Chart-Parsing

Übersicht

- Motivation: Bisher vorgestellte Verfahren sind nicht effizient
- Grundidee des Chart-Parsing
- Datenstruktur
  - Knoten
  - passive und aktive Kanten
  - gepunktete Regeln *(dotted rules)*
- Fundamentalregel

Ziel

- Verstehen dieser für die Computerlinguistik *sehr* wichtigen Technik

Ineffizienz anderer Verfahren

Das Problem

- bei den bisher vorgestellten Verfahren kann es geschehen, dass derselbe Teil eines Satzes mehrfach analysiert wird
- das ist sehr ineffizient

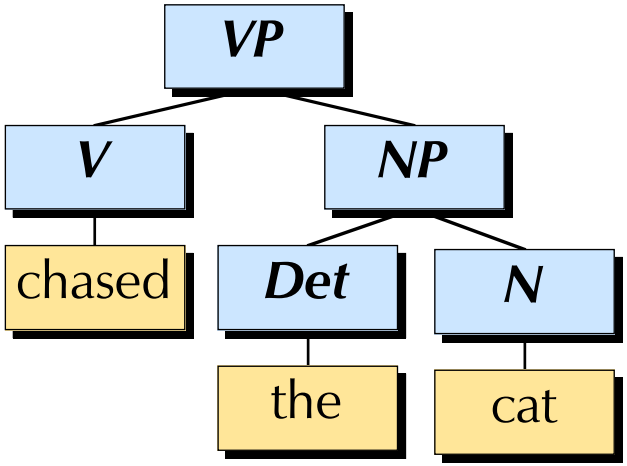
Die Frage

- wie könnte diese wiederholte Berechnung vermieden werden, die ja doch jedesmal zum selben Ergebnis führt?

Chart-Parsing: Grundidee

Beispiel-Grammatik (Ausschnitt):

- *VP → V NP*
- *VP → V NP PP*



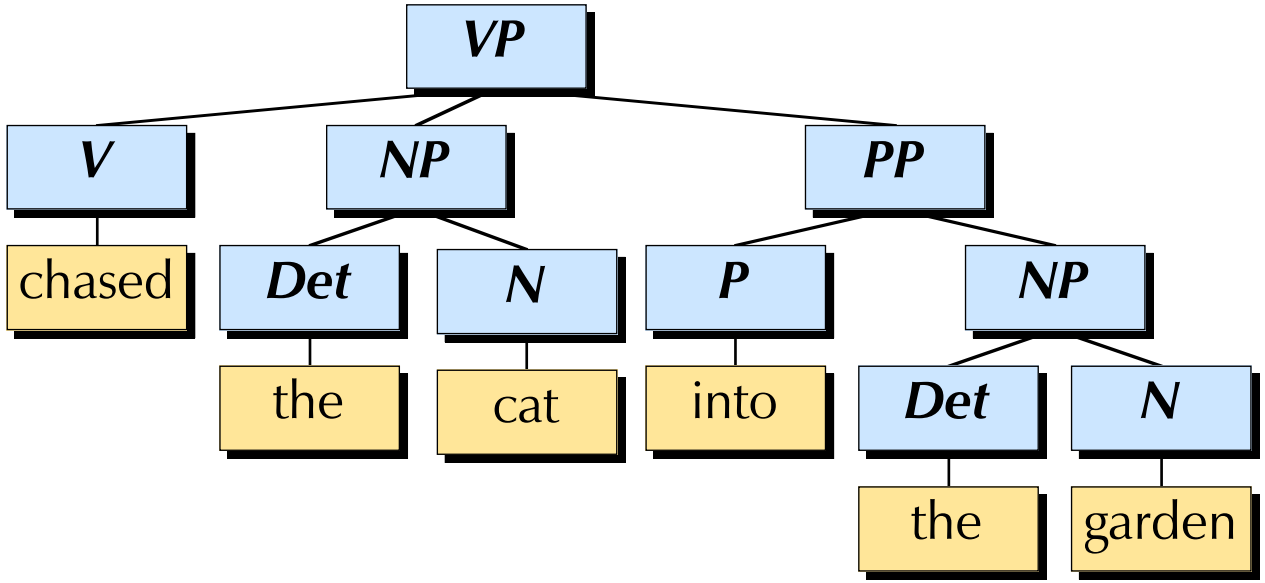


Chart-Parsing: Grundidee

Was geschieht, wenn ein Top-Down-Parser die Eingabekette »chased the cat into the garden« als VP analysiert?

