Informatik III

Albert-Ludwigs-Universität Freiburg
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

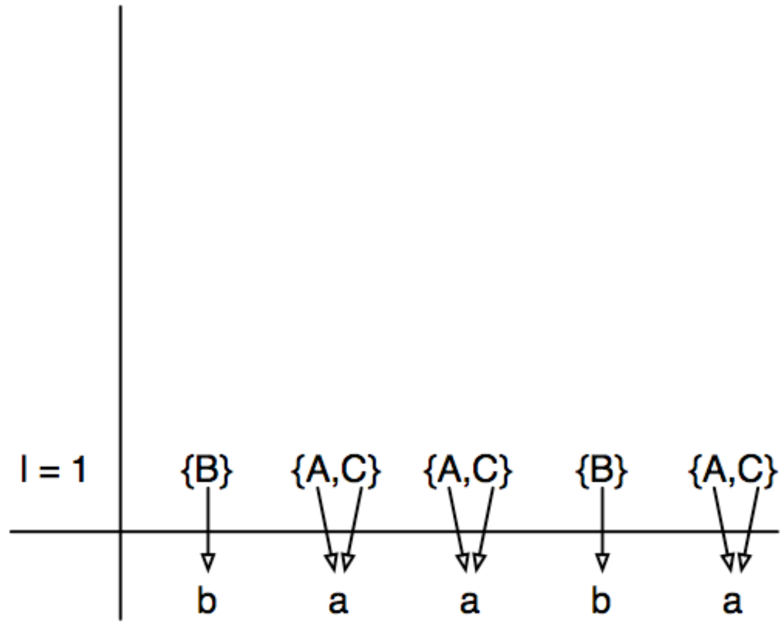
Christian Schindelhauer
Wintersemester 2006/07
7. Vorlesung
16.11.2006
schindel@informatik.uni-freiburg.de

CYK-Algorithmus

Bei Eingabe $w = w_1 \cdots w_n$:

1. Falls $w = \epsilon$ und $S \to \epsilon$ eine Regel ist, akzeptiere.
2. Für $i = 1$ bis n
3.   Für jede Variable v
4.   Teste, ob $v \to b$ eine Regel ist, wobei $b = w_i$
5.   Falls ja, füge v zu $T(i, i)$ hinzu.
6. Für $\ell = 2$ bis n
7.   Für $i = 1$ bis $n - \ell + 1$
8.   Setze $j = i + \ell - 1$
9.   Für $k = i$ bis $j - 1$
10.   Für jede Regel $v \to uz$
11.   Falls $u \in T(i, k)$ und $z \in T(k + 1, j)$, füge v zu $T(i, j)$ hinzu.
12. Falls S in $T(1, n)$ enthalten ist, akzeptiere. Sonst lehne ab.

l = Länge
i = Startindex
j = Schlussindex



G = (V , Σ, S, P), wobei V = {S, A, B, C } und Σ = {a, b}.

P = { S → AB, S → BC,
A → BA
B → CC,
C → AB,
A → a, B → b, C → a }

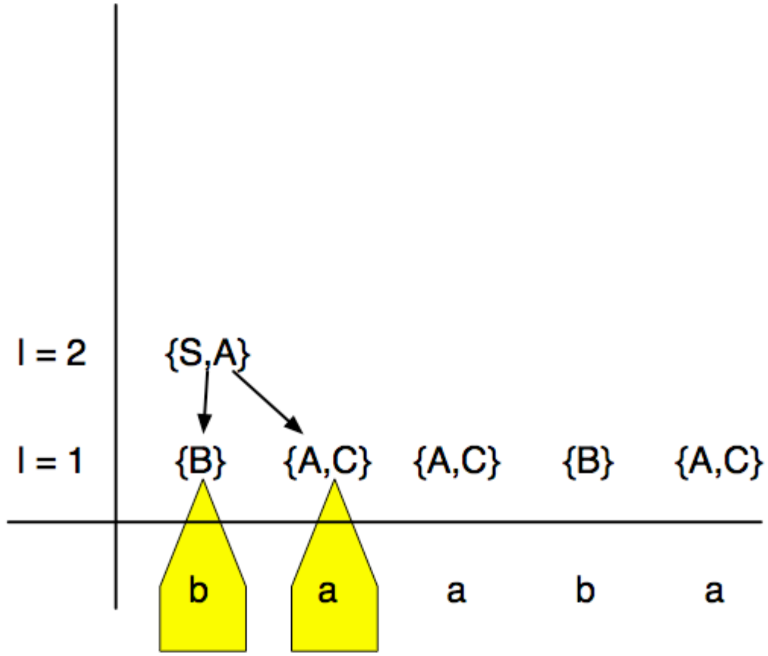
Cocke-Younger-Kasami-Algorithmus
Beispiel

CYK-Algorithmus

Bei Eingabe $w = w_1 \cdots w_n$:

1. Falls $w = \epsilon$ und $S \to \epsilon$ eine Regel ist, akzeptiere.
2. Für $i = 1$ bis n
3. Für jede Variable v
4. Teste, ob $v \to b$ eine Regel ist, wobei $b = w_i$
5. Falls ja, füge v zu $T(i,i)$ hinzu.
6. Für $\ell = 2$ bis n
7. Für $i = 1$ bis $n - \ell + 1$
8. Setze $j = i + \ell - 1$
9. Für $k = i$ bis $j - 1$
10. Für jede Regel $v \to uz$
11. Falls $u \in T(i,k)$ und $z \in T(k+1,j)$, füge v zu $T(i,j)$ hinzu.
12. Falls S in $T(1,n)$ enthalten ist, akzeptiere. Sonst lehne ab.

l = Länge
i = Startindex
j = Schlussindex



G = (V , Σ, S, P), wobei V = {S, A, B, C } und Σ = {a, b}.

P = { **S → AB, S → BC,**
A → BA
B → CC,
C → AB,
A → a, B → b, C → a }

Cocke-Younger-Kasami-Algorithmus
Beispiel

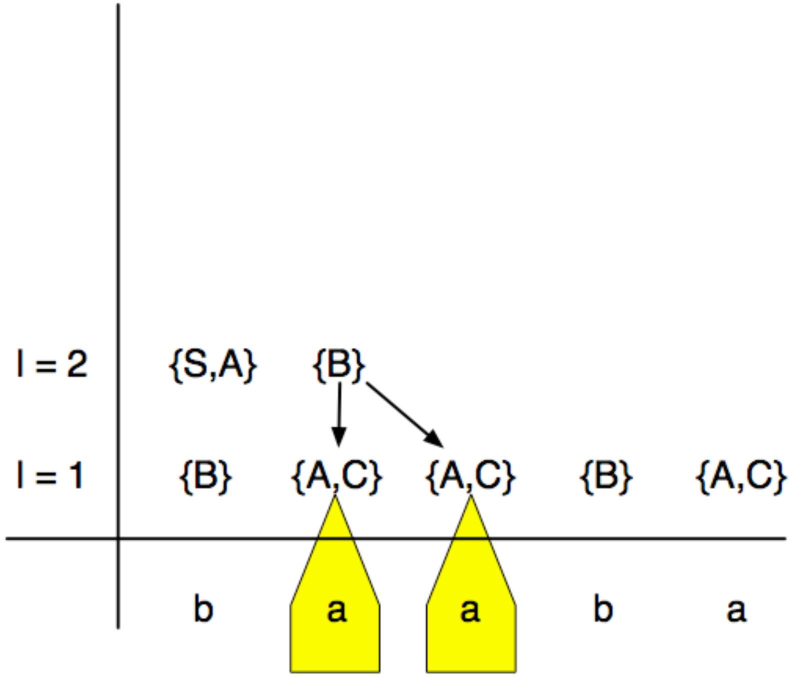
Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

CYK-Algorithmus

Bei Eingabe $w = w_1 \cdots w_n$:

1. Falls $w = \epsilon$ und $S \rightarrow \epsilon$ eine Regel ist, akzeptiere.
2. Für $i = 1$ bis n
3. Für jede Variable v
4. Teste, ob $v \rightarrow b$ eine Regel ist, wobei $b = w_i$
5. Falls ja, füge v zu $T(i, i)$ hinzu.
6. Für $\ell = 2$ bis n
7. Für $i = 1$ bis $n - \ell + 1$
8. Setze $j = i + \ell - 1$
9. Für $k = i$ bis $j - 1$
10. Für jede Regel $v \rightarrow uz$
11. Falls $u \in T(i, k)$ und $z \in T(k + 1, j)$, füge v zu $T(i, j)$ hinzu.
12. Falls S in $T(1, n)$ enthalten ist, akzeptiere. Sonst lehne ab.

l = Länge
i = Startindex
j = Schlussindex



G = (V , Σ, S, P), wobei V = {S, A, B, C } und Σ = {a, b}.

P = {
S → AB, S → BC,
A → BA
B → CC,
C → AB,
A → a, B → b, C → a }

Cocke-Younger-Kasami-Algorithmus
Beispiel

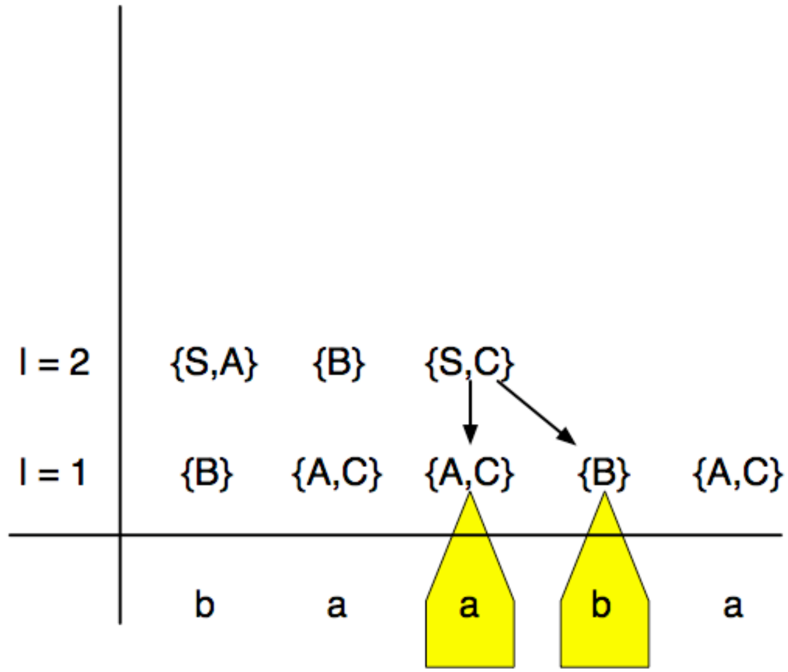
Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

CYK-Algorithmus

Bei Eingabe $w = w_1 \cdots w_n$:

1. Falls $w = \epsilon$ und $S \rightarrow \epsilon$ eine Regel ist, akzeptiere.
2. Für $i = 1$ bis n
3. Für jede Variable v
4. Teste, ob $v \rightarrow b$ eine Regel ist, wobei $b = w_i$
5. Falls ja, füge v zu $T(i,i)$ hinzu.
6. Für $\ell = 2$ bis n
7. Für $i = 1$ bis $n - \ell + 1$
8. Setze $j = i + \ell - 1$
9. Für $k = i$ bis $j - 1$
10. Für jede Regel $v \rightarrow uz$
11. Falls $u \in T(i,k)$ und $z \in T(k+1,j)$, füge v zu $T(i,j)$ hinzu.
12. Falls S in $T(1,n)$ enthalten ist, akzeptiere. Sonst lehne ab.

l = Länge
i = Startindex
j = Schlussindex



G = (V , Σ, S, P), wobei V = {S, A, B, C } und Σ = {a, b}.

P = { S → AB, S → BC,
A → BA
B → CC,
C → AB,
A → a, B → b, C → a }

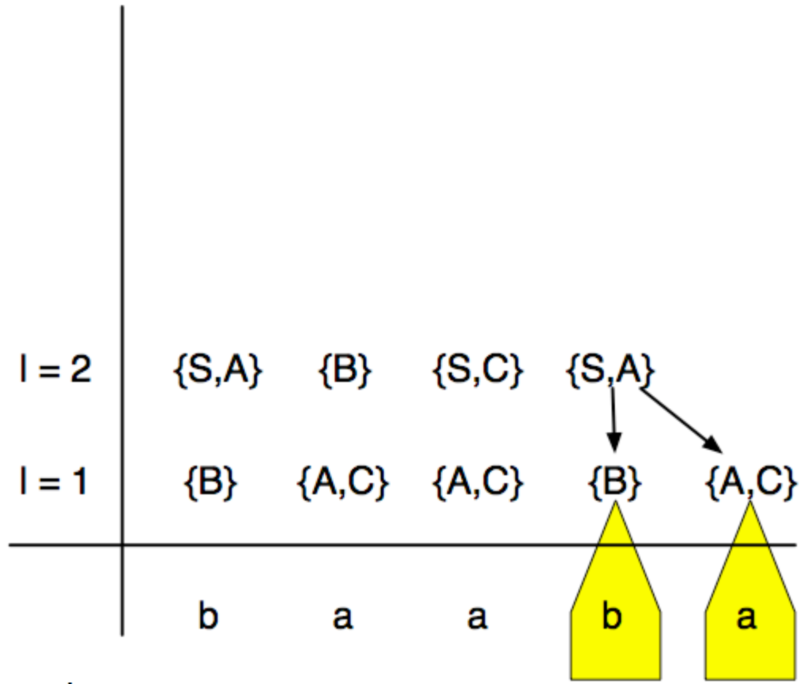
Cocke-Younger-Kasami-Algorithmus
Beispiel

CYK-Algorithmus

Bei Eingabe $w = w_1 \cdots w_n$:

1. Falls $w = \epsilon$ und $S \rightarrow \epsilon$ eine Regel ist, akzeptiere.
2. Für $i = 1$ bis n
3. Für jede Variable v
4. Teste, ob $v \rightarrow b$ eine Regel ist, wobei $b = w_i$
5. Falls ja, füge v zu $T(i,i)$ hinzu.
6. Für $\ell = 2$ bis n
7. Für $i = 1$ bis $n - \ell + 1$
8. Setze $j = i + \ell - 1$
9. Für $k = i$ bis $j - 1$
10. Für jede Regel $v \rightarrow uz$
11. Falls $u \in T(i,k)$ und $z \in T(k+1,j)$, füge v zu $T(i,j)$ hinzu.
12. Falls S in $T(1,n)$ enthalten ist, akzeptiere. Sonst lehne ab.

l = Länge
i = Startindex
j = Schlussindex



G = (V , Σ, S, P), wobei V = {S, A, B, C } und Σ = {a, b}.

