

SS 2005

Einführung in die Informatik IV

Ernst W. Mayr

Fakultät für Informatik
TU München

<http://www14.in.tum.de/lehre/2005SS/info4/index.html.de>

27. Mai 2005

4.10 $LR(k)$ -Grammatiken

Beispiel 91 (Grammatik für Arithmetische Ausdrücke)

Regeln:

$$S \rightarrow A$$

$$A \rightarrow E \mid A + E \mid A - E$$

$$E \rightarrow P \mid E * P \mid E / P$$

$$P \rightarrow (A) \mid a$$

4.10 $LR(k)$ -Grammatiken

Beispiel 91 (Grammatik für Arithmetische Ausdrücke)

Regeln:

$$S \rightarrow A$$

$$A \rightarrow E \mid A + E \mid A - E$$

$$E \rightarrow P \mid E * P \mid E / P$$

$$P \rightarrow (A) \mid a$$

Wir betrachten für das Parsen einen **bottom-up**-Ansatz, wobei die Reduktionen von links nach rechts angewendet werden.

4.10 $LR(k)$ -Grammatiken

Beispiel 91 (Grammatik für Arithmetische Ausdrücke)

Dabei können sich bei naivem Vorgehen allerdings Sackgassen ergeben, die dann aufgrund des Backtracking zu einem ineffizienten Algorithmus führen:

Regeln:

$$S \rightarrow A$$

$$A \rightarrow E \mid A + E \mid A - E$$

$$E \rightarrow P \mid E * P \mid E / P$$

$$P \rightarrow (A) \mid a$$

Ableitung:

$$a \quad + \quad a \quad * \quad a$$

$$P \quad + \quad a \quad * \quad a$$

$$E \quad + \quad a \quad * \quad a$$

$$A \quad + \quad a \quad * \quad a$$

$$S \quad + \quad a \quad * \quad a$$

4.10 $LR(k)$ -Grammatiken

Beispiel 91 (Grammatik für Arithmetische Ausdrücke)

Dabei können sich bei naivem Vorgehen allerdings Sackgassen ergeben, die dann aufgrund des Backtracking zu einem ineffizienten Algorithmus führen:

Regeln:

$$S \rightarrow A$$

$$A \rightarrow E \mid A + E \mid A - E$$

$$E \rightarrow P \mid E * P \mid E / P$$

$$P \rightarrow (A) \mid a$$

Ableitung:

$$a \quad + \quad a \quad * \quad a$$

$$P \quad + \quad a \quad * \quad a$$

$$E \quad + \quad a \quad * \quad a$$

$$A \quad + \quad a \quad * \quad a$$

$$S \quad + \quad a \quad * \quad a$$

4.10 $LR(k)$ -Grammatiken

Beispiel 91 (Grammatik für Arithmetische Ausdrücke)

Dabei können sich bei naivem Vorgehen allerdings Sackgassen ergeben, die dann aufgrund des Backtracking zu einem ineffizienten Algorithmus führen:

Regeln:

$$S \rightarrow A$$

$$A \rightarrow E \mid A + E \mid A - E$$

$$E \rightarrow P \mid E * P \mid E / P$$

$$P \rightarrow (A) \mid a$$

Ableitung:

$$a \quad + \quad a \quad * \quad a$$

$$P \quad + \quad a \quad * \quad a$$

$$E \quad + \quad a \quad * \quad a$$

$$A \quad + \quad a \quad * \quad a$$

$$S \quad + \quad a \quad * \quad a$$

4.10 $LR(k)$ -Grammatiken

Beispiel 91 (Grammatik für Arithmetische Ausdrücke)

Dabei können sich bei naivem Vorgehen allerdings Sackgassen ergeben, die dann aufgrund des Backtracking zu einem ineffizienten Algorithmus führen:

Regeln:

$$S \rightarrow A$$

$$A \rightarrow E \mid A + E \mid A - E$$

$$E \rightarrow P \mid E * P \mid E / P$$

$$P \rightarrow (A) \mid a$$

Ableitung:

$$a \quad + \quad a \quad * \quad a$$

$$P \quad + \quad a \quad * \quad a$$

$$E \quad + \quad a \quad * \quad a$$

$$A \quad + \quad a \quad * \quad a$$

$$S \quad + \quad a \quad * \quad a$$

4.10 LR(k)-Grammatiken

Beispiel 91 (Grammatik für Arithmetische Ausdrücke)

Dabei können sich bei naivem Vorgehen allerdings Sackgassen ergeben, die dann aufgrund des Backtracking zu einem ineffizienten Algorithmus führen:

Regeln:

$$S \rightarrow A$$

$$A \rightarrow E \mid A + E \mid A - E$$

$$E \rightarrow P \mid E * P \mid E / P$$

$$P \rightarrow (A) \mid a$$

Ableitung:

$$a \quad + \quad a \quad * \quad a$$

$$P \quad + \quad a \quad * \quad a$$

$$E \quad + \quad a \quad * \quad a$$

$$A \quad + \quad a \quad * \quad a$$

$$S \quad + \quad a \quad * \quad a$$

4.10 LR(k)-Grammatiken

Beispiel 91 (Grammatik für Arithmetische Ausdrücke)

Dabei können sich bei naivem Vorgehen allerdings Sackgassen ergeben, die dann aufgrund des Backtracking zu einem ineffizienten Algorithmus führen:

Regeln:

$$S \rightarrow A$$

$$A \rightarrow E \mid A + E \mid A - E$$

$$E \rightarrow P \mid E * P \mid E / P$$

$$P \rightarrow (A) \mid a$$

Ableitung:

$$a \quad + \quad a \quad * \quad a$$

$$P \quad + \quad a \quad * \quad a$$

$$E \quad + \quad a \quad * \quad a$$

$$A \quad + \quad a \quad * \quad a$$

$$S \quad + \quad a \quad * \quad a \quad \text{Sackgasse!}$$

4.10 $LR(k)$ -Grammatiken

Beispiel 91 (Grammatik für Arithmetische Ausdrücke)