- suche nach VP, nimm die erste Regel: *VP → V NP*
- (potentiell aufwendige) Analyse; schliesslich VP »chased the cat« gefunden
- es sind aber noch weitere Wörter da ⇒ Backtracking
- suche nach VP, nimm die zweite Regel: *VP → V NP PP*
- (potentiell aufwendige) Analyse
  - dass z.B. »the cat« eine NP ist, muss ein zweites Mal herausgefunden werden

Chart-Parsing: Grundidee

Das Problem

- bei den bisher vorgestellten Verfahren kann es geschehen, dass derselbe Teil eines Satzes mehrfach analysiert wird

Die Lösung

- der Parser sollte sich bereits gefundene Teilstrukturen merken
- auch wenn Backtracking erfolgt
  - vgl. Beispiel von vorhin: »the cat« ist eine NP, unabhängig davon, welche VP-Regel die richtige ist
- zum »Merken« dient die Chart

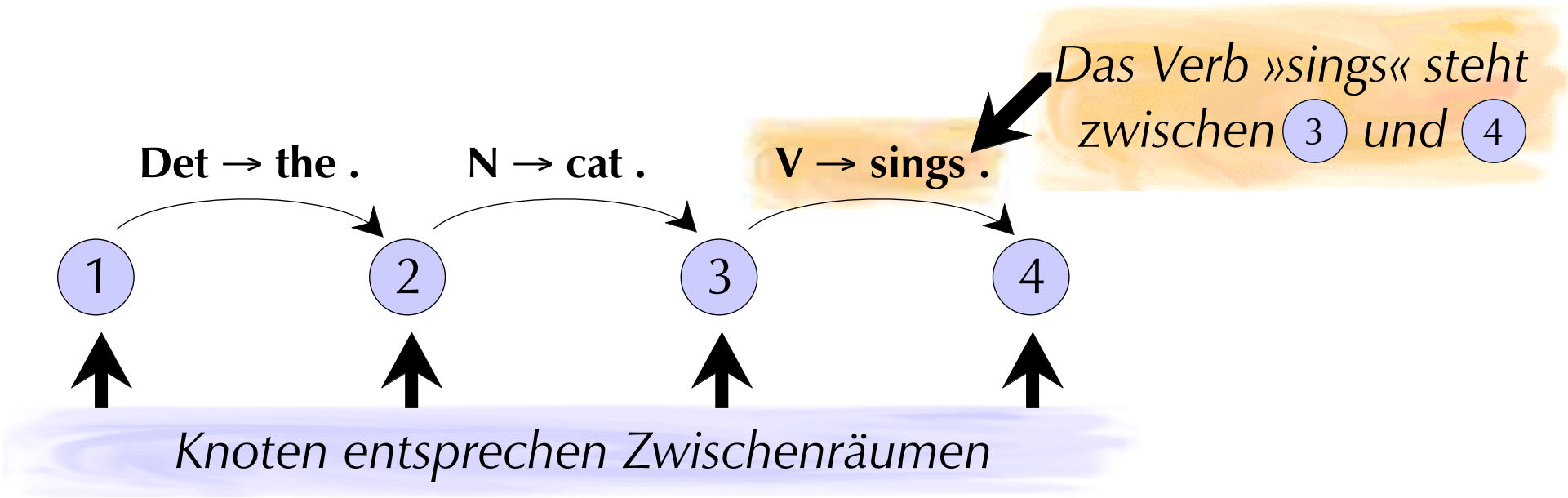
Chart-Parsing

Die Chart ist eine Datenstruktur

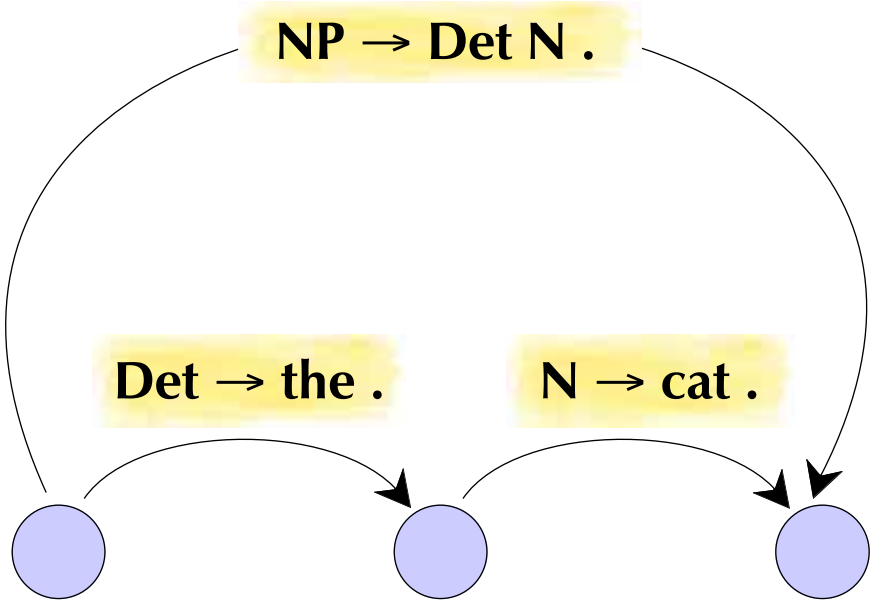
- nimmt alle Zwischenresultate der syntaktischen Analyse auf
- gerichteter Graph mit Kantenbeschriftungen
- wird im Verlauf der syntaktischen Analyse dynamisch erweitert
- hingegen wird nie etwas von der Chart entfernt → Monotonie
- unabhängig von Parsingstrategien und spezifischen Parsing-verfahren

Die Chart: Knoten

Die Knoten entsprechen Wort-Zwischenräumen



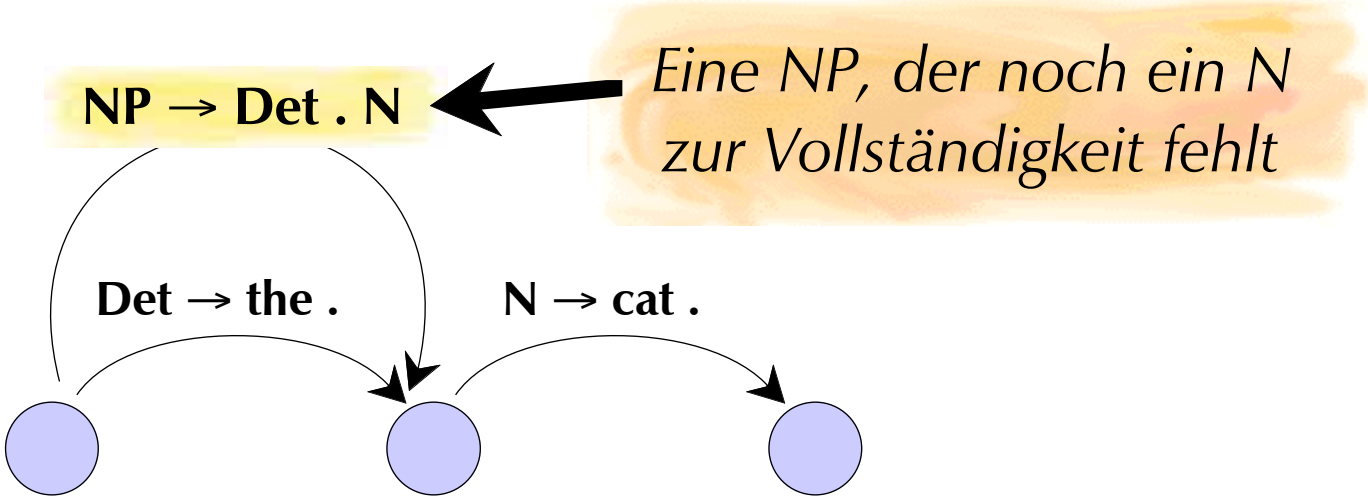
Die Chart: Inaktive/passive Kanten



Inaktive (auch: »passive«) Kanten zeigen an, welche Strukturen vollständig erkannt wurden.