P = { S → AB, S → BC,
A → BA
B → CC,
C → AB,
A → a, B → b, C → a }

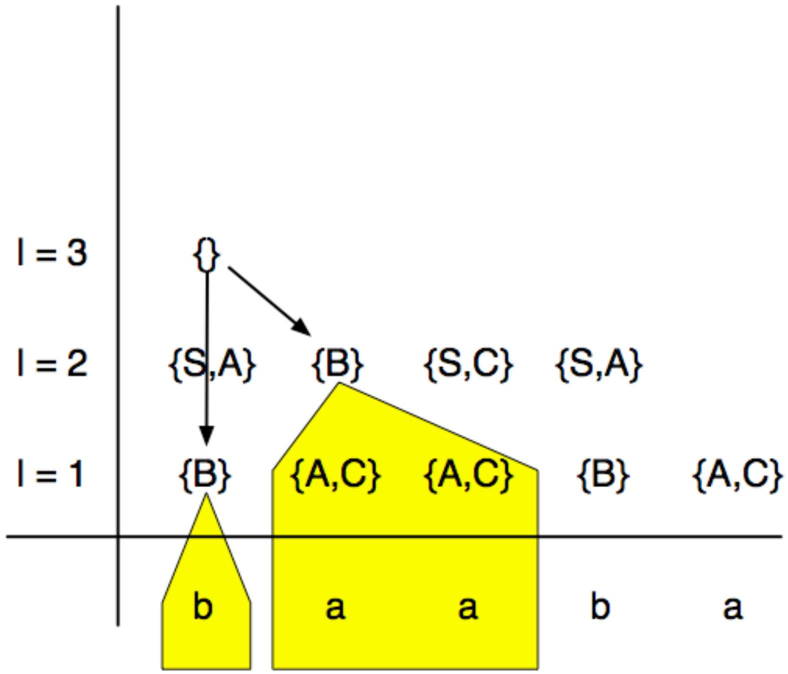
Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

CYK-Algorithmus

Bei Eingabe $w = w_1 \cdots w_n$:

1. Falls $w = \epsilon$ und $S \rightarrow \epsilon$ eine Regel ist, akzeptiere.
2. Für $i = 1$ bis n
3. Für jede Variable v
4. Teste, ob $v \rightarrow b$ eine Regel ist, wobei $b = w_i$
5. Falls ja, füge v zu $T(i,i)$ hinzu.
6. Für $\ell = 2$ bis n
7. Für $i = 1$ bis $n - \ell + 1$
8. Setze $j = i + \ell - 1$
9. Für $k = i$ bis $j - 1$
10. Für jede Regel $v \rightarrow uz$
11. Falls $u \in T(i,k)$ und $z \in T(k+1,j)$, füge v zu $T(i,j)$ hinzu.
12. Falls S in $T(1,n)$ enthalten ist, akzeptiere. Sonst lehne ab.

l = Länge
i = Startindex
j = Schlussindex



G = (V , Σ, S, P), wobei V = {S, A, B, C } und Σ = {a, b}.

P = { S → AB, S → BC,
A → BA
B → CC,
C → AB,
A → a, B → b, C → a }

Cocke-Younger-Kasami-Algorithmus
Beispiel

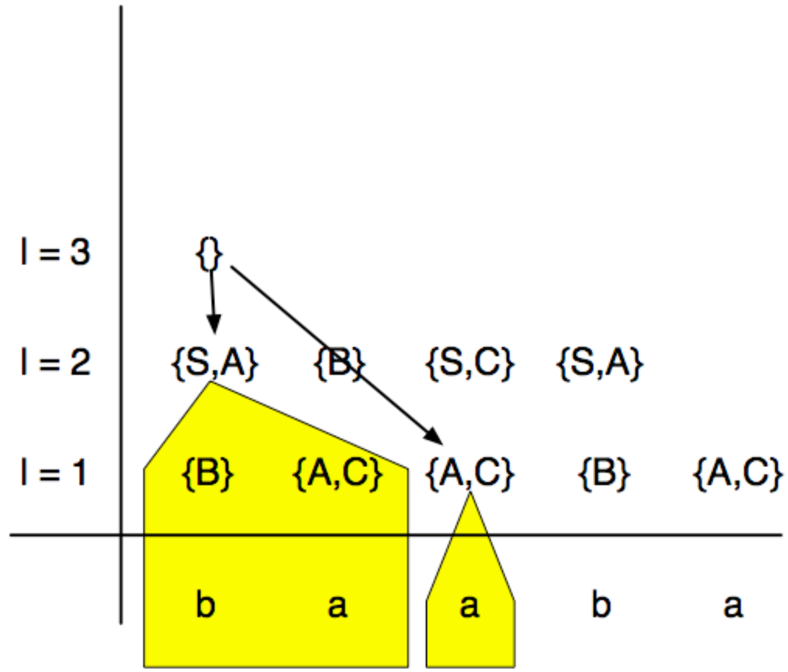
Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

CYK-Algorithmus

Bei Eingabe $w = w_1 \cdots w_n$:

1. Falls $w = \epsilon$ und $S \rightarrow \epsilon$ eine Regel ist, akzeptiere.
2. Für $i = 1$ bis n
3.   Für jede Variable v
4.   Teste, ob $v \rightarrow b$ eine Regel ist, wobei $b = w_i$
5.   Falls ja, füge v zu $T(i, i)$ hinzu.
6. Für $\ell = 2$ bis n
7.   Für $i = 1$ bis $n - \ell + 1$
8.   Setze $j = i + \ell - 1$
9.   Für $k = i$ bis $j - 1$
10.   Für jede Regel $v \rightarrow uz$
11.   Falls $u \in T(i, k)$ und $z \in T(k + 1, j)$, füge v zu $T(i, j)$ hinzu.
12. Falls S in $T(1, n)$ enthalten ist, akzeptiere. Sonst lehne ab.

l = Länge
i = Startindex
j = Schlussindex



G = (V , Σ, S, P), wobei V = {S, A, B, C } und Σ = {a, b}.

P = {
S → AB, S → BC,
A → BA
B → CC,
C → AB,
A → a, B → b, C → a }

Cocke-Younger-Kasami-Algorithmus
Beispiel

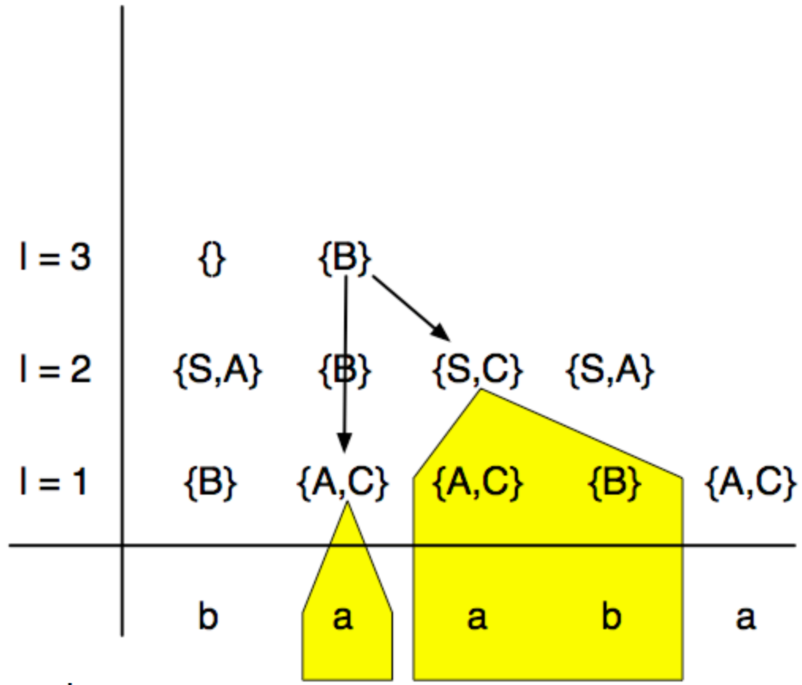
Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

CYK-Algorithmus

Bei Eingabe $w = w_1 \cdots w_n$:

1. Falls $w = \epsilon$ und $S \rightarrow \epsilon$ eine Regel ist, akzeptiere.
2. Für $i = 1$ bis n
3. Für jede Variable v
4. Teste, ob $v \rightarrow b$ eine Regel ist, wobei $b = w_i$
5. Falls ja, füge v zu $T(i, i)$ hinzu.
6. Für $\ell = 2$ bis n
7. Für $i = 1$ bis $n - \ell + 1$
8. Setze $j = i + \ell - 1$
9. Für $k = i$ bis $j - 1$
10. Für jede Regel $v \rightarrow uz$
11. Falls $u \in T(i, k)$ und $z \in T(k + 1, j)$, füge v zu $T(i, j)$ hinzu.
12. Falls S in $T(1, n)$ enthalten ist, akzeptiere. Sonst lehne ab.

l = Länge
i = Startindex
j = Schlussindex



G = (V , Σ, S, P), wobei V = {S, A, B, C } und Σ = {a, b}.

P = {
S → AB, S → BC,
A → BA
B → CC,
C → AB,
A → a, B → b, C → a }

Cocke-Younger-Kasami-Algorithmus
Beispiel

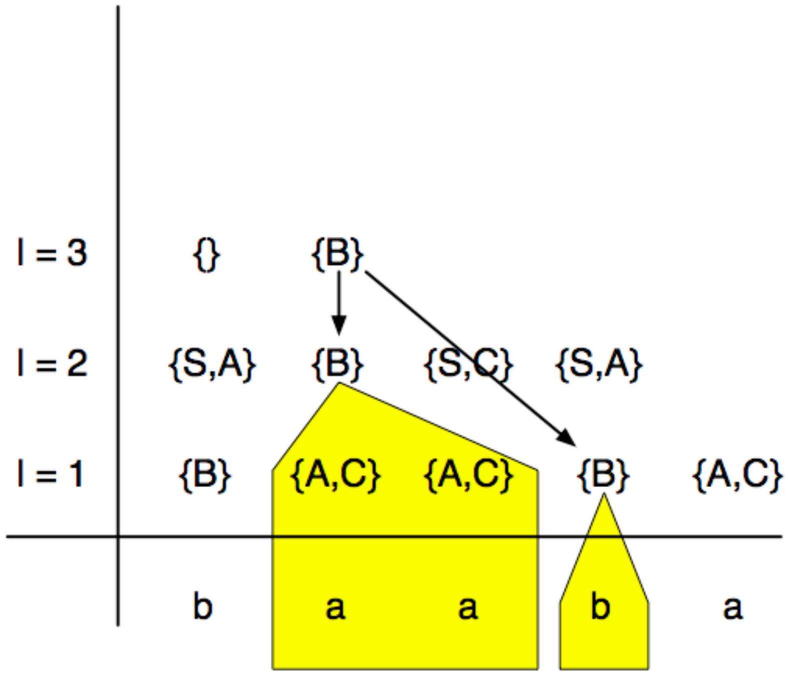
Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

CYK-Algorithmus

Bei Eingabe $w = w_1 \cdots w_n$:

1. Falls $w = \epsilon$ und $S \rightarrow \epsilon$ eine Regel ist, akzeptiere.
2. Für $i = 1$ bis n
3. Für jede Variable v
4. Teste, ob $v \rightarrow b$ eine Regel ist, wobei $b = w_i$
5. Falls ja, füge v zu $T(i, i)$ hinzu.
6. Für $\ell = 2$ bis n
7. Für $i = 1$ bis $n - \ell + 1$
8. Setze $j = i + \ell - 1$
9. Für $k = i$ bis $j - 1$
10. Für jede Regel $v \rightarrow uz$
11. Falls $u \in T(i, k)$ und $z \in T(k + 1, j)$, füge v zu $T(i, j)$ hinzu.
12. Falls S in $T(1, n)$ enthalten ist, akzeptiere. Sonst lehne ab.

l = Länge
i = Startindex
j = Schlussindex



G = (V , Σ, S, P), wobei V = {S, A, B, C } und Σ = {a, b}.

P = { S → AB, S → BC,
A → BA
B → CC,
C → AB,
A → a, B → b, C → a }

Cocke-Younger-Kasami-Algorithmus
Beispiel

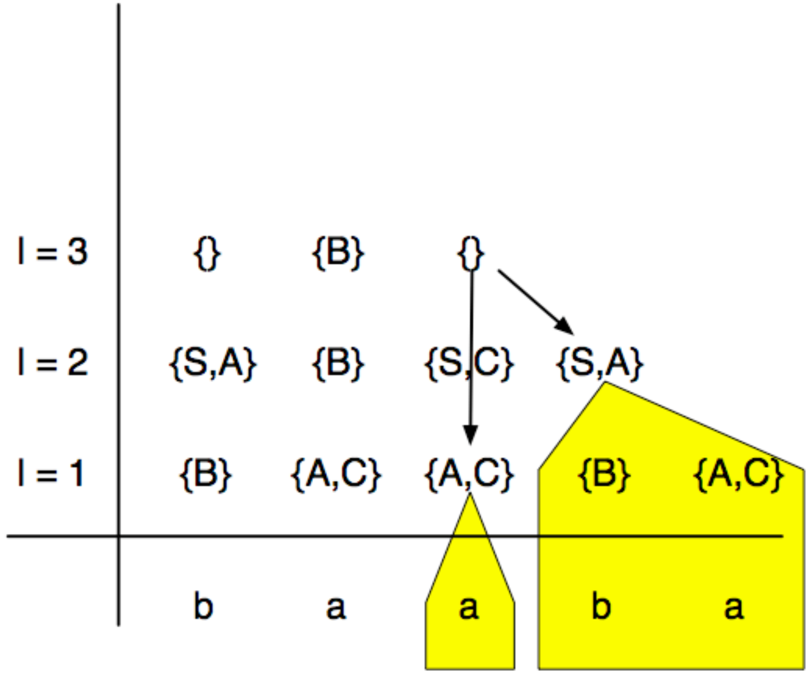
Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

CYK-Algorithmus

Bei Eingabe $w = w_1 \cdots w_n$:

1. Falls $w = \epsilon$ und $S \rightarrow \epsilon$ eine Regel ist, akzeptiere.
2. Für $i = 1$ bis n
3.   Für jede Variable v
4.   Teste, ob $v \rightarrow b$ eine Regel ist, wobei $b = w_i$
5.   Falls ja, füge v zu $T(i, i)$ hinzu.
6. Für $\ell = 2$ bis n
7.   Für $i = 1$ bis $n - \ell + 1$
8.   Setze $j = i + \ell - 1$
9.   Für $k = i$ bis $j - 1$
10.   Für jede Regel $v \rightarrow uz$
11.   Falls $u \in T(i, k)$ und $z \in T(k + 1, j)$, füge v zu $T(i, j)$ hinzu.
12. Falls S in $T(1, n)$ enthalten ist, akzeptiere. Sonst lehne ab.

l = Länge
i = Startindex
j = Schlussindex



G = (V , Σ, S, P), wobei V = {S, A, B, C } und Σ = {a, b}.