... oder auch

Regeln:

$$S \rightarrow A$$

$$A \rightarrow E \mid A + E \mid A - E$$

$$E \rightarrow P \mid E * P \mid E / P$$

$$P \rightarrow (A) \mid a$$

Ableitung:

$$a \quad + \quad a \quad * \quad a$$

$$P \quad + \quad a \quad * \quad a$$

$$E \quad + \quad a \quad * \quad a$$

$$A \quad + \quad a \quad * \quad a$$

$$A \quad + \quad P \quad * \quad a$$

$$A \quad + \quad E \quad * \quad a$$

$$A \quad * \quad a$$

$$A \quad * \quad P$$

$$A \quad * \quad E$$

4.10 LR(k)-Grammatiken

Beispiel 91 (Grammatik für Arithmetische Ausdrücke)

... oder auch

Regeln:

$$S \rightarrow A$$

$$A \rightarrow E \mid A + E \mid A - E$$

$$E \rightarrow P \mid E * P \mid E / P$$

$$P \rightarrow (A) \mid a$$

Ableitung:

$$a \quad + \quad a \quad * \quad a$$

$$P \quad + \quad a \quad * \quad a$$

$$E \quad + \quad a \quad * \quad a$$

$$A \quad + \quad a \quad * \quad a$$

$$A \quad + \quad P \quad * \quad a$$

$$A \quad + \quad E \quad * \quad a$$

$$A \quad * \quad a$$

$$A \quad * \quad P$$

$$A \quad * \quad E$$

4.10 LR(k)-Grammatiken

Beispiel 91 (Grammatik für Arithmetische Ausdrücke)

... oder auch

Regeln:

$$S \rightarrow A$$

$$A \rightarrow E \mid A + E \mid A - E$$

$$E \rightarrow P \mid E * P \mid E / P$$

$$P \rightarrow (A) \mid a$$

Ableitung:

$$a \quad + \quad a \quad * \quad a$$

$$P \quad + \quad a \quad * \quad a$$

$$E \quad + \quad a \quad * \quad a$$

$$A \quad + \quad a \quad * \quad a$$

$$A \quad + \quad P \quad * \quad a$$

$$A \quad + \quad E \quad * \quad a$$

$$A \quad * \quad a$$

$$A \quad * \quad P$$

$$A \quad * \quad E$$

4.10 LR(k)-Grammatiken

Beispiel 91 (Grammatik für Arithmetische Ausdrücke)

... oder auch

Regeln:

$$S \rightarrow A$$

$$A \rightarrow E \mid A + E \mid A - E$$

$$E \rightarrow P \mid E * P \mid E / P$$

$$P \rightarrow (A) \mid a$$

Ableitung:

$$a \quad + \quad a \quad * \quad a$$

$$P \quad + \quad a \quad * \quad a$$

$$E \quad + \quad a \quad * \quad a$$

$$A \quad + \quad a \quad * \quad a$$

$$A \quad + \quad P \quad * \quad a$$

$$A \quad + \quad E \quad * \quad a$$

$$A \quad * \quad a$$

$$A \quad * \quad P$$

$$A \quad * \quad E$$

4.10 LR(k)-Grammatiken

Beispiel 91 (Grammatik für Arithmetische Ausdrücke)

... oder auch

Regeln:

$$S \rightarrow A$$

$$A \rightarrow E \mid A + E \mid A - E$$

$$E \rightarrow P \mid E * P \mid E / P$$

$$P \rightarrow (A) \mid a$$

Ableitung:

$$a \quad + \quad a \quad * \quad a$$

$$P \quad + \quad a \quad * \quad a$$

$$E \quad + \quad a \quad * \quad a$$

$$A \quad + \quad a \quad * \quad a$$

$$A \quad + \quad P \quad * \quad a$$

$$A \quad + \quad E \quad * \quad a$$

$$A \quad * \quad a$$

$$A \quad * \quad P$$

$$A \quad * \quad E$$

4.10 LR(k)-Grammatiken

Beispiel 91 (Grammatik für Arithmetische Ausdrücke)

... oder auch

Regeln:

$$S \rightarrow A$$

$$A \rightarrow E \mid A + E \mid A - E$$

$$E \rightarrow P \mid E * P \mid E / P$$

$$P \rightarrow (A) \mid a$$

Ableitung:

$$a \quad + \quad a \quad * \quad a$$

$$P \quad + \quad a \quad * \quad a$$

$$E \quad + \quad a \quad * \quad a$$

$$A \quad + \quad a \quad * \quad a$$

$$A \quad + \quad P \quad * \quad a$$

$$A \quad + \quad E \quad * \quad a$$

$$A \quad * \quad a$$

$$A \quad * \quad P$$

$$A \quad * \quad E$$

4.10 LR(k)-Grammatiken

Beispiel 91 (Grammatik für Arithmetische Ausdrücke)

... oder auch

Regeln:

$$S \rightarrow A$$

$$A \rightarrow E \mid A + E \mid A - E$$

$$E \rightarrow P \mid E * P \mid E / P$$

$$P \rightarrow (A) \mid a$$

Ableitung:

$$a \quad + \quad a \quad * \quad a$$

$$P \quad + \quad a \quad * \quad a$$

$$E \quad + \quad a \quad * \quad a$$

$$A \quad + \quad a \quad * \quad a$$

$$A \quad + \quad P \quad * \quad a$$

$$A \quad + \quad E \quad * \quad a$$

$$A \quad * \quad a$$

$$A \quad * \quad P$$

$$A \quad * \quad E$$

4.10 LR(k)-Grammatiken

Beispiel 91 (Grammatik für Arithmetische Ausdrücke)