- der Punkt steht ganz rechts

Die Chart: Aktive Kanten



Aktive Kanten zeigen an, welche Strukturen erst teilweise erkannt wurden.

- vor dem Punkt: bereits gefundener Teil
- nach dem Punkt: noch fehlender Teil

Gepunktete Regeln

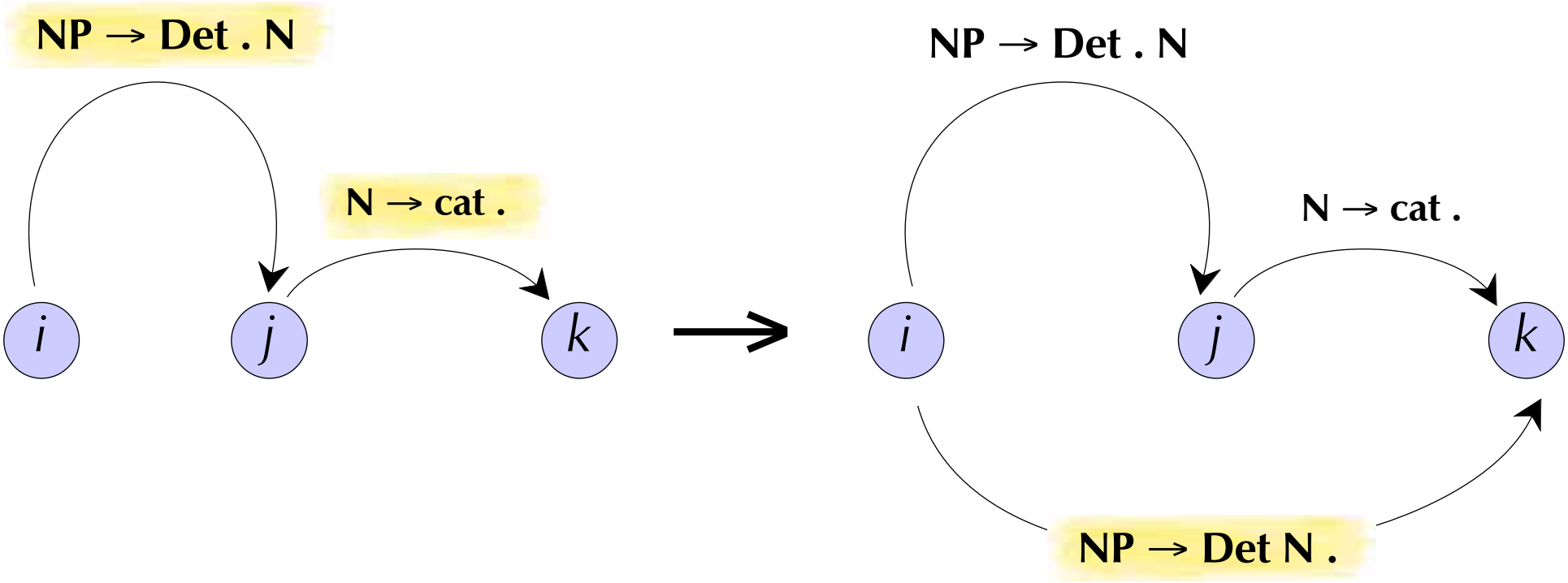
Eine gepunktete Regel *(dotted rule)* steht für ein Zwischenresultat des Parsers.

