

Grundlagen der Informatik

Prof. Dr. Stefan Enderle
NTA Isny

1. Automaten und Sprachen

1.1 Endlicher Automat

- Einen endlichen Automaten stellen wir uns als Black-Box vor, die sich aufgrund einer Folge von Eingaben in einem bestimmten Zustand befindet.

Zustandsübergänge

- In Abhängigkeit von seinem aktuellen Zustand und einer Eingabe geht der EA in einen Folgezustand über.

Endzustand

- Kommt der EA in einen ausgezeichneten Zustand (Endzustand), dann handelt es sich um eine *akzeptierte* Eingabe.

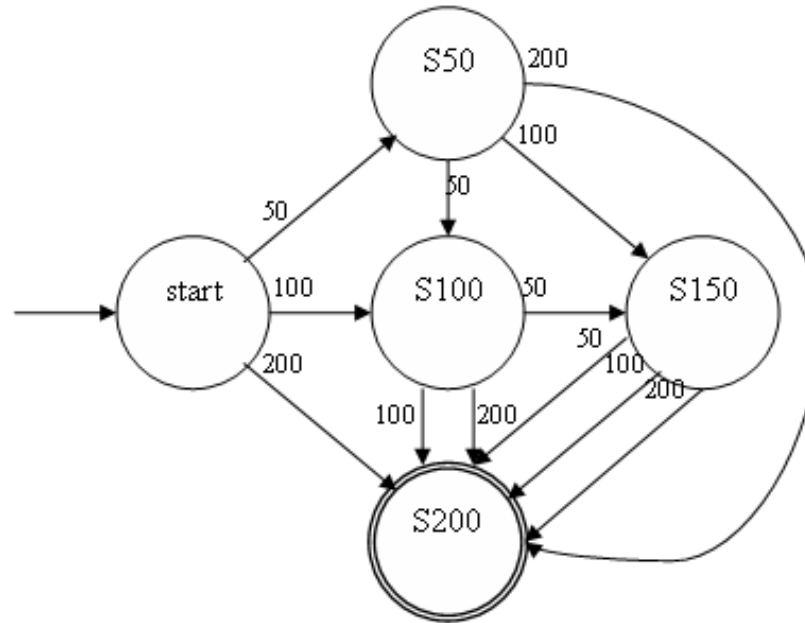
Beispiel

- Der Automat „Eintritt“ soll den Eintritt in einem Schwimmbad kontrollieren.
- Aufgabe:
 - Der Eintritt kostet EUR 2,-
 - Er soll 50ct, 1,- und 2,- EUR Münzen akzeptieren
 - Sind mindestens EUR 2,- eingeworfen, öffnet der Automat die Schranke

Realisierung

- Automat ist anfangs in einem Startzustand
- Nach Einwurf geht er in Zwischenzustände über
- Nach Einwurf von mind. 2 EUR geht er in Endzustand
- → Automat entwerfen!

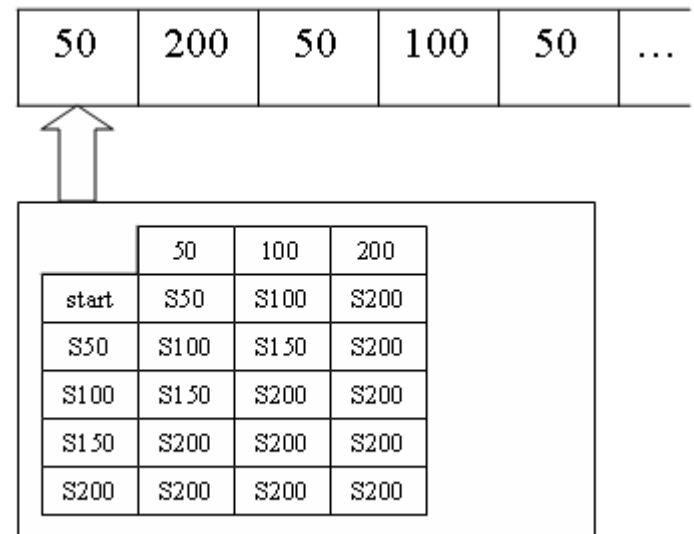
Zustandsdiagramm



- Zustände = Kreise
- Zustandsübergänge = Pfeile
- Startzustand = Kreis mit Eingangspfeil
- Endzustand = Doppelkreis