P = {   **S → AB, S → BC,**
A → BA
B → CC,
C → AB,
A → a, B → b, C → a }

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

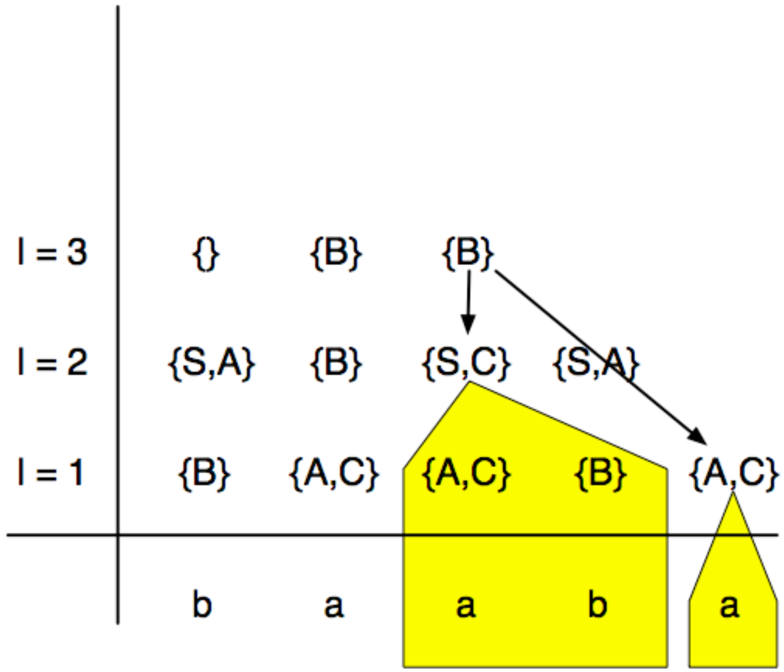
Cocke-Younger-Kasami-Algorithmus
Beispiel

CYK-Algorithmus

Bei Eingabe $w = w_1 \cdots w_n$:

1. Falls $w = \epsilon$ und $S \to \epsilon$ eine Regel ist, akzeptiere.
2. Für $i = 1$ bis n
3. Für jede Variable v
4. Teste, ob $v \to b$ eine Regel ist, wobei $b = w_i$
5. Falls ja, füge v zu $T(i,i)$ hinzu.
6. Für $\ell = 2$ bis n
7. Für $i = 1$ bis $n - \ell + 1$
8. Setze $j = i + \ell - 1$
9. Für $k = i$ bis $j - 1$
10. Für jede Regel $v \to uz$
11. Falls $u \in T(i,k)$ und $z \in T(k+1,j)$, füge v zu $T(i,j)$ hinzu.
12. Falls S in $T(1,n)$ enthalten ist, akzeptiere. Sonst lehne ab.

l = Länge
i = Startindex
j = Schlussindex



G = (V , Σ, S, P), wobei V = {S, A, B, C } und Σ = {a, b}.

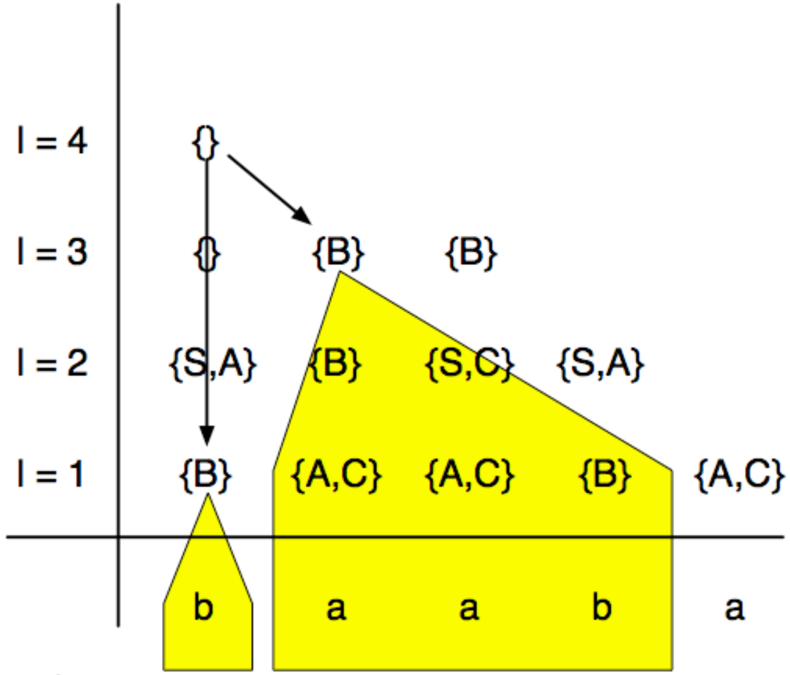
P = { S → AB, S → BC,
A → BA
B → CC,
C → AB,
A → a, B → b, C → a }

CYK-Algorithmus

Bei Eingabe $w = w_1 \cdots w_n$:

1. Falls $w = \epsilon$ und $S \to \epsilon$ eine Regel ist, akzeptiere.
2. Für $i = 1$ bis n
3. Für jede Variable v
4. Teste, ob $v \to b$ eine Regel ist, wobei $b = w_i$
5. Falls ja, füge v zu $T(i,i)$ hinzu.
6. Für $\ell = 2$ bis n
7. Für $i = 1$ bis $n - \ell + 1$
8. Setze $j = i + \ell - 1$
9. Für $k = i$ bis $j - 1$
10. Für jede Regel $v \to uz$
11. Falls $u \in T(i,k)$ und $z \in T(k+1,j)$, füge v zu $T(i,j)$ hinzu.
12. Falls S in $T(1,n)$ enthalten ist, akzeptiere. Sonst lehne ab.

l = Länge
i = Startindex
j = Schlussindex



G = (V , Σ, S, P), wobei V = {S, A, B, C } und Σ = {a, b}.

P = { S → AB, S → BC,
A → BA
B → CC,
C → AB,
A → a, B → b, C → a }

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

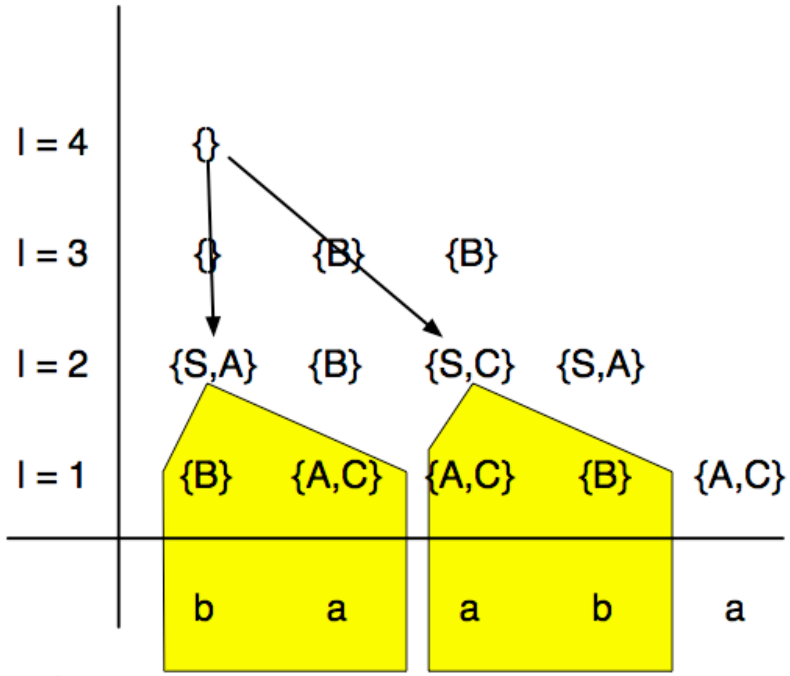
Cocke-Younger-Kasami-Algorithmus Beispiel

CYK-Algorithmus

Bei Eingabe $w = w_1 \cdots w_n$:

1. Falls $w = \epsilon$ und $S \to \epsilon$ eine Regel ist, akzeptiere.
2. Für $i = 1$ bis n
3. Für jede Variable v
4. Teste, ob $v \to b$ eine Regel ist, wobei $b = w_i$
5. Falls ja, füge v zu $T(i,i)$ hinzu.
6. Für $\ell = 2$ bis n
7. Für $i = 1$ bis $n - \ell + 1$
8. Setze $j = i + \ell - 1$
9. Für $k = i$ bis $j - 1$
10. Für jede Regel $v \to uz$
11. Falls $u \in T(i,k)$ und $z \in T(k+1,j)$, füge v zu $T(i,j)$ hinzu.
12. Falls S in $T(1,n)$ enthalten ist, akzeptiere. Sonst lehne ab.

l = Länge
i = Startindex
j = Schlussindex



G = (V , Σ, S, P), wobei V = {S, A, B, C } und Σ = {a, b}.

P = { S → AB, S → BC,
A → BA
B → CC,
C → AB,
A → a, B → b, C → a }

Cocke-Younger-Kasami-Algorithmus
Beispiel

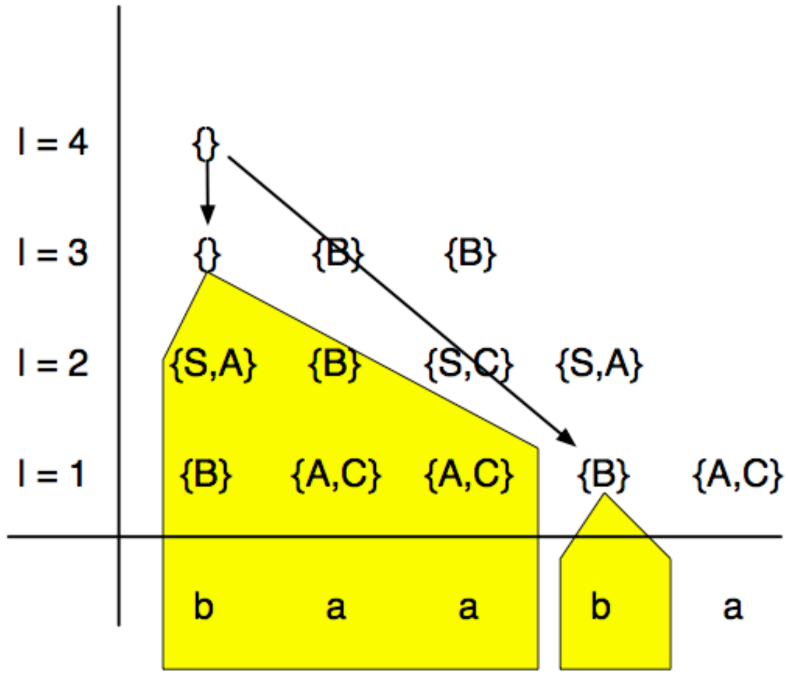
Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

CYK-Algorithmus

Bei Eingabe $w = w_1 \cdots w_n$:

1. Falls $w = \epsilon$ und $S \to \epsilon$ eine Regel ist, akzeptiere.
2. Für $i = 1$ bis n
3. Für jede Variable v
4. Teste, ob $v \to b$ eine Regel ist, wobei $b = w_i$
5. Falls ja, füge v zu $T(i,i)$ hinzu.
6. Für $\ell = 2$ bis n
7. Für $i = 1$ bis $n - \ell + 1$
8. Setze $j = i + \ell - 1$
9. Für $k = i$ bis $j - 1$
10. Für jede Regel $v \to uz$
11. Falls $u \in T(i,k)$ und $z \in T(k+1,j)$, füge v zu $T(i,j)$ hinzu.
12. Falls S in $T(1,n)$ enthalten ist, akzeptiere. Sonst lehne ab.

l = Länge
i = Startindex
j = Schlussindex



G = (V , Σ, S, P), wobei V = {S, A, B, C } und Σ = {a, b}.

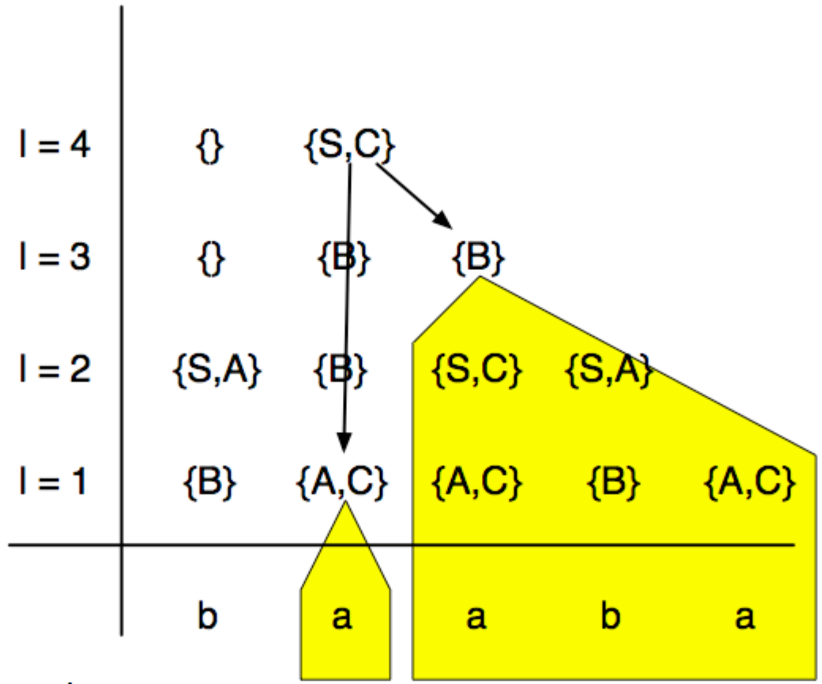
P = { S → AB, S → BC,
A → BA
B → CC,
C → AB,
A → a, B → b, C → a }

CYK-Algorithmus

Bei Eingabe $w = w_1 \cdots w_n$:

1. Falls $w = \epsilon$ und $S \to \epsilon$ eine Regel ist, akzeptiere.
2. Für $i = 1$ bis n
3. Für jede Variable v
4. Teste, ob $v \to b$ eine Regel ist, wobei $b = w_i$
5. Falls ja, füge v zu $T(i,i)$ hinzu.
6. Für $\ell = 2$ bis n
7. Für $i = 1$ bis $n - \ell + 1$
8. Setze $j = i + \ell - 1$
9. Für $k = i$ bis $j - 1$
10. Für jede Regel $v \to uz$
11. Falls $u \in T(i,k)$ und $z \in T(k+1,j)$, füge v zu $T(i,j)$ hinzu.
12. Falls S in $T(1,n)$ enthalten ist, akzeptiere. Sonst lehne ab.

l = Länge
i = Startindex
j = Schlussindex



G = (V , Σ, S, P), wobei V = {S, A, B, C } und Σ = {a, b}.