... oder auch

Regeln:

$$S \rightarrow A$$

$$A \rightarrow E \mid A + E \mid A - E$$

$$E \rightarrow P \mid E * P \mid E / P$$

$$P \rightarrow (A) \mid a$$

Ableitung:

$$a \quad + \quad a \quad * \quad a$$

$$P \quad + \quad a \quad * \quad a$$

$$E \quad + \quad a \quad * \quad a$$

$$A \quad + \quad a \quad * \quad a$$

$$A \quad + \quad P \quad * \quad a$$

$$A \quad + \quad E \quad * \quad a$$

$$A \quad * \quad a$$

$$A \quad * \quad P$$

$$A \quad * \quad E$$

4.10 LR(k)-Grammatiken

Beispiel 91 (Grammatik für Arithmetische Ausdrücke)

... oder auch

Regeln:

$$S \rightarrow A$$

$$A \rightarrow E \mid A + E \mid A - E$$

$$E \rightarrow P \mid E * P \mid E / P$$

$$P \rightarrow (A) \mid a$$

Ableitung:

$$a \quad + \quad a \quad * \quad a$$

$$P \quad + \quad a \quad * \quad a$$

$$E \quad + \quad a \quad * \quad a$$

$$A \quad + \quad a \quad * \quad a$$

$$A \quad + \quad P \quad * \quad a$$

$$A \quad + \quad E \quad * \quad a$$

$$A \quad * \quad a$$

$$A \quad * \quad P$$

$$A \quad * \quad E$$

4.10 LR(k)-Grammatiken

Beispiel 91 (Grammatik für Arithmetische Ausdrücke)

... oder auch

Regeln:

$$S \rightarrow A$$

$$A \rightarrow E \mid A + E \mid A - E$$

$$E \rightarrow P \mid E * P \mid E / P$$

$$P \rightarrow (A) \mid a$$

Ableitung:

$$a \quad + \quad a \quad * \quad a$$

$$P \quad + \quad a \quad * \quad a$$

$$E \quad + \quad a \quad * \quad a$$

$$A \quad + \quad a \quad * \quad a$$

$$A \quad + \quad P \quad * \quad a$$

$$A \quad + \quad E \quad * \quad a$$

$$A \quad * \quad a$$

$$A \quad * \quad P$$

$$A \quad * \quad E$$

Sackgasse!

4.10 $LR(k)$ -Grammatiken

Beispiel 91 (Grammatik für Arithmetische Ausdrücke)

Zur Behebung des Problems führen wir für jede Ableitungsregel (besser hier: **Reduktionsregel**) einen **Lookahead** (der Länge k) ein (in unserem Beispiel $k = 1$) und legen fest, dass eine Ableitungsregel nur dann angewendet werden darf, wenn die nächsten k Zeichen mit den erlaubten Lookaheads übereinstimmen.

4.10 $LR(k)$ -Grammatiken

Beispiel 91 (Grammatik für Arithmetische Ausdrücke)

Produktion	Lookaheads (der Länge 1)
$S \rightarrow A$	ϵ
$A \rightarrow E$	$+, -,), \epsilon$
$A \rightarrow A + E$	$+, -,), \epsilon$
$A \rightarrow A - E$	$+, -,), \epsilon$
$E \rightarrow P$	beliebig
$E \rightarrow E * P$	beliebig
$E \rightarrow E / P$	beliebig
$P \rightarrow (A)$	beliebig
$P \rightarrow a$	beliebig

4.10 $LR(k)$ -Grammatiken

Beispiel 91 (Grammatik für Arithmetische Ausdrücke)

Damit ergibt sich

Produktion	Lookaheads
$S \rightarrow A$	ϵ
$A \rightarrow E$	$+, -,), \epsilon$
$A \rightarrow A + E$	$+, -,), \epsilon$
$A \rightarrow A - E$	$+, -,), \epsilon$
$E \rightarrow P$	beliebig
$E \rightarrow E * P$	beliebig
$E \rightarrow E / P$	beliebig
$P \rightarrow (A)$	beliebig
$P \rightarrow a$	beliebig

Ableitung:

$a \quad + \quad a \quad * \quad a$
 $P \quad + \quad a \quad * \quad a$
 $E \quad + \quad a \quad * \quad a$
 $A \quad + \quad a \quad * \quad a$
 $A \quad + \quad P \quad * \quad a$
 $A \quad + \quad E \quad * \quad a$
 $A \quad + \quad E \quad * \quad P$
 $A \quad + \quad \quad \quad E$
 $\quad \quad \quad A$
 $\quad \quad \quad S$

4.10 LR(k)-Grammatiken

Beispiel 91 (Grammatik für Arithmetische Ausdrücke)

Damit ergibt sich

Produktion	Lookaheads
$S \rightarrow A$	ϵ
$A \rightarrow E$	$+, -,), \epsilon$
$A \rightarrow A + E$	$+, -,), \epsilon$
$A \rightarrow A - E$	$+, -,), \epsilon$
$E \rightarrow P$	beliebig
$E \rightarrow E * P$	beliebig
$E \rightarrow E / P$	beliebig
$P \rightarrow (A)$	beliebig
$P \rightarrow a$	beliebig

Ableitung:

a + a * a
 P + a * a
 E + a * a
 A + a * a
 A + P * a
 A + E * a
 A + E * P
 A + E
 A
 S

4.10 LR(k)-Grammatiken

Beispiel 91 (Grammatik für Arithmetische Ausdrücke)