Abstraktes Rechnermodell

- Zwei Speicher:
 - Eingabeband (mit Folge von Geldstücken) wird von links nach rechts gelesen
 - Zustandspeicher (eine Speicherzelle) mit aktuellem Zustand
- Zustandsübergänge sind in einer Zustands-tabelle gespeichert



Ablauf

- Ist der aktuelle Zustand s ,
- und der Lesekopf über Zeichen a ,
- dann geht der Automat in den Folgezustand, der in Zeile s und Spalte a steht.
- Außerdem wird der Lesekopf um eine Stelle nach rechts bewegt.
- (Beispiel)

Akzeptierte Folgen

- Ist die Eingabefolge komplett gelesen,
- und beinhaltet der Zustandspeicher einen Endzustand,
- dann ist die Zeichenfolge *akzeptiert*.

„Sprache“ des Automaten

- Definition:
Die Menge aller von einem Automaten akzeptierten Eingaben heißt die „Sprache“ des Automaten.
- Beispiel: Sprache des Eintrittsautomaten:
 - 50,50,50,50 50,50,50,100 50,50,50,200
 - 50,50,100 50,50,200
 - 50,100,50 50,100,100 50,100,200 50,200
 - 100,50,50 100,50,100 100,50,200
 - 100,100 100,200
 - 200

1.2 Alphabete, Wörter, Sprachen

Symbole

- Die bisherigen Eingaben (50, 100, 200) waren atomare *Symbole* für Geldstücke.
- Der Lesekopf liest das Eingabeband *symbolweise*.
- Die Eingabesymbole zusammen ergeben ein *Eingabewort*.
- Statt *Symbole* sagt man auch *Buchstaben*.

Alphabete

- Die Menge der möglichen Eingabesymbole bilden das *Eingabealphabet*.
- Formal wird meist Σ verwendet:
also: $\Sigma = \{50, 100, 200\}$
- Allgemein verwenden wir die Symbole a_1 bis a_n : $\Sigma = \{a_1, a_2, \dots, a_n\}$

Wörter

- Die endlich langen Zeichenfolgen, die über einem Alphabet Σ gebildet werden können, heißen *Wörter über Σ* .
- Wörter entstehen, indem Symbole oder bereits erzeugte Wörter aneinandergereiht werden.

Operationen auf Wörtern

- Konkatenation: vw
- i -te Potenz: $w^0 = \varepsilon$ $w^i = ww^{i-1}$
- Reverse: $w^R = w$ rückwärts

Menge aller Wörter

- Σ^* , die Menge aller Wörter über Σ ist folgendermaßen definiert:
 - Jeder Buchstabe $a \in \Sigma$ ist auch ein Wort über Σ , also $a \in \Sigma^*$
 - Sind v und $w \in \Sigma^*$, so ist auch $vw \in \Sigma^*$
 - ε is das *leere Wort* über jedem Alphabet Σ .
Es gilt $w\varepsilon = \varepsilon w = w$ für alle $w \in \Sigma^*$

Beispiel

- Sei $\Sigma = \{a,b\}$
- Dann ist $\Sigma^* =$
 $\{\varepsilon, a, b, aa, ab, ba, bb, aaa, aab, aba, abb, baa, b$
 $ab, bba, bbb, aaaa \dots\}$

Beispiel

- Beim Eintrittsautomaten ist $\Sigma = \{50, 100, 200\}$
- Σ^* ist gleich der Menge aller Geldstückfolgen, die mit 50ct, 1 EUR und 2 EUR Münzen gebildet werden kann.

