A, z_1))\} \\
 \\
5) & \delta(\cdot, b, (z_0, A, z_1)) & = & \{(\cdot, \varepsilon)\} \\
 \\
6) & \delta(\cdot, b, (z_1, A, z_1)) & = & \{(\cdot, \varepsilon)\} \\
 \\
7) & \delta(\cdot, c, (z_1, C, z_1)) & = & \{(\cdot, \varepsilon)\} \\
 \\
 & \text{Kellerstartsymbol} & & \\
0) & \delta(\cdot, \varepsilon, \#) & = & \{(\cdot, (z_0, S, z_0)), (\cdot, (z_0, S, z_1))\}
\end{array}
$$

Die Grammatik zu diesem Automaten sieht dann so aus.

$$
\begin{array}{lrcl}
0) & S & \to & (z_0, S, z_0) \mid (z_0, S, z_1) \\
\\
1) & (z_0, S, z_0) & \to & c(z_0, C, z_0) \\
 & (z_0, S, z_1) & \to & c(z_0, C, z_1) \\
\\
2) & (z_0, C, z_0) & \to & c \\
\\
3) & (z_0, C, z_0) & \to & a(z_0, A, z_0)(z_0, C, z_0) \mid a(z_0, A, z_1)(z_1, C, z_0) \\
 & (z_0, C, z_1) & \to & a(z_0, A, z_0)(z_0, C, z_1) \mid a(z_0, A, z_1)(z_1, C, z_1) \\
\\
4) & (z_0, A, z_0) & \to & a(z_0, A, z_0)(z_0, A, z_0) \mid a(z_0, A, z_1)(z_1, A, z_0) \\
 & (z_0, A, z_1) & \to & a(z_0, A, z_0)(z_0, A, z_1) \mid a(z_0, A, z_1)(z_1, A, z_1) \\
\\
5) & (z_0, A, z_1) & \to & b \\
\\
6) & (z_1, A, z_1) & \to & b \\
\\
7) & (z_1, C, z_1) & \to & c
\end{array}
$$

Eventuell lassen sich von hier aus noch Regeln löschen, die keine (Teil-)Worte erzeugen können. Wie?

Noch zwei Beispiele: Die Ableitung

$$S \to (z_0, S, z_0) \to c(z_0, C, z_0) \to ca(z_0, A, z_1)(z_1, C, z_0) \to cab(z_1, C, z_0)$$

führt zu keinem Wort und entspricht der *nichtakzeptierenden* Rechnung

$$\# \xrightarrow{0} (z_0, S, z_0) \xrightarrow{1} (z_0, C, z_0) \xrightarrow{3} (z_0, A, z_1)(z_1, C, z_0) \xrightarrow{5} (z_1, C, z_0)$$

auf dem Wort *cabc*. Aber: *cabc* ist ein gültiges Wort.
Rechnung:

$$\# \xrightarrow{0} (z_0, S, z_1) \xrightarrow{1} (z_0, C, z_1) \xrightarrow{3} (z_0, A, z_1)(z_1, C, z_1) \xrightarrow{5} (z_1, C, z_1) \xrightarrow{7} \varepsilon$$

Ableitung:

$$S \to (z_0, S, z_1) \to c(z_0, C, z_1) \to ca(z_0, A, z_1)(z_1, C, z_1) \to cab(z_1, C, z_1) \to cabc$$

Grammatik

$$
\begin{array}{rcl}
S & \rightarrow & cc \mid cTc \\
T & \rightarrow & ab \mid aTb
\end{array}
$$

Grammatik in Chomsky-Normalform umwandeln.

$$
\begin{array}{rcl}
S & \rightarrow & CC \mid CD \\
A & \rightarrow & a \\
B & \rightarrow & b \\
C & \rightarrow & c \\
D & \rightarrow & TC \\
E & \rightarrow & TB \\
T & \rightarrow & AB \mid AE
\end{array}
$$