P = { S → AB, S → BC,
A → BA
B → CC,
C → AB,
A → a, B → b, C → a }

Cocke-Younger-Kasami-Algorithmus
Beispiel

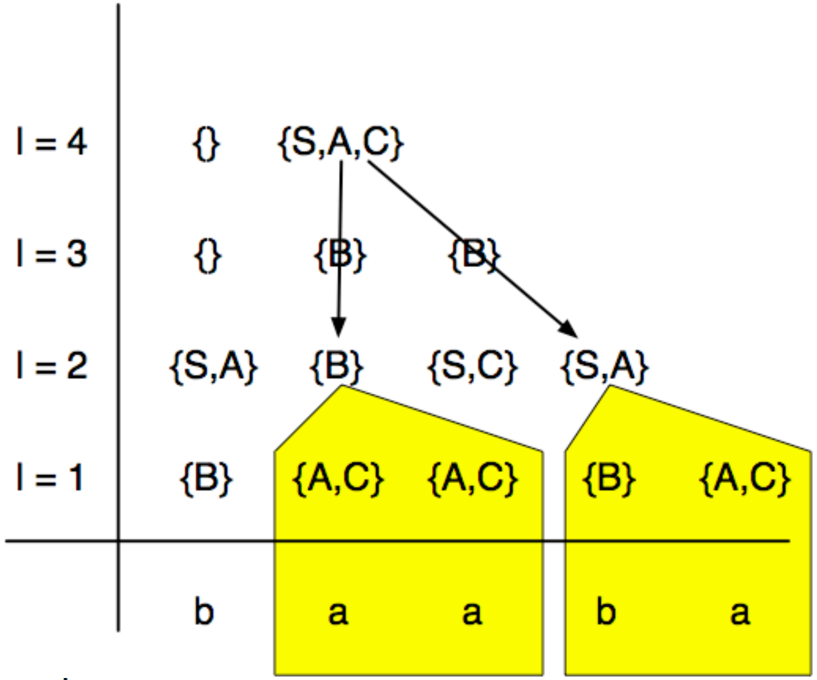
Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

CYK-Algorithmus

Bei Eingabe $w = w_1 \cdots w_n$:

1. Falls $w = \epsilon$ und $S \to \epsilon$ eine Regel ist, akzeptiere.
2. Für $i = 1$ bis n
3. Für jede Variable v
4. Teste, ob $v \to b$ eine Regel ist, wobei $b = w_i$
5. Falls ja, füge v zu $T(i,i)$ hinzu.
6. Für $\ell = 2$ bis n
7. Für $i = 1$ bis $n - \ell + 1$
8. Setze $j = i + \ell - 1$
9. Für $k = i$ bis $j - 1$
10. Für jede Regel $v \to uz$
11. Falls $u \in T(i,k)$ und $z \in T(k+1,j)$, füge v zu $T(i,j)$ hinzu.
12. Falls S in $T(1,n)$ enthalten ist, akzeptiere. Sonst lehne ab.

l = Länge
i = Startindex
j = Schlussindex



G = (V , Σ, S, P), wobei V = {S, A, B, C } und Σ = {a, b}.

P = { S → AB, S → BC,
A → BA
B → CC,
C → AB,
A → a, B → b, C → a }

Cocke-Younger-Kasami-Algorithmus
Beispiel

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

CYK-Algorithmus

Bei Eingabe $w = w_1 \cdots w_n$:

1. Falls $w = \epsilon$ und $S \to \epsilon$ eine Regel ist, akzeptiere.
2. Für $i = 1$ bis n
3. Für jede Variable v
4. Teste, ob $v \to b$ eine Regel ist, wobei $b = w_i$
5. Falls ja, füge v zu $T(i,i)$ hinzu.
6. Für $\ell = 2$ bis n
7. Für $i = 1$ bis $n - \ell + 1$
8. Setze $j = i + \ell - 1$
9. Für $k = i$ bis $j - 1$
10. Für jede Regel $v \to uz$
11. Falls $u \in T(i,k)$ und $z \in T(k+1,j)$, füge v zu $T(i,j)$ hinzu.
12. Falls S in $T(1,n)$ enthalten ist, akzeptiere. Sonst lehne ab.

l = Länge
i = Startindex
j = Schlussindex

l = 4	{}	{S,A,C}			
l = 3	{}	{B}	{B}		
l = 2	{S,A}	{B}	{S,C}	{S,A}	
l = 1	{B}	{A,C}	{A,C}	{B}	{A,C}
	b	a	a	b	a

G = (V , Σ, S, P), wobei V = {S, A, B, C } und Σ = {a, b}.

P = { S → AB, S → BC,
A → BA
B → CC,
C → AB,
A → a, B → b, C → a }

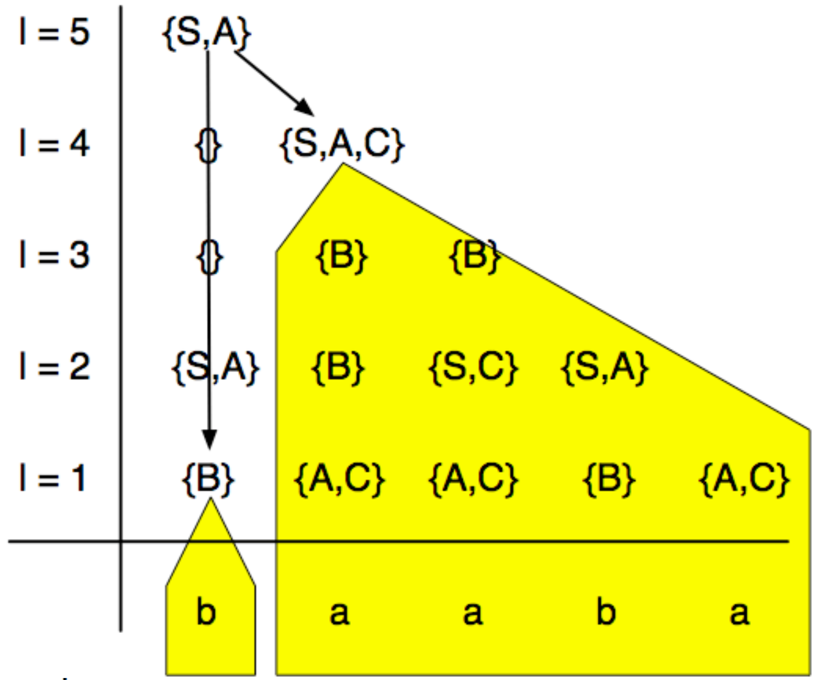
Cocke-Younger-Kasami-Algorithmus Beispiel

CYK-Algorithmus

Bei Eingabe $w = w_1 \cdots w_n$:

1. Falls $w = \epsilon$ und $S \to \epsilon$ eine Regel ist, akzeptiere.
2. Für $i = 1$ bis n
3. Für jede Variable v
4. Teste, ob $v \to b$ eine Regel ist, wobei $b = w_i$
5. Falls ja, füge v zu $T(i,i)$ hinzu.
6. Für $\ell = 2$ bis n
7. Für $i = 1$ bis $n - \ell + 1$
8. Setze $j = i + \ell - 1$
9. Für $k = i$ bis $j - 1$
10. Für jede Regel $v \to uz$
11. Falls $u \in T(i,k)$ und $z \in T(k+1,j)$, füge v zu $T(i,j)$ hinzu.
12. Falls S in $T(1,n)$ enthalten ist, akzeptiere. Sonst lehne ab.

l = Länge
i = Startindex
j = Schlussindex



G = (V , Σ, S, P), wobei V = {S, A, B, C } und Σ = {a, b}.

P = { S → AB, S → BC,
A → BA
B → CC,
C → AB,
A → a, B → b, C → a }

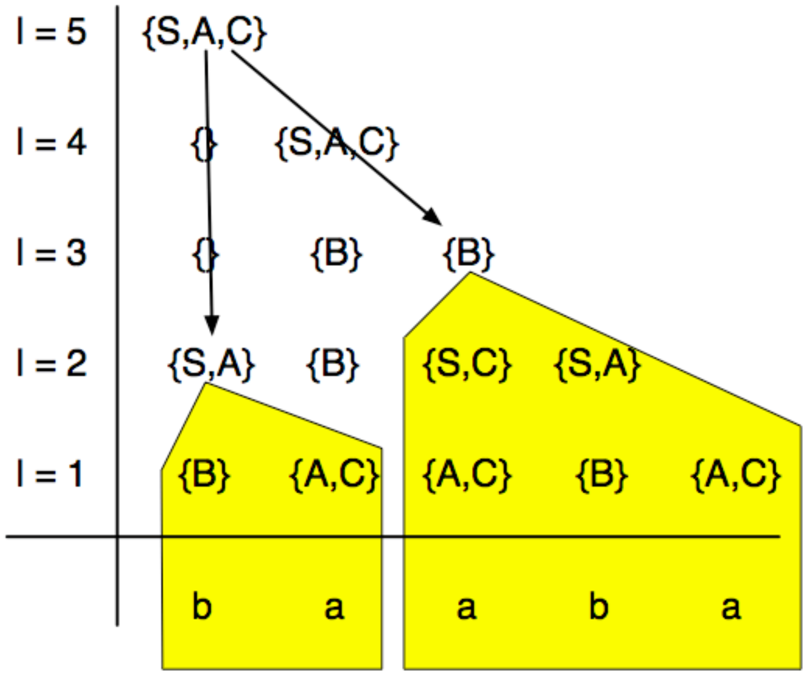
Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

CYK-Algorithmus

Bei Eingabe $w = w_1 \cdots w_n$:

1. Falls $w = \epsilon$ und $S \to \epsilon$ eine Regel ist, akzeptiere.
2. Für $i = 1$ bis n
3. Für jede Variable v
4. Teste, ob $v \to b$ eine Regel ist, wobei $b = w_i$
5. Falls ja, füge v zu $T(i,i)$ hinzu.
6. Für $\ell = 2$ bis n
7. Für $i = 1$ bis $n - \ell + 1$
8. Setze $j = i + \ell - 1$
9. Für $k = i$ bis $j - 1$
10. Für jede Regel $v \to uz$
11. Falls $u \in T(i,k)$ und $z \in T(k+1,j)$, füge v zu $T(i,j)$ hinzu.
12. Falls S in $T(1,n)$ enthalten ist, akzeptiere. Sonst lehne ab.

l = Länge
i = Startindex
j = Schlussindex



G = (V , Σ, S, P), wobei V = {S, A, B, C } und Σ = {a, b}.

P = { S → AB, S → BC,
A → BA
B → CC,
C → AB,
A → a, B → b, C → a }

Cocke-Younger-Kasami-Algorithmus
Beispiel

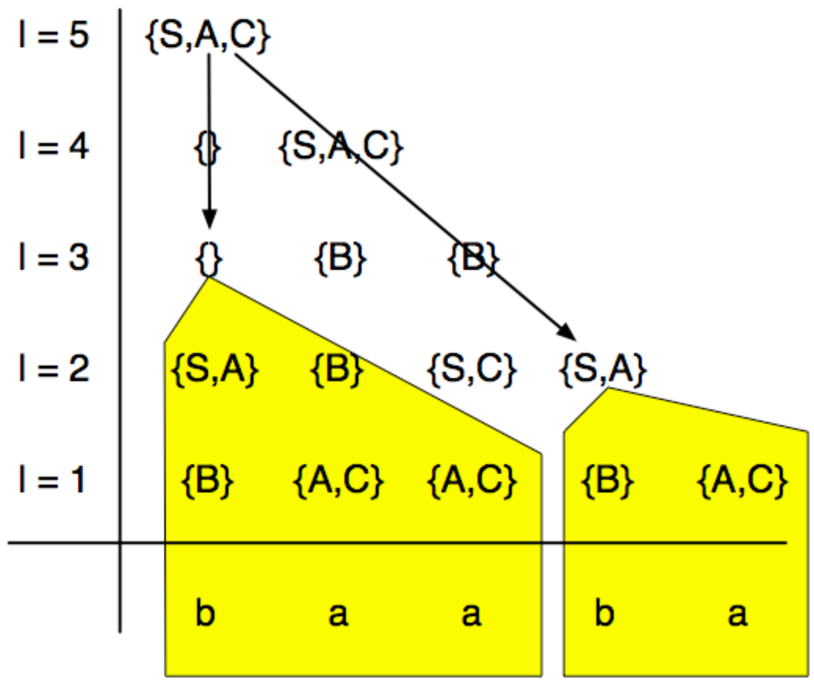
Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

CYK-Algorithmus

Bei Eingabe $w = w_1 \cdots w_n$:

1. Falls $w = \epsilon$ und $S \to \epsilon$ eine Regel ist, akzeptiere.
2. Für $i = 1$ bis n
3. Für jede Variable v
4. Teste, ob $v \to b$ eine Regel ist, wobei $b = w_i$
5. Falls ja, füge v zu $T(i,i)$ hinzu.
6. Für $\ell = 2$ bis n
7. Für $i = 1$ bis $n - \ell + 1$
8. Setze $j = i + \ell - 1$
9. Für $k = i$ bis $j - 1$
10. Für jede Regel $v \to uz$
11. Falls $u \in T(i,k)$ und $z \in T(k+1,j)$, füge v zu $T(i,j)$ hinzu.
12. Falls S in $T(1,n)$ enthalten ist, akzeptiere. Sonst lehne ab.

l = Länge
i = Startindex
j = Schlussindex



G = (V , Σ, S, P), wobei V = {S, A, B, C } und Σ = {a, b}.