Folgerung

- Ist ein Alphabet Σ leer, so ist $\Sigma^* = \{\varepsilon\}$
- Ist ein Alphabet nicht leer, dann besitzt Σ^* unendlich viele Wörter.

Länge von Wörtern

- $|w|$ bezeichnet die Länge des Wortes w
- Beispiel:
 - $|a|=1$
 - $|\varepsilon|=0$
 - $|abc|=3$

Lexikografische Ordnung

- Für einzelne Buchstaben v_1 und v_2 gilt $v_1 < v_2$, wenn v_1 vor v_2 im Alphabet vorkommt.
- Wir definieren eine lexikografische Ordnung „ $<$ “ auf Wörtern wie folgt:
 - Sei $v = v_1, \dots, v_i, \dots, v_m$ und $w = w_1, \dots, w_j, \dots, w_n$
 - Dann ist $v < w$ genau dann, wenn
 1. $m < n$, oder
 2. $m = n$ und es gibt ein k , so dass $v_1, \dots, v_k = w_1, \dots, w_k$ und $v_{k+1} < w_{k+1}$

Beispiel

- Sei $\Sigma = \{a, b\}$ (mit $a < b$)
- dann gilt $bb < aaa$ (wegen 1.)
- und es gilt $baab < babb$ (wegen 2.)

D.h., Wörter werden erst ihrer Länge nach und bei gleicher Länge nach dem ersten unterschiedlichen Buchstaben geordnet.

Sprache

- Definition:
Eine Sprache L über einem Alphabet Σ ist eine Menge von Wörtern über Σ .
- D.h. L ist eine Teilmenge von Σ^* .

Beispiel

- Alphabet $\Sigma = \{0, 1\}$
- Menge aller Wörter:
 $\Sigma^* = \{0, 1, 00, 01, 10, 11, 000, 001, 010, 011, \dots\}$
- Sprache $L = \{0, 1, 10, 11, 100, 101, 110, \dots\}$
= Menge der natürlichen Zahlen in
Binärdarstellung

Beispiel

- Alphabet $\Sigma = \{a, +, -, *, /, (,)\}$
- Menge aller Wörter:
 $\Sigma^* = \{a, a+, -+-aa,)^*(a-, ******, \dots)\}$
- Sprache $L = \{\varepsilon, a, a+a, (a+a), (a+(a^*a)), \dots\}$
= Menge der korrekt geklammerten
Ausdrücke

Operationen auf Sprachen

- Konkatenation: $LM = \{vw \mid v \in L, w \in M\}$
- I-te Potenz: $L^0 = \{\varepsilon\}$ $L^i = LL^{i-1}$
- Reverse: $L^R = \{w^R : w \in L\}$

Kleene Hülle

- $L^* = L^0 \cup L^1 \cup L^2 \dots$

- Variante:

$$L^+ = L^1 \cup L^2 \dots$$

Definition einer Sprache

- Frage: Wie kann eine Sprache (mit unendlich vielen Wörtern) sinnvoll definiert werden?
Sinnvoll heißt, durch eine endliche Menge von Zeichen.
- Beispielsweise durch einen
„regulären Ausdruck“

1.3 Reguläre Ausdrücke und Grammatiken

Regulärer Ausdruck

- Die Menge $\text{Reg}(\Sigma)$ der regulären Ausdrücke (über dem Alphabet Σ) ist definiert durch:
 1. \emptyset ist ein regulärer Ausdruck
 2. Für jedes $a \in \Sigma$ ist a ein regulärer Ausdruck
 3. Sind r und s reguläre Ausdrücke, so auch
 - $(r|s)$ (Vereinigung)
 - (rs) (Konkatenation)
 - (r^*) (Kleene Stern)
- Wenn Klammern weggelassen werden gilt $*$ vor $|$ und $|$ vor $|$