Damit ergibt sich

Produktion	Lookaheads
$S \rightarrow A$	ϵ
$A \rightarrow E$	$+, -,), \epsilon$
$A \rightarrow A + E$	$+, -,), \epsilon$
$A \rightarrow A - E$	$+, -,), \epsilon$
$E \rightarrow P$	beliebig
$E \rightarrow E * P$	beliebig
$E \rightarrow E / P$	beliebig
$P \rightarrow (A)$	beliebig
$P \rightarrow a$	beliebig

Ableitung:

$a + a * a$
 $P + a * a$
 $E + a * a$
 $A + a * a$
 $A + P * a$
 $A + E * a$
 $A + E * P$
 $A + E$
 A
 S

4.10 $LR(k)$ -Grammatiken

Beispiel 91 (Grammatik für Arithmetische Ausdrücke)

Damit ergibt sich

Produktion	Lookaheads
$S \rightarrow A$	ϵ
$A \rightarrow E$	$+, -,), \epsilon$
$A \rightarrow A + E$	$+, -,), \epsilon$
$A \rightarrow A - E$	$+, -,), \epsilon$
$E \rightarrow P$	beliebig
$E \rightarrow E * P$	beliebig
$E \rightarrow E / P$	beliebig
$P \rightarrow (A)$	beliebig
$P \rightarrow a$	beliebig

Ableitung:

$a + a * a$
 $P + a * a$
 $E + a * a$
 $A + a * a$
 $A + P * a$
 $A + E * a$
 $A + E * P$
 $A + E$
 A
 S

4.10 LR(k)-Grammatiken

Beispiel 91 (Grammatik für Arithmetische Ausdrücke)

Damit ergibt sich

Produktion	Lookaheads
$S \rightarrow A$	ϵ
$A \rightarrow E$	$+, -,), \epsilon$
$A \rightarrow A + E$	$+, -,), \epsilon$
$A \rightarrow A - E$	$+, -,), \epsilon$
$E \rightarrow P$	beliebig
$E \rightarrow E * P$	beliebig
$E \rightarrow E / P$	beliebig
$P \rightarrow (A)$	beliebig
$P \rightarrow a$	beliebig

Ableitung:

$a + a * a$
 $P + a * a$
 $E + a * a$
 $A + a * a$
 $A + P * a$
 $A + E * a$
 $A + E * P$
 $A + E$
 A
 S

4.10 $LR(k)$ -Grammatiken

Beispiel 91 (Grammatik für Arithmetische Ausdrücke)

Damit ergibt sich

Produktion	Lookaheads
$S \rightarrow A$	ϵ
$A \rightarrow E$	$+, -,), \epsilon$
$A \rightarrow A + E$	$+, -,), \epsilon$
$A \rightarrow A - E$	$+, -,), \epsilon$
$E \rightarrow P$	beliebig
$E \rightarrow E * P$	beliebig
$E \rightarrow E / P$	beliebig
$P \rightarrow (A)$	beliebig
$P \rightarrow a$	beliebig

Ableitung:

$a + a * a$
 $P + a * a$
 $E + a * a$
 $A + a * a$
 $A + P * a$
 $A + E * a$
 $A + E * P$
 $A + E$
 A
 S

4.10 LR(k)-Grammatiken

Beispiel 91 (Grammatik für Arithmetische Ausdrücke)

Damit ergibt sich

Produktion	Lookaheads
$S \rightarrow A$	ϵ
$A \rightarrow E$	$+, -,), \epsilon$
$A \rightarrow A + E$	$+, -,), \epsilon$
$A \rightarrow A - E$	$+, -,), \epsilon$
$E \rightarrow P$	beliebig
$E \rightarrow E * P$	beliebig
$E \rightarrow E / P$	beliebig
$P \rightarrow (A)$	beliebig
$P \rightarrow a$	beliebig

Ableitung:

$a + a * a$
 $P + a * a$
 $E + a * a$
 $A + a * a$
 $A + P * a$
 $A + E * a$
 $A + E * P$
 $A + \quad E$
 $\quad A$
 $\quad S$

4.10 LR(k)-Grammatiken

Beispiel 91 (Grammatik für Arithmetische Ausdrücke)

Damit ergibt sich

Produktion	Lookaheads
$S \rightarrow A$	ϵ
$A \rightarrow E$	$+, -,), \epsilon$
$A \rightarrow A + E$	$+, -,), \epsilon$
$A \rightarrow A - E$	$+, -,), \epsilon$
$E \rightarrow P$	beliebig
$E \rightarrow E * P$	beliebig
$E \rightarrow E / P$	beliebig
$P \rightarrow (A)$	beliebig
$P \rightarrow a$	beliebig

Ableitung:

$a + a * a$
 $P + a * a$
 $E + a * a$
 $A + a * a$
 $A + P * a$
 $A + E * a$
 $A + E * P$
 $A + E$
 A
 S

4.10 $LR(k)$ -Grammatiken

Beispiel 91 (Grammatik für Arithmetische Ausdrücke)

Damit ergibt sich

Produktion	Lookaheads
$S \rightarrow A$	ϵ
$A \rightarrow E$	$+, -,), \epsilon$
$A \rightarrow A + E$	$+, -,), \epsilon$
$A \rightarrow A - E$	$+, -,), \epsilon$
$E \rightarrow P$	beliebig
$E \rightarrow E * P$	beliebig
$E \rightarrow E / P$	beliebig
$P \rightarrow (A)$	beliebig
$P \rightarrow a$	beliebig

Ableitung:

$$\begin{array}{cccccc}
 a & + & a & * & a & \\
 P & + & a & * & a & \\
 E & + & a & * & a & \\
 A & + & a & * & a & \\
 A & + & P & * & a & \\
 A & + & E & * & a & \\
 A & + & E & * & P & \\
 A & + & & & E & \\
 & & & & A & \\
 & & & & S &
 \end{array}$$