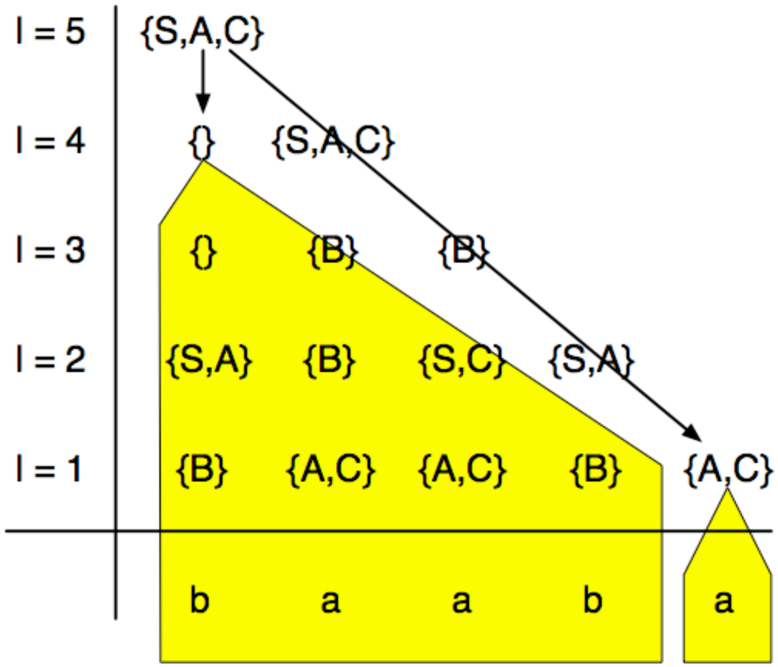
P = { S → AB, S → BC,
A → BA
B → CC,
C → AB,
A → a, B → b, C → a }

CYK-Algorithmus

Bei Eingabe $w = w_1 \cdots w_n$:

1. Falls $w = \epsilon$ und $S \to \epsilon$ eine Regel ist, akzeptiere.
2. Für $i = 1$ bis n
3. Für jede Variable v
4. Teste, ob $v \to b$ eine Regel ist, wobei $b = w_i$
5. Falls ja, füge v zu $T(i,i)$ hinzu.
6. Für $\ell = 2$ bis n
7. Für $i = 1$ bis $n - \ell + 1$
8. Setze $j = i + \ell - 1$
9. Für $k = i$ bis $j - 1$
10. Für jede Regel $v \to uz$
11. Falls $u \in T(i,k)$ und $z \in T(k+1,j)$, füge v zu $T(i,j)$ hinzu.
12. Falls S in $T(1,n)$ enthalten ist, akzeptiere. Sonst lehne ab.

l = Länge
i = Startindex
j = Schlussindex



G = (V , Σ, S, P), wobei V = {S, A, B, C } und Σ = {a, b}.

P = { S → AB, S → BC,
A → BA
B → CC,
C → AB,
A → a, B → b, C → a }

Cocke-Younger-Kasami-Algorithmus
Beispiel

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

CYK-Algorithmus

Bei Eingabe $w = w_1 \cdots w_n$:

1. Falls $w = \epsilon$ und $S \to \epsilon$ eine Regel ist, akzeptiere.
2. Für $i = 1$ bis n
3. Für jede Variable v
4. Teste, ob $v \to b$ eine Regel ist, wobei $b = w_i$
5. Falls ja, füge v zu $T(i,i)$ hinzu.
6. Für $\ell = 2$ bis n
7. Für $i = 1$ bis $n - \ell + 1$
8. Setze $j = i + \ell - 1$
9. Für $k = i$ bis $j - 1$
10. Für jede Regel $v \to uz$
11. Falls $u \in T(i,k)$ und $z \in T(k+1,j)$, füge v zu $T(i,j)$ hinzu.
12. Falls S in $T(1,n)$ enthalten ist, akzeptiere. Sonst lehne ab.

l = Länge
i = Startindex
j = Schlussindex

l = 5	{S,A,C}				
l = 4	{}	{S,A,C}			
l = 3	{}	{B}	{B}		
l = 2	{S,A}	{B}	{S,C}	{S,A}	
l = 1	{B}	{A,C}	{A,C}	{B}	{A,C}
	b	a	a	b	a

G = (V , Σ, S, P), wobei V = {S, A, B, C } und Σ = {a, b}.

P = { S → AB, S → BC,
A → BA
B → CC,
C → AB,
A → a, B → b, C → a }

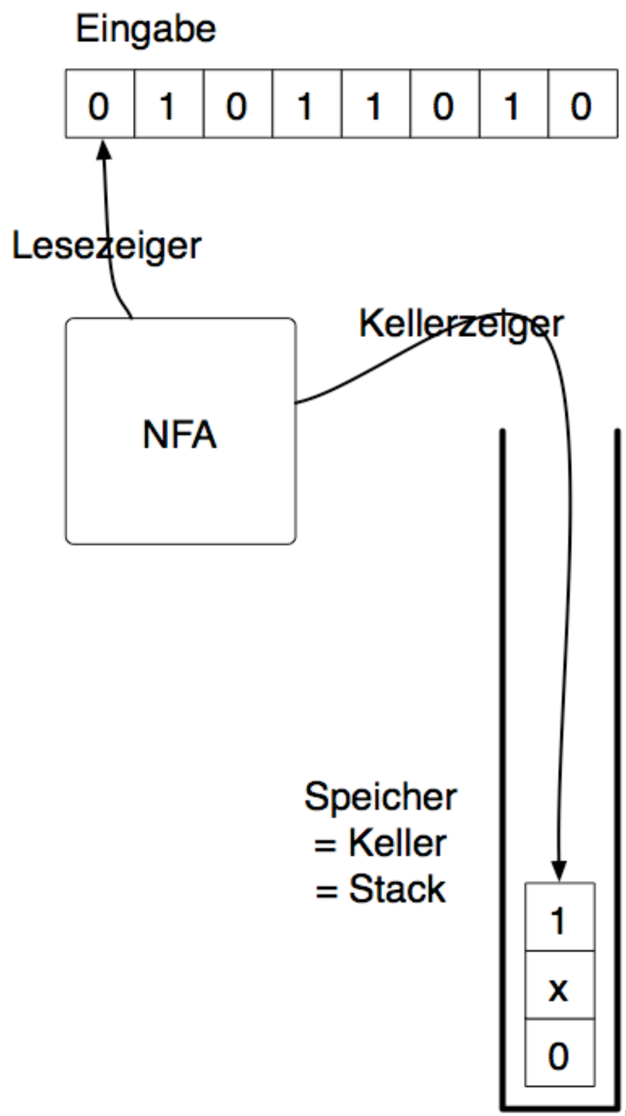
baaba ist in der Sprache

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

Keller-automaten

Prinzip des Kellerautomats
Push-Down-Automaton (PDA)

- **Ein Kellerautomat vereinigt einen**
  - NFA mit einem
  - Keller (Stack)
- **Der Keller kann potenziell beliebig viel Zeichen speichern**
- **Zugriff ist eingeschränkt:**
  - Pop: Auslesen des obersten Zeichens (inklusive Entfernen)
  - Push: Hineinlegen eines Zeichens
- **Auf den Übergängen des NFA wird der Zugriff auf den Keller festgelegt**
  - zusätzlich zum aktuellen Zeichen der Eingabe,
  - die weiterhin von links nach rechts gelesen wird.



Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

Keller-Automat: Formale Definition

- **Definition**
  - Ein Kellerautomat (pushdown automaton - PDA) wird durch ein Sechser-Tupel $(Q, \Sigma, \Gamma, \delta, q_0, F)$, wobei Q, Σ, Γ, F endliche Mengen sind und
  1. Q ist die Menge der Zustände
  2. Σ ist das Eingabealphabet
  3. Γ ist das Kelleralphabet
  4. $\delta : Q \times \Sigma_\varepsilon \times \Gamma_\varepsilon \rightarrow \mathbf{P}(Q \times \Gamma_\varepsilon)$ ist die Übergangsfunktion
  5. q_0 ist der Startzustand
  6. $F \subseteq Q$ ist die Menge der akzeptierenden Zustände

- **Ein PDA akzeptiert die Eingabe w,**
  - wenn es eine Darstellung $w = w_1 w_2 \ldots w_m$ mit $w_i \in \Sigma_\varepsilon$ gibt
  - wenn es eine Zustandsfolge $q = r_0 r_2 \ldots r_m$ mit $s_i \in Q$ gibt
  - wenn es Zeichenketten $s_0, s_1, \ldots, s_m \in \Gamma_\varepsilon^*$ gibt, so dass

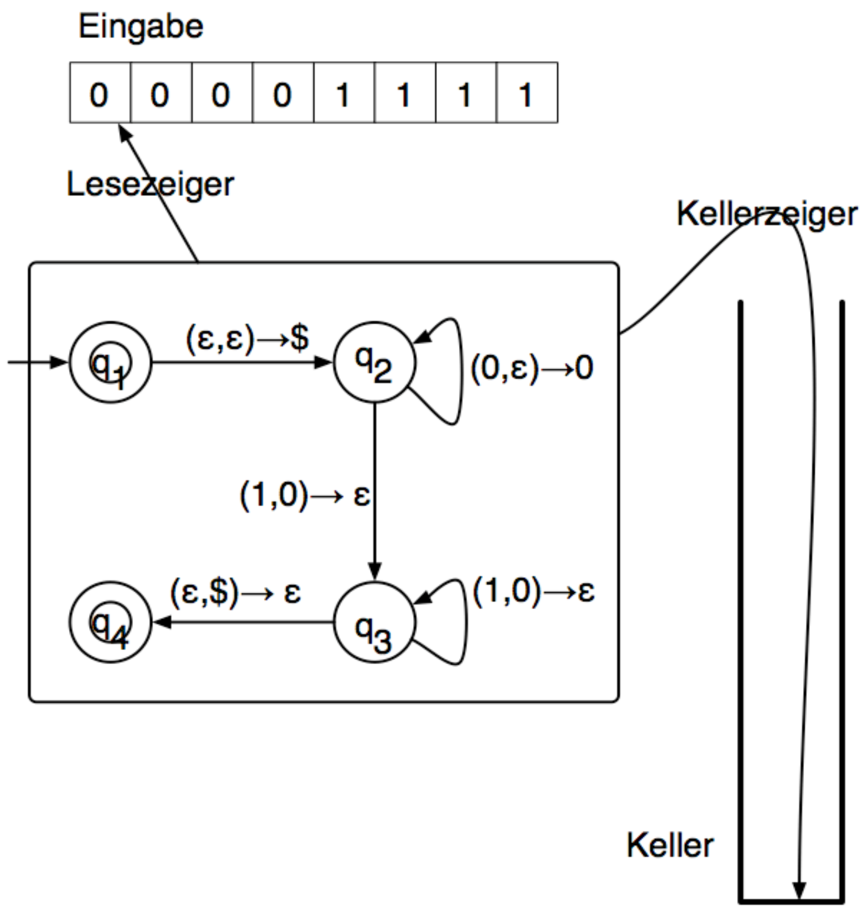
1. $r_0 = q_0$ und $s_0 = \varepsilon$
  - Startzustand mit leeren Keller

2. Für $i = 0, \ldots, m-1$ gilt:
  - $(r_{i+1}, b) \in \delta(r_i, w_{i+1}, a)$, wobei
  - $s_i = at$ und $s_{i+1} = bt$
    für passende $a, b \in \Gamma_\varepsilon$, $t \in \Gamma_\varepsilon^*$
  - Übergang mit Kellerverhalten:
    - Lies a, Schreibe b

3. $r_m \in F$
  - Ein akzeptierender Zustand erscheint als Endzustand

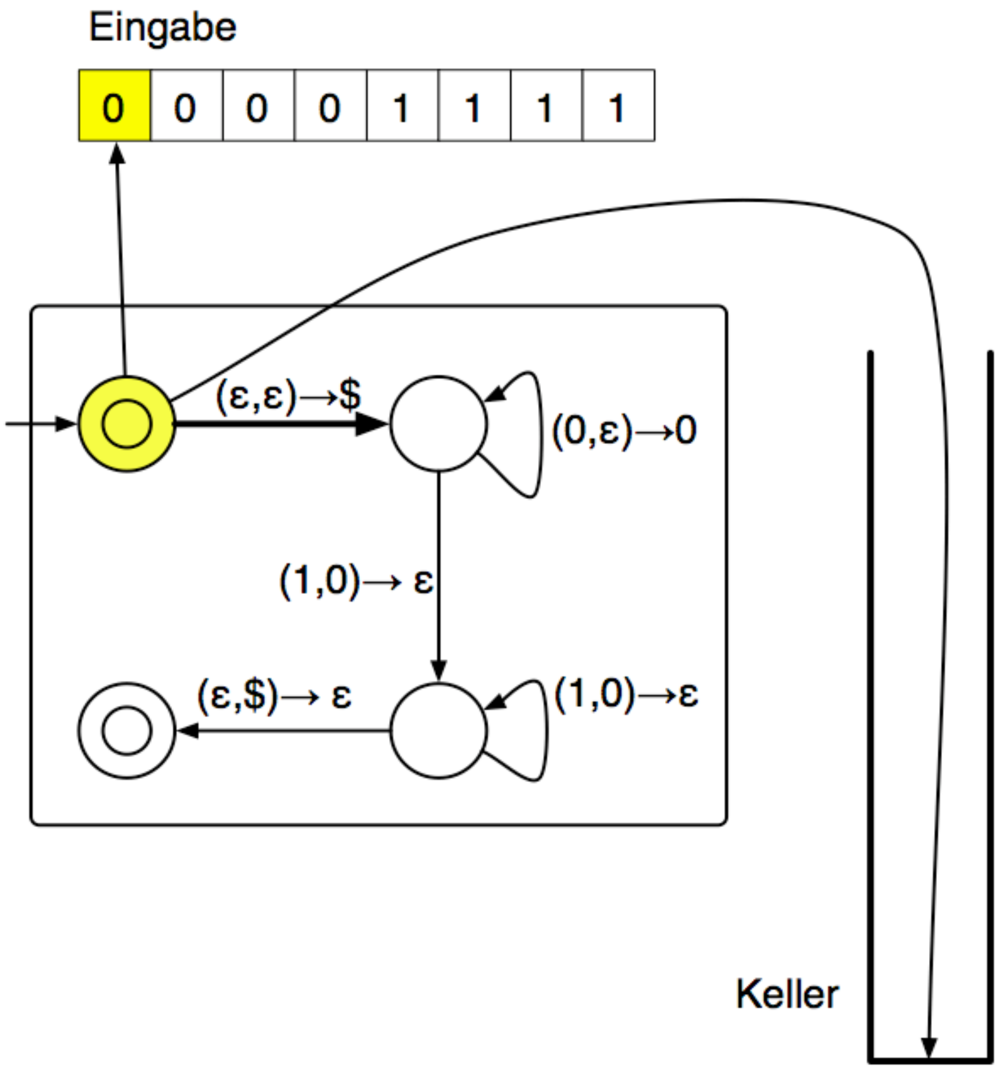
Keller-Automat: Beispielberechnung

- **Der PDA M = $(Q, \Sigma, \Gamma, \delta, q_1, F)$ akzeptiert die Sprache $\{0^n 1^n \mid n \geq 0\}$**
- **$Q = \{q_1, q_2, q_3, q_4\}$**
- **$\Sigma = \{0,1\}$**
- **$\Gamma = \{0,\$\}$**
- **$F = \{q_1, q_4\}$**
- **δ:**
  - $\delta(q_1, \varepsilon, \varepsilon) = \{(q_2,\$)\}$
    - q_1: Push \$, Gehe zu q_2
  - $\delta(q_2, 0, \varepsilon) = \{(q_2,0)\}$
    - q_2: Falls Lese = 0: Push 0; Gehe zu q_2
  - $\delta(q_2, 1, 0) = \{(q_3,\varepsilon)\}$
    - q_2: Falls Lese=1 und Pop=0 gehe zu q_3
  - $\delta(q_3, 1,0) = \{(q_3,\varepsilon)\}$
    - q_3: Falls Lese=1 gehe zu q_3
  - $\delta(q_3, \varepsilon, \$) = \{(q_4,\varepsilon)\}$
    - q_3: Falls Pop=\$ gehe zu q_4
  - $\delta(q, a) = \{\}$, für alle anderen Kombinationen $q \in Q, a \in \Sigma_\varepsilon$



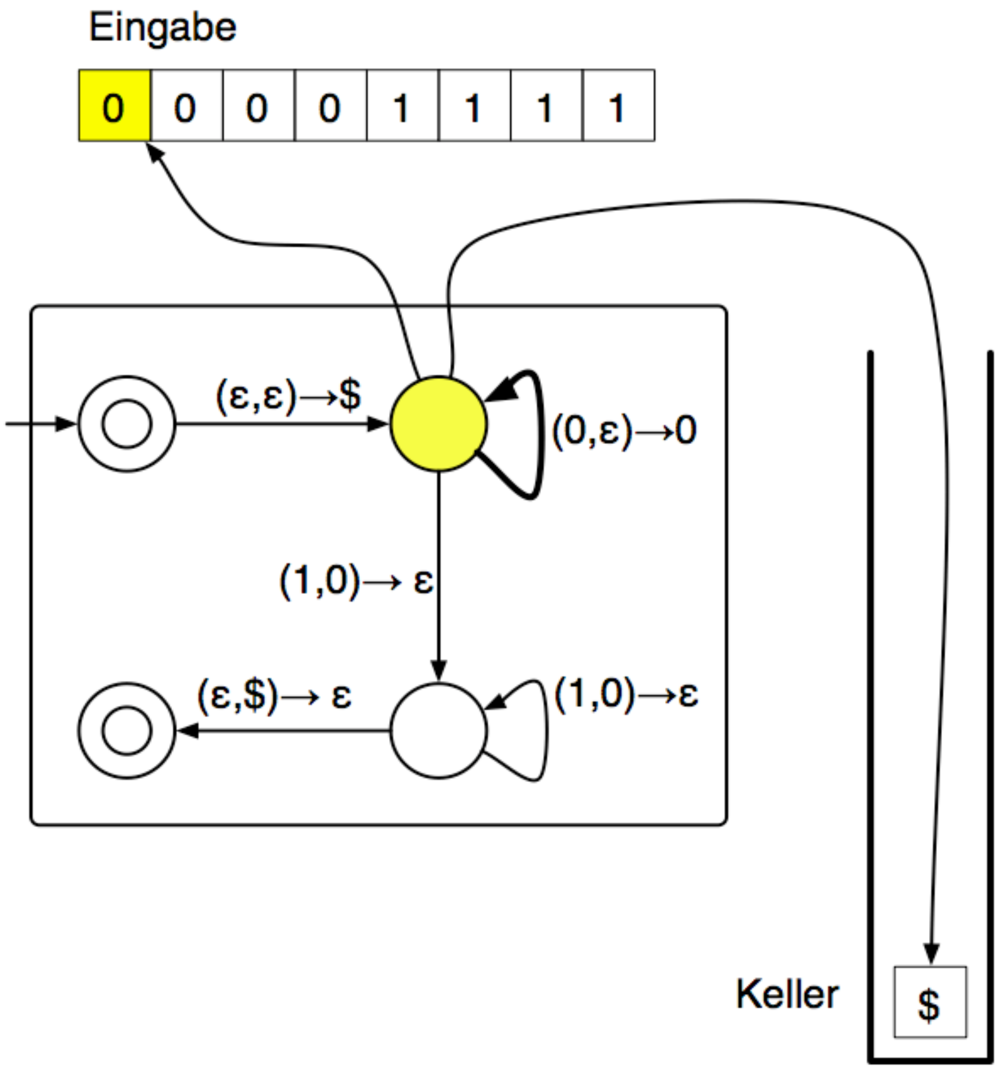
Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

Keller-Automat: Beispielberechnung



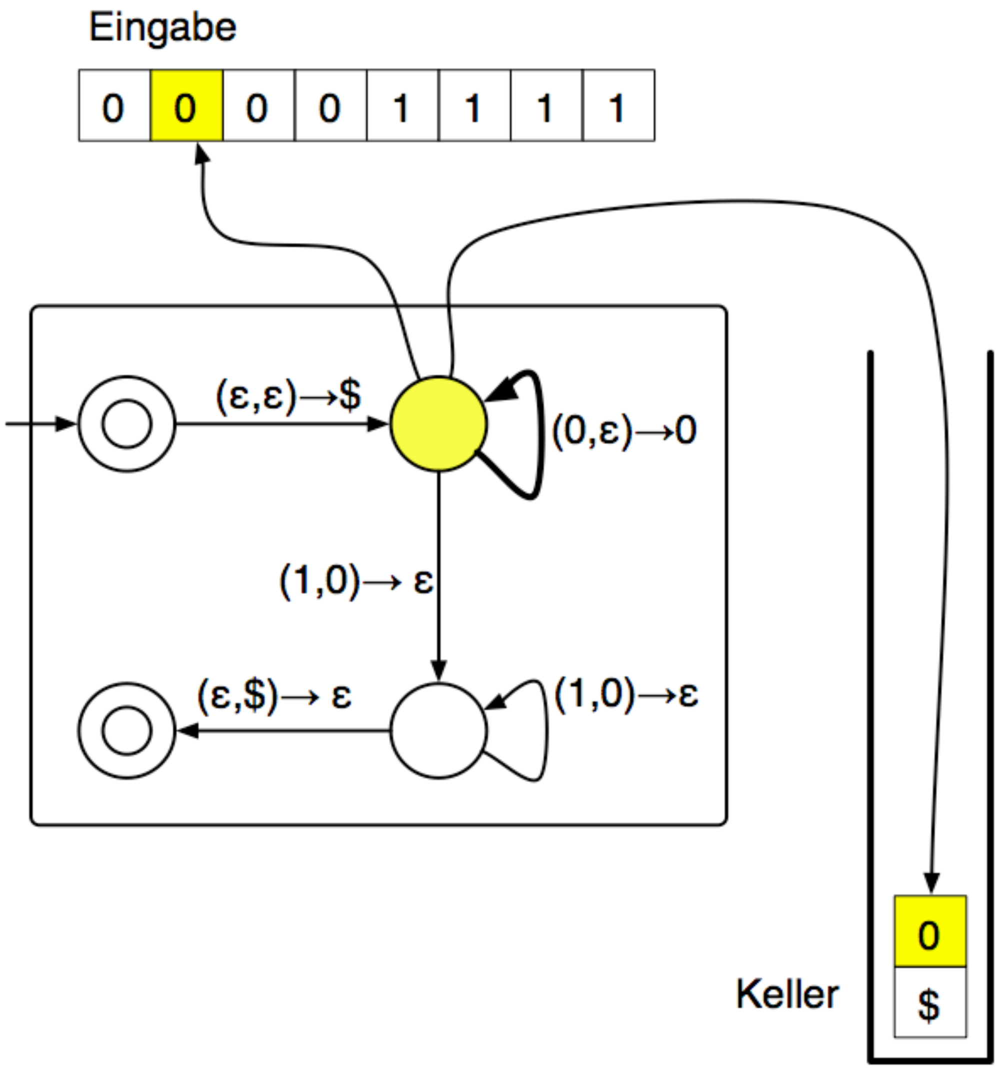
Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

Keller-Automat: Beispielberechnung



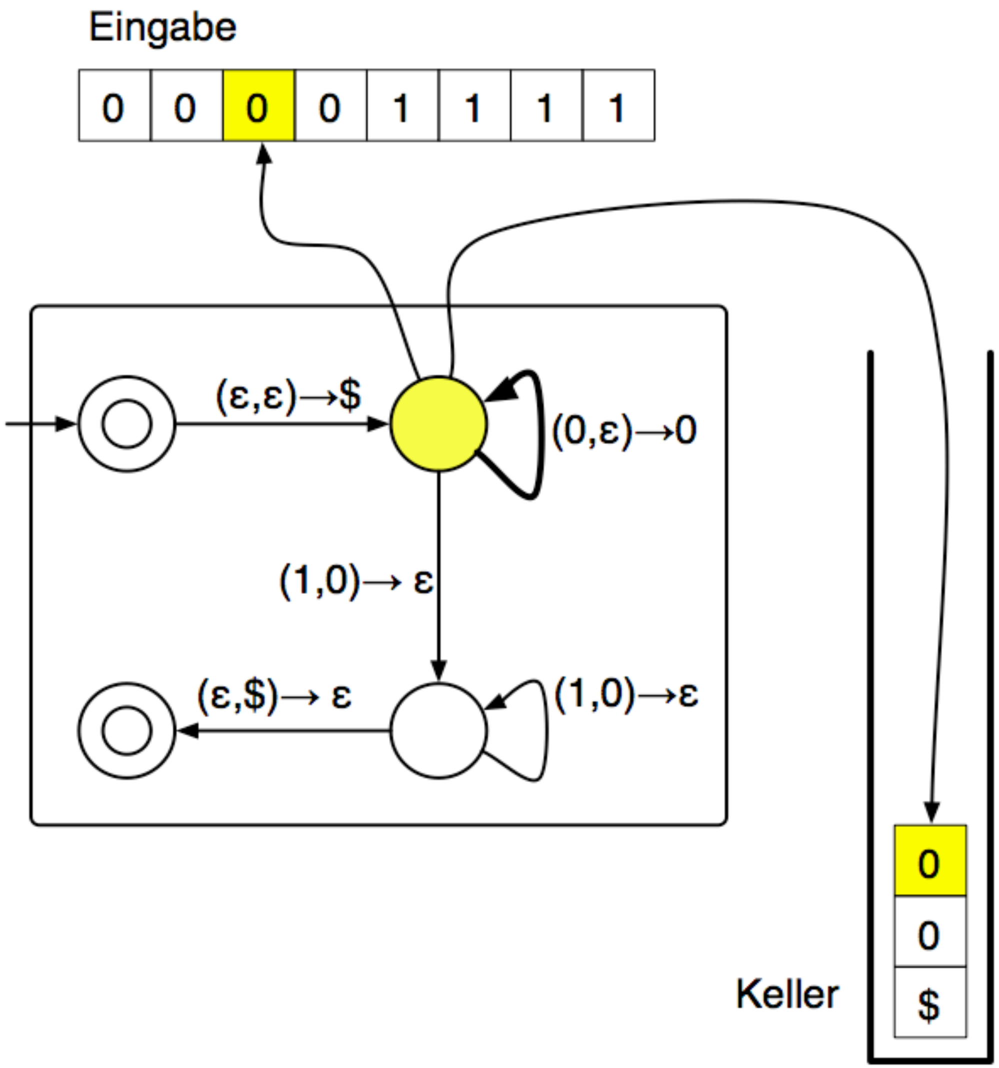
Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

Keller-Automat: Beispielberechnung



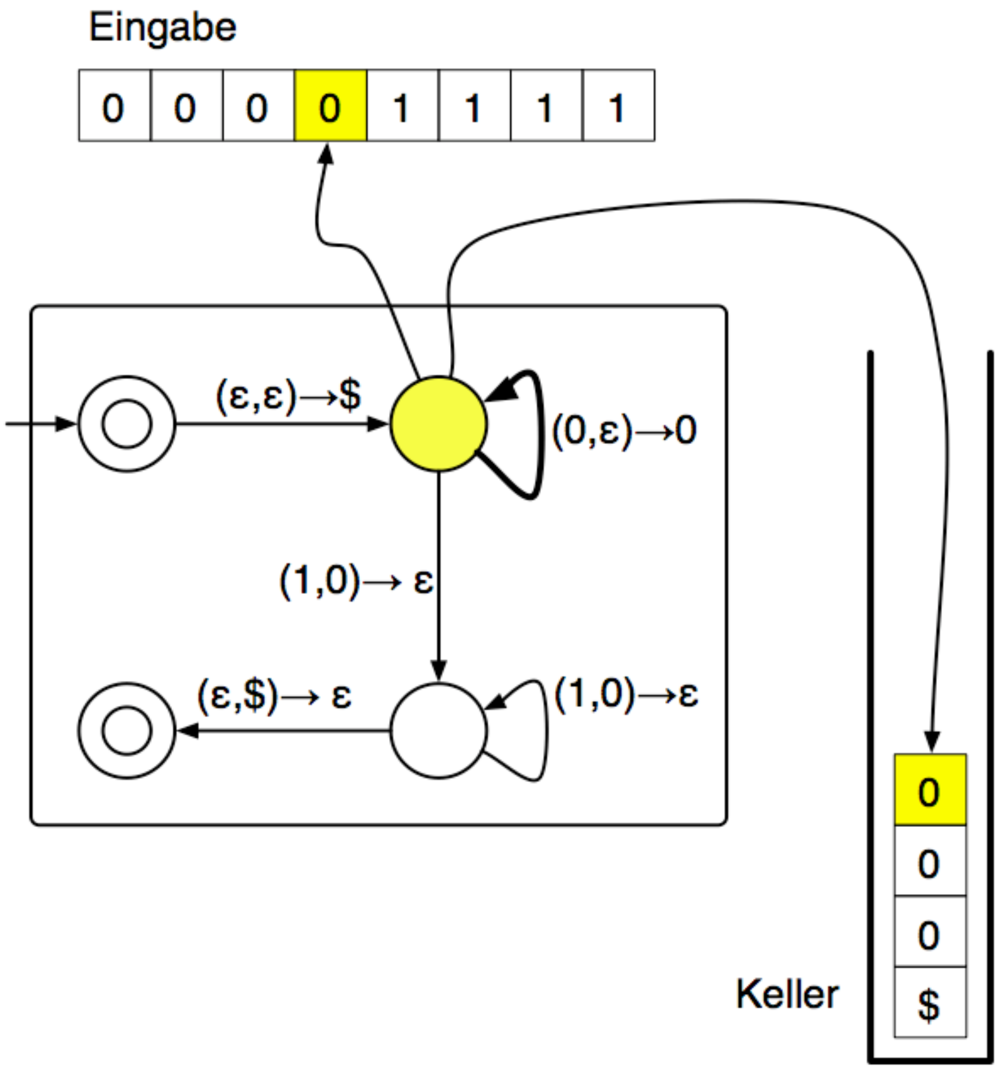
Keller-Automat: Beispielberechnung

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer



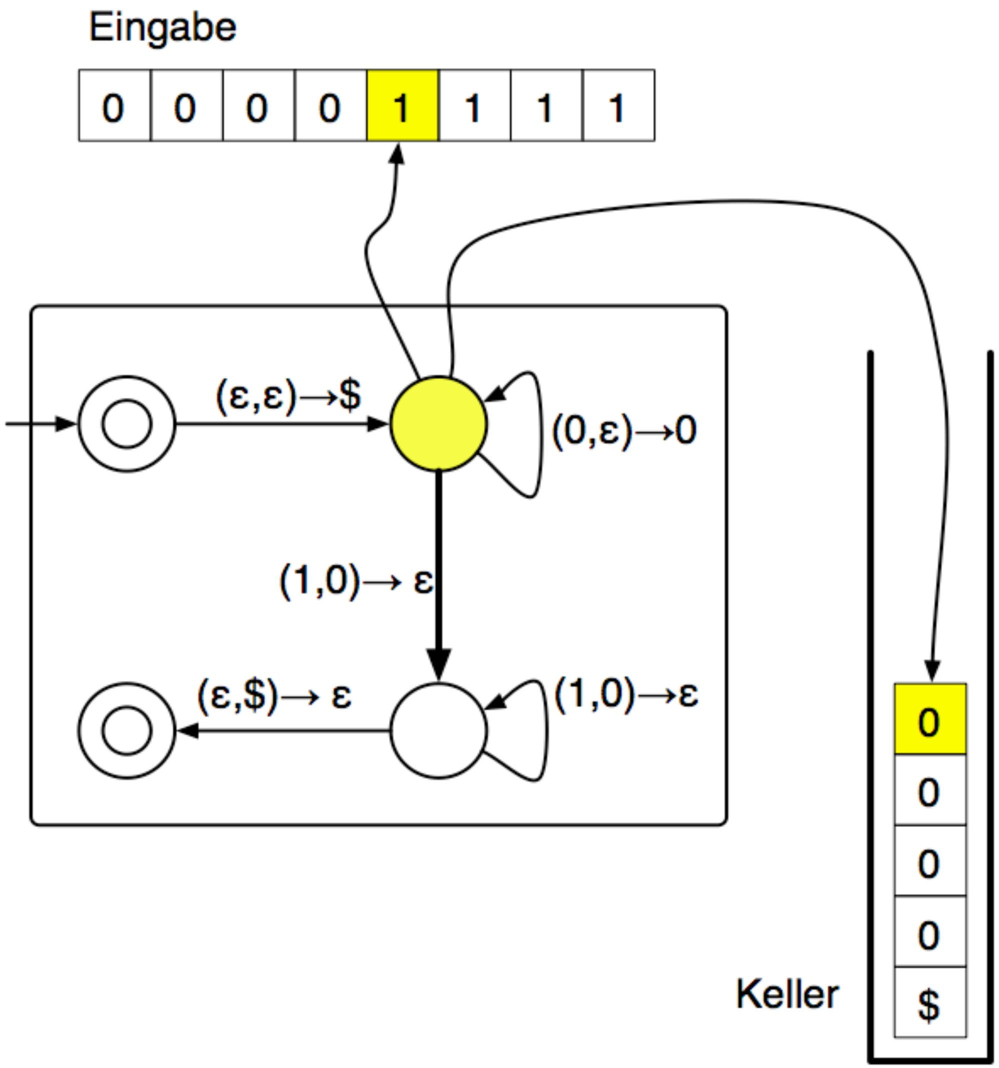
Keller-Automat: Beispielberechnung

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer



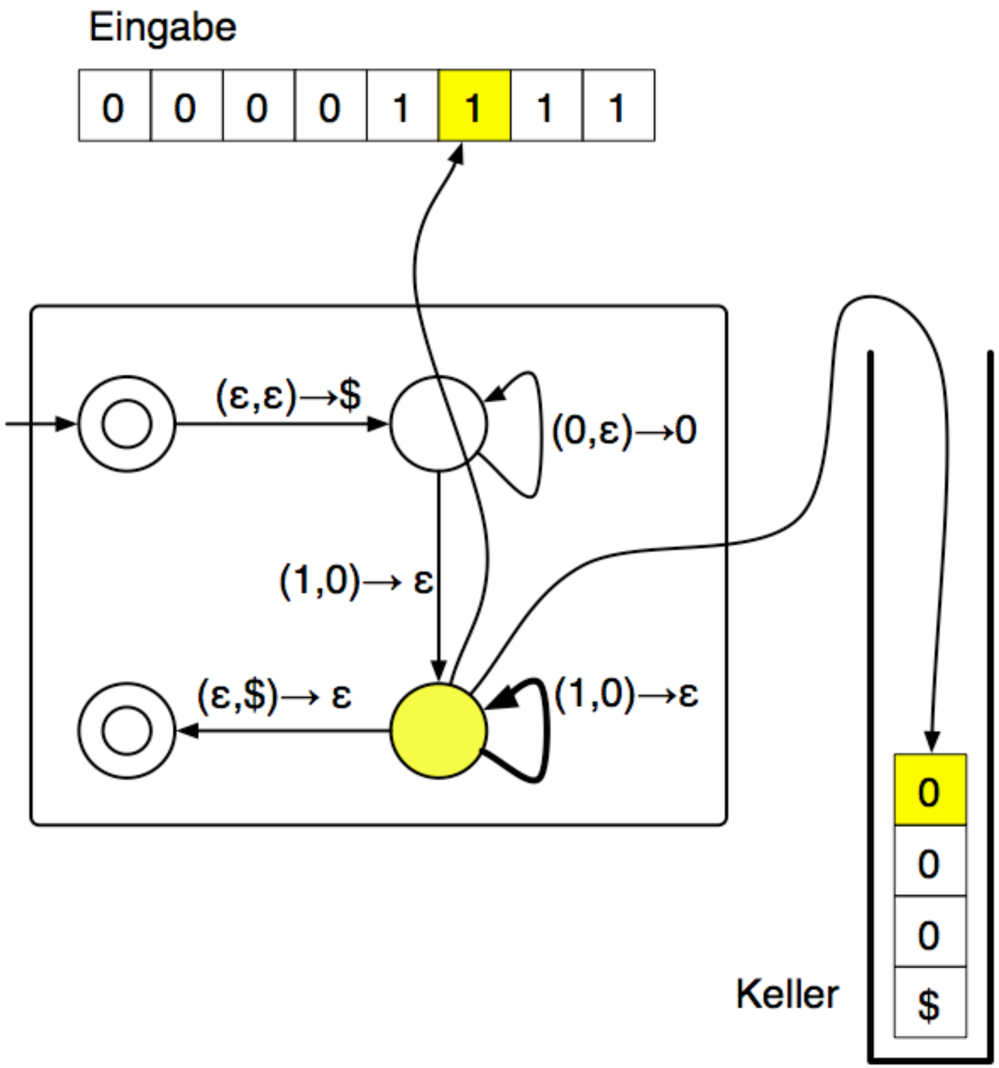
Keller-Automat: Beispielberechnung

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer



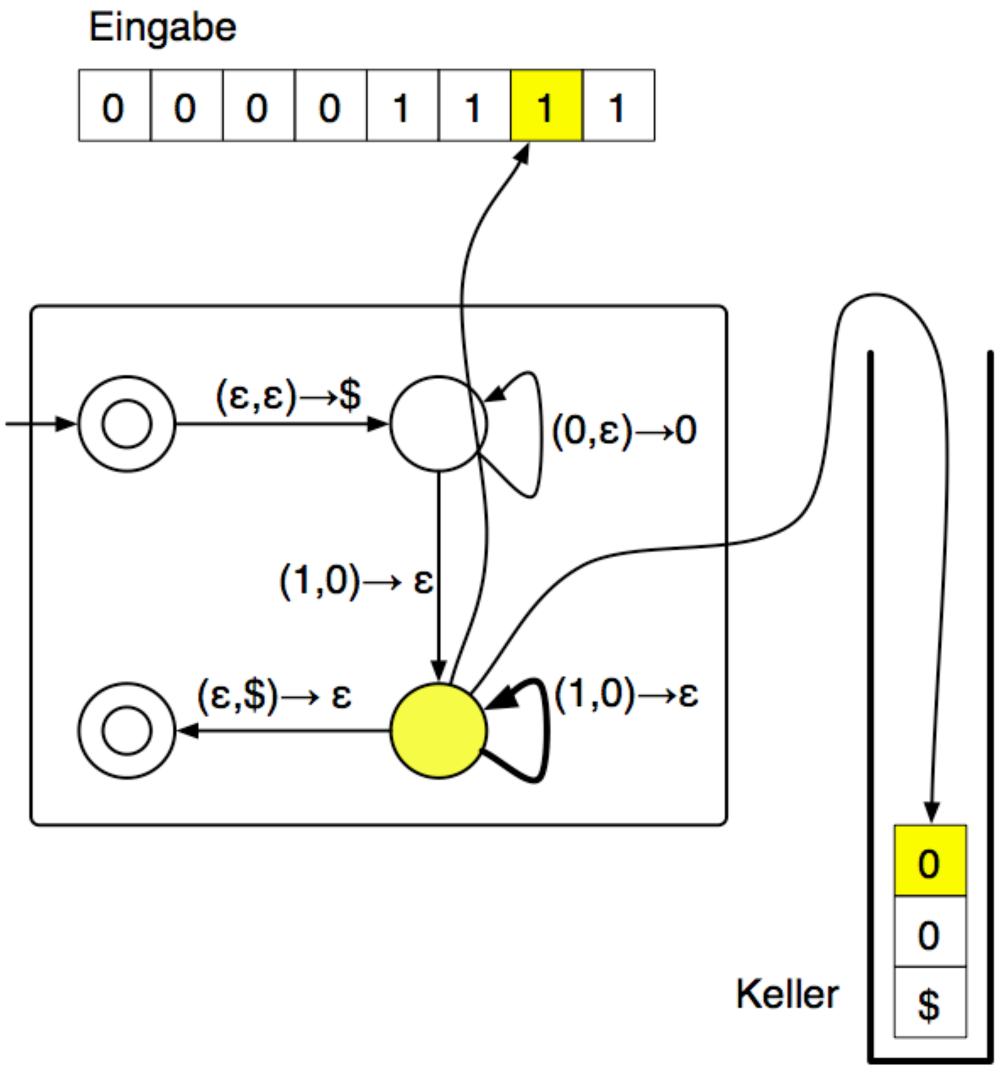
Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

Keller-Automat: Beispielberechnung

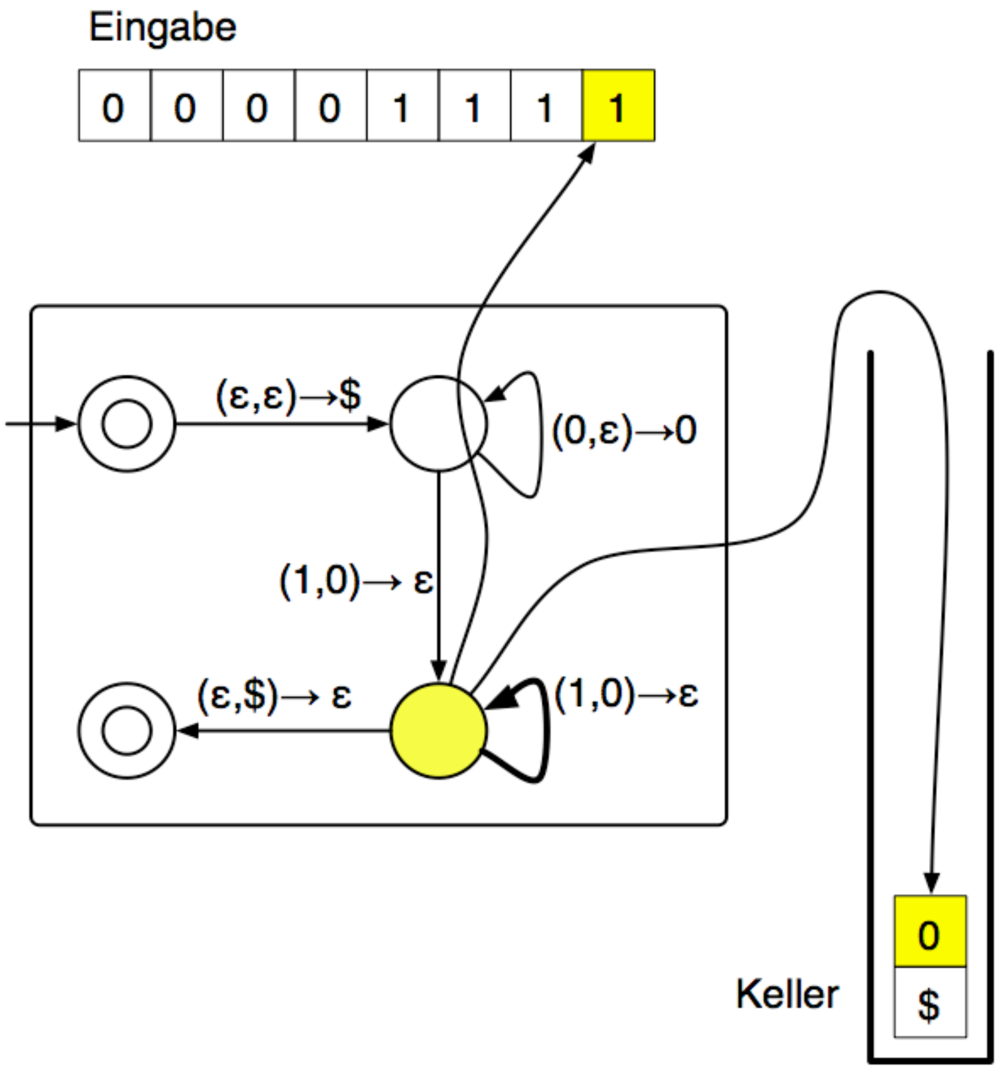


Keller-Automat: Beispielberechnung

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

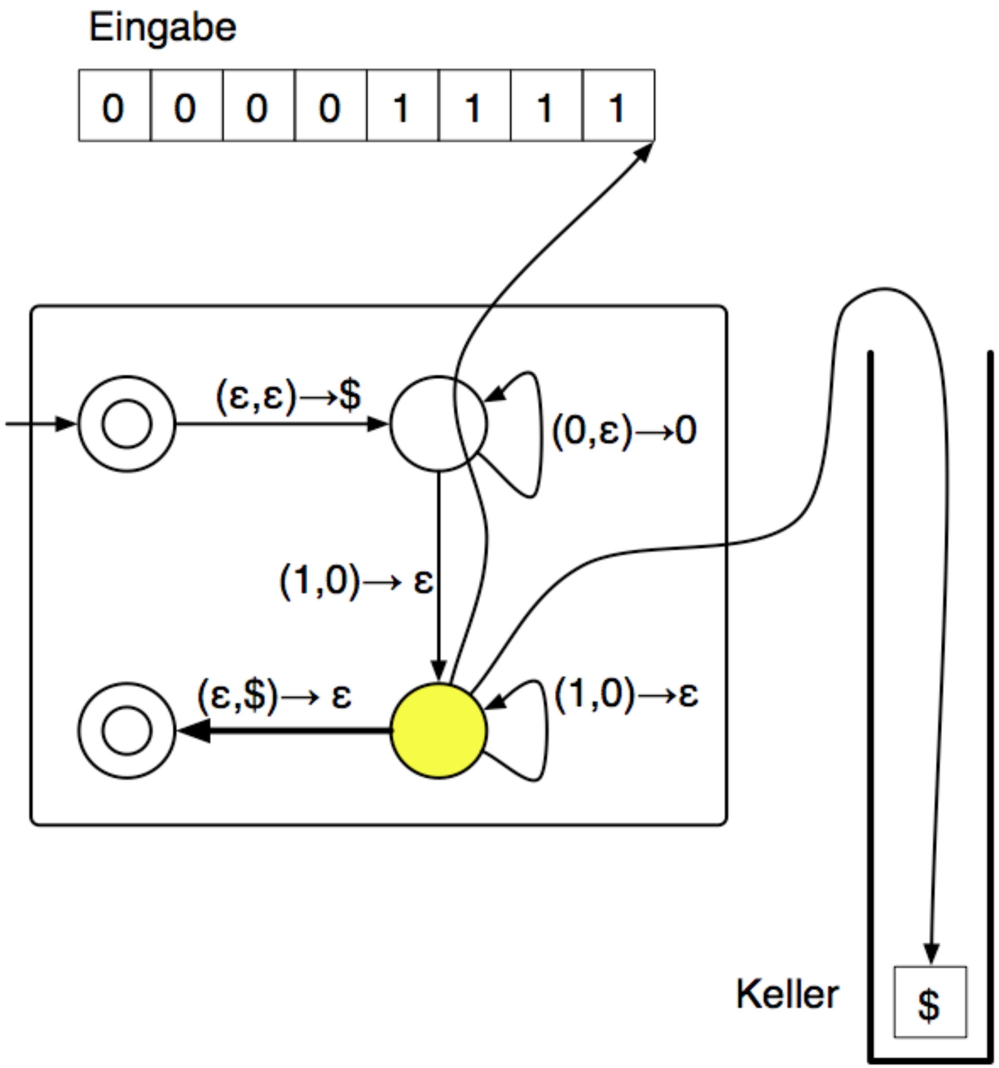


Keller-Automat: Beispielberechnung

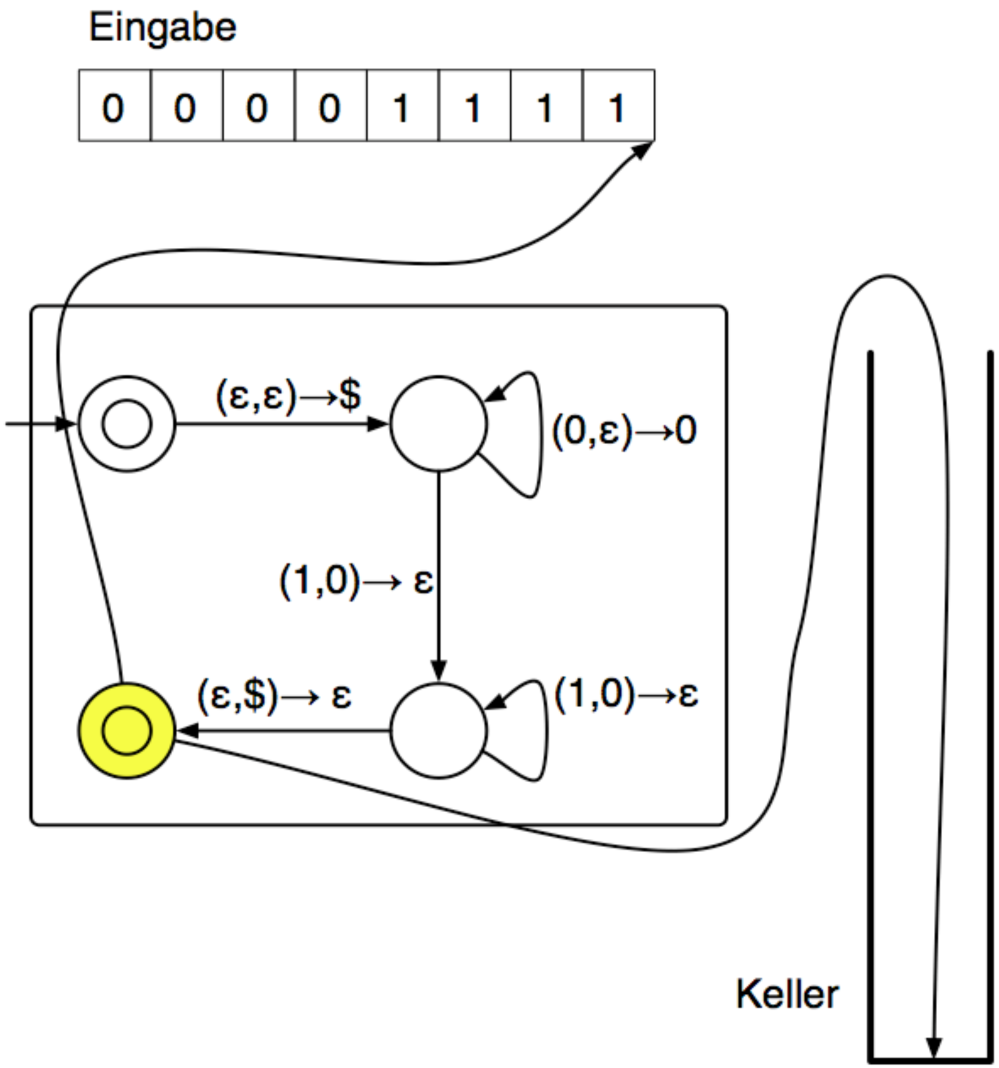


Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

Keller-Automat: Beispielberechnung



Keller-Automat: Beispielberechnung Automat akzeptiert!



Keller-Automaten beschreiben genau die kontextfreien Sprachen

- **Theorem 6.1**
  - Eine Sprache ist genau dann kontextfrei, wenn ein Kellerautomat sie erkennt
- **Lemma 6.1**
  - Ist eine Sprache kontextfrei, dann erkennt Sie ein Kellerautomat.
- **Beweisidee**
  - Die Mehrdeutigkeit der kontextfreien Sprache wird durch den Nichtdeterminismus des PDA gelöst
  - Die Ableitung der Worte steht im Keller
  - Im Keller werden die möglichen Übergänge der Ableitung geraten
  - Oben im Keller = Links, Unten im Keller = Rechts.
  - Sobald oben im Keller ein Terminal steht, wird es mit der Eingabe verglichen und jeweils gestrichen
  - Sobald oben im Keller eine Variable steht, wird eine mögliche Ersetzung im Keller durchgeführt.

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

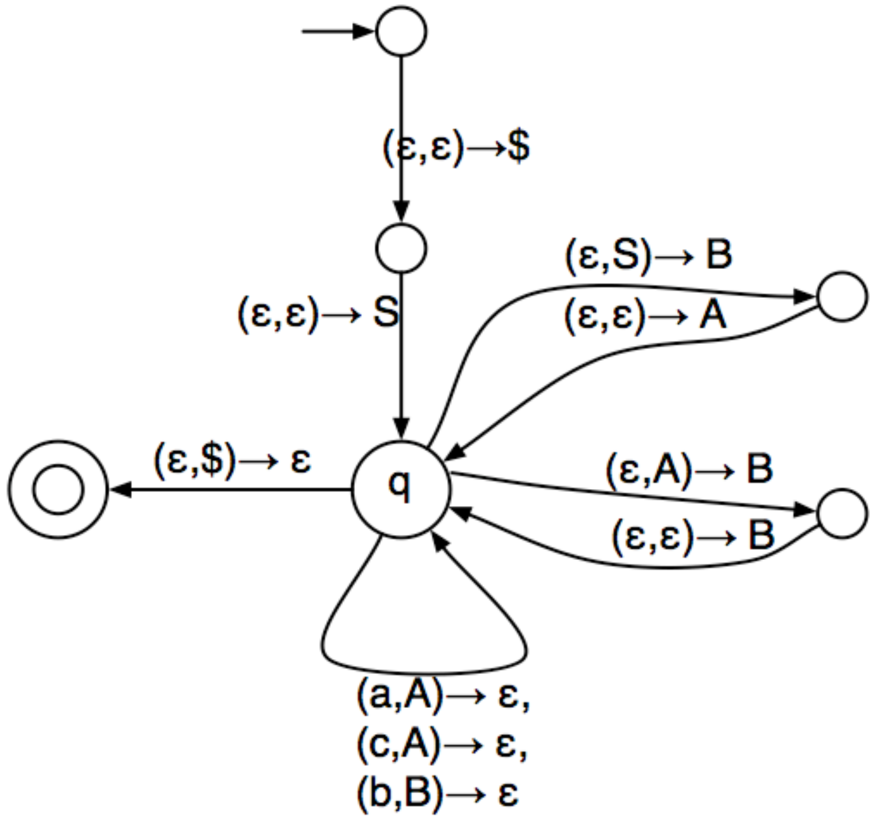
Jede kontextfreie Sprache kann von einem PDA erkannt werden

- **Lemma 6.1**
  - Ist eine Sprache kontextfrei, dann gibt es einen Kellerautomat für sie.
- **Konstruktion des Kellerautomaten**
  - Wir können davon ausgehen, dass die Sprache als kontextfreie Grammatik $G=(V,\Sigma,R,S)$ in Chomsky-Normalform vorliegt.
  - Ist $S \rightarrow \varepsilon$ in der Grammatik so gibt es einen ist der Startzustand des PDA akzeptierend
  - Zuerst wird der Stack mit den Symbolen \$S initialisiert
  - Sei q der Zustand nach der Initialisierung
  - Für jede Regel der Form
    - $A \rightarrow BC$
    - wird ein Übergang von q zu Zustand q[BC] hergestellt, der A vom Keller geholt, ε von der Eingabe liest und C auf den Keller legt
    - Der Übergang von q[BC] nach q liest ε von der Eingabe und legt B auf den Keller ablegt.
  - Für jede Regel der Form
    - $A \rightarrow a$
    - wird ein Übergang von q zu q anglegt, der von der Eingabe a liest und A vom Keller holt
  - Eine weitere Regel liest \$ vom Keller und geht in den einzigen akzeptierenden Zustand q[akz]

Beispielkonstruktion

- **Grammatik: G=(V,Σ,R,S) mit**
  - –V={S,,A,B}
  - –Σ={a,b,c}

- **R = { S → AB,**
  A → BB,
  A → a,
  A → c,
  B → b }



- **Beispiel:**

–	S	Keller: \$S	Eingabe: cb
	⇒ AB	Keller: \$BA	Eingabe: cb
	⇒ cB	Keller: \$B	Eingabe: b
	⇒ cb	Keller: \$	Eingabe: -

Ende der 7. Vorlesung

CoNe Freiburg

Albert-Ludwigs-Universität Freiburg
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

Informatik III

Christian Schindelhauer
schindel@informatik.uni-freiburg.de