Kontextfreie Grammatiken

- Regeln der Form $A \to w$
  $w \in (N \cup T)^*$

- Chomsky-Normalform:
  Reglen der Form $A \to BC$ oder $A \to a$
  $B, C \in N$ und $a \in T$

- Greibach-Normalform:
  Regeln der Form $A \to aX_1...X_n$
  $a \in T$ und $X_1, ..., X_n \in N$

Parsing kontextfreier Sprachen

- Erster Algorithmus von Kuno und Oettinger:
  Ausrechnen aller Berechnungspfade eines äquivalenten, nichtdeterministischen Kellerautomaten
  exponentielle Laufzeit

- Dann J. Cocke (und unabhängig Kasami ('65) und Younger ('67)):
  basiert auf Chomsky-Normalform
  $O(n^3)$

- Später Earley (1970):
  Bester praktischer Algorithmus,
  keine Normalform, $O(n^3)$,
  $O(n^2)$ auf eindeutigen Grammatiken,
  auf vielen Grammatiken lineare Laufzeit

- Theoretische Verbesserungen:
  Valiant (1975): $O(n^{2.78})$ (Matrixmultiplikation)
  Graham, Harrison, Ruzzo (1976) $O(n^3/\log n)$

CYK Parsing

- Cocke-Younger-Kasami
- Annahme: Grammatik in Chomsky-Normalform
- Musterbeispiel für *dynamisches Programmieren*
- siehe auch *Memoization*

CYK Algorithmus

- Aufbau einer 2-dimensionalen Matrix M

- Nichtterminal A in $M_{i,j}$:

  Es gibt Ableitung $A \overset{*}{\vdash} x_{i+1}...x_j$

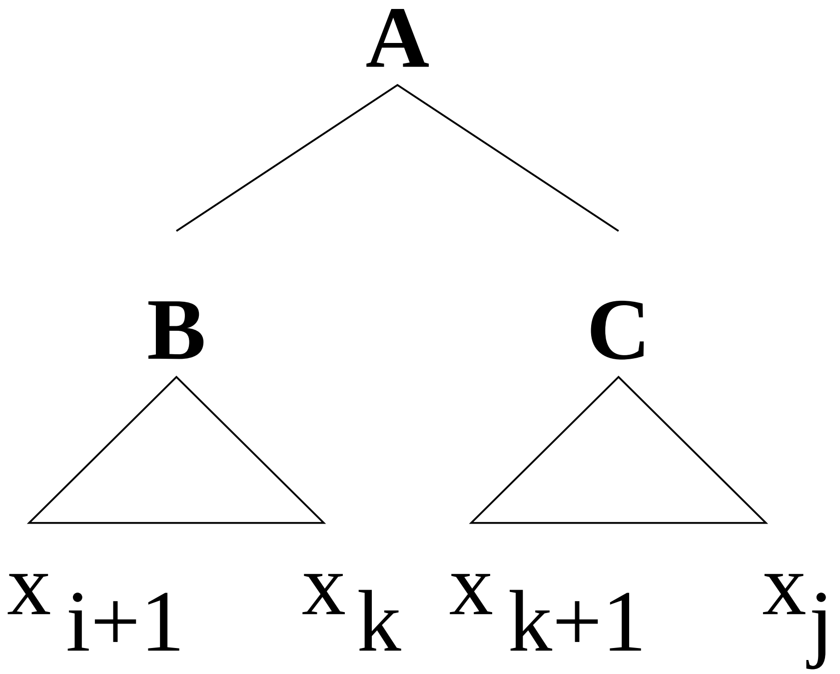
- Iterativer Aufbau dieser Matrix

CYK Algorithmus

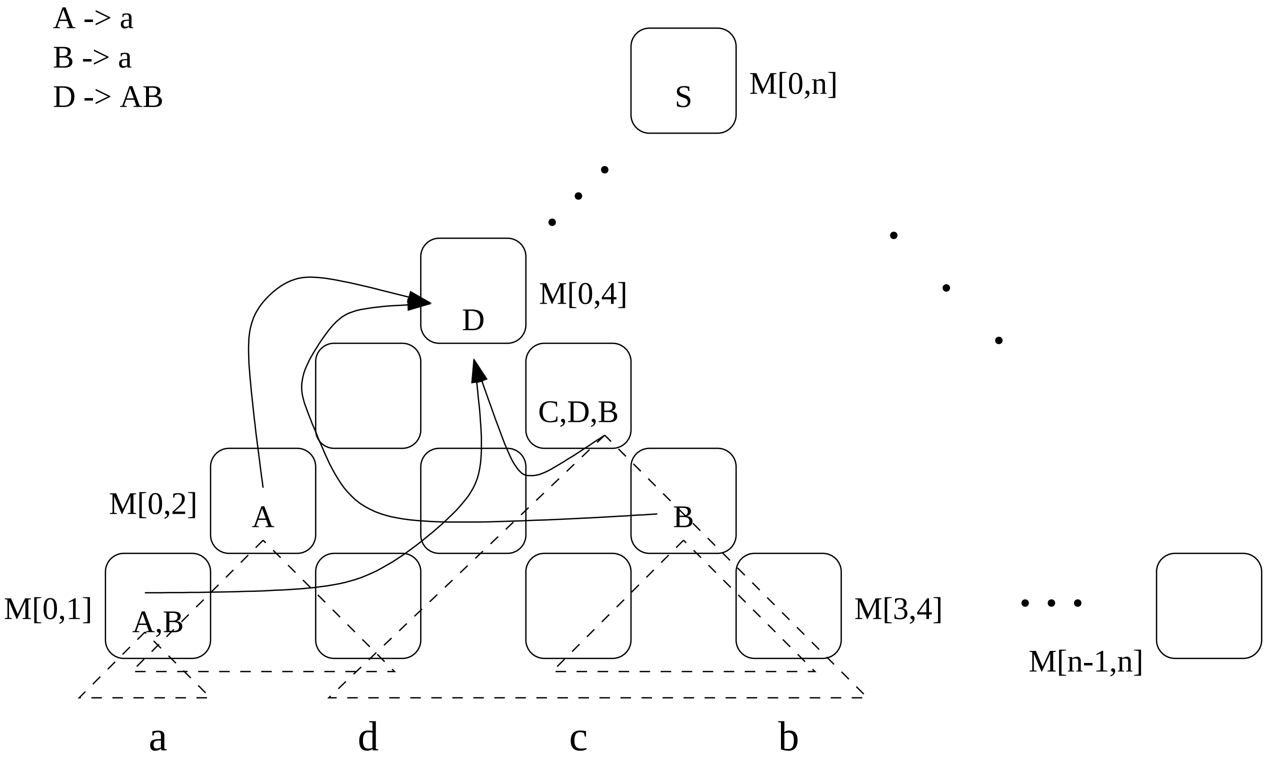
- $B \in M_{i,k}$, $C \in M_{k,j}$

  dann auch

  $A \rightarrow BC \in M_{i,j}$



CYK Algorithmus



CYK Algorithmus

- Input = $x_1 ... x_n$

- `for` $i := 0$ `to` $n-1$ `do`

  $M_{i,i+1} := \{A | A \rightarrow x_{i+1} \in P\}$

  `for` $d := 2$ `to` n `do`

  `for` $i := 0$ `to` $n-d$ `do`

  $j := i + d$

  $M_{i,j} := \emptyset$

  `for` $k := i+1$ `to` $j-1$ `do`

  $$M_{i,j} := M_{i,j} \cup \{A | A \rightarrow BC \in P, \; B \in M_{i,k}, \; C \in M_{k,j}\}$$

  `accept if` $S \in M_{0,n}$

CYK Algorithmus

- Speicherkomplexität: $O(n^2)$
- Zeitkomplexität: $O(n^3)$
- Nachteile:
  best-case = worst-case
  braucht Normalform
- Vorteil: sehr einfach