4.10 $LR(k)$ -Grammatiken

Beispiel 91 (Grammatik für Arithmetische Ausdrücke)

Damit ergibt sich

Produktion	Lookaheads
$S \rightarrow A$	ϵ
$A \rightarrow E$	$+, -,), \epsilon$
$A \rightarrow A + E$	$+, -,), \epsilon$
$A \rightarrow A - E$	$+, -,), \epsilon$
$E \rightarrow P$	beliebig
$E \rightarrow E * P$	beliebig
$E \rightarrow E / P$	beliebig
$P \rightarrow (A)$	beliebig
$P \rightarrow a$	beliebig

Ableitung:

$$\begin{array}{cccccc}
 a & + & a & * & a & \\
 P & + & a & * & a & \\
 E & + & a & * & a & \\
 A & + & a & * & a & \\
 A & + & P & * & a & \\
 A & + & E & * & a & \\
 A & + & E & * & P & \\
 A & + & & E & & \\
 & & & & A & \\
 & & & & S &
 \end{array}$$

4.10 LR(k)-Grammatiken

Beispiel 91 (Grammatik für Arithmetische Ausdrücke)

Damit ergibt sich

Produktion	Lookaheads
$S \rightarrow A$	ϵ
$A \rightarrow E$	$+, -,), \epsilon$
$A \rightarrow A + E$	$+, -,), \epsilon$
$A \rightarrow A - E$	$+, -,), \epsilon$
$E \rightarrow P$	beliebig
$E \rightarrow E * P$	beliebig
$E \rightarrow E / P$	beliebig
$P \rightarrow (A)$	beliebig
$P \rightarrow a$	beliebig

Ableitung:

$a + a * a$
 $P + a * a$
 $E + a * a$
 $A + a * a$
 $A + P * a$
 $A + E * a$
 $A + E * P$
 $A + E$
 A
 S

Definition 92

Eine kontextfreie Grammatik ist eine $LR(k)$ -Grammatik, wenn man durch Lookaheads der Länge k erreichen kann, dass bei einer Reduktion von links nach rechts in jedem Schritt höchstens eine Produktion/Reduktion anwendbar ist.

Korollar 93

Jede kontextfreie Sprache, für die es eine $LR(k)$ -Grammatik gibt, ist deterministisch kontextfrei.

Definition 92

Eine kontextfreie Grammatik ist eine $LR(k)$ -Grammatik, wenn man durch Lookaheads der Länge k erreichen kann, dass bei einer Reduktion von links nach rechts in jedem Schritt höchstens eine Produktion/Reduktion anwendbar ist.

Korollar 93

Jede kontextfreie Sprache, für die es eine $LR(k)$ -Grammatik gibt, ist deterministisch kontextfrei.

Bemerkung:

Es gibt eine (im allgemeinen nicht effiziente) Konstruktion, um aus einer $LR(k)$ -Grammatik, $k > 1$, eine äquivalente $LR(1)$ -Grammatik zu machen.

Korollar 94

Die folgenden Familien von Sprachen sind gleich:

- *die Familie DCFL,*
- *die $LR(1)$ -Sprachen.*

Bemerkung:

Es gibt eine (im allgemeinen nicht effiziente) Konstruktion, um aus einer $LR(k)$ -Grammatik, $k > 1$, eine äquivalente $LR(1)$ -Grammatik zu machen.

Korollar 94

Die folgenden Familien von Sprachen sind gleich:

- *die Familie DCFL,*
- *die $LR(1)$ -Sprachen.*

Bemerkung:

Es gibt eine (im allgemeinen nicht effiziente) Konstruktion, um aus einer $LR(k)$ -Grammatik, $k > 1$, eine äquivalente $LR(1)$ -Grammatik zu machen.

Korollar 94

Die folgenden Familien von Sprachen sind gleich:

- *die Familie DCFL,*
- *die $LR(1)$ -Sprachen.*

Bemerkung:

Es gibt eine (im allgemeinen nicht effiziente) Konstruktion, um aus einer $LR(k)$ -Grammatik, $k > 1$, eine äquivalente $LR(1)$ -Grammatik zu machen.

Korollar 94

Die folgenden Familien von Sprachen sind gleich:

- *die Familie DCFL,*
- *die $LR(1)$ -Sprachen.*

4.11 $LL(k)$ -Grammatiken

Beispiel 95 (Noch eine Grammatik)

Regeln:

$$S \rightarrow A$$

$$A \rightarrow E \mid E + A$$

$$E \rightarrow P \mid P * E$$

$$P \rightarrow (A) \mid a$$

$$S \rightarrow A$$

$$A \rightarrow EA'$$

$$A' \rightarrow +A \mid \epsilon$$

$$E \rightarrow PE'$$

$$E' \rightarrow *E \mid \epsilon$$

$$P \rightarrow (A) \mid a$$

4.11 $LL(k)$ -Grammatiken

Beispiel 95 (Noch eine Grammatik)

Regeln:

$$S \rightarrow A$$

$$A \rightarrow E \mid E + A$$

$$E \rightarrow P \mid P * E$$

$$P \rightarrow (A) \mid a$$

$$S \rightarrow A$$

$$A \rightarrow EA'$$

$$A' \rightarrow +A \mid \epsilon$$

$$E \rightarrow PE'$$

$$E' \rightarrow *E \mid \epsilon$$

$$P \rightarrow (A) \mid a$$

Wir betrachten nun für das **Parse**n einen **top-down**-Ansatz, wobei die Produktionen in Form einer Linksableitung angewendet werden.