Beispiele für die Phrasenstruktur-Regel *S* → *NP VP*:

- *S* → . *NP VP*
  erst initialisiert; vom Satz wurde noch gar nichts gefunden
- *S* → *NP* . *VP*
  vom Satz wurde bereits die NP gefunden, aber noch keine VP
- *S* → *NP VP* .
  der Satz wurde vollständig gefunden

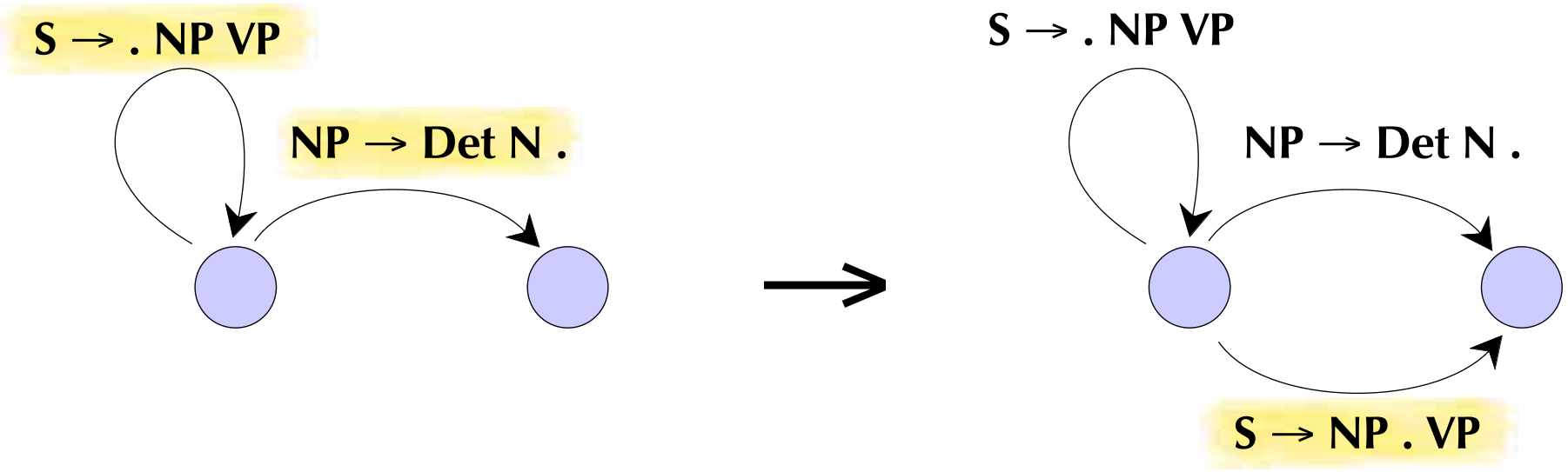
Fundamental-Regel des Chart-Parsing

Beispiel für die Anwendung der Fundamental-Regel:



Fundamental-Regel des Chart-Parsing

Beispiel für die Anwendung der Fundamental-Regel:



Fundamental-Regel des Chart-Parsing

Wenn folgende Kanten in der Chart sind:

- zwischen Knoten i und j: $A \rightarrow \alpha \ . \ B \ \gamma$
- zwischen Knoten j und k: $B \rightarrow \beta \ .$

Dann füge folgende Kante zur Chart hinzu:

- zwischen Knoten i und k: $A \rightarrow \alpha \ B \ . \ \gamma$

Dabei sind

- $A, B \in (V - \Sigma)$ — Nichtterminale
- $\alpha, \beta, \gamma \in V^*$ — beliebig lange Ketten von Nichtterminal- und Terminalsymbolen (evtl. auch leer)

Chart-Parsing-Verfahren

Die Chart ist eine Datenstruktur

- … und kein bestimmtes Parsing-Verfahren!
- Es gibt ganz verschiedene Verfahren, die mit einer Chart arbeiten
  - Top-Down-Chart-Parsing
  - Bottom-Up-Chart-Parsing
  - Earley-Algorithmus
  - Stolcke-Algorithmus (= Earley-Algorithmus mit Erweiterung um Stochastik)

    Andreas Stolcke: An Efficient Probabilistic Context-Free Parsing Algorithm that Computes Prefix Probabilities. http://xxx.lanl.gov/cmp-lg/abs/9411029
  - … und zahlreiche mehr