Semantik regulärer Ausdrücke

- Ein regulärer Ausdruck r stellt eine Sprache $L(r)$ wie folgt dar:
 - $L(\emptyset) := \{ \}$
 - $L(a) := \{a\}$ für alle $a \in \Sigma$
 - $L(r|s) := L(r) \cup L(s)$
 - $L(rs) := L(r)L(s)$
 - $L(r^*) := L(r)^* = L^0 \cup L^1 \cup L^2 \dots$

Beispiel

- Sei $\Sigma = \{a, b\}$
- Welche Sprachen werden durch folgende regulären Ausdrücke dargestellt?
 - aa
 - $(a|b)^*$
 - $aa^* | bb^*$

Anwendungen

- Bankautomat: Erkenner für Währungsangaben in Eur, USD oder Pfund:
 - Schema: *währung vorzeichen betrag*
 - Alphabet: $\Sigma = \{\text{€}, \text{\$}, \text{\pounds}, +, -, ., 0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$

Anwendungen

- Definition von Teilausdrücken:
 - *einzahlung* = *währung vorzeichen betrag*
 - *währung* = $(\text{€}|\text{\$}|\text{£})$
 - *vorzeichen* = $(+|-|\varepsilon)$
 - *betrag* = *ganz* (ε | . *dez*)
 - *ganz* = $(0 | (1|2|\dots|9) (0|1|2|\dots|9)^*)$
 - *dez* = $(0|1|2|\dots|9) (0|1|2|\dots|9)$

Anwendungen

- Gesamter Ausdruck:

$$\begin{aligned} - \textit{einzahlung} &= (\epsilon | \$ | \pounds) (+ | - | \epsilon) \\ &\quad (0 | (1 | 2 | \dots | 9) (0 | 1 | 2 | \dots | 9)^*) \\ &\quad (\epsilon | . (0 | 1 | 2 | \dots | 9) (0 | 1 | 2 | \dots | 9)) \end{aligned}$$

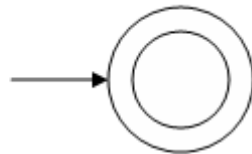
Äquivalenz von reg. Ausdrücken und endlichen Automaten

- Satz: Zu jedem regulären Ausdruck a über Σ existiert ein endlicher Automaten A über Σ , so dass $L(A)=L(a)$.

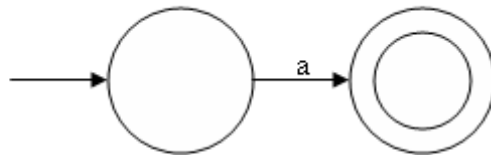
Konstruktion

- Konstruktion eines endlichen Automaten A aus einem regulärer Ausdruck:

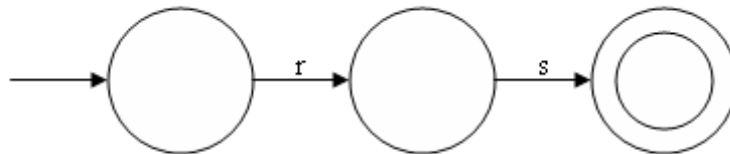
- $L(0)$



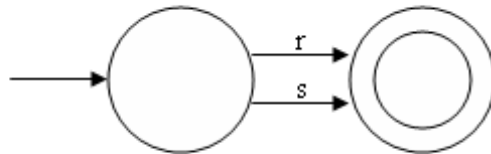
- $L(a)$



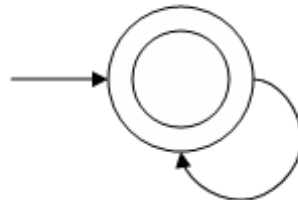
- $L(rs)$



- $L(r|s)$



- $L(r^*)$

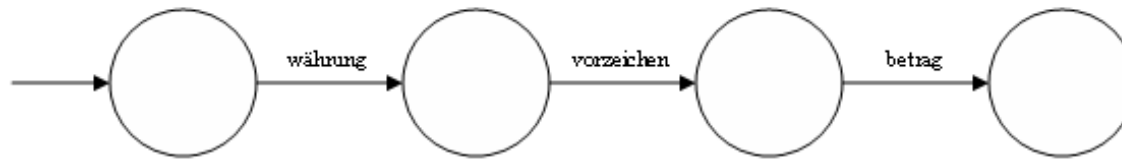


Beispiel

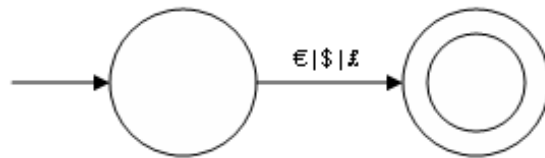
- Aufgabe:
Endliche Automatenen für:
 - *einzahlung* = *währung vorzeichen betrag*
 - *währung* = $(\epsilon|\$|\pounds)$
 - *vorzeichen* = $(+|-|\epsilon)$
 - *betrag* = *ganz* ($\epsilon|.$ *dez*)
 - *ganz* = $(0 | (1|2|\dots|9) (0|1|2|\dots|9)^*)$
 - *dez* = $(0|1|2|\dots|9) (0|1|2|\dots|9)$

Beispiel

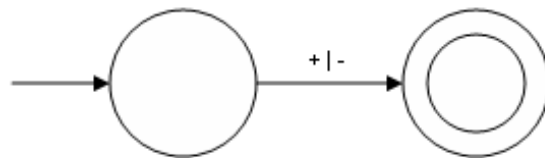
- einzahlung:



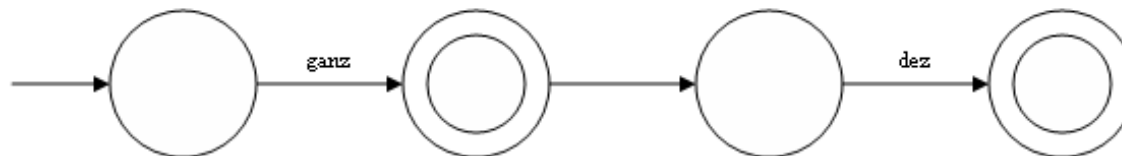
- währung:



- vorzeichen:

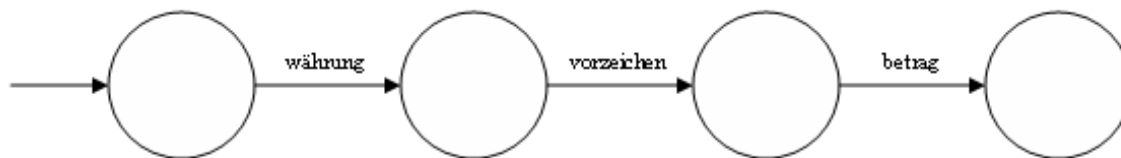


- betrag:

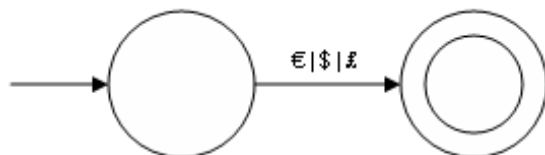


Beispiel

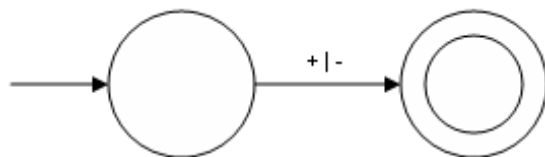
- einzahlung:



- währung:



- vorzeichen:



- betrag:

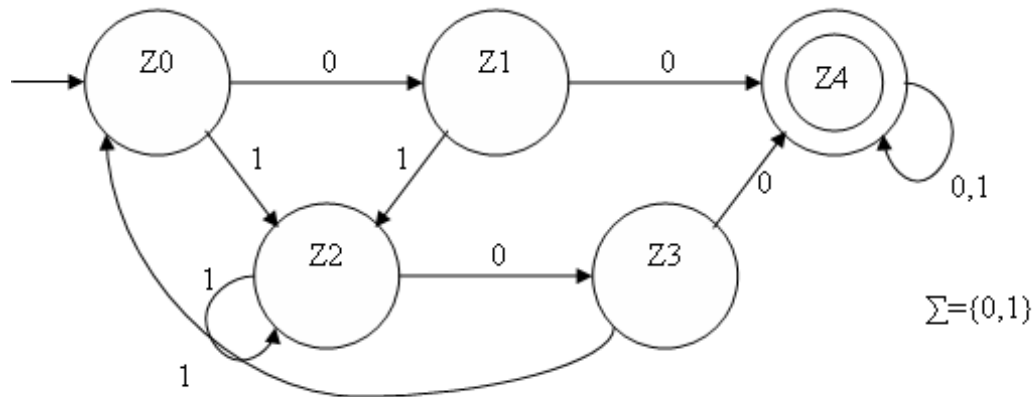


Minimalautomat

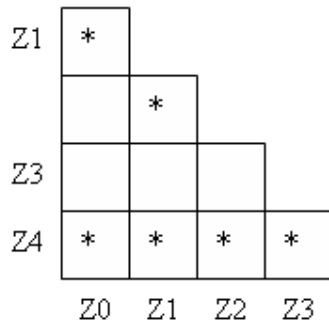
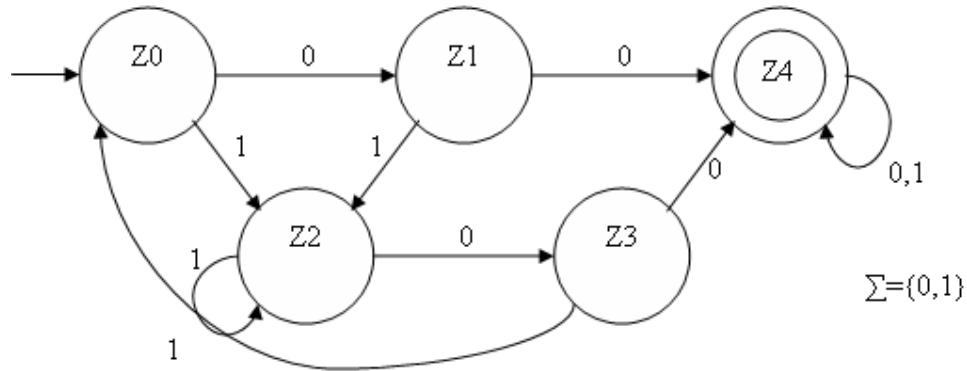
- Eingabe: Endlicher Automat M
- Ausgabe: Minimaler endlicher Automat
- Algorithmus:
 - 1) Stelle eine Tabelle aller Zustandspaare (z, z') von M auf, mit z ungleich z' .
 - 2) Markiere alle Paare mit genau einem Endzustand
 - 3) Für jedes unmarkierte Paar (z, z') , teste für jedes Symbol a : $(z \rightarrow a, z' \rightarrow a)$ markiert?
 - Ja, dann markiere auch (z, z')
 - 4) Wiederhole 3) bis sich keine Änderung mehr ergibt
 - 5) Verschmelze alle unmarkierten Zustandspaare (z, z') zu einem neuen Zustand

Beispiel

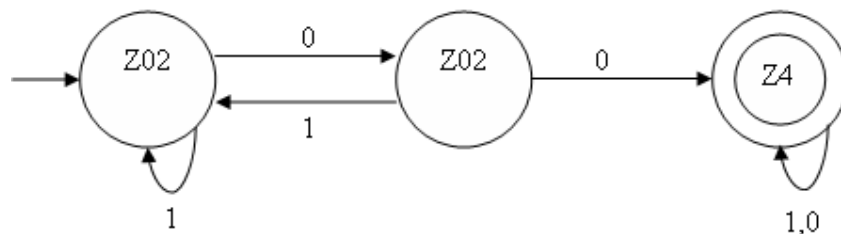
- Aufgabe:
Minimieren Sie diesen Graphen:



Beispiel



(Z0, Z1): 0: (Z1, Z4)
 1:
 (Z0, Z2): 0: (Z1, Z3)
 1: (Z2, Z2)
 (Z1, Z2): 0: (Z4, Z3)
 1:
 (Z2, Z3): 0: (Z1, Z4)
 (Z1, Z3): 0: (Z4, Z4)
 1: (Z2, Z0)
 (Z2, Z3): 0: (Z3, Z4)



Grenzen von regulären Sprachen

- Ein endlicher Automat kann sich nur endlich viele Zustände „merken“. D.h., Sprachen, bei denen man zählen muss und die Obergrenze nicht bekannt ist, lassen sich nicht darstellen.
- Beispiele:
 - $L = \{a^k b^k \mid k > 0\}$
 - $L = \{w \in \{a, b\}^* \mid |w_a| = |w_b|\}$ (gleichviele a's wie b's)

1.4 Syntaxdiagramme

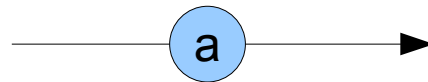
Graphische Darstellung

- Mit einem Syntaxdiagramm lässt sich ein regulärer Ausdruck graphisch darstellen:

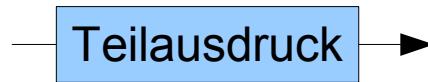
- \emptyset



- $a \in \Sigma$