4.11 $LL(k)$ -Grammatiken

Beispiel 95 (Noch eine Grammatik)

Regeln:

$$S \rightarrow A$$

$$A \rightarrow E \mid E + A$$

$$E \rightarrow P \mid P * E$$

$$P \rightarrow (A) \mid a$$

Ableitung:

S

A

E

P

$a + a * a$

4.11 $LL(k)$ -Grammatiken

Beispiel 95 (Noch eine Grammatik)

Regeln:

$$S \rightarrow A$$

$$A \rightarrow E \mid E + A$$

$$E \rightarrow P \mid P * E$$

$$P \rightarrow (A) \mid a$$

Ableitung:

S

A

E

P

$a + a * a$

4.11 $LL(k)$ -Grammatiken

Beispiel 95 (Noch eine Grammatik)

Regeln:

$$S \rightarrow A$$

$$A \rightarrow E \mid E + A$$

$$E \rightarrow P \mid P * E$$

$$P \rightarrow (A) \mid a$$

Ableitung:

S

A

E

P

$a + a * a$

4.11 $LL(k)$ -Grammatiken

Beispiel 95 (Noch eine Grammatik)

Regeln:

$$S \rightarrow A$$

$$A \rightarrow E \mid E + A$$

$$E \rightarrow P \mid P * E$$

$$P \rightarrow (A) \mid a$$

Ableitung:

S

A

E

P

$a + a * a$

4.11 $LL(k)$ -Grammatiken

Beispiel 95 (Noch eine Grammatik)

Regeln:

$$S \rightarrow A$$

$$A \rightarrow E \mid E + A$$

$$E \rightarrow P \mid P * E$$

$$P \rightarrow (A) \mid a$$

Ableitung:

S

A

E

P

$a + a * a$

4.11 $LL(k)$ -Grammatiken

Beispiel 95 (Noch eine Grammatik)

Regeln:

$$S \rightarrow A$$

$$A \rightarrow E \mid E + A$$

$$E \rightarrow P \mid P * E$$

$$P \rightarrow (A) \mid a$$

Ableitung:

S

A

E

P

$a + a * a$ Sackgasse!

4.11 $LL(k)$ -Grammatiken

Beispiel 95 (Noch eine Grammatik)

Wir bestimmen nun für jede Produktion $A \rightarrow \alpha$ ihre **Auswahlmenge**, das heißt die Menge aller terminalen Präfixe der Länge $\leq k$ der von α ableitbaren Zeichenreihen.

4.11 $LL(k)$ -Grammatiken

Beispiel 95 (Noch eine Grammatik)

Wir bestimmen nun für jede Produktion $A \rightarrow \alpha$ ihre **Auswahlmenge**, das heißt die Menge aller terminalen Präfixe der Länge $\leq k$ der von α ableitbaren Zeichenreihen.

Es ergibt sich (der Einfachheit lassen wir ϵ -Produktionen zu):

S	$\rightarrow A$	$\{a, (\}$
A	$\rightarrow EA'$	$\{a, (\}$
A'	$\rightarrow +A$	$\{+\}$
A'	$\rightarrow \epsilon$	$\{)\}$
E	$\rightarrow PE'$	$\{a(\}$
E'	$\rightarrow *E$	$\{*\}$
E'	$\rightarrow \epsilon$	$\{+)\}$
P	$\rightarrow (A)$	$\{(\}$
P	$\rightarrow a$	$\{a\}$

4.11 $LL(k)$ -Grammatiken

Beispiel 95 (Noch eine Grammatik)

Damit ergibt sich

S	$\rightarrow A$	$\{a, (\}$
A	$\rightarrow EA'$	$\{a, (\}$
A'	$\rightarrow +A$	$\{+\}$
A'	$\rightarrow \epsilon$	$\{)\}$
E	$\rightarrow PE'$	$\{a(\}$
E'	$\rightarrow *E$	$\{*\}$
E'	$\rightarrow \epsilon$	$\{+)\}$
P	$\rightarrow (A)$	$\{(\}$
P	$\rightarrow a$	$\{a\}$

Ableitung:

S
 A
 \vdots
 $E + E$
 \vdots
 $E + P * P$
 \vdots
 $a + a * a$

4.11 $LL(k)$ -Grammatiken

Beispiel 95 (Noch eine Grammatik)

Bemerkung:

Parser für $LL(k)$ -Grammatiken entsprechen der Methode des rekursiven Abstiegs (recursive descent).

5. Kontextsensitive und Typ-0-Sprachen

5.1 Turingmaschinen

Turingmaschinen sind das grundlegende Modell, das wir für Computer/Rechenmaschinen verwenden. Es geht auf **Alan Turing** (1912–1954) zurück.

Definition 96

Eine **nichtdeterministische Turingmaschine** (kurz TM) wird durch ein 7-Tupel $M = (Q, \Sigma, \Gamma, \delta, q_0, \square, F)$ beschrieben, das folgende Bedingungen erfüllt:

- 1 Q ist eine endliche Menge von Zuständen.
- 2 Σ ist eine endliche Menge, das Eingabealphabet.
- 3 Γ ist eine endliche Menge, das Bandalphabet.
- 4 $\delta : Q \times \Gamma \rightarrow \mathcal{P}(Q \times \Gamma \times \{L, R, N\})$ ist die Übergangsfunktion.
- 5 $q_0 \in Q$ ist der Startzustand.
- 6 $\square \in \Gamma \setminus \Sigma$ ist das Leerzeichen.
- 7 $F \subseteq Q$ ist die Menge der (akzeptierenden) Endzustände.

Eine Turingmaschine heißt **deterministisch**, falls gilt

$$|\delta(q, a)| \leq 1 \quad \text{für alle } q \in Q, a \in \Gamma.$$

Definition 96

Eine **nichtdeterministische Turingmaschine** (kurz TM) wird durch ein 7-Tupel $M = (Q, \Sigma, \Gamma, \delta, q_0, \square, F)$ beschrieben, das folgende Bedingungen erfüllt:

- 1 Q ist eine endliche Menge von **Zuständen**.
- 2 Σ ist eine **endliche Menge, das Eingabealphabet**.
- 3 Γ ist eine endliche Menge, das **Bandalphabet**.
- 4 $\delta : Q \times \Gamma \rightarrow \mathcal{P}(Q \times \Gamma \times \{L, R, N\})$ ist die **Übergangsfunktion**.
- 5 $q_0 \in Q$ ist der **Startzustand**.
- 6 $\square \in \Gamma \setminus \Sigma$ ist das **Leerzeichen**.
- 7 $F \subseteq Q$ ist die Menge der (**akzeptierenden**) **Endzustände**.

Eine Turingmaschine heißt **deterministisch**, falls gilt

$$|\delta(q, a)| \leq 1 \quad \text{für alle } q \in Q, a \in \Gamma.$$

Definition 96

Eine **nichtdeterministische Turingmaschine** (kurz TM) wird durch ein 7-Tupel $M = (Q, \Sigma, \Gamma, \delta, q_0, \square, F)$ beschrieben, das folgende Bedingungen erfüllt:

- 1 Q ist eine endliche Menge von **Zuständen**.
- 2 Σ ist eine endliche Menge, das **Eingabealphabet**.
- 3 Γ ist eine **endliche Menge**, das **Bandalphabet**.
- 4 $\delta : Q \times \Gamma \rightarrow \mathcal{P}(Q \times \Gamma \times \{L, R, N\})$ ist die **Übergangsfunktion**.
- 5 $q_0 \in Q$ ist der **Startzustand**.
- 6 $\square \in \Gamma \setminus \Sigma$ ist das **Leerzeichen**.
- 7 $F \subseteq Q$ ist die Menge der (akzeptierenden) **Endzustände**.

Eine Turingmaschine heißt **deterministisch**, falls gilt

$$|\delta(q, a)| \leq 1 \quad \text{für alle } q \in Q, a \in \Gamma.$$

Definition 96

Eine **nichtdeterministische Turingmaschine** (kurz TM) wird durch ein 7-Tupel $M = (Q, \Sigma, \Gamma, \delta, q_0, \square, F)$ beschrieben, das folgende Bedingungen erfüllt:

- 1 Q ist eine endliche Menge von **Zuständen**.
- 2 Σ ist eine endliche Menge, das **Eingabealphabet**.
- 3 Γ ist eine endliche Menge, das **Bandalphabet**.
- 4 $\delta : Q \times \Gamma \rightarrow \mathcal{P}(Q \times \Gamma \times \{L, R, N\})$ ist die **Übergangsfunktion**.
- 5 $q_0 \in Q$ ist der **Startzustand**.
- 6 $\square \in \Gamma \setminus \Sigma$ ist das **Leerzeichen**.
- 7 $F \subseteq Q$ ist die Menge der (akzeptierenden) **Endzustände**.

Eine Turingmaschine heißt **deterministisch**, falls gilt

$$|\delta(q, a)| \leq 1 \quad \text{für alle } q \in Q, a \in \Gamma.$$

Definition 96

Eine **nichtdeterministische Turingmaschine** (kurz TM) wird durch ein 7-Tupel $M = (Q, \Sigma, \Gamma, \delta, q_0, \square, F)$ beschrieben, das folgende Bedingungen erfüllt:

- 1 Q ist eine endliche Menge von **Zuständen**.
- 2 Σ ist eine endliche Menge, das **Eingabealphabet**.
- 3 Γ ist eine endliche Menge, das **Bandalphabet**.
- 4 $\delta : Q \times \Gamma \rightarrow \mathcal{P}(Q \times \Gamma \times \{L, R, N\})$ ist die **Übergangsfunktion**.
- 5 $q_0 \in Q$ ist der **Startzustand**.
- 6 $\square \in \Gamma \setminus \Sigma$ ist das **Leerzeichen**.
- 7 $F \subseteq Q$ ist die Menge der (akzeptierenden) **Endzustände**.

Eine Turingmaschine heißt **deterministisch**, falls gilt

$$|\delta(q, a)| \leq 1 \quad \text{für alle } q \in Q, a \in \Gamma.$$

Definition 96

Eine **nichtdeterministische Turingmaschine** (kurz TM) wird durch ein 7-Tupel $M = (Q, \Sigma, \Gamma, \delta, q_0, \square, F)$ beschrieben, das folgende Bedingungen erfüllt:

- 1 Q ist eine endliche Menge von **Zuständen**.
- 2 Σ ist eine endliche Menge, das **Eingabealphabet**.
- 3 Γ ist eine endliche Menge, das **Bandalphabet**.
- 4 $\delta : Q \times \Gamma \rightarrow \mathcal{P}(Q \times \Gamma \times \{L, R, N\})$ ist die **Übergangsfunktion**.
- 5 $q_0 \in Q$ ist der **Startzustand**.
- 6 $\square \in \Gamma \setminus \Sigma$ ist das **Leerzeichen**.
- 7 $F \subseteq Q$ ist die Menge der (akzeptierenden) **Endzustände**.

Eine Turingmaschine heißt **deterministisch**, falls gilt

$$|\delta(q, a)| \leq 1 \quad \text{für alle } q \in Q, a \in \Gamma.$$

Definition 96

Eine **nichtdeterministische Turingmaschine** (kurz TM) wird durch ein 7-Tupel $M = (Q, \Sigma, \Gamma, \delta, q_0, \square, F)$ beschrieben, das folgende Bedingungen erfüllt:

- 1 Q ist eine endliche Menge von **Zuständen**.
- 2 Σ ist eine endliche Menge, das **Eingabealphabet**.
- 3 Γ ist eine endliche Menge, das **Bandalphabet**.
- 4 $\delta : Q \times \Gamma \rightarrow \mathcal{P}(Q \times \Gamma \times \{L, R, N\})$ ist die **Übergangsfunktion**.
- 5 $q_0 \in Q$ ist der **Startzustand**.
- 6 $\square \in \Gamma \setminus \Sigma$ ist das **Leerzeichen**.
- 7 $F \subseteq Q$ ist die **Menge der (akzeptierenden) Endzustände**.

Eine Turingmaschine heißt **deterministisch**, falls gilt

$$|\delta(q, a)| \leq 1 \quad \text{für alle } q \in Q, a \in \Gamma.$$

Definition 96

Eine **nichtdeterministische Turingmaschine** (kurz TM) wird durch ein 7-Tupel $M = (Q, \Sigma, \Gamma, \delta, q_0, \square, F)$ beschrieben, das folgende Bedingungen erfüllt:

- 1 Q ist eine endliche Menge von **Zuständen**.
- 2 Σ ist eine endliche Menge, das **Eingabealphabet**.
- 3 Γ ist eine endliche Menge, das **Bandalphabet**.
- 4 $\delta : Q \times \Gamma \rightarrow \mathcal{P}(Q \times \Gamma \times \{L, R, N\})$ ist die **Übergangsfunktion**.
- 5 $q_0 \in Q$ ist der **Startzustand**.
- 6 $\square \in \Gamma \setminus \Sigma$ ist das **Leerzeichen**.
- 7 $F \subseteq Q$ ist die Menge der **(akzeptierenden) Endzustände**.

Eine Turingmaschine heißt **deterministisch**, falls gilt

$$|\delta(q, a)| \leq 1 \quad \text{für alle } q \in Q, a \in \Gamma.$$

Definition 96

Eine **nichtdeterministische Turingmaschine** (kurz TM) wird durch ein 7-Tupel $M = (Q, \Sigma, \Gamma, \delta, q_0, \square, F)$ beschrieben, das folgende Bedingungen erfüllt:

- 1 Q ist eine endliche Menge von **Zuständen**.
- 2 Σ ist eine endliche Menge, das **Eingabealphabet**.
- 3 Γ ist eine endliche Menge, das **Bandalphabet**.
- 4 $\delta : Q \times \Gamma \rightarrow \mathcal{P}(Q \times \Gamma \times \{L, R, N\})$ ist die **Übergangsfunktion**.
- 5 $q_0 \in Q$ ist der **Startzustand**.
- 6 $\square \in \Gamma \setminus \Sigma$ ist das **Leerzeichen**.
- 7 $F \subseteq Q$ ist die Menge der **(akzeptierenden) Endzustände**.

Eine Turingmaschine heißt **deterministisch**, falls gilt

$$|\delta(q, a)| \leq 1 \quad \text{für alle } q \in Q, a \in \Gamma.$$

Erläuterung:

Intuitiv bedeutet $\delta(q, a) = (q', b, d)$ bzw. $\delta(q, a) \ni (q', b, d)$:
Wenn sich M im Zustand q befindet und unter dem Schreib-/Lesekopf das Zeichen a steht, so geht M im nächsten Schritt in den Zustand q' über, schreibt an die Stelle des a 's das Zeichen b und bewegt danach den Schreib-/Lesekopf um eine Position nach **rechts** (falls $d = R$), **links** (falls $d = L$) bzw. lässt ihn **unverändert** (falls $d = N$).

Beispiel 97

Es soll eine TM angegeben werden, die eine gegebene Zeichenreihe aus $\{0, 1\}^+$ als Binärzahl interpretiert und zu dieser Zahl 1 addiert. Folgende Vorgehensweise bietet sich an:

- 1 Gehe ganz nach rechts bis ans Ende der Zahl. Dieses Ende kann durch das erste Auftreten eines Leerzeichens gefunden werden.
- 2 Gehe wieder nach links bis zur ersten 0 und ändere diese zu einer 1. Ersetze dabei auf dem Weg alle 1en durch 0.

Beispiel 97

Es soll eine TM angegeben werden, die eine gegebene Zeichenreihe aus $\{0, 1\}^+$ als Binärzahl interpretiert und zu dieser Zahl 1 addiert. Folgende Vorgehensweise bietet sich an:

- 1 Gehe ganz nach rechts bis ans Ende der Zahl. Dieses Ende kann durch das erste Auftreten eines Leerzeichens gefunden werden.
- 2 Gehe wieder nach links bis zur ersten 0 und ändere diese zu einer 1. Ersetze dabei auf dem Weg alle 1en durch 0.

Beispiel 97

Es soll eine TM angegeben werden, die eine gegebene Zeichenreihe aus $\{0, 1\}^+$ als Binärzahl interpretiert und zu dieser Zahl 1 addiert. Folgende Vorgehensweise bietet sich an:

- 1 Gehe ganz nach rechts bis ans Ende der Zahl. Dieses Ende kann durch das erste Auftreten eines Leerzeichens gefunden werden.
- 2 Gehe wieder nach links bis zur ersten 0 und ändere diese zu einer 1. Ersetze dabei auf dem Weg alle 1en durch 0.

Also:

$$\begin{array}{ll} \delta(q_0, 0) = (q_0, 0, R) & \delta(q_1, 1) = (q_1, 0, L) \\ \delta(q_0, 1) = (q_0, 1, R) & \delta(q_1, 0) = (q_f, 1, N) \\ \delta(q_0, \square) = (q_1, \square, L) & \delta(q_1, \square) = (q_f, 1, N) \end{array}$$

Damit ist $Q = \{q_0, q_1, q_f\}$ und $F = \{q_f\}$.