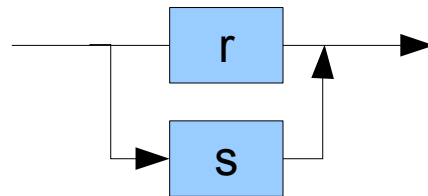
- Teilausdruck



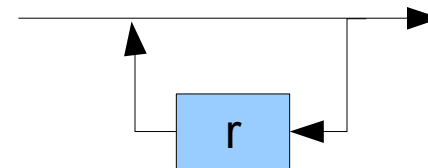
- rs



- $r|s$



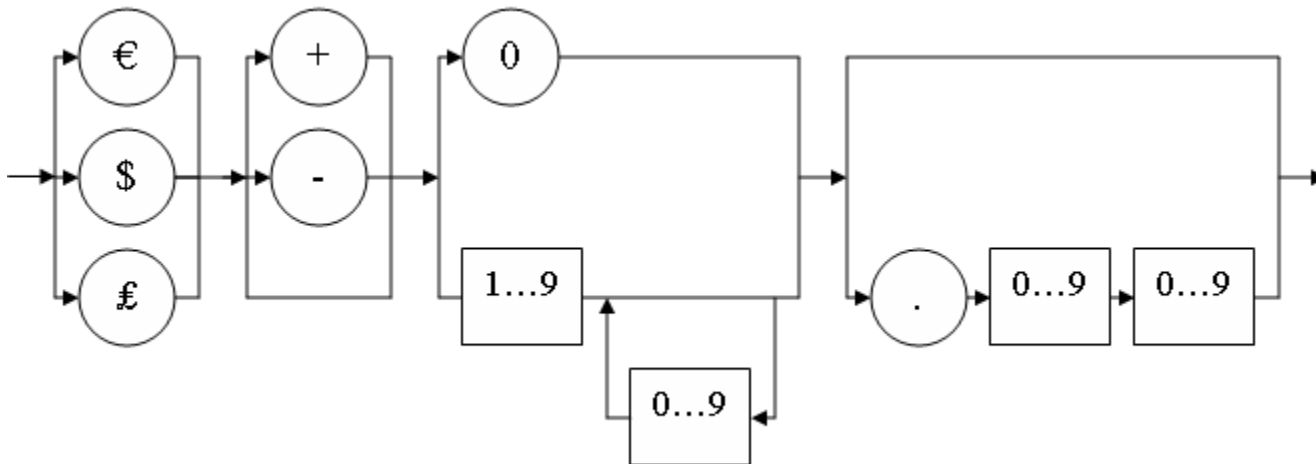
- r^*



Beispiel

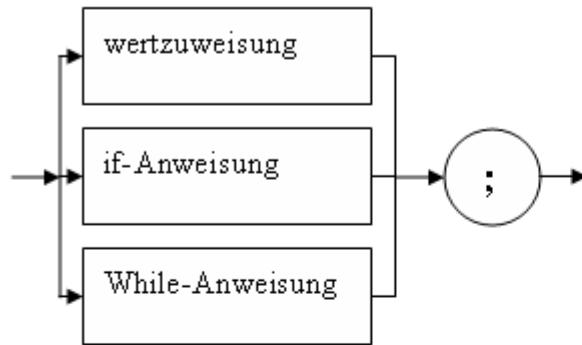
- Syntaxdiagramm für:

– *einzahlung* = $(\text{€}|\text{\$}|\text{£}) (+|-|\varepsilon)$
 $(0 | (1|2|\dots|9) (0|1|2|\dots|9)^*)$
 $(\varepsilon|. (0|1|2|\dots|9) (0|1|2|\dots|9))$

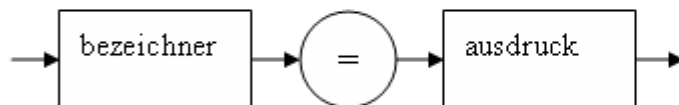


Beispiel: C-Anweisungen

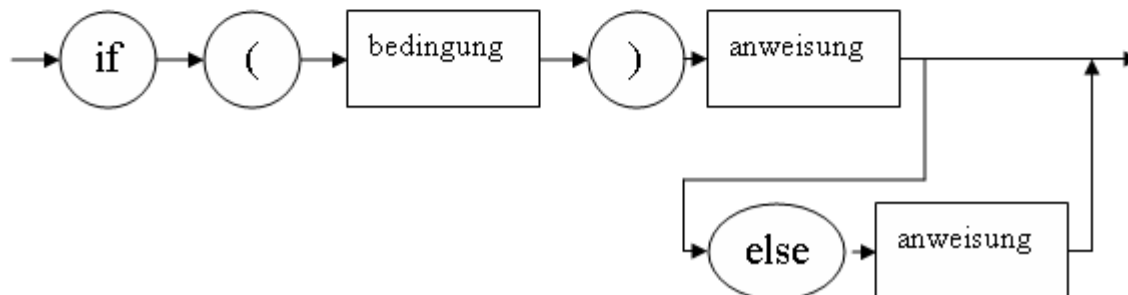
- anweisung:



- wertzuweisung:

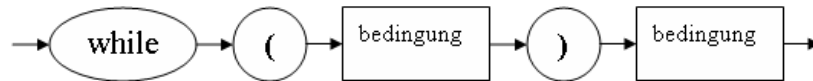


- if-anweisung:

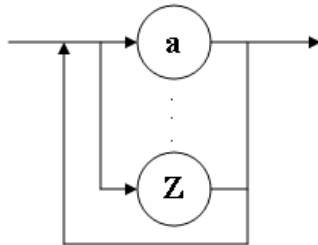


Beispiel: C-Anweisungen

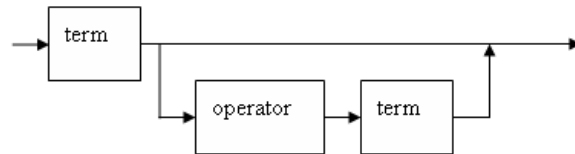
- while-anweisung:



- bezeichner:



- ausdrück:



- term:

