Generatives, lexikalisiertes, statistisches Parsing von Michael John Collins (1996, 1997, 2003)

Éva Mújdricza

Übersicht

- Wiederholung
- Eugene Charniak (1997)
  - Lexikalisiertes Parsing
  - Generatives Beispiel
- Michael John Collins (1996, 1997, 2003)
  - Hauptmerkmale
  - Generatives Parsing bei Collins
  - 3 Modelle – Ziel: bessere Ergebnisse
    - Ergebnisse der Trainings auf der Penn Treebank
- Fehleranalyse

Wiederholung

- Generelles Ziel: möglichst fehlerfreie Parsebäume/Analysen zu den verschiedensten Sätzen generieren. Bei jedem Input eine wahrscheinlichste Analyse ausgeben lassen.
- PCFG – probabilistisches Modell
- Wahrscheinlichkeiten:
  - P: Wahrscheinlichkeit
  - β: Nichtterminale
  - X: Terminale

$$P(\beta \mid X) = \frac{\text{count}(X \rightarrow \beta)}{\text{count}(X)}$$

- T_{Best}: die beste Analyse ~ der wahrscheinlichste Parsebaum

$$T_{Best} = \text{argmax}\, P(T \mid S) = \text{argmax}\, \frac{P(T,S)}{P(S)} = P(T,S)$$

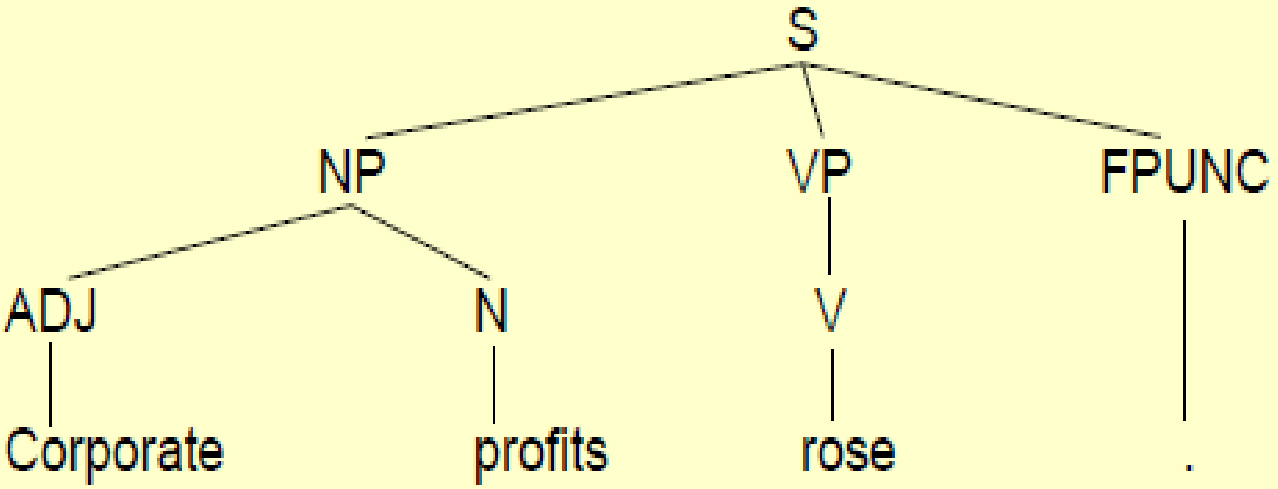
  - T: Baum
  - S: Satz

Wiederholung

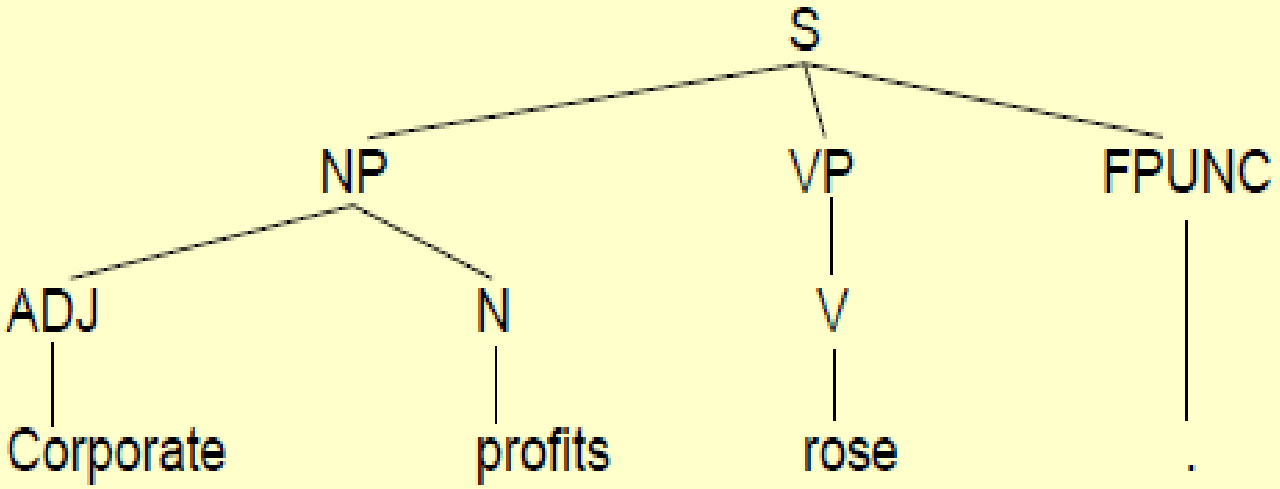
- Charniak (1997)
  - statistisches Modell
  - generatives Modell
  - lexikalisiertes Modell

    → Performanzvorteil von ca. 10%
  - explizite Grammatik
  - kein explizites Tagging
  - Smoothing
  - Große Performanz: 86-87% Korrektheit
- Beispielsatz: *Corporate profits rose.*

Charniak: Ausgangsparsebaum



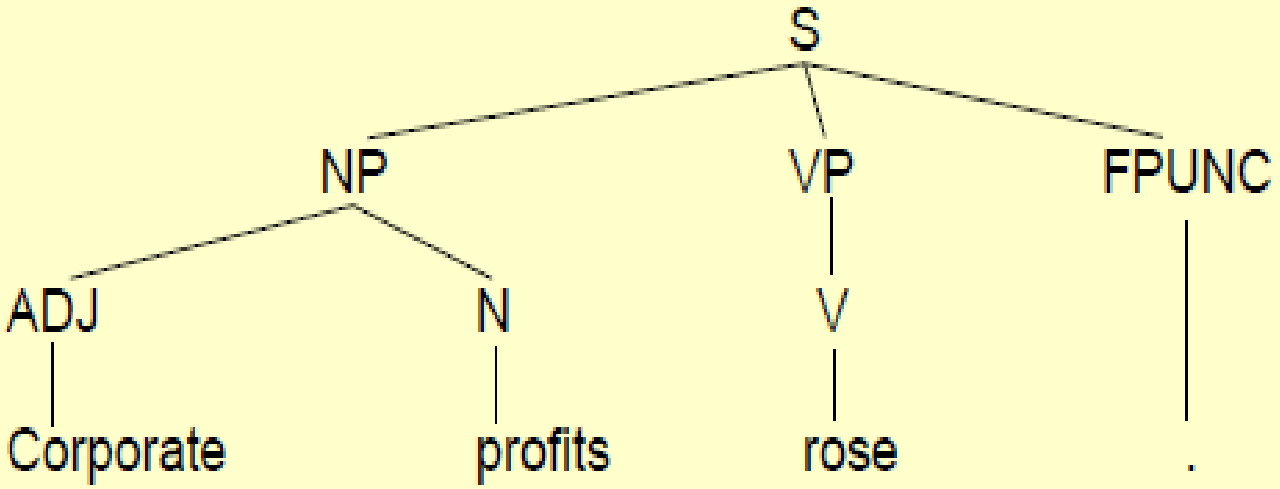
Charniak: Ausgangsparsebaum



Baumwahrscheinlichkeit:

P_{Baum} =

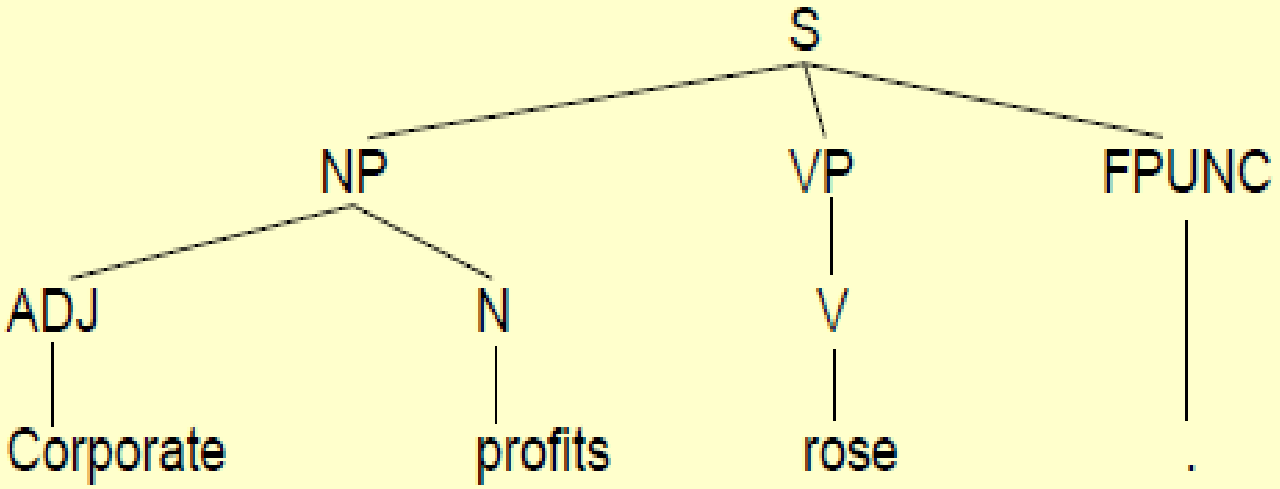
Charniak: Ausgangsparsebaum



Baumwahrscheinlichkeit:

$P_{Baum} = p(S)$

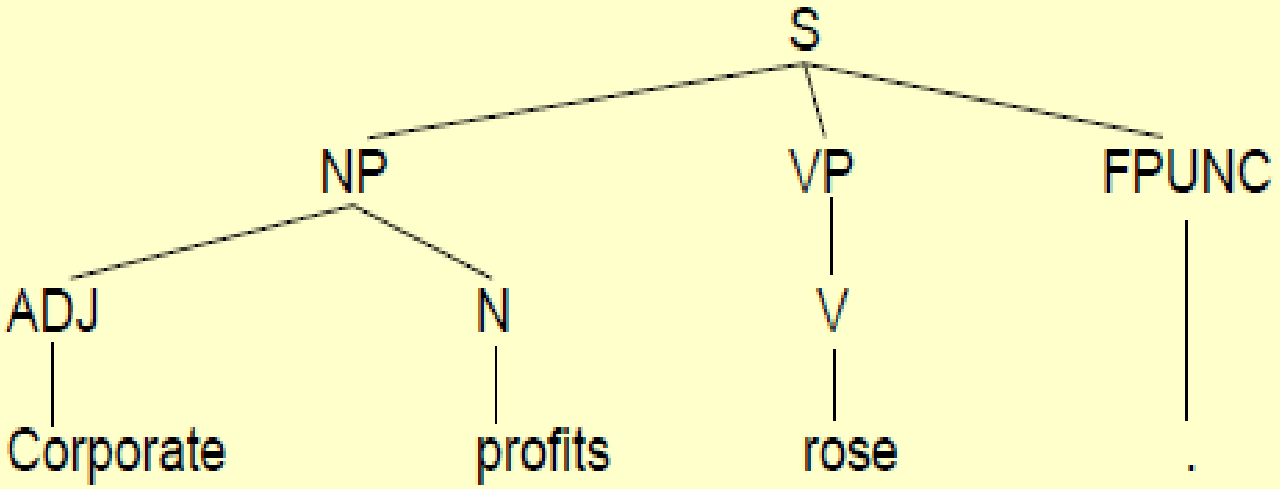
Charniak: Ausgangsparsebaum



Baumwahrscheinlichkeit:

$P_{Baum} = p(S) * p(S \rightarrow NP\ VP\ FPUNC)$

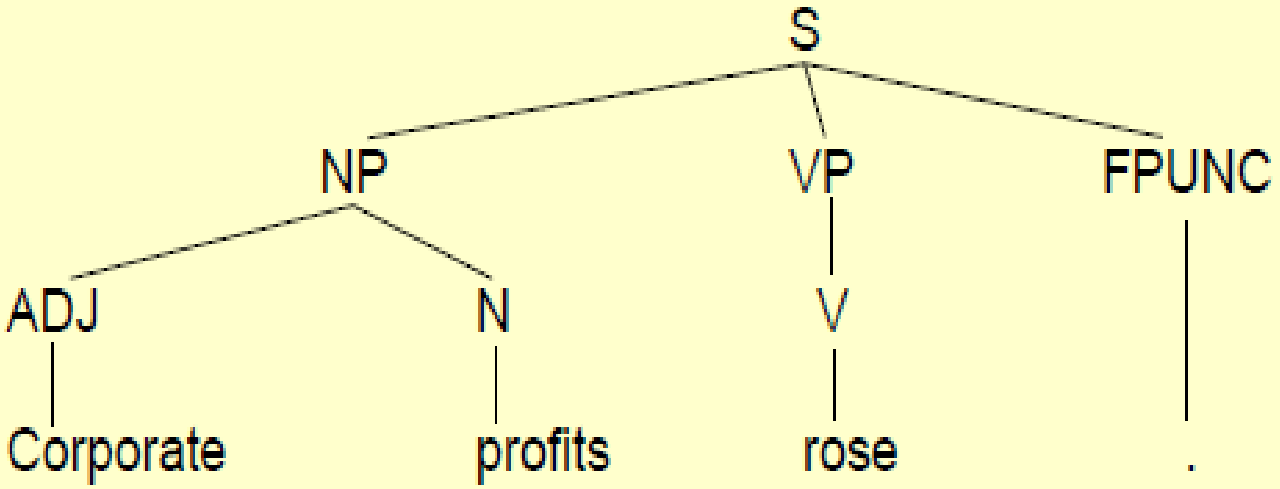
Charniak: Ausgangsparsebaum



Baumwahrscheinlichkeit:

$P_{Baum} = p(S) * p(S \rightarrow NP\ VP\ FPUNC) * p(NP \rightarrow ADJ\ N)$

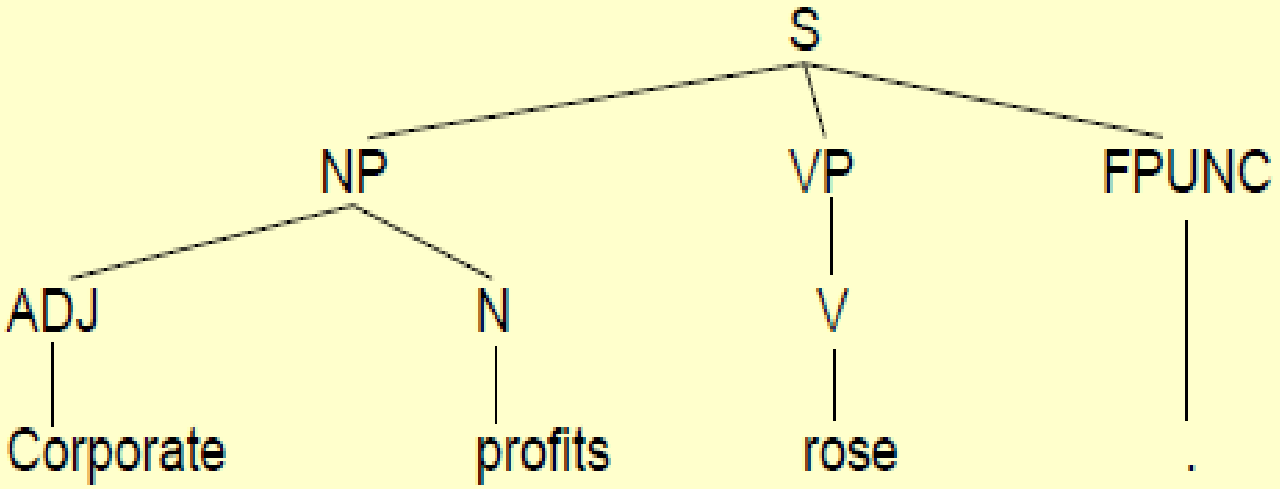
Charniak: Ausgangsparsebaum



Baumwahrscheinlichkeit:

$P_{Baum} = p(S) * p(S \rightarrow NP\ VP\ FPUNC) * p(NP \rightarrow ADJ\ N) * p(VP \rightarrow V)$

Charniak: Ausgangsparsebaum



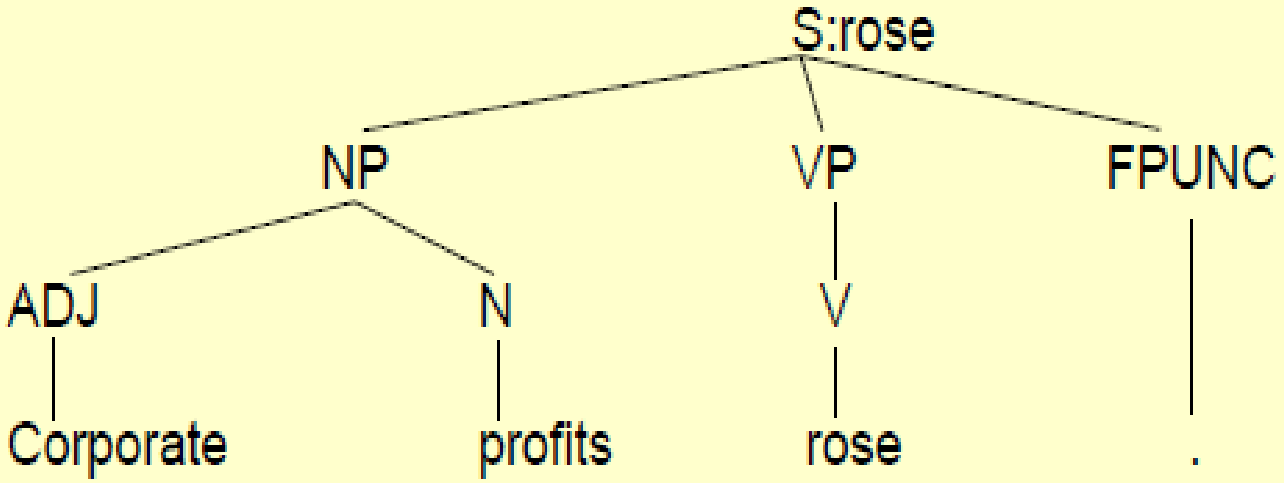
Baumwahrscheinlichkeit:

P_{Baum} = p(S) * p(S → NP VP FPUNC) * p(NP → ADJ N) * p(VP → V)

* p(ADJ → corporate) * p(N → profits) * p(V → rose) * p(FPUNC → .)

Charniak: Lexikalisierter Parsebaum

Charniak: Lexikalisierter Parsebaum

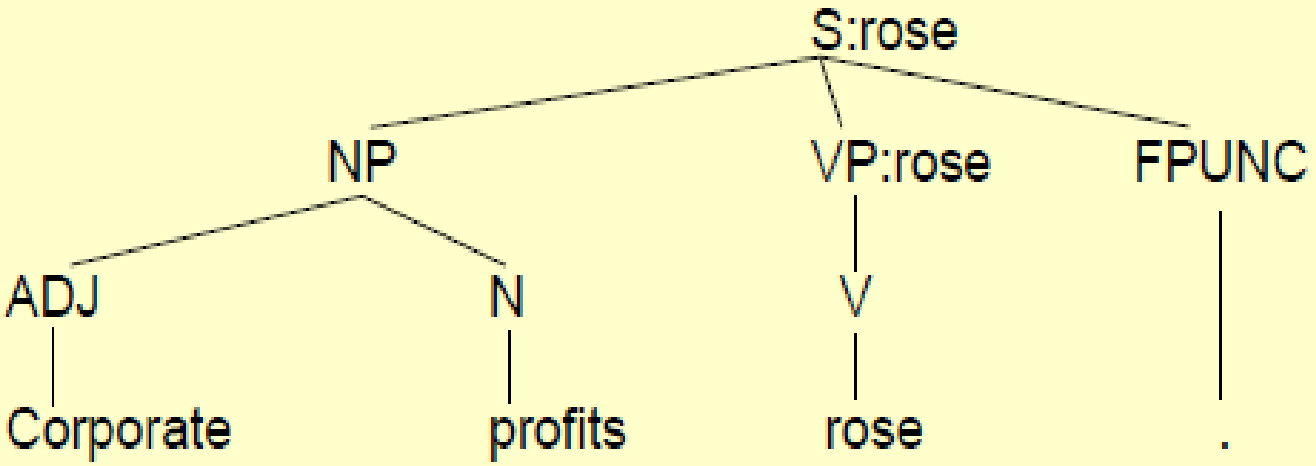


Die Mutterknoten (P) werden mit dem Blatt (~ Wort), auf das der Kopf der unmittelbaren Konstituente von P abgeleitet wird, versehen. → Der Baum enthält lexikalische Informationen.

P:h

S:rose

Charniak: Lexikalisierter Parsebaum

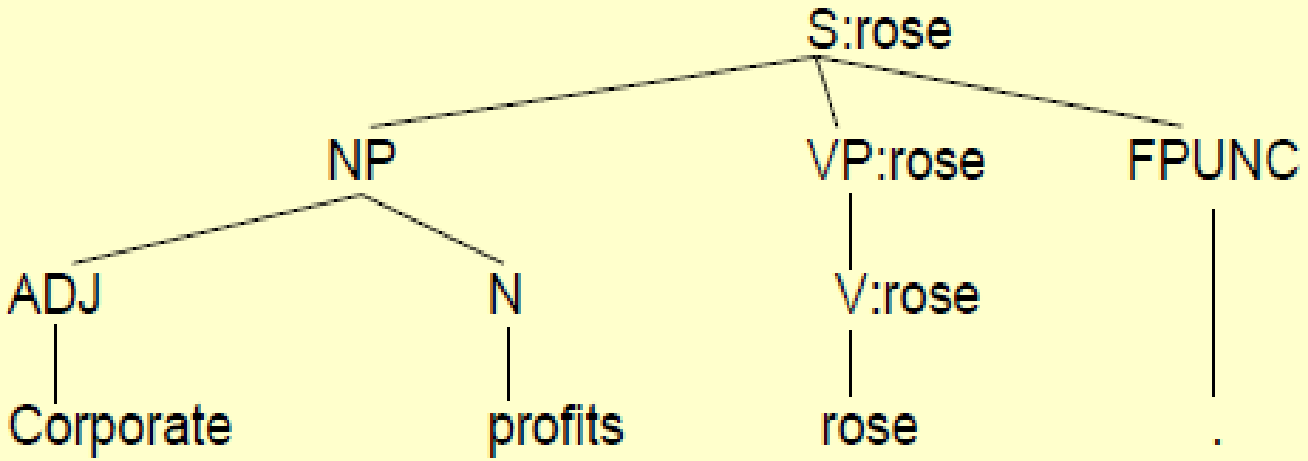


Die Mutterknoten (P) werden mit dem Blatt (~ Wort), auf das der Kopf der unmittelbaren Konstituente von P abgeleitet wird, versehen. → Der Baum enthält lexikalische Informationen.

P:h

S:rose - VP:rose

Charniak: Lexikalisierter Parsebaum

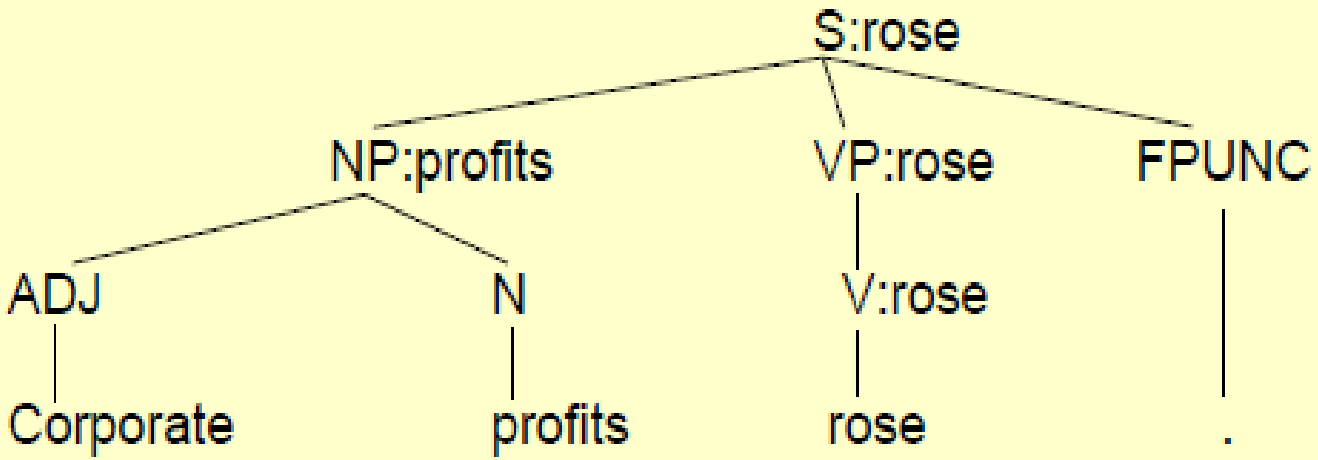


Die Mutterknoten (P) werden mit dem Blatt (~ Wort), auf das der Kopf der unmittelbaren Konstituente von P abgeleitet wird, versehen. → Der Baum enthält lexikalische Informationen.

P:h

S:rose – VP:rose – V:rose → rose

Charniak: Lexikalisierter Parsebaum



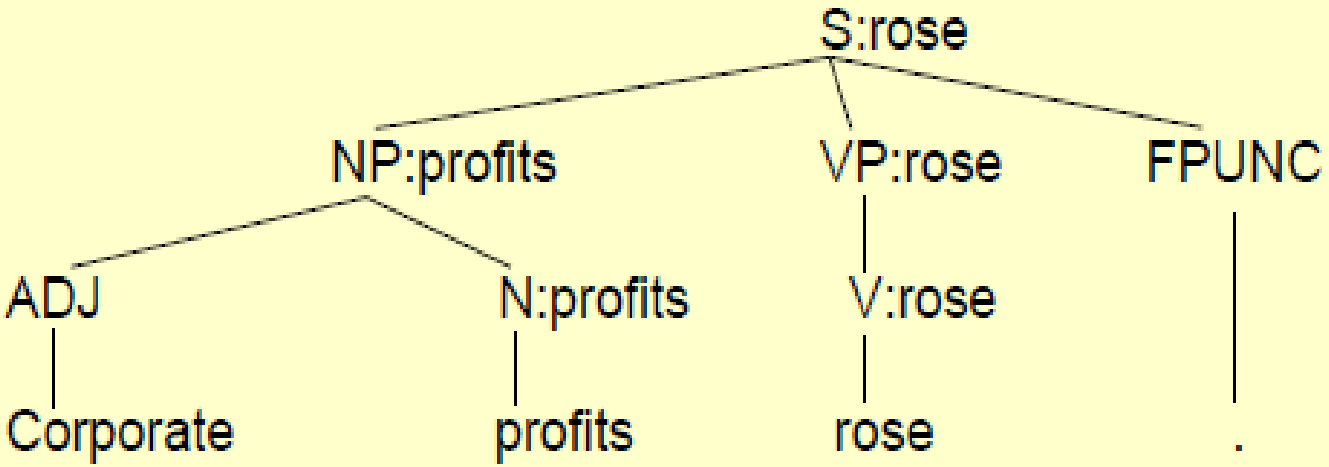
Die Mutterknoten (P) werden mit dem Blatt (~ Wort), auf das der Kopf der unmittelbaren Konstituente von P abgeleitet wird, versehen. → Der Baum enthält lexikalische Informationen.

P:h

S:rose – VP:rose – V:rose → rose

NP:profits

Charniak: Lexikalisierter Parsebaum



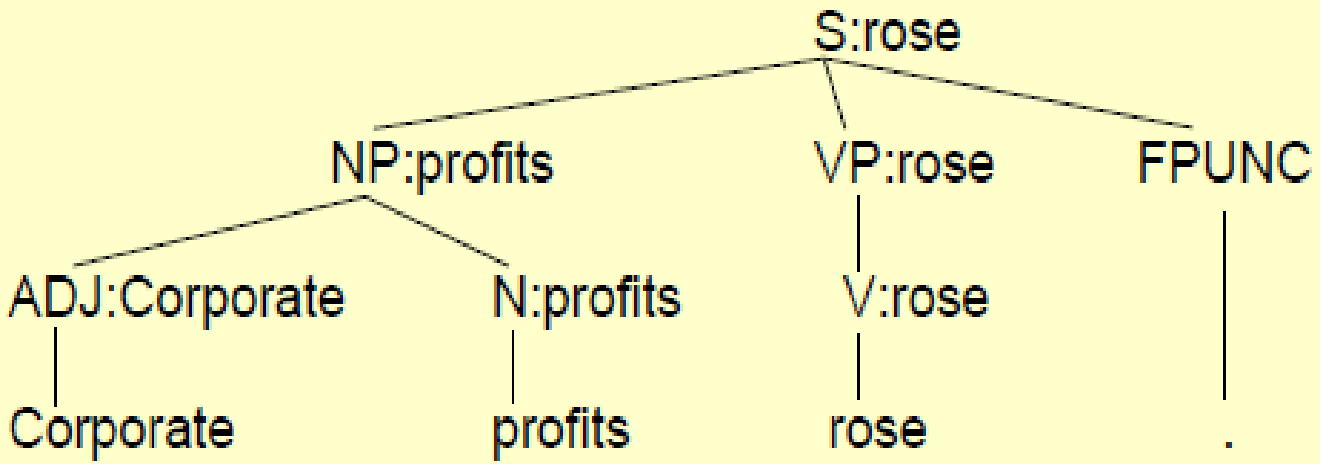
Die Mutterknoten (P) werden mit dem Blatt (~ Wort), auf das der Kopf der unmittelbaren Konstituente von P abgeleitet wird, versehen. → Der Baum enthält lexikalische Informationen.

P:h

S:rose – VP:rose – V:rose → rose

NP:profits – N:profits → profits

Charniak: Lexikalisierter Parsebaum



Die Mutterknoten (P) werden mit dem Blatt (~ Wort), auf das der Kopf der unmittelbaren Konstituente von P abgeleitet wird, versehen. → Der Baum enthält lexikalische Informationen.

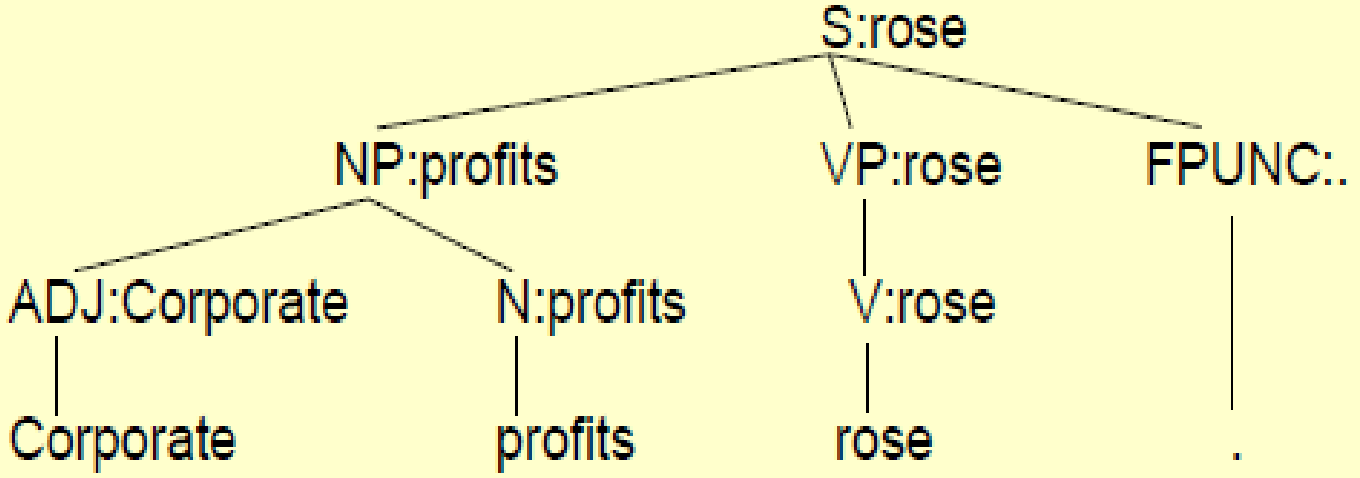
P:h

S:rose – VP:rose – V:rose → rose

NP:profits – N:profits → profits

ADJ:corporate → corporate

Charniak: Lexikalisierter Parsebaum



Die Mutterknoten (P) werden mit dem Blatt (~ Wort), auf das der Kopf der unmittelbaren Konstituente von P abgeleitet wird, versehen. → Der Baum enthält lexikalische Informationen.

P:h

S:rose – VP:rose – V:rose → rose

NP:profits – N:profits → profits

ADJ:corporate → corporate

FPUNC:. → .

Charniak: Lexikalisierten Parsebaum generieren

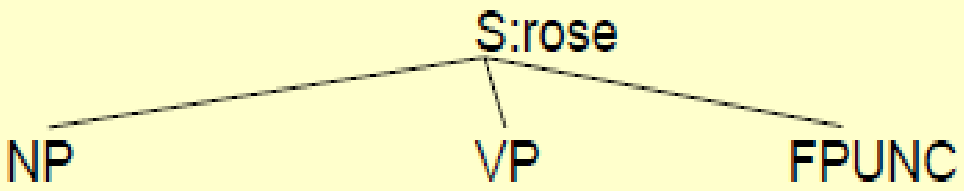
Charniak: Lexikalisierten Parsebaum generieren

S:rose

Baumwahrscheinlichkeit:

P_{Baum} = p(S:rose)

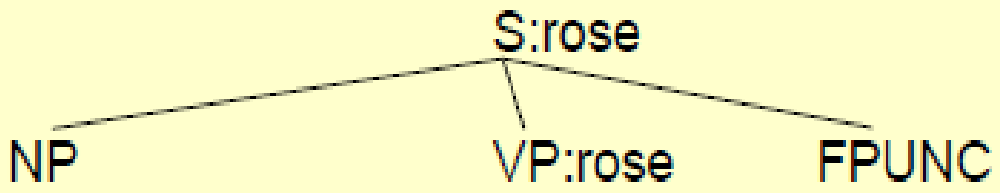
Charniak: Lexikalisierten Parsebaum generieren



Baumwahrscheinlichkeit:

P_{Baum} = p(S:rose) * p(S --> NP VP FPUNC | S:rose)

Charniak: Lexikalisierten Parsebaum generieren

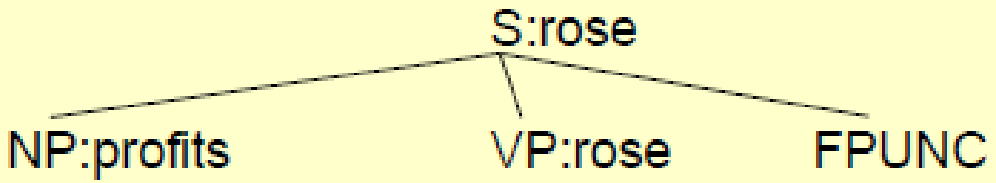


Baumwahrscheinlichkeit:

P_{Baum} = p(S:rose) * p(S --> NP VP FPUNC | S:rose)

(deterministisch)

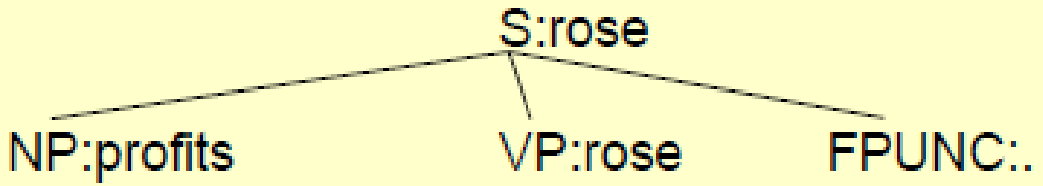
Charniak: Lexikalisierten Parsebaum generieren



Baumwahrscheinlichkeit:

P_{Baum} = p(S:rose) * p(S --> NP VP FPUNC | S:rose) * p(profits | NP, S:rose)

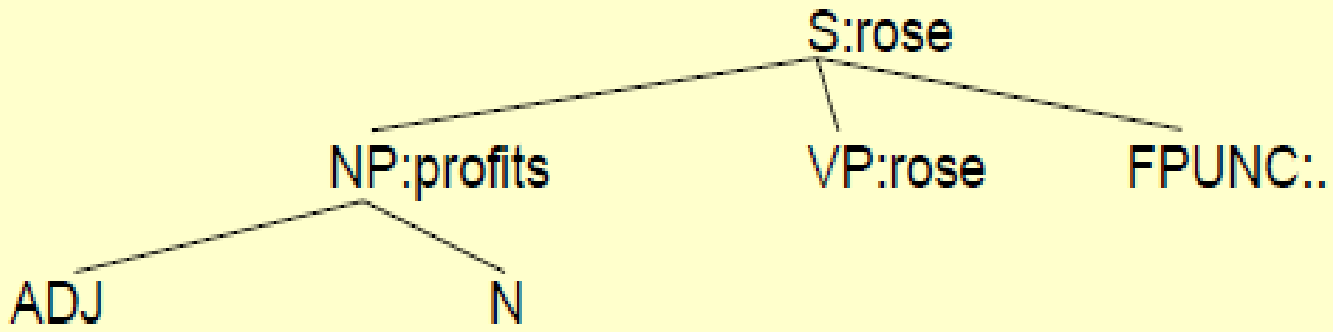
Charniak: Lexikalisierten Parsebaum generieren



Baumwahrscheinlichkeit:

P_{Baum} = p(S:rose) * p(S --> NP VP FPUNC | S:rose) * p(profits | NP, S:rose) * p(. | FPUNC, S:rose)

Charniak: Lexikalisierten Parsebaum generieren

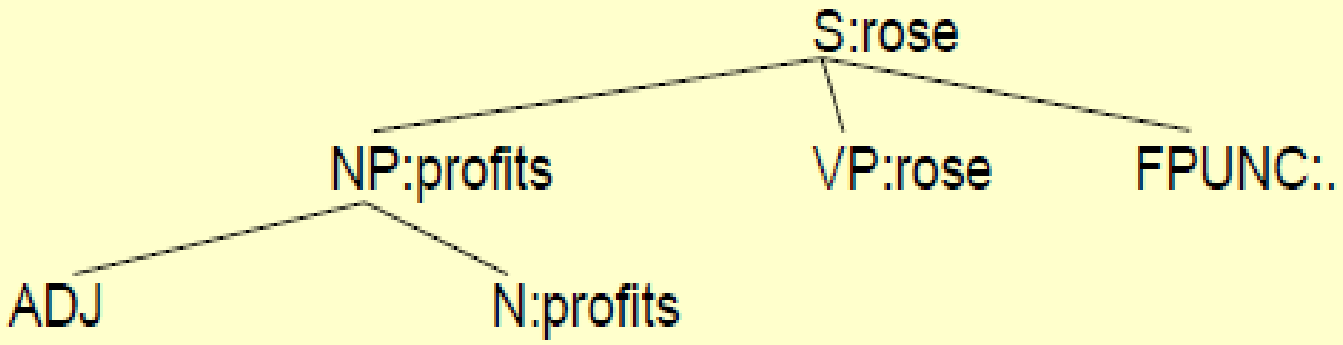


Baumwahrscheinlichkeit:

P_{Baum} = p(S:rose) * p(S --> NP VP FPUNC | S:rose) * p(profits | NP, S:rose) * p(. | FPUNC, S:rose)

* p(NP --> ADJ N | NP:profits)

Charniak: Lexikalisierten Parsebaum generieren



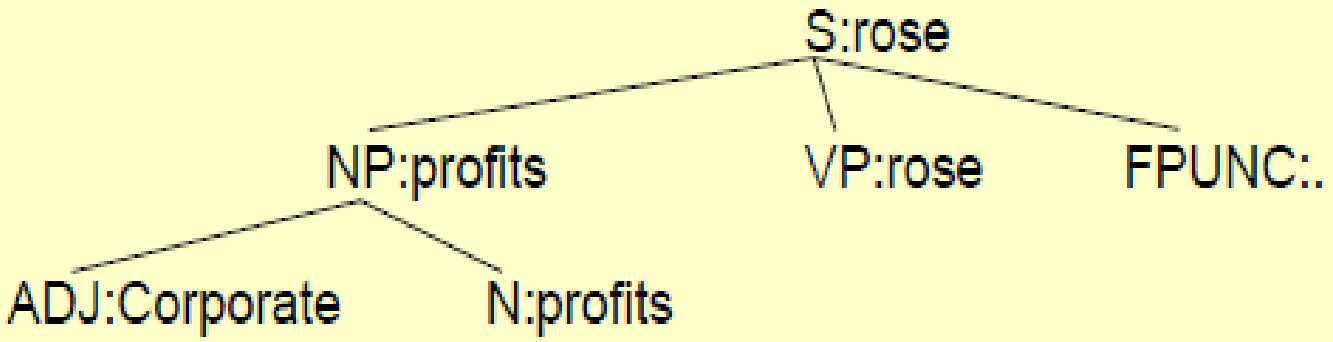
Baumwahrscheinlichkeit:

P_{Baum} = p(S:rose) * p(S --> NP VP FPUNC | S:rose) * p(profits | NP, S:rose) * p(. | FPUNC, S:rose)

* p(NP --> ADJ N | NP:profits)

(deterministisch)

Charniak: Lexikalisierten Parsebaum generieren

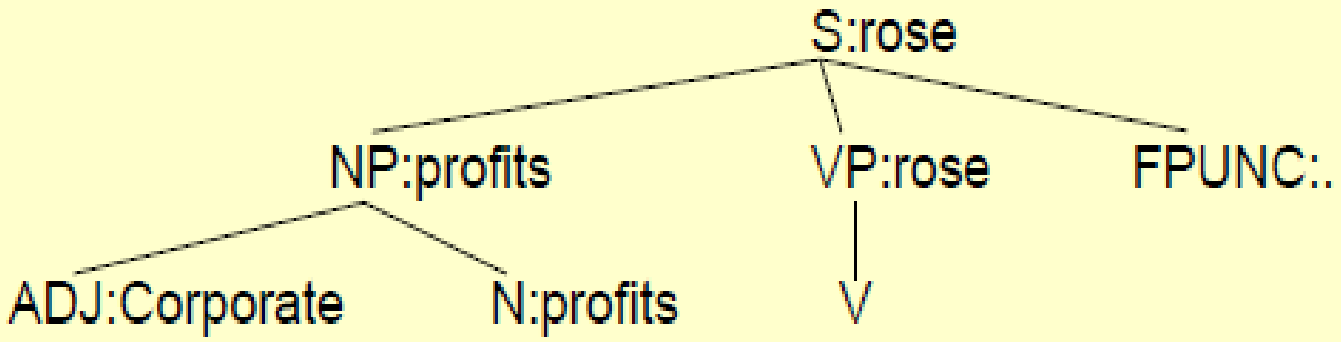


Baumwahrscheinlichkeit:

P_{Baum} = p(S:rose) * p(S --> NP VP FPUNC | S:rose) * p(profits | NP, S:rose) * p(. | FPUNC, S:rose)

* p(NP --> ADJ N | NP:profits) * p(corporate | ADJ, NP:profits)

Charniak: Lexikalisierten Parsebaum generieren

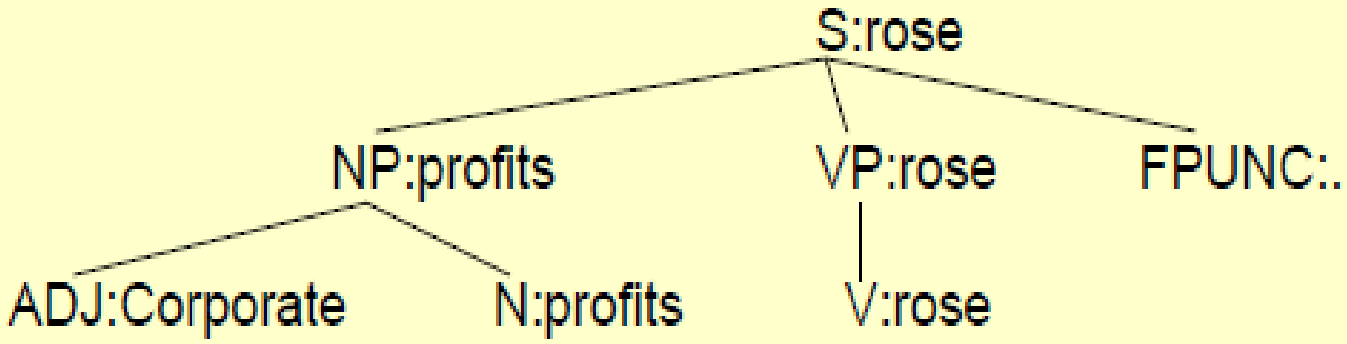


Baumwahrscheinlichkeit:

P_{Baum} = p(S:rose) * p(S --> NP VP FPUNC | S:rose) * p(profits | NP, S:rose) * p(. | FPUNC, S:rose)

* p(NP --> ADJ N | NP:profits) * p(corporate | ADJ, NP:profits) * p(VP --> V | VP:rose)

Charniak: Lexikalisierten Parsebaum generieren



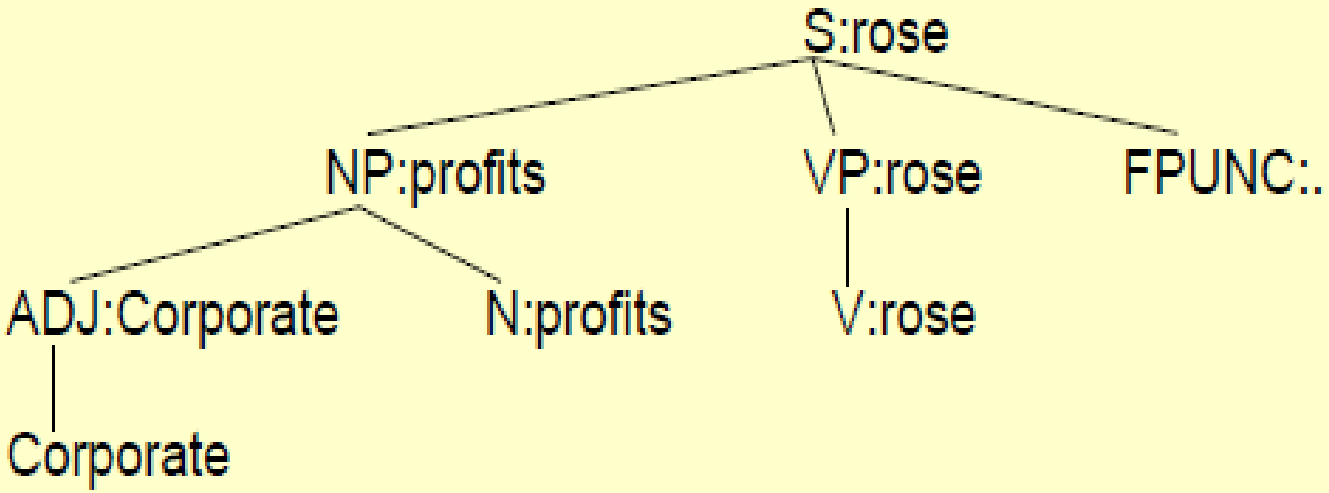
Baumwahrscheinlichkeit:

P_{Baum} = p(S:rose) * p(S --> NP VP FPUNC | S:rose) * p(profits | NP, S:rose) * p(. | FPUNC, S:rose)

* p(NP --> ADJ N | NP:profits) * p(corporate | ADJ, NP:profits) * p(VP --> V | VP:rose)

(deterministisch)

Charniak: Lexikalisierten Parsebaum generieren



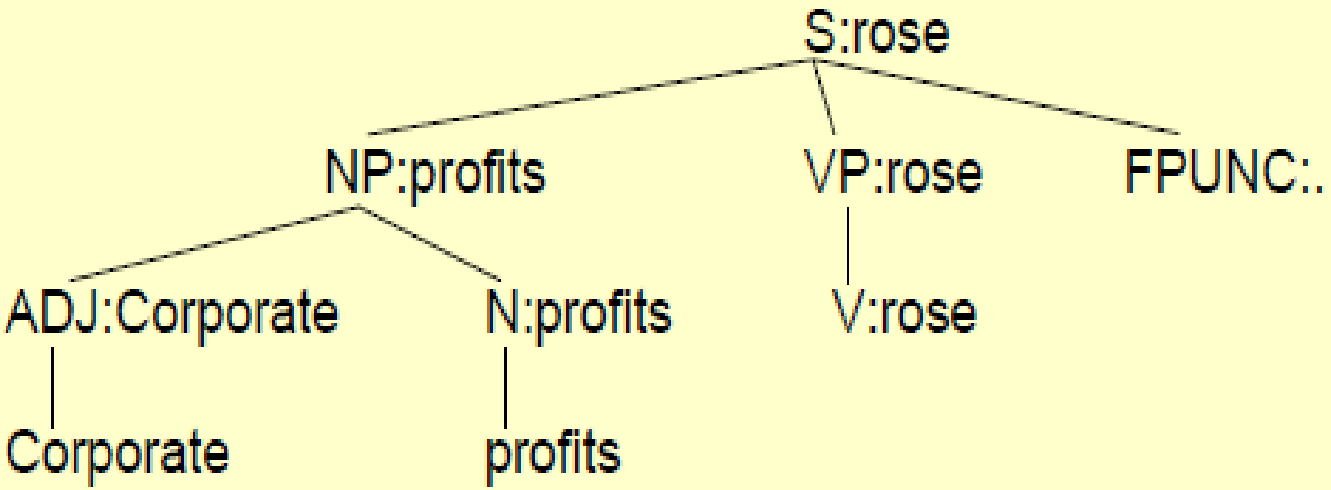
Baumwahrscheinlichkeit:

P_{Baum} = p(S:rose) * p(S --> NP VP FPUNC | S:rose) * p(profits | NP, S:rose) * p(. | FPUNC, S:rose)

* p(NP --> ADJ N | NP:profits) * p(corporate | ADJ, NP:profits) * p(VP --> V | VP:rose)

* p(ADJ --> corporate | ADJ:corporate)

Charniak: Lexikalisierten Parsebaum generieren



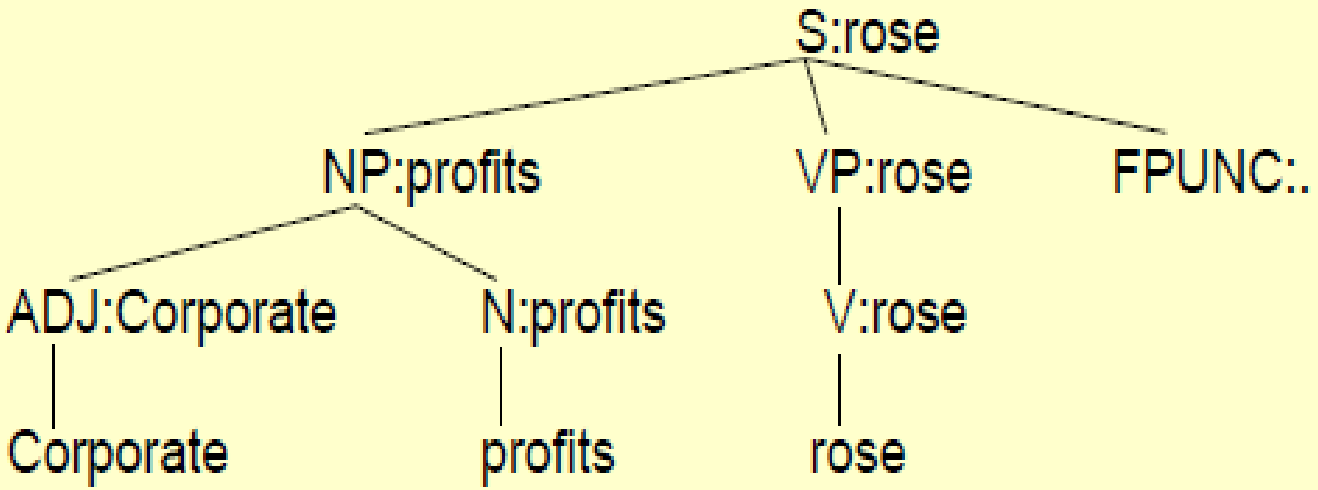
Baumwahrscheinlichkeit:

P_{Baum} = p(S:rose) * p(S --> NP VP FPUNC | S:rose) * p(profits | NP, S:rose) * p(. | FPUNC, S:rose)

* p(NP --> ADJ N | NP:profits) * p(corporate | ADJ, NP:profits) * p(VP --> V | VP:rose)

* p(ADJ --> corporate | ADJ:corporate) * p(N --> profits | N:profits)

Charniak: Lexikalisierten Parsebaum generieren



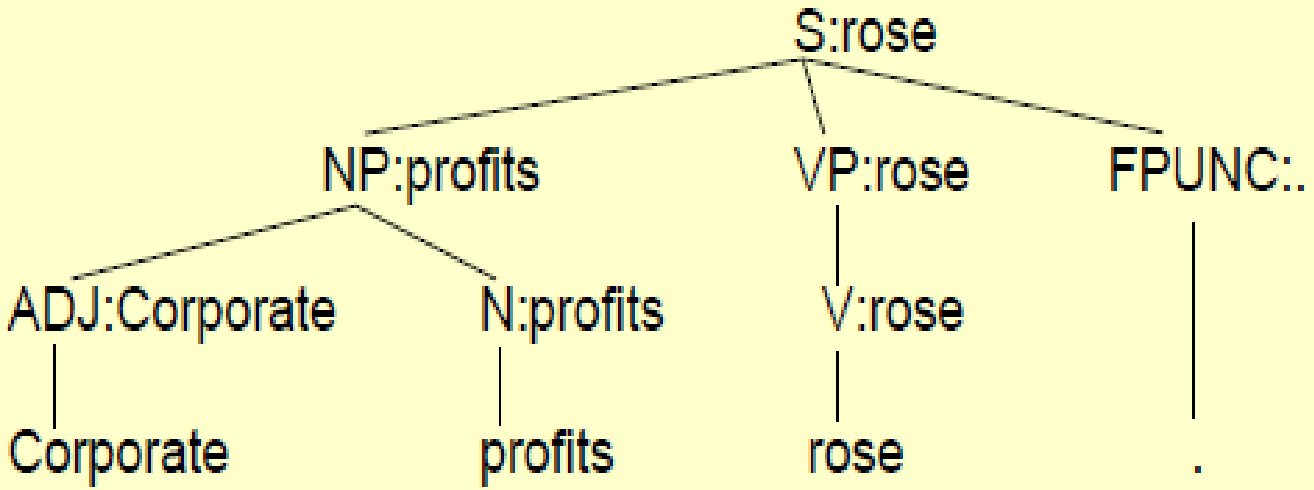
Baumwahrscheinlichkeit:

P_{Baum} = p(S:rose) * p(S --> NP VP FPUNC | S:rose) * p(profits | NP, S:rose) * p(. | FPUNC, S:rose)

* p(NP --> ADJ N | NP:profits) * p(corporate | ADJ, NP:profits) * p(VP --> V | VP:rose)

* p(ADJ --> corporate | ADJ:corporate) * p(N --> profits | N:profits) * p(V --> rose | V:rose)

Charniak: Lexikalisierten Parsebaum generieren



Baumwahrscheinlichkeit:

P_{Baum} = p(S:rose) * p(S --> NP VP FPUNC | S:rose) * p(profits | NP, S:rose) * p(. | FPUNC, S:rose)

* p(NP --> ADJ N | NP:profits) * p(corporate | ADJ, NP:profits) * p(VP --> V | VP:rose)

* p(ADJ --> corporate | ADJ:corporate) * p(N --> profits | N:profits) * p(V --> rose | V:rose)

* p(FPUNC --> . | FPUNC:.)

Michael John Collins

Michael John Collins

- Charniak (1997)
  - statistisches Modell
  - generatives Modell
  - lexikalisiertes Modell

    → Performanzvorteil von ca. 10%
  - explizite Grammatik
  - kein explizites Tagging
  - Smoothing
  - Große Performanz: 86-87% Korrektheit
- Beispielsatz: *Corporate profits rose.*

Collins (1996, 1997 – vor und nach Charniak)

Michael John Collins

- Charniak (1997)
  - statistisches Modell
  - generatives Modell
  - lexikalisiertes Modell

    → Performanzvorteil von ca. 10%
  - explizite Grammatik
  - kein explizites Tagging
  - Smoothing
  - Große Performanz: 86-87% Korrektheit
- Beispielsatz: *Corporate profits rose.*

Collins (1996, 1997)

|| Collins

Michael John Collins

- Charniak (1997)
  - statistisches Modell
  - generatives Modell
  - lexikalisiertes Modell

    → Performanzvorteil von ca. 10%
  - explizite Grammatik
  - kein explizites Tagging
  - Smoothing
  - Große Performanz: 86-87% Korrektheit
- Beispielsatz: *Corporate profits rose.*

Collins (1996, 1997)

|| Collins

|| Collins

Michael John Collins

- Charniak (1997) Collins (1996, 1997)
  - statistisches Modell || Collins
  - generatives Modell || Collins
  - lexikalisiertes Modell || Collins

    → Performanzvorteil von ca. 10%
  - explizite Grammatik
  - kein explizites Tagging
  - Smoothing
  - Große Performanz: 86-87% Korrektheit
- Beispielsatz: *Corporate profits rose.*

Michael John Collins

- Charniak (1997) — Collins (1996, 1997)
  - statistisches Modell || Collins
  - generatives Modell || Collins
  - lexikalisiertes Modell || Collins
    → Performanzvorteil von ca. 10%
  - explizite Grammatik ↔ Collins: neue Grammatik
  - kein explizites Tagging
  - Smoothing
  - Große Performanz: 86-87% Korrektheit
- Beispielsatz: *Corporate profits rose.*

Michael John Collins

- Charniak (1997) — Collins (1996, 1997)
  - statistisches Modell || Collins
  - generatives Modell || Collins
  - lexikalisiertes Modell || Collins
    → Performanzvorteil von ca. 10%
  - explizite Grammatik ↔ Collins: neue Grammatik
  - kein explizites Tagging ↔ Collins: Tagging-Phase
  - Smoothing
  - Große Performanz: 86-87% Korrektheit
- Beispielsatz: *Corporate profits rose.*

Michael John Collins

- Charniak (1997) — Collins (1996, 1997)
  - statistisches Modell || Collins
  - generatives Modell || Collins
  - lexikalisiertes Modell || Collins
    → Performanzvorteil von ca. 10%
  - explizite Grammatik ↔ Collins: neue Grammatik
  - kein explizites Tagging ↔ Collins: Tagging-Phase
  - Smoothing || Collins
  - Große Performanz: 86-87% Korrektheit
- Beispielsatz: *Corporate profits rose.*

Collins: Hauptmerkmale (1)

- Statistisches, generatives, lexikalistisches Modell (|| Charniak)

Collins: Hauptmerkmale (1)

- Statistisches, generatives, lexikalistisches Modell (|| Charniak)
- Markovisierung der Regeln (↔ Charniak)
    - Die Regeln werden aufgebrochen („zerhackt”) und während des Generierungsprozesses nach bestimmten Wahrscheinlichkeiten erzeugt!

Collins: Hauptmerkmale (1)

- Statistisches, generatives, lexikalistisches Modell (|| Charniak)
- Markovisierung der Regeln (↔ Charniak)
    - Die Regeln werden aufgebrochen („zerhackt”) und während des Generierungsprozesses nach bestimmten Wahrscheinlichkeiten erzeugt!
        - 1. Kopf + 2. rechte Argumente + 3. linke Argumente

Collins: Hauptmerkmale (1)

- Statistisches, generatives, lexikalistisches Modell (|| Charniak)
- Markovisierung der Regeln (↔ Charniak)
    - Die Regeln werden aufgebrochen („zerhackt”) und während des Generierungsprozesses nach bestimmten Wahrscheinlichkeiten erzeugt!
        - 1. Kopf + 2. rechte Argumente + 3. linke Argumente
    - (↔ Charniak: explizite Grammatik ~ Die Regeln werden von der Baumbank abgelesen und als Ganzes verwendet.)
- ...

Collins: aufgebrochene Regeln Schritt für Schritt generieren

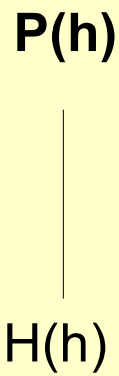
Collins: aufgebrochene Regeln Schritt für Schritt generieren

P(h)

Regelwahrscheinlichkeit:

p_{Regel} =

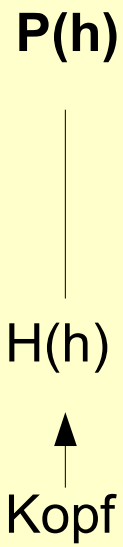
Collins: aufgebrochene Regeln Schritt für Schritt generieren



Regelwahrscheinlichkeit:

p_{Regel} =

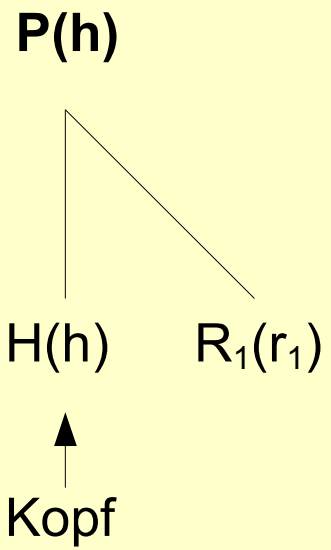
Collins: aufgebrochene Regeln Schritt für Schritt generieren



Regelwahrscheinlichkeit:

$p_{Regel} = p_{Kopf}(H \mid P, h)$

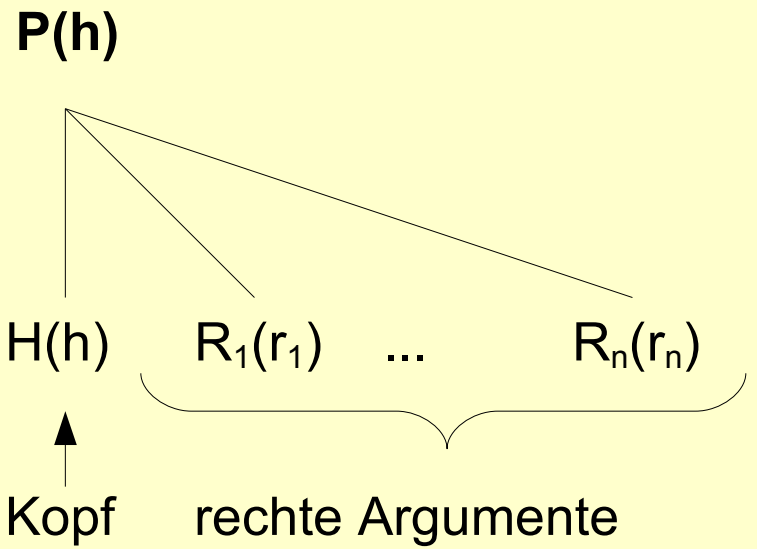
Collins: aufgebrochene Regeln Schritt für Schritt generieren



Regelwahrscheinlichkeit:

$p_{Regel} = p_{Kopf}(H \mid P, h)$

Collins: aufgebrochene Regeln Schritt für Schritt generieren

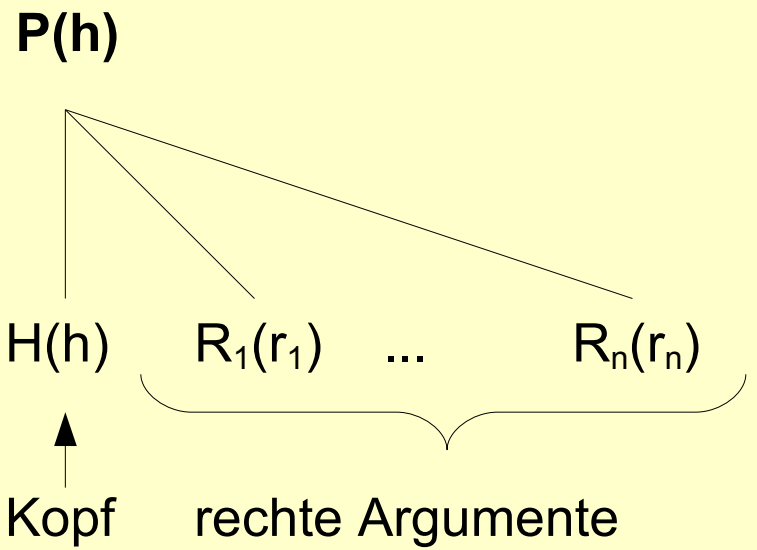


Regelwahrscheinlichkeit:

$p_{Regel} = p_{Kopf}(H \mid P, h)$

$$* \prod_{i=0}^{n} p_{Rechts}(R_i(r_i) \mid P, H, h)$$

Collins: aufgebrochene Regeln Schritt für Schritt generieren

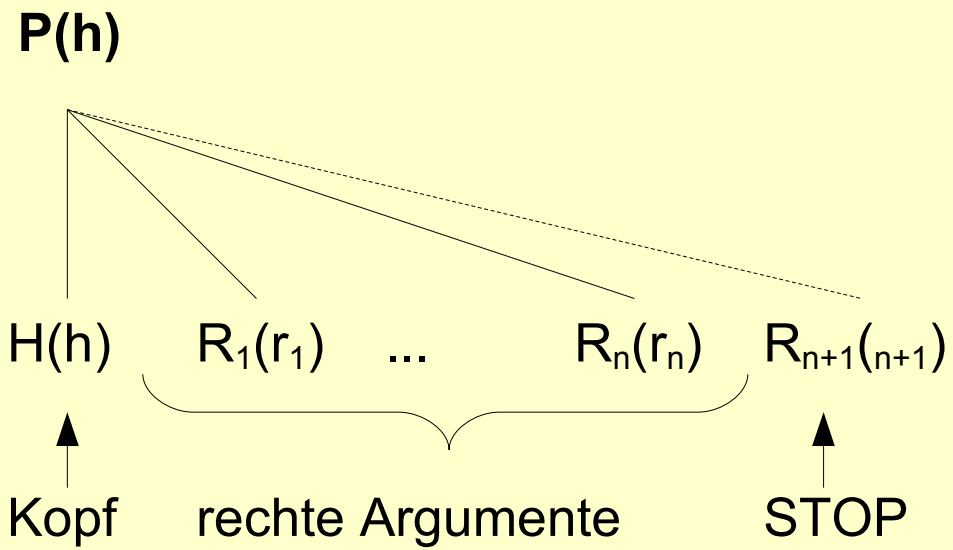


Regelwahrscheinlichkeit:

Wo hört man auf?

$$p_{Regel} = p_{Kopf}(H \mid P, h) * \prod_{i=0}^{n} p_{Rechts}(R_i(r_i) \mid P, H, h)$$

Collins: aufgebrochene Regeln Schritt für Schritt generieren

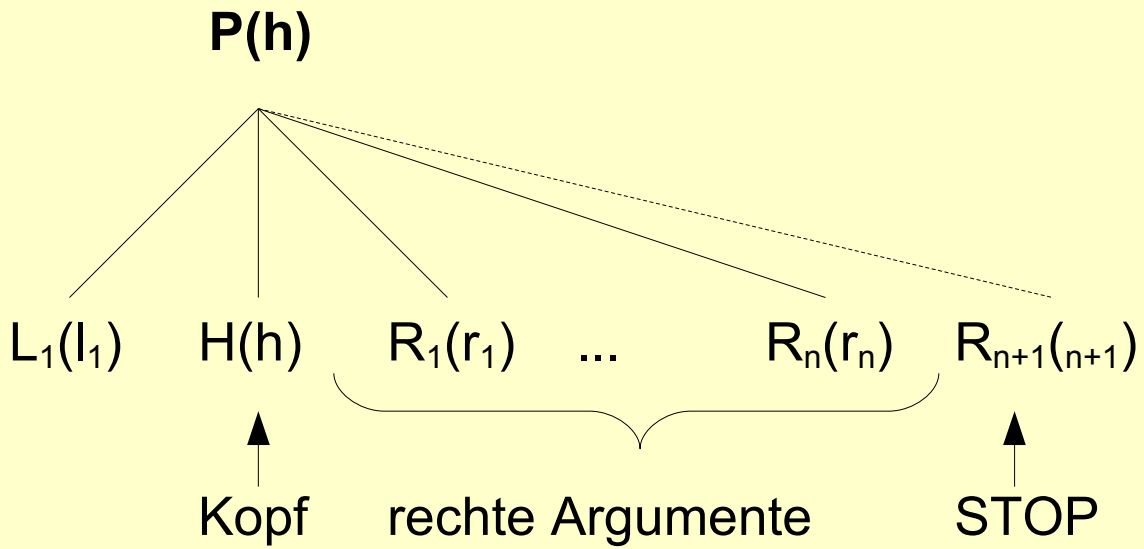


Regelwahrscheinlichkeit:

$p_{Regel} = p_{Kopf}(H \mid P, h)$

$$* \prod_{i=0}^{n} p_{Rechts}(R_i(r_i) \mid P, H, h) * p_{Rechts}(STOP \mid P, H, h)$$

Collins: aufgebrochene Regeln Schritt für Schritt generieren

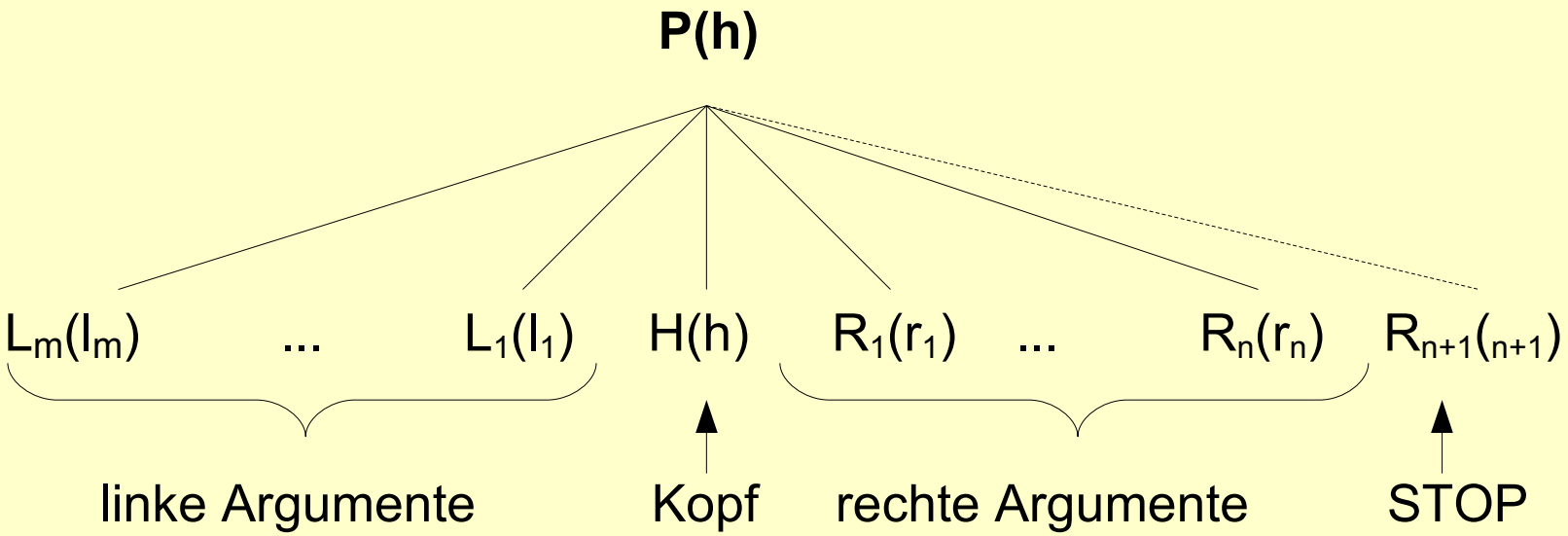


Regelwahrscheinlichkeit:

$p_{Regel} = p_{Kopf}(H \mid P, h)$

$$* \prod_{i=0}^{n} p_{Rechts}(R_i(r_i) \mid P, H, h) * p_{Rechts}(STOP \mid P, H, h)$$

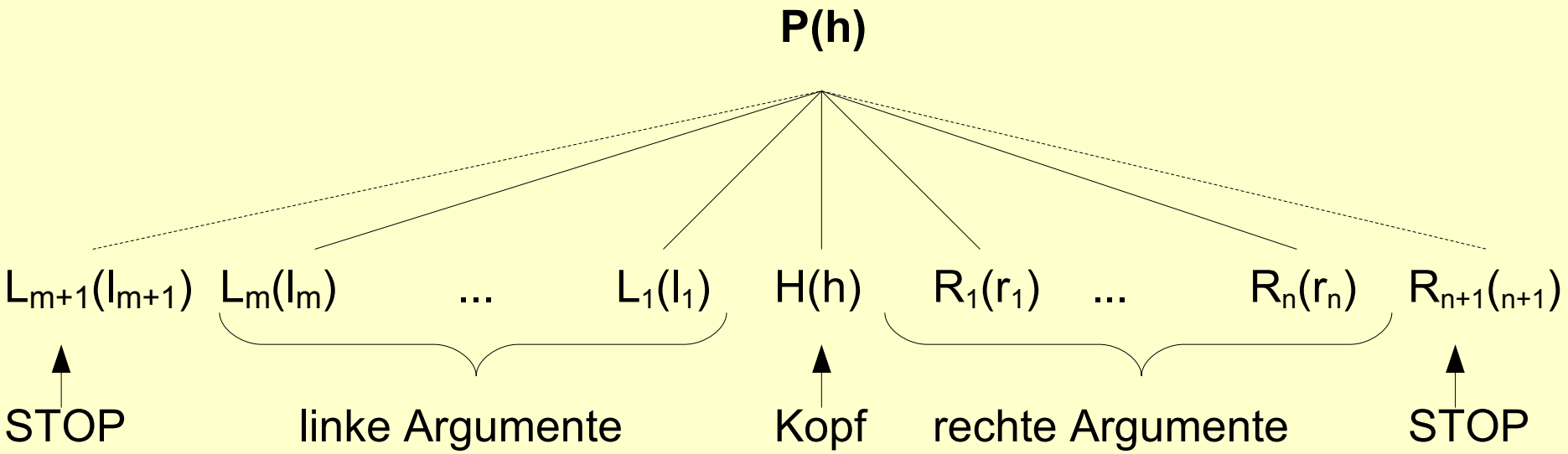
Collins: aufgebrochene Regeln Schritt für Schritt generieren



Regelwahrscheinlichkeit:

$$p_{Regel} = p_{Kopf}(H \mid P, h) * \prod_{i=0}^{n} p_{Rechts}(R_i(r_i) \mid P, H, h) * p_{Rechts}(STOP \mid P, H, h) * \prod_{j=0}^{m} p_{Links}(L_j(l_j) \mid P, H, h)$$

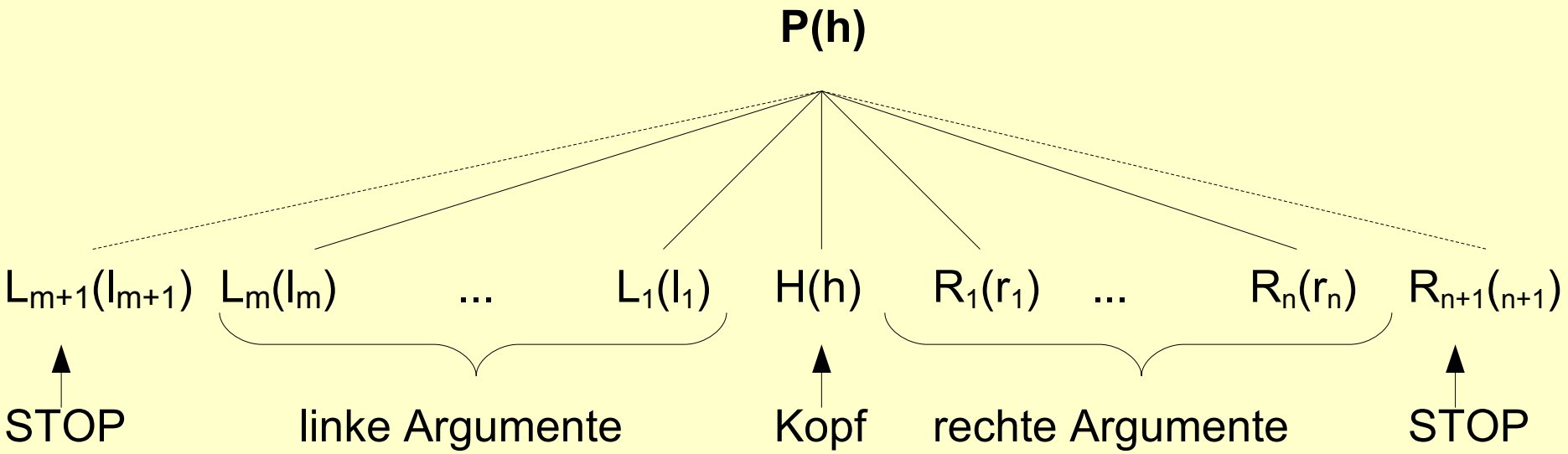
Collins: aufgebrochene Regeln Schritt für Schritt generieren



Regelwahrscheinlichkeit:

$$p_{Regel} = p_{Kopf}(H \mid P, h) * {}_{i=0}\Pi^{n}\, p_{Rechts}(R_i(r_i) \mid P, H, h) * p_{Rechts}(STOP \mid P, H, h) * {}_{j=0}\Pi^{m}\, p_{Links}(L_j(l_j) \mid P, H, h) * p_{Links}(STOP \mid P, H, h)$$

Collins: aufgebrochene Regeln Schritt für Schritt generieren



Regelwahrscheinlichkeit:

$p_{Regel} = p_{Kopf}(H \mid P, h)$

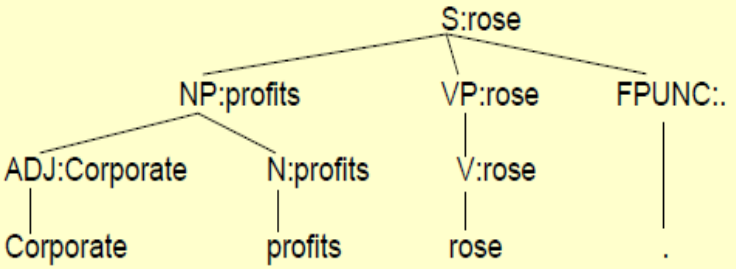
$$* \prod_{i=0}^{n} p_{Rechts}(R_i(r_i) \mid P, H, h) * p_{Rechts}(STOP \mid P, H, h)$$

$$* \prod_{j=0}^{m} p_{Links}(L_j(l_j) \mid P, H, h) * p_{Links}(STOP \mid P, H, h)$$

Es werden jeweils nur der Mutterknoten, der Kopf und die Kategorie des Kopfes als angegeben betrachtet.

Collins: *Corporate profits rose.* – Parsebaum generieren

Collins: Parsebaum generieren

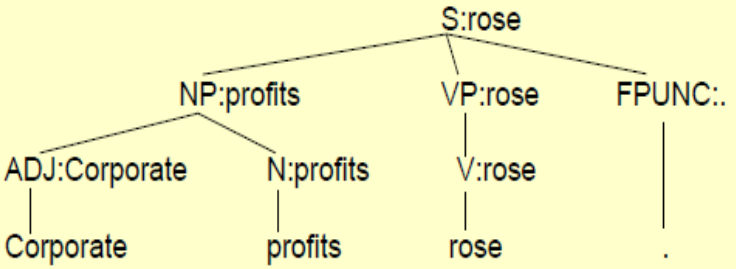


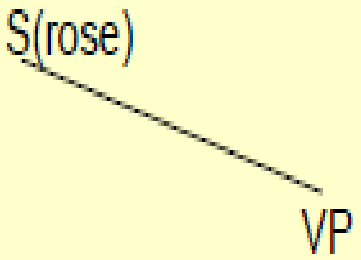
S(rose)

Baumwahrscheinlichkeit:

- P_{Baum} = p(S, rose)

Collins: Parsebaum generieren

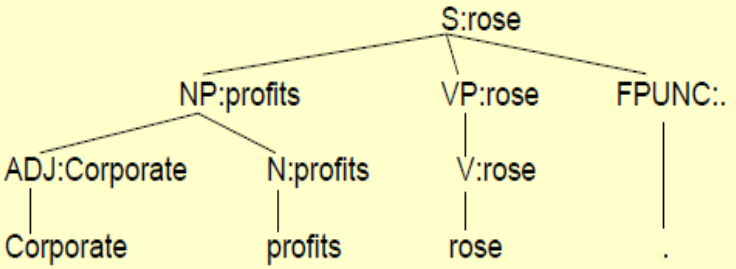


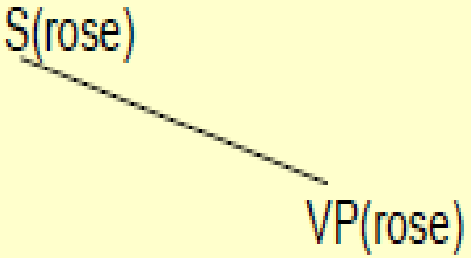


Baumwahrscheinlichkeit:

- $P_{Baum} = p(S, rose) * p_H(VP \mid S, rose)$

Collins: Parsebaum generieren



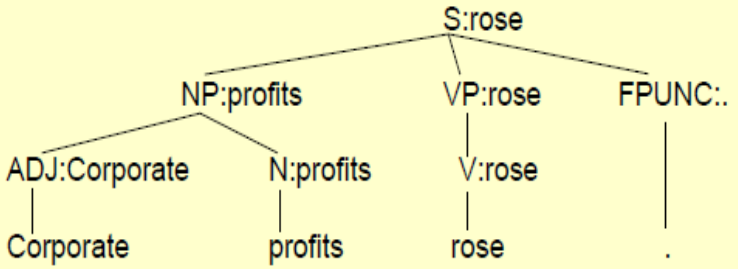


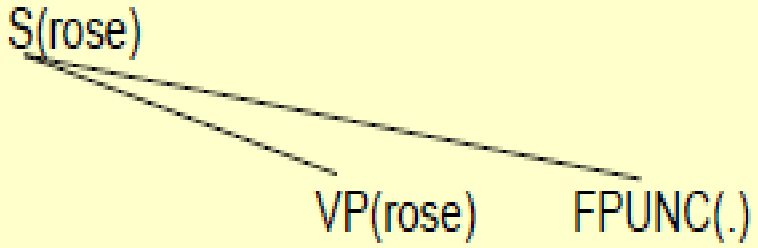
Baumwahrscheinlichkeit:

- $P_{Baum} = p(S, rose) * p_H(VP \mid S, rose)$

(deterministisch)

Collins: Parsebaum generieren

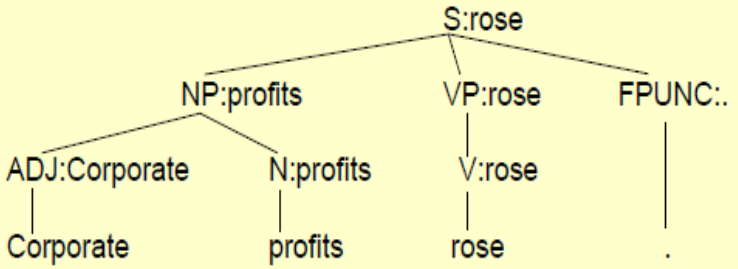


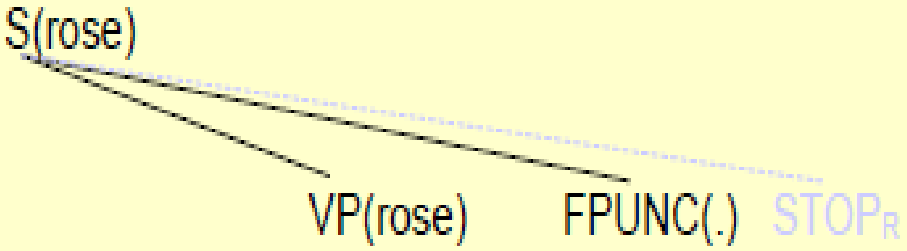


Baumwahrscheinlichkeit:

- P_{Baum} = p(S, rose) * p_H(VP | S, rose) * p_R(FPUNC(.) | S, rose, VP)

Collins: Parsebaum generieren

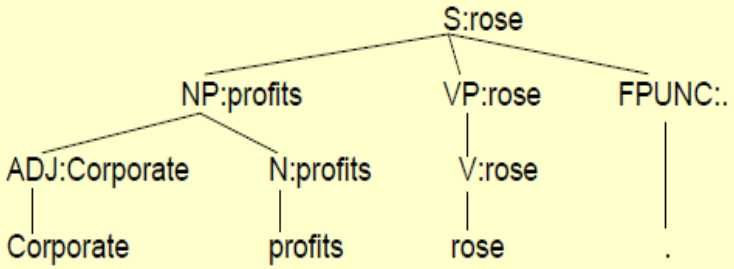


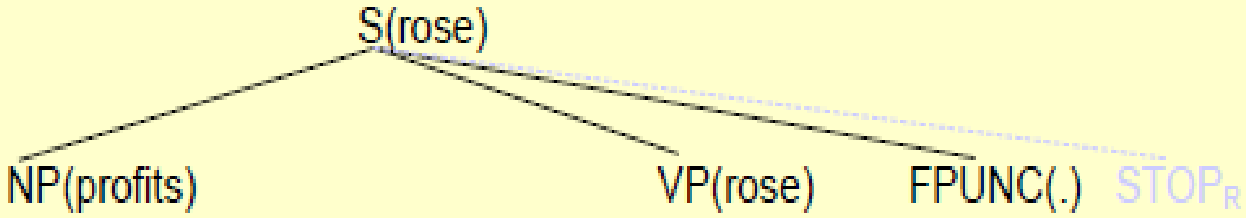


Baumwahrscheinlichkeit:

- P_{Baum} = p(S, rose) * p_H(VP | S, rose) * p_R(FPUNC(.) | S, rose, VP)* p_R($STOP_R$ | S, rose, VP)

Collins: Parsebaum generieren

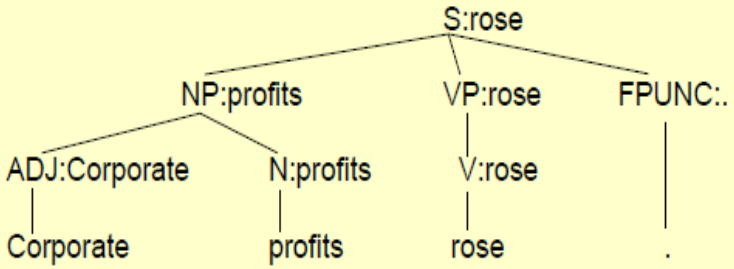


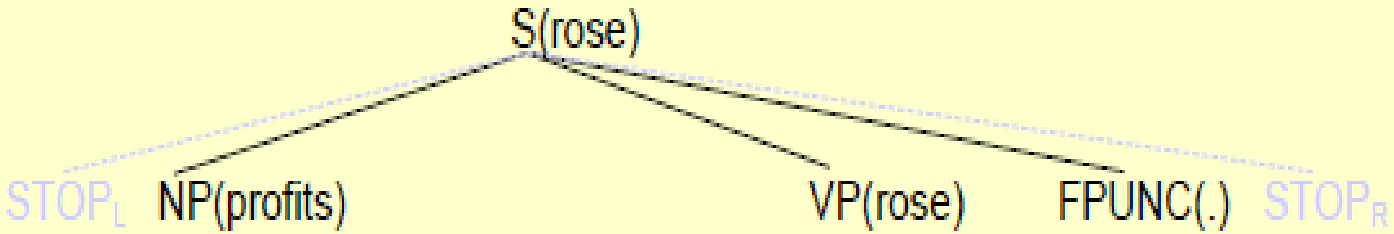


Baumwahrscheinlichkeit:

- P_{Baum} = p(S, rose) * p_H(VP | S, rose) * p_R(FPUNC(.) | S, rose, VP)* p_R($STOP_R$ | S, rose, VP) * p_L(NP(profits) | S, rose, VP)

Collins: Parsebaum generieren

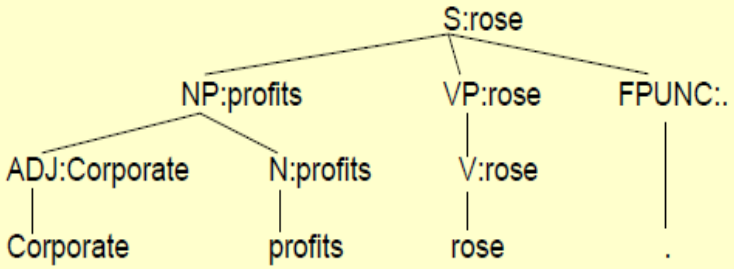


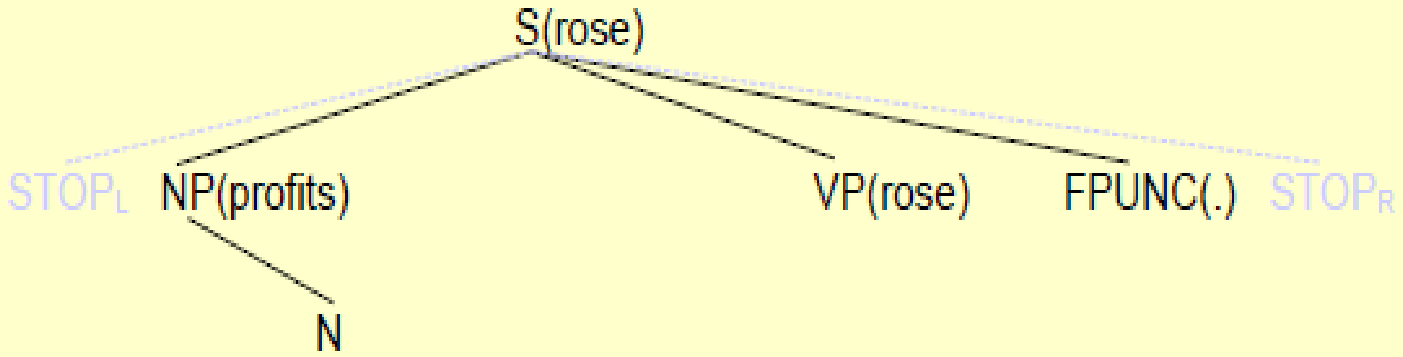


Baumwahrscheinlichkeit:

- P_{Baum} = p(S, rose) * p_H(VP | S, rose) * p_R(FPUNC(.) | S, rose, VP)* p_R($STOP_R$ | S, rose, VP) * p_L(NP(profits) | S, rose, VP) * p_L($STOP_L$ | S, rose, VP)

Collins: Parsebaum generieren

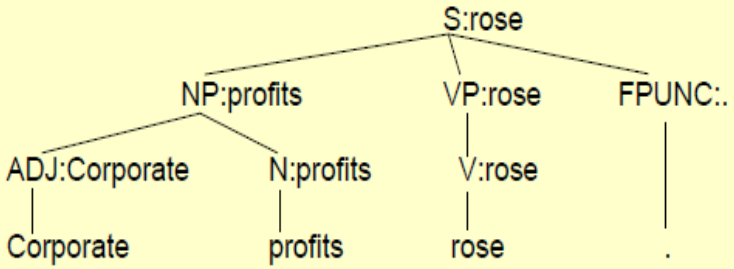


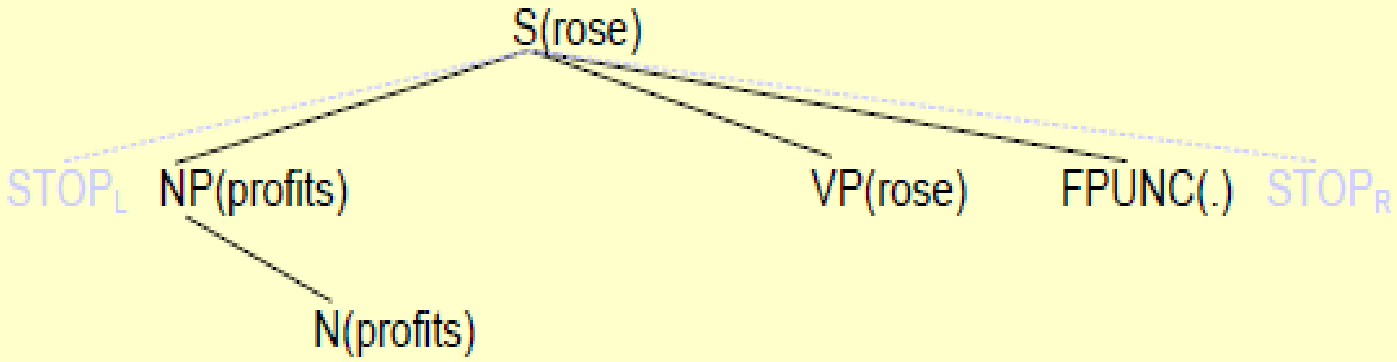


Baumwahrscheinlichkeit:

- P_{Baum} = p(S, rose) * p_H(VP | S, rose) * p_R(FPUNC(.) | S, rose, VP)* p_R($STOP_R$ | S, rose, VP) * p_L(NP(profits) | S, rose, VP) * p_L($STOP_L$ | S, rose, VP) * p_H(N | NP, profits)

Collins: Parsebaum generieren



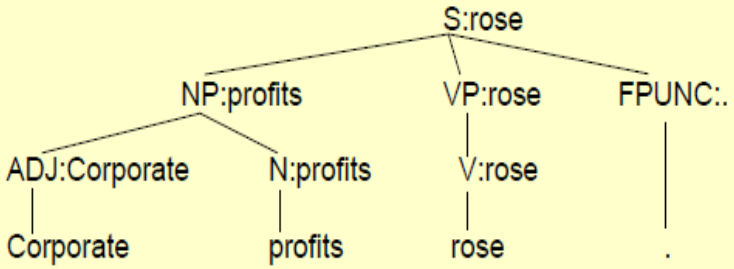


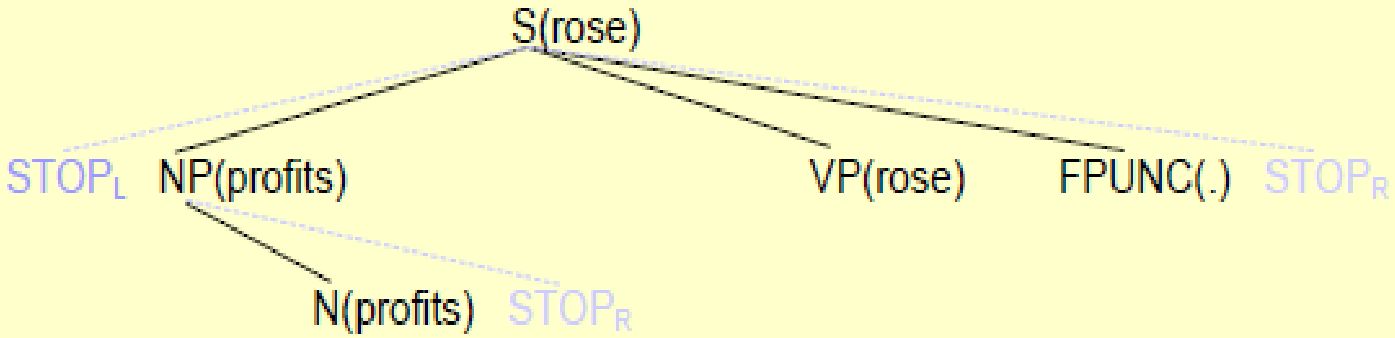
Baumwahrscheinlichkeit:

- P_{Baum} = p(S, rose) * p_H(VP | S, rose) * p_R(FPUNC(.) | S, rose, VP)* p_R($STOP_R$ | S, rose, VP) * p_L(NP(profits) | S, rose, VP) * p_L($STOP_L$ | S, rose, VP) * p_H(N | NP, profits)

(deterministisch)

Collins: Parsebaum generieren

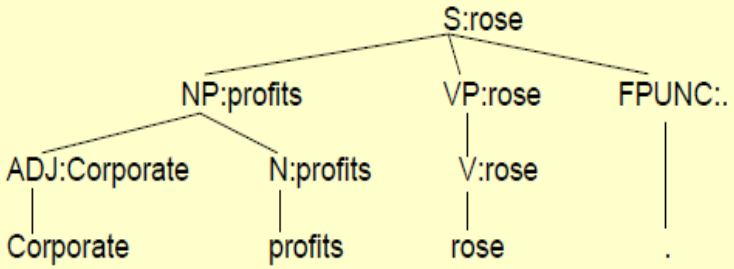


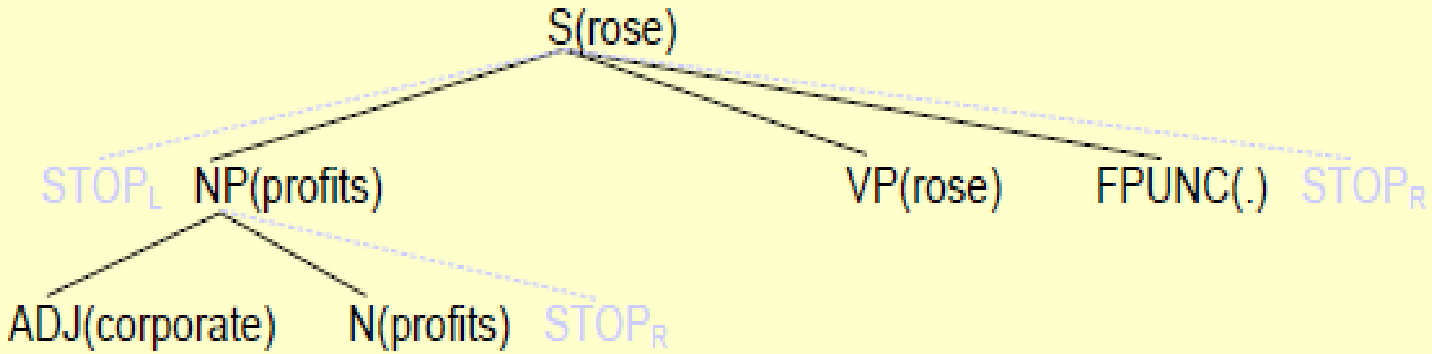


Baumwahrscheinlichkeit:

- P_{Baum} = p(S, rose) * p_H(VP | S, rose) * p_R(FPUNC(.) | S, rose, VP)* p_R($STOP_R$ | S, rose, VP) * p_L(NP(profits) | S, rose, VP) * p_L($STOP_L$ | S, rose, VP) * p_H(N | NP, profits) * p_R($STOP_R$ | NP, profits, N)

Collins: Parsebaum generieren

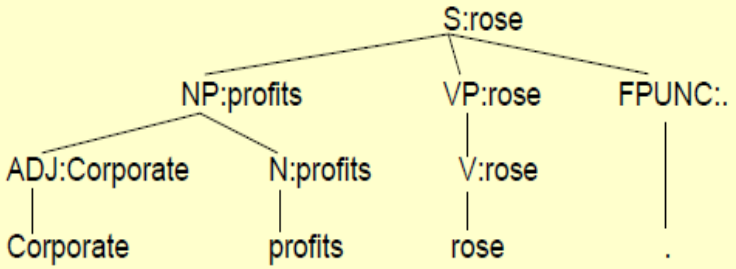


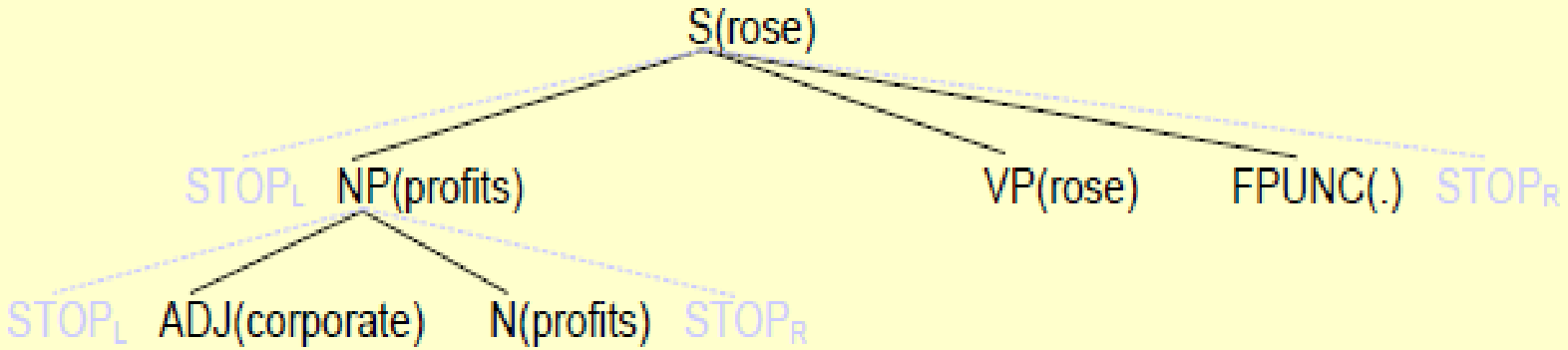


Baumwahrscheinlichkeit:

- P_{Baum} = p(S, rose) * p_H(VP | S, rose) * p_R(FPUNC(.) | S, rose, VP)* p_R($STOP_R$ | S, rose, VP) * p_L(NP(profits) | S, rose, VP) * p_L($STOP_L$ | S, rose, VP) * p_H(N | NP, profits) * p_R($STOP_R$ | NP, profits, N) * p_L(ADJ, corporate | NP, profits, N)

Collins: Parsebaum generieren

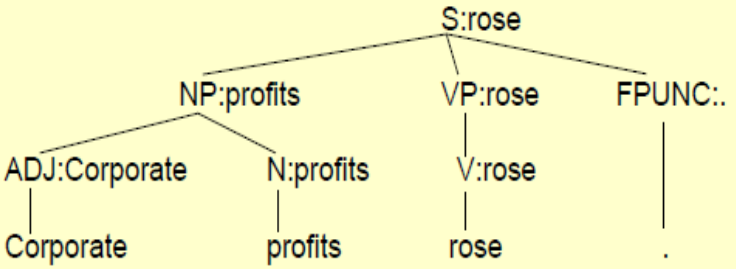


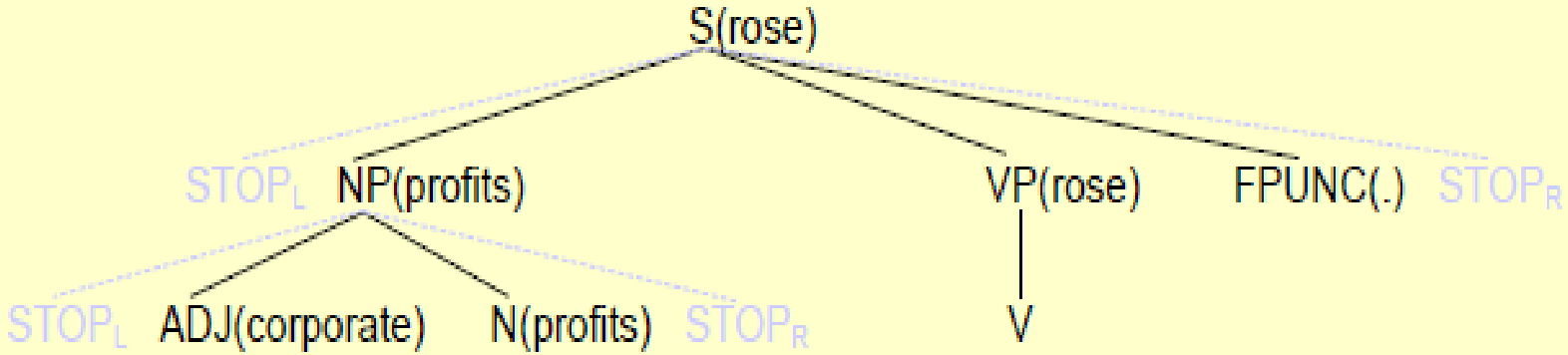


Baumwahrscheinlichkeit:

- P_{Baum} = p(S, rose) * p_H(VP | S, rose) * p_R(FPUNC(.) | S, rose, VP) * p_R($STOP_R$ | S, rose, VP) * p_L(NP(profits) | S, rose, VP) * p_L($STOP_L$ | S, rose, VP) * p_H(N | NP, profits) * p_R($STOP_R$ | NP, profits, N) * p_L(ADJ, corporate | NP, profits, N) * p_L($STOP_L$ | NP, profits, N)

Collins: Parsebaum generieren

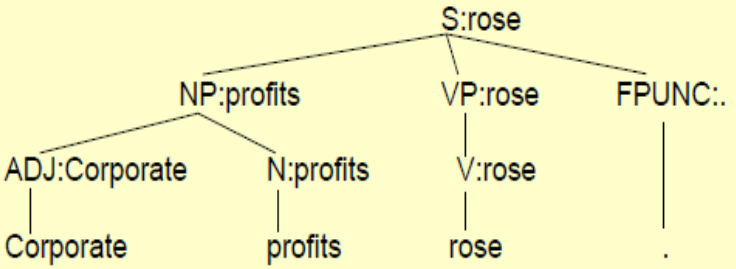


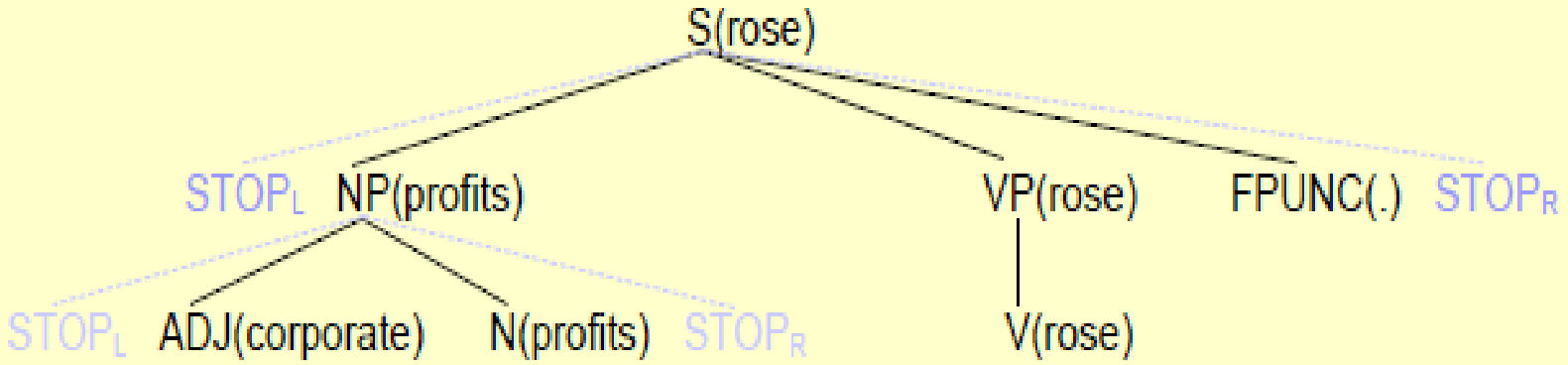


Baumwahrscheinlichkeit:

- P_{Baum} = p(S, rose) * p_H(VP | S, rose) * p_R(FPUNC(.) | S, rose, VP) * p_R($STOP_R$ | S, rose, VP) * p_L(NP(profits) | S, rose, VP) * p_L($STOP_L$ | S, rose, VP) * p_H(N | NP, profits) * p_R($STOP_R$ | NP, profits, N) * p_L(ADJ, corporate | NP, profits, N) * p_L($STOP_L$ | NP, profits, N) * p_H(V | VP, rose)

Collins: Parsebaum generieren



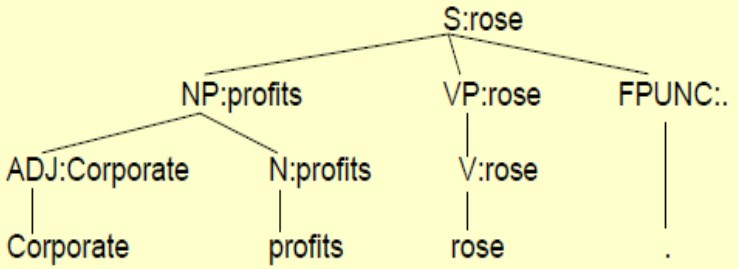


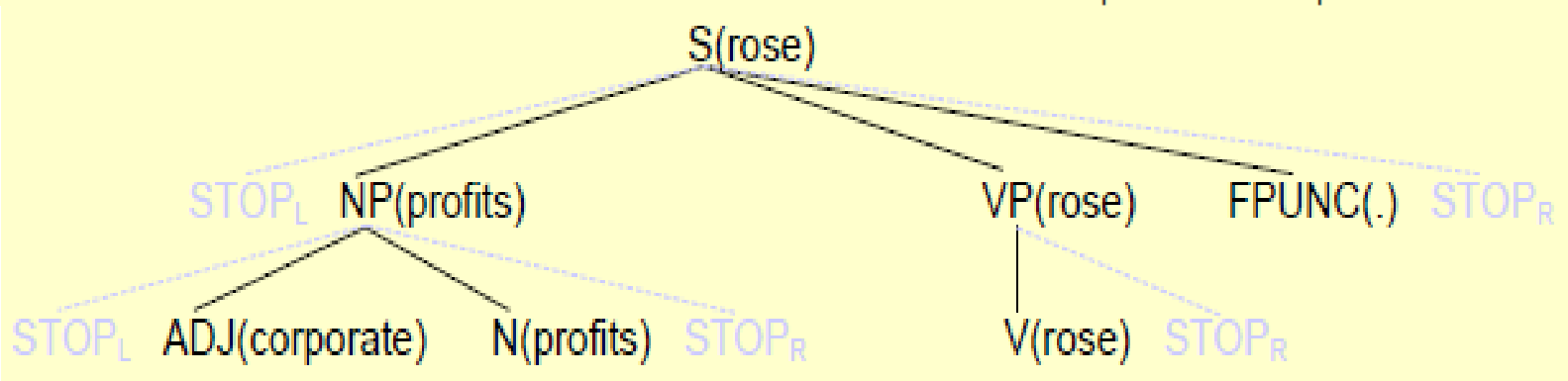
Baumwahrscheinlichkeit:

- P_{Baum} = p(S, rose) * p_H(VP | S, rose) * p_R(FPUNC(.) | S, rose, VP) * p_R($STOP_R$ | S, rose, VP) * p_L(NP(profits) | S, rose, VP) * p_L($STOP_L$ | S, rose, VP) * p_H(N | NP, profits) * p_R($STOP_R$ | NP, profits, N) * p_L(ADJ, corporate | NP, profits, N) * p_L($STOP_L$ | NP, profits, N) * p_H(V | VP, rose)

(deterministisch)

Collins: Parsebaum generieren

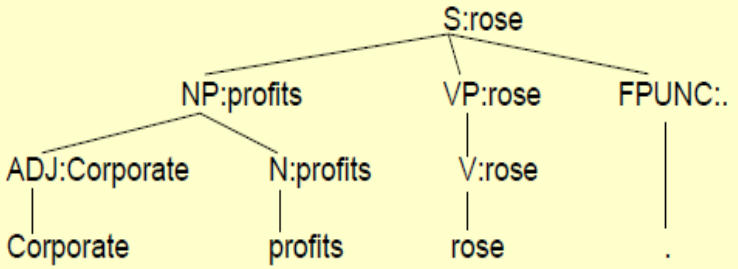


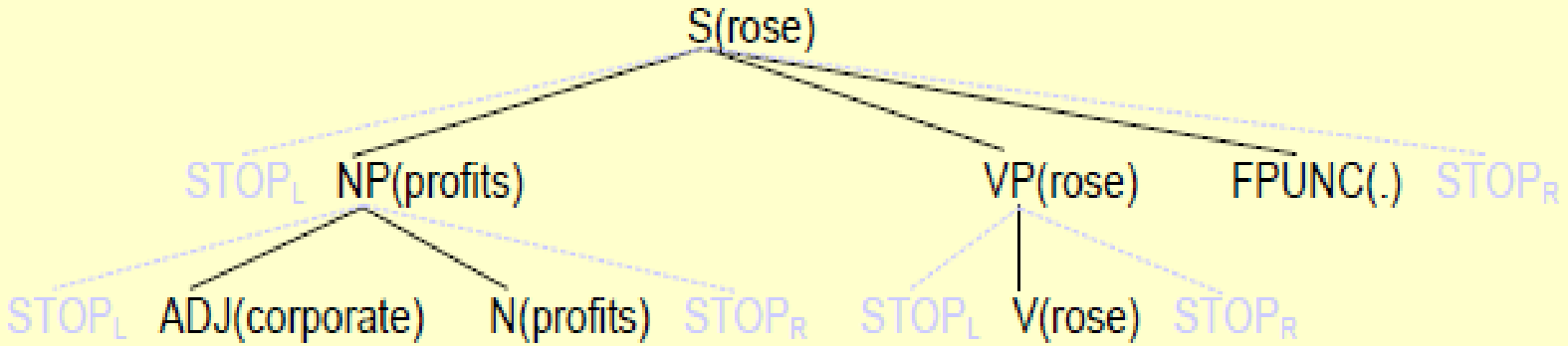


Baumwahrscheinlichkeit:

- P_{Baum} = p(S, rose) * p_H(VP | S, rose) * p_R(FPUNC(.) | S, rose, VP) * p_R($STOP_R$ | S, rose, VP) * p_L(NP(profits) | S, rose, VP) * p_L($STOP_L$ | S, rose, VP) * p_H(N | NP, profits) * p_R($STOP_R$ | NP, profits, N) * p_L(ADJ, corporate | NP, profits, N) * p_L($STOP_L$ | NP, profits, N) * p_H(V | VP, rose) * p_R($STOP_R$ | VP, rose, V)

Collins: Parsebaum generieren

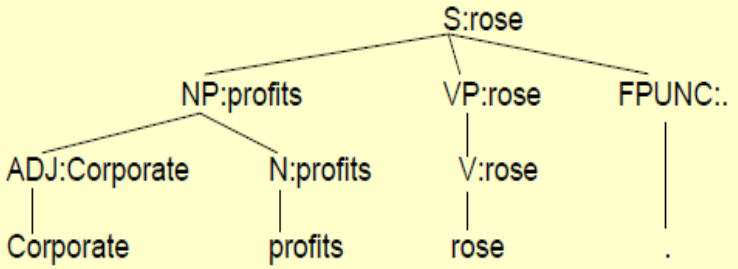


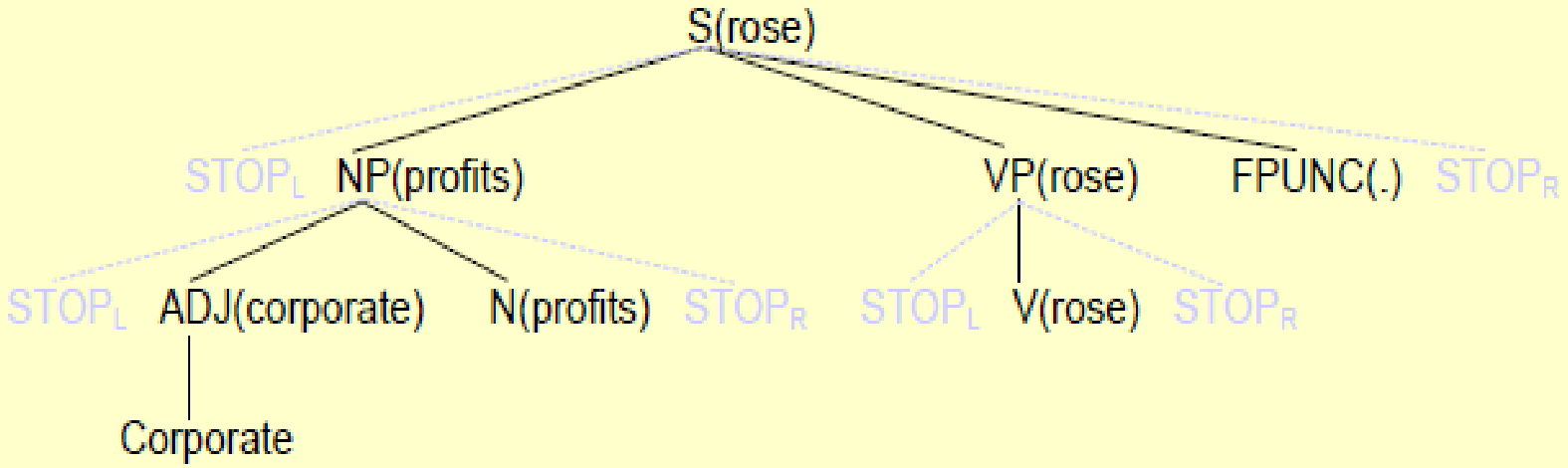


Baumwahrscheinlichkeit:

- P_{Baum} = p(S, rose) * p_H(VP | S, rose) * p_R(FPUNC(.) | S, rose, VP) * p_R($STOP_R$ | S, rose, VP) * p_L(NP(profits) | S, rose, VP) * p_L($STOP_L$ | S, rose, VP) * p_H(N | NP, profits) * p_R($STOP_R$ | NP, profits, N) * p_L(ADJ, corporate | NP, profits, N) * p_L($STOP_L$ | NP, profits, N) * p_H(V | VP, rose) * p_R($STOP_R$ | VP, rose, V) * p_L($STOP_L$ | VP, rose, V)

Collins: Parsebaum generieren

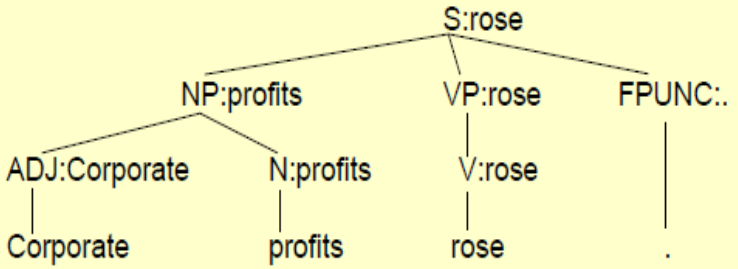


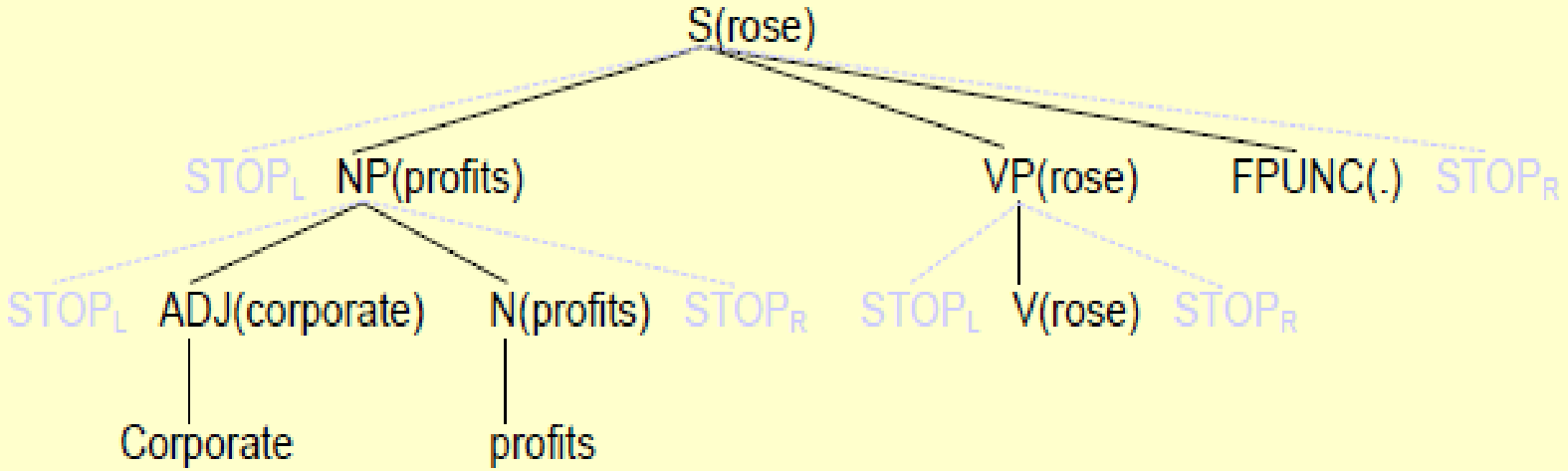


Baumwahrscheinlichkeit:

- P_{Baum} = p(S, rose) * p_H(VP | S, rose) * p_R(FPUNC(.) | S, rose, VP) * p_R($STOP_R$ | S, rose, VP) * p_L(NP(profits) | S, rose, VP) * p_L($STOP_L$ | S, rose, VP) * p_H(N | NP, profits) * p_R($STOP_R$ | NP, profits, N) * p_L(ADJ, corporate | NP, profits, N) * p_L($STOP_L$ | NP, profits, N) * p_H(V | VP, rose) * p_R($STOP_R$ | VP, rose, V) * p_L($STOP_L$ | VP, rose, V)

Collins: Parsebaum generieren

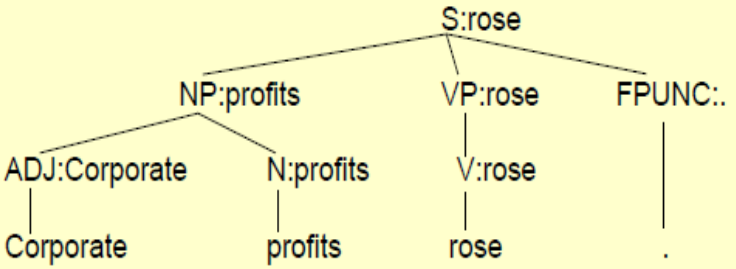


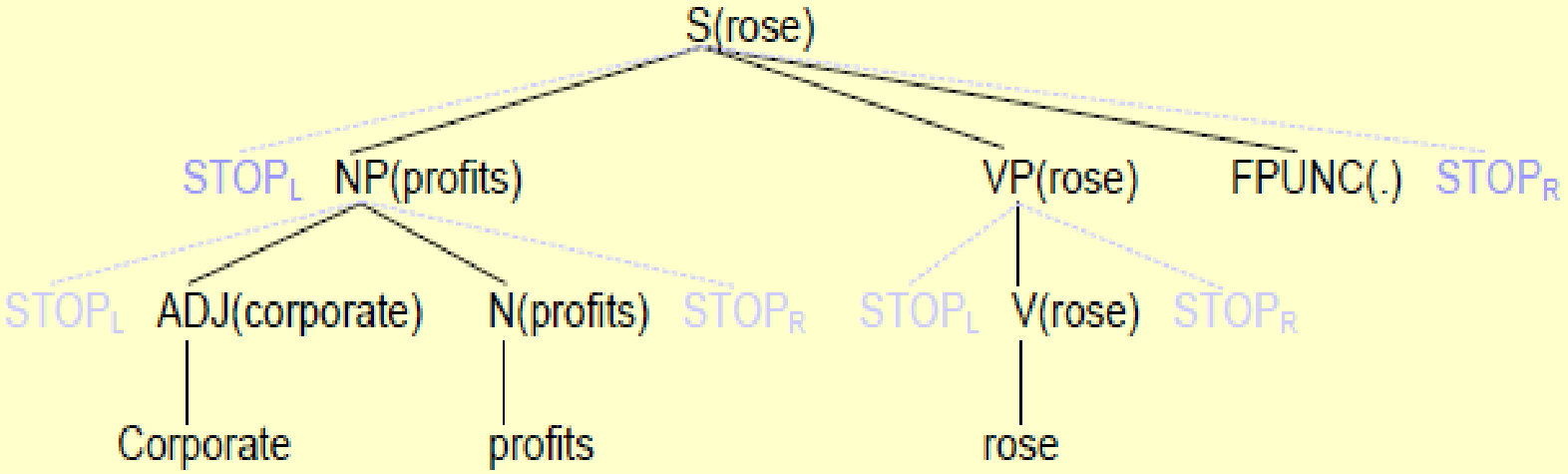


Baumwahrscheinlichkeit:

- P_{Baum} = p(S, rose) * p_H(VP | S, rose) * p_R(FPUNC(.) | S, rose, VP) * p_R($STOP_R$ | S, rose, VP) * p_L(NP(profits) | S, rose, VP) * p_L($STOP_L$ | S, rose, VP) * p_H(N | NP, profits) * p_R($STOP_R$ | NP, profits, N) * p_L(ADJ, corporate | NP, profits, N) * p_L($STOP_L$ | NP, profits, N) * p_H(V | VP, rose) * p_R($STOP_R$ | VP, rose, V) * p_L($STOP_L$ | VP, rose, V)

Collins: Parsebaum generieren

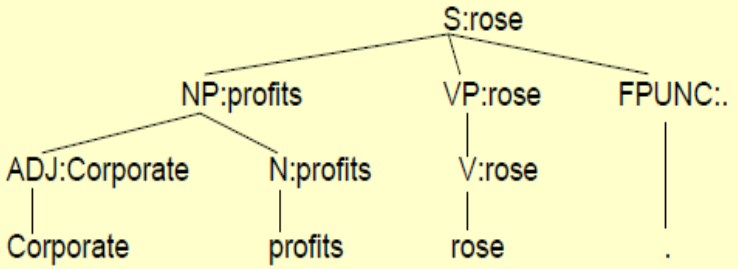


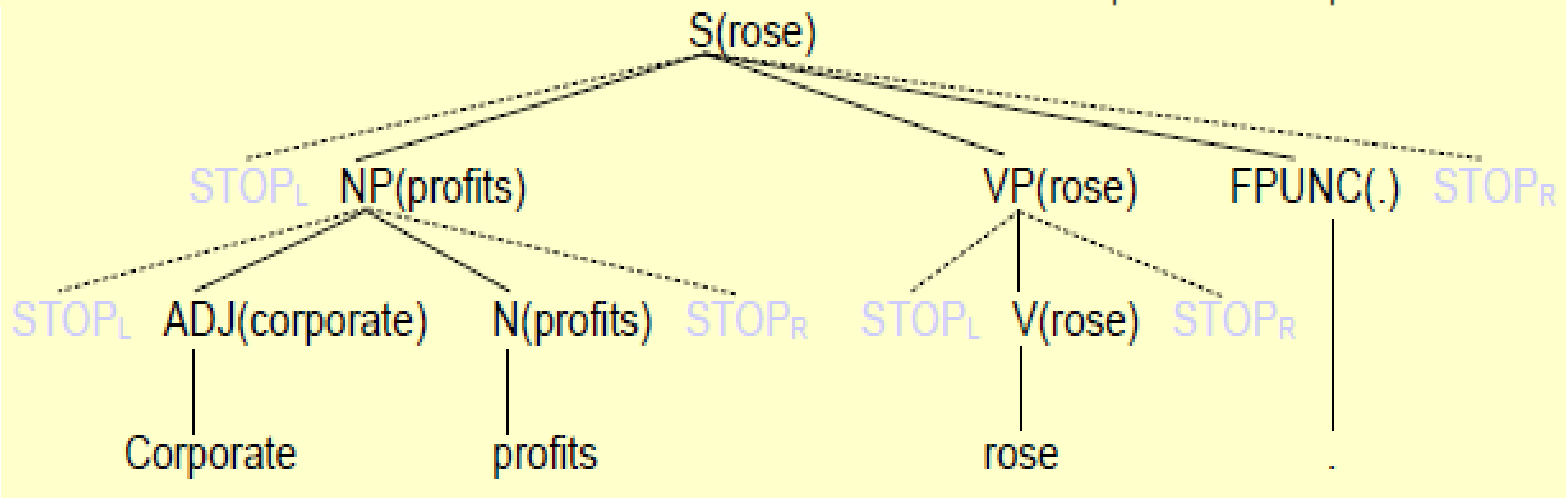


Baumwahrscheinlichkeit:

- P_{Baum} = p(S, rose) * p_H(VP | S, rose) * p_R(FPUNC(.) | S, rose, VP) * p_R($STOP_R$ | S, rose, VP) * p_L(NP(profits) | S, rose, VP) * p_L($STOP_L$ | S, rose, VP) * p_H(N | NP, profits) * p_R($STOP_R$ | NP, profits, N) * p_L(ADJ, corporate | NP, profits, N) * p_L($STOP_L$ | NP, profits, N) * p_H(V | VP, rose) * p_R($STOP_R$ | VP, rose, V) * p_L($STOP_L$ | VP, rose, V)

Collins: Parsebaum generieren





Baumwahrscheinlichkeit:

- P_{Baum} = p(S, rose) * p_H(VP | S, rose) * p_R(FPUNC(.) | S, rose, VP) * p_R($STOP_R$ | S, rose, VP) * p_L(NP(profits) | S, rose, VP) * p_L($STOP_L$ | S, rose, VP) * p_H(N | NP, profits) * p_R($STOP_R$ | NP, profits, N) * p_L(ADJ, corporate | NP, profits, N) * p_L($STOP_L$ | NP, profits, N) * p_H(V | VP, rose) * p_R($STOP_R$ | VP, rose, V) * p_L($STOP_L$ | VP, rose, V)

Collins: Hauptmerkmale (2)

- Statistisches, generatives, lexikalistisches Modell (|| Charniak)
- Markovisierung der Regeln (↔ Charniak)
    - Die Regeln werden aufgebrochen („zerhackt”) und während des Generierungsprozesses nach bestimmten Wahrscheinlichkeiten erzeugt!
        - 1. Kopf + 2. rechte Argumente + 3. linke Argumente
    - (↔ Charniak: explizite Grammatik ~ Die Regeln werden von der Baumbank abgelesen und als Ganzes verwendet.)
- ...

Collins: Hauptmerkmale (2)

- Statistisches, generatives, lexikalistisches Modell (|| Charniak)
- Markovisierung der Regeln (↔ Charniak)
    - Die Regeln werden aufgebrochen („zerhackt”) und während des Generierungsprozesses nach bestimmten Wahrscheinlichkeiten erzeugt!
        - 1. Kopf + 2. rechte Argumente + 3. linke Argumente
    - (↔ Charniak: explizite Grammatik ~ Die Regeln werden von der Baumbank abgelesen und als Ganzes verwendet.)
- Intensive, zusätzliche linguistische Annotation der Grammatik

Collins: Hauptmerkmale (2)

- Statistisches, generatives, lexikalistisches Modell (|| Charniak)
- Markovisierung der Regeln (↔ Charniak)
    - Die Regeln werden aufgebrochen („zerhackt”) und während des Generierungsprozesses nach bestimmten Wahrscheinlichkeiten erzeugt!
        - 1. Kopf + 2. rechte Argumente + 3. linke Argumente
    - (↔ Charniak: explizite Grammatik ~ Die Regeln werden von der Baumbank abgelesen und als Ganzes verwendet.)
- Intensive, zusätzliche linguistische Annotation der Grammatik
    - Basis-NP: nicht-rekursive NP → *Modell 1*
        - NP hat keine NP als unmittelbare Konstituente

Collins: Hauptmerkmale (2)

- Statistisches, generatives, lexikalistisches Modell (|| Charniak)
- Markovisierung der Regeln (↔ Charniak)
    - Die Regeln werden aufgebrochen („zerhackt”) und während des Generierungsprozesses nach bestimmten Wahrscheinlichkeiten erzeugt!
        - 1. Kopf + 2. rechte Argumente + 3. linke Argumente
    - (↔ Charniak: explizite Grammatik ~ Die Regeln werden von der Baumbank abgelesen und als Ganzes verwendet.)
- Intensive, zusätzliche linguistische Annotation der Grammatik
    - Basis-NP: nicht-rekursive NP → *Modell 1*
        - NP hat keine NP als unmittelbare Konstituente
    - Subkategorisierung *und*

Collins: Hauptmerkmale (2)

- Statistisches, generatives, lexikalistisches Modell (|| Charniak)
- Markovisierung der Regeln (↔ Charniak)
    - Die Regeln werden aufgebrochen („zerhackt”) und während des Generierungsprozesses nach bestimmten Wahrscheinlichkeiten erzeugt!
        - 1. Kopf + 2. rechte Argumente + 3. linke Argumente
    - (↔ Charniak: explizite Grammatik ~ Die Regeln werden von der Baumbank abgelesen und als Ganzes verwendet.)
- Intensive, zusätzliche linguistische Annotation der Grammatik
    - Basis-NP: nicht-rekursive NP → *Modell 1*
        - NP hat keine NP als unmittelbare Konstituente
    - Subkategorisierung *und*
    - Komplement/Adjunkt-Unterscheidung → *Modell 2*

Collins: Hauptmerkmale (2)

- Statistisches, generatives, lexikalistisches Modell (|| Charniak)
- Markovisierung der Regeln (↔ Charniak)
    - Die Regeln werden aufgebrochen („zerhackt”) und während des Generierungsprozesses nach bestimmten Wahrscheinlichkeiten erzeugt!
        - 1. Kopf + 2. rechte Argumente + 3. linke Argumente
    - (↔ Charniak: explizite Grammatik ~ Die Regeln werden von der Baumbank abgelesen und als Ganzes verwendet.)
- Intensive, zusätzliche linguistische Annotation der Grammatik
    - Basis-NP: nicht-rekursive NP → *Modell 1*
        - NP hat keine NP als unmittelbare Konstituente
    - Subkategorisierung *und*
    - Komplement/Adjunkt-Unterscheidung → *Modell 2*
    - Wh-Bewegungen → *Modell 3*

Collins: Hauptmerkmale (2)

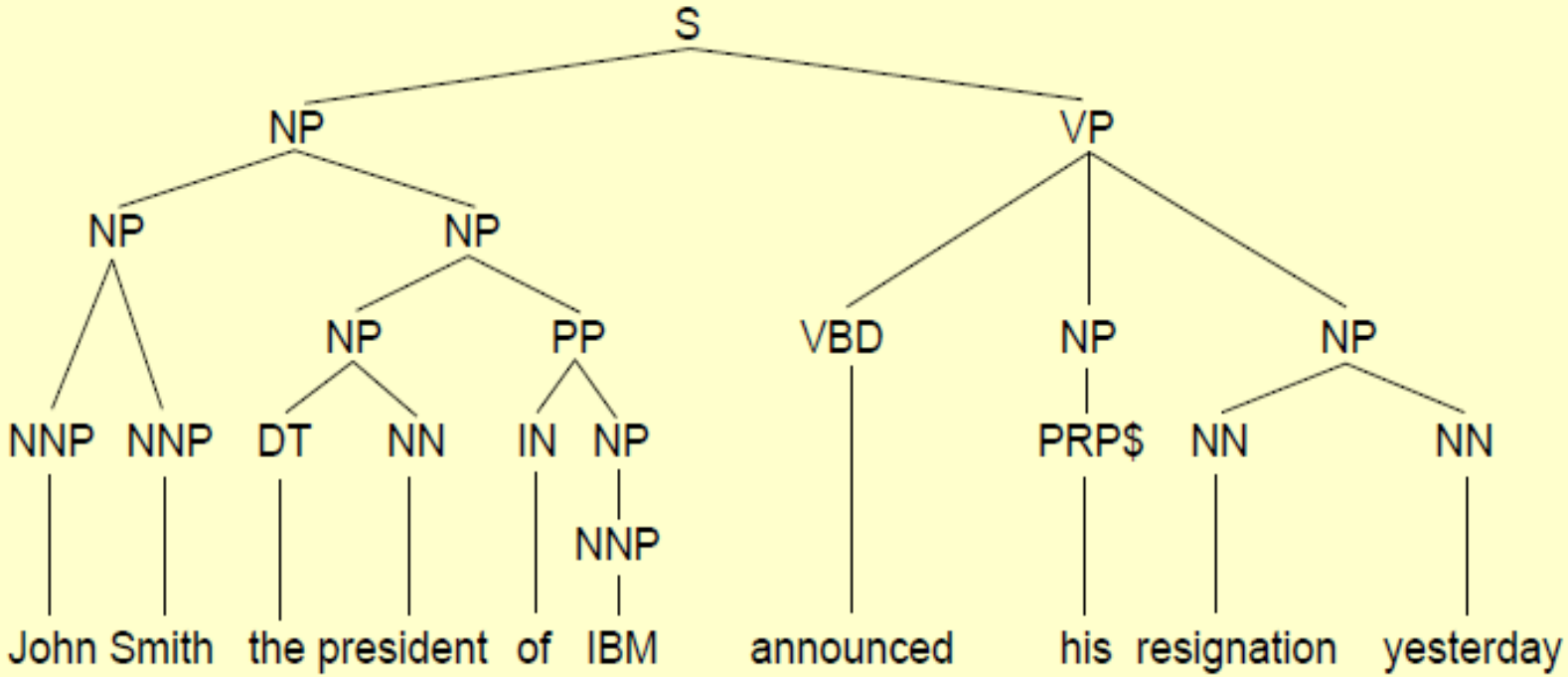
- Statistisches, generatives, lexikalistisches Modell (|| Charniak)
- Markovisierung der Regeln (↔ Charniak)
    - Die Regeln werden aufgebrochen („zerhackt”) und während des Generierungsprozesses nach bestimmten Wahrscheinlichkeiten erzeugt!
        - 1. Kopf + 2. rechte Argumente + 3. linke Argumente
    - (↔ Charniak: explizite Grammatik ~ Die Regeln werden von der Baumbank abgelesen und als Ganzes verwendet.)
- Intensive, zusätzliche linguistische Annotation der Grammatik
    - Basis-NP: nicht-rekursive NP → *Modell 1*
        - NP hat keine NP als unmittelbare Konstituente
    - Subkategorisierung *und*
    - Komplement/Adjunkt-Unterscheidung → *Modell 2*
    - Wh-Bewegungen → *Modell 3*
- ...

Collins: Drei statistische, generative Modelle (1)

- *Modell 1*
  - generatives Verfahren
  - Lexikalisierung
  - Basis-NP [base NP] und Abhängigkeiten [dependencies]
- *...*

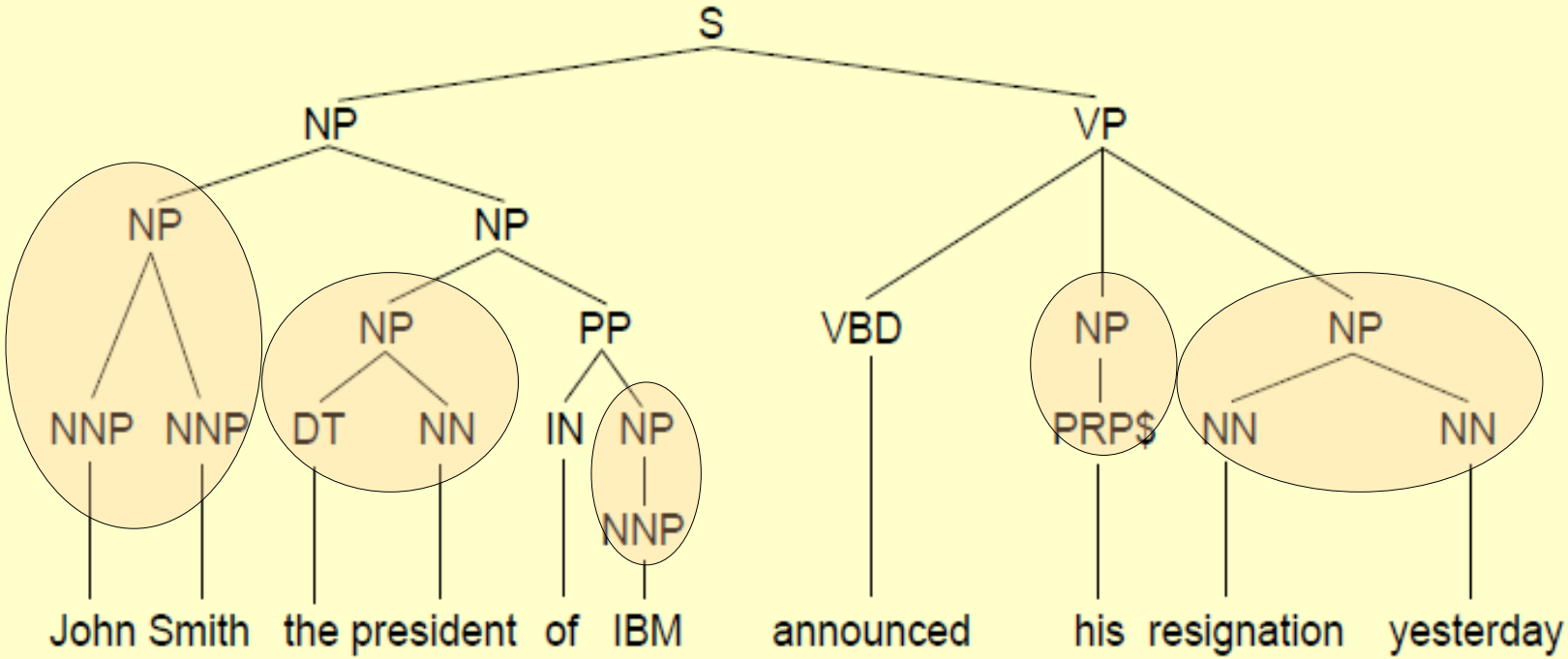
Collins: Modell 1

- Basis-NP [base NP]: nicht-rekursive NP ~ NP hat keine NP als unmittelbare Konstituente



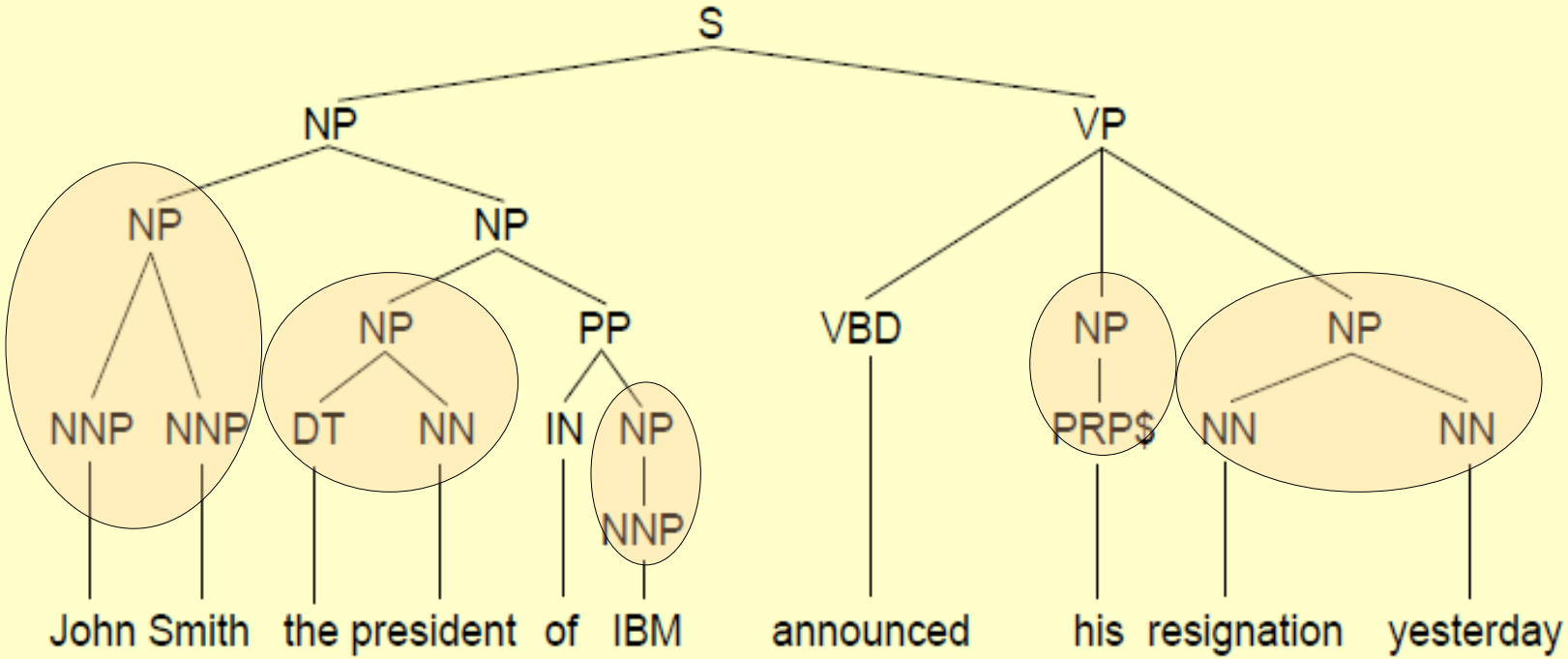
Collins: Modell 1

- Basis-NP [base NP]: nicht-rekursive NP ~ NP hat keine NP als unmittelbare Konstituente



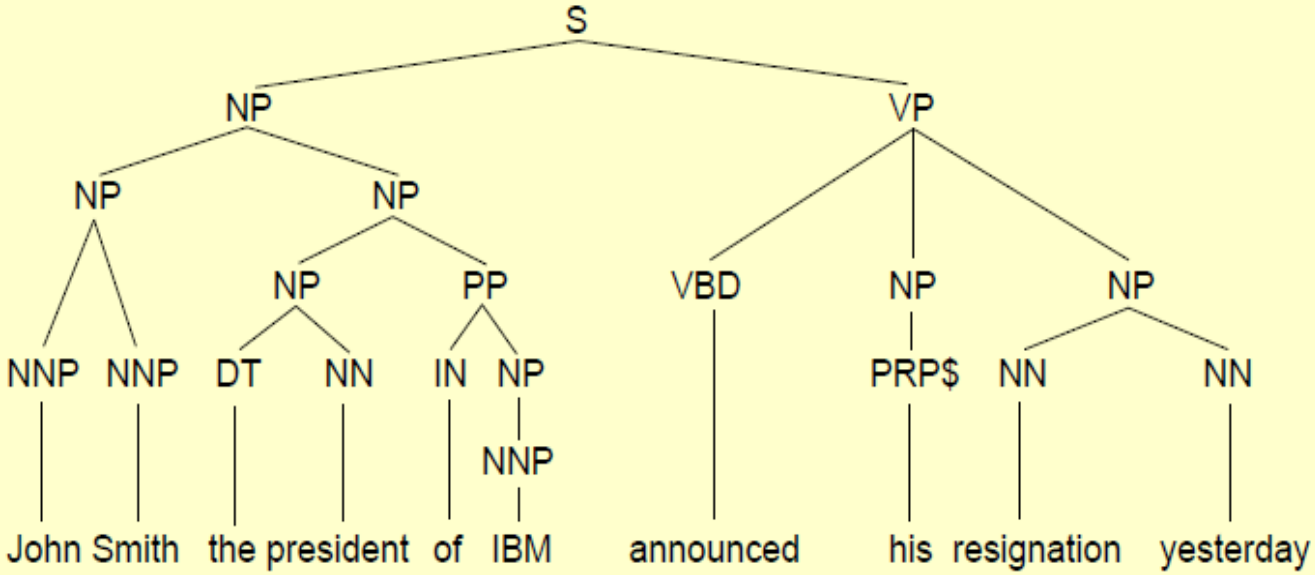
Collins: Modell 1

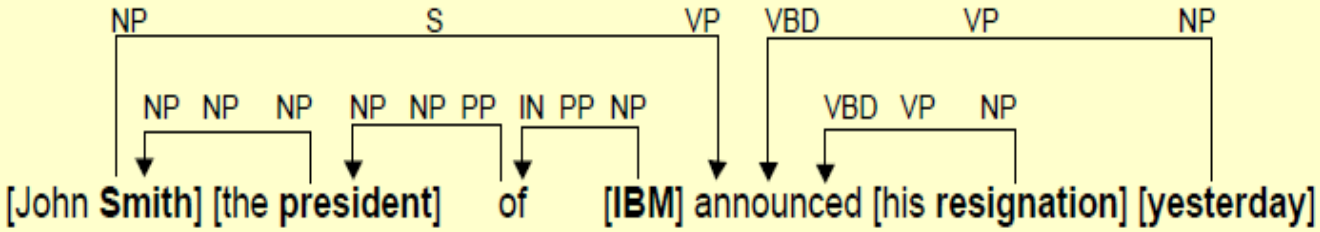
- Basis-NP [base NP]: nicht-rekursive NP ~ NP hat keine NP als unmittelbare Konstituente
- POS-Tagging:
  - John/NNP Smith/NNP, the/DT president/NN of/IN IBM/NNP, announced/VBD his/PRP$ resignation/NN yesterday/NN .



Collins: Modell 1

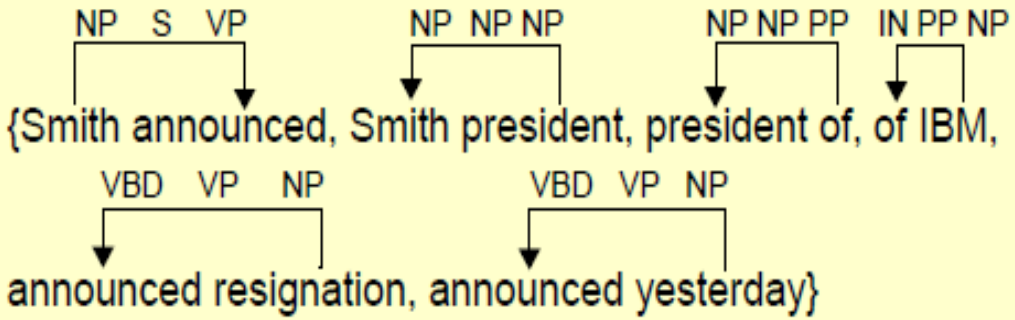
- Basis-NPs
- Dependencies





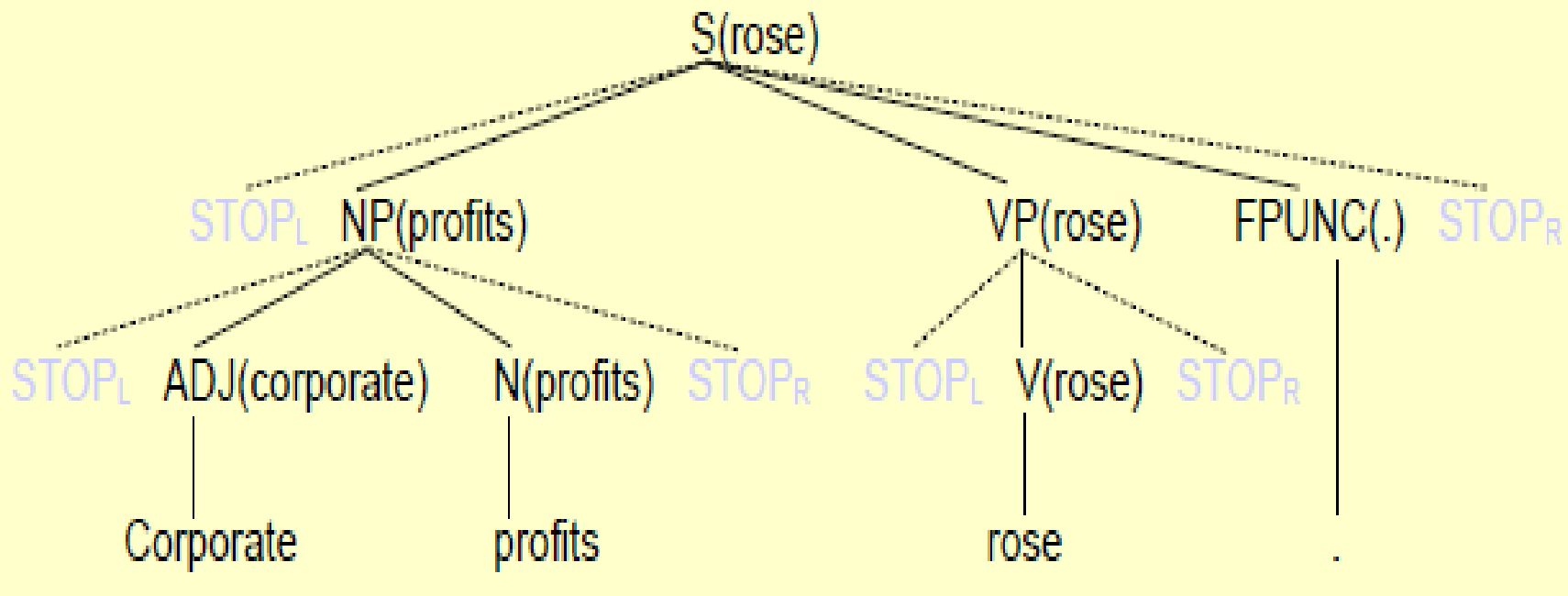
Base NP = { [John Smith], [the president], [IBM], [his resignation], [yesterday]}

Dependencies =



Collins: Modell 1

- Lexikalisierung

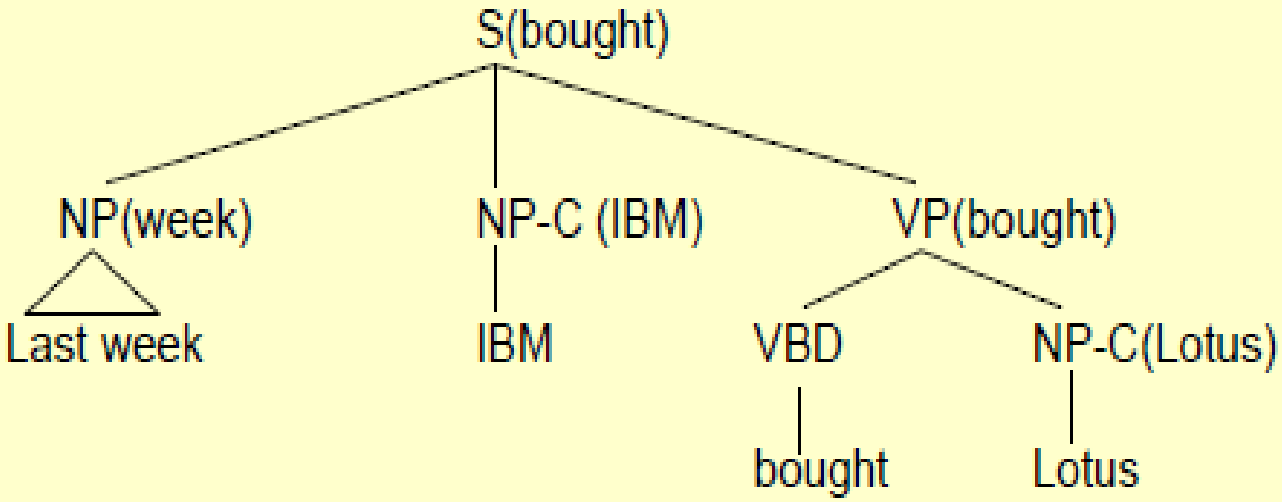


Collins: Drei statistische, generative Modelle (2)

- *Modell 1*
  - generatives Verfahren
  - Lexikalisierung
  - Basis-NP [base NP] und Abhängigkeiten [dependencies]
- *Modell 2*
  - Modell 1 +
  - Subkategorisierungsrahmen (Wahrscheinlichkeiten den möglichen Modifizierern/Argumenten des Kopfes zuordnen)
  - Komplemente [complement] markieren: „-C” (Adjunkte bleiben unmarkiert)
- *...*

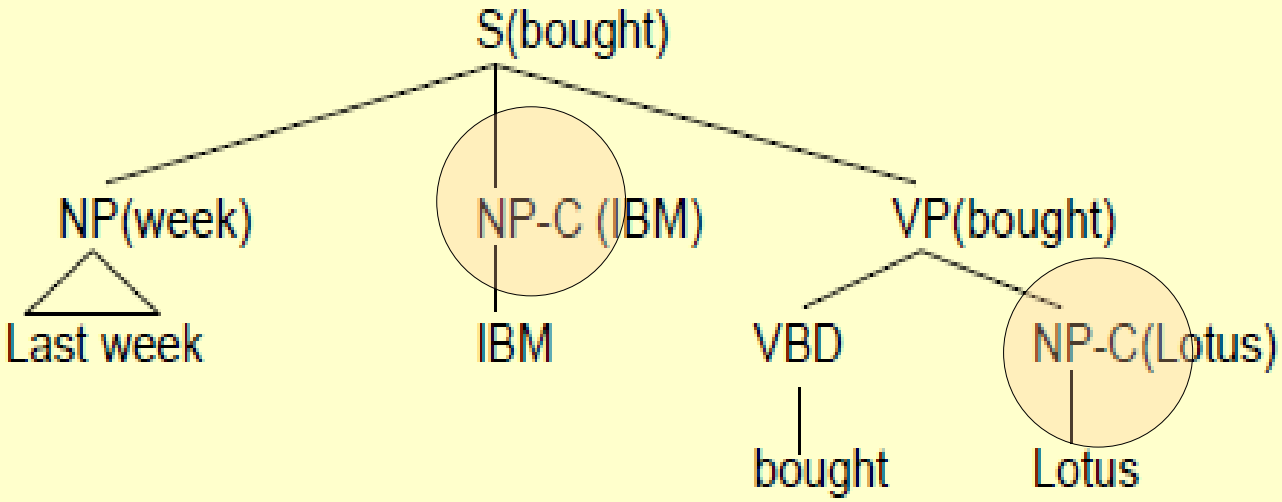
Collins: Modell 2

- Subkategorisierungsrahmen [subcategorisation frame]
- Komplemente [complement] markieren: „-C”



Collins: Modell 2

- Subkategorisierungsrahmen [subcategorisation frame]
- Komplemente [complement] markieren: „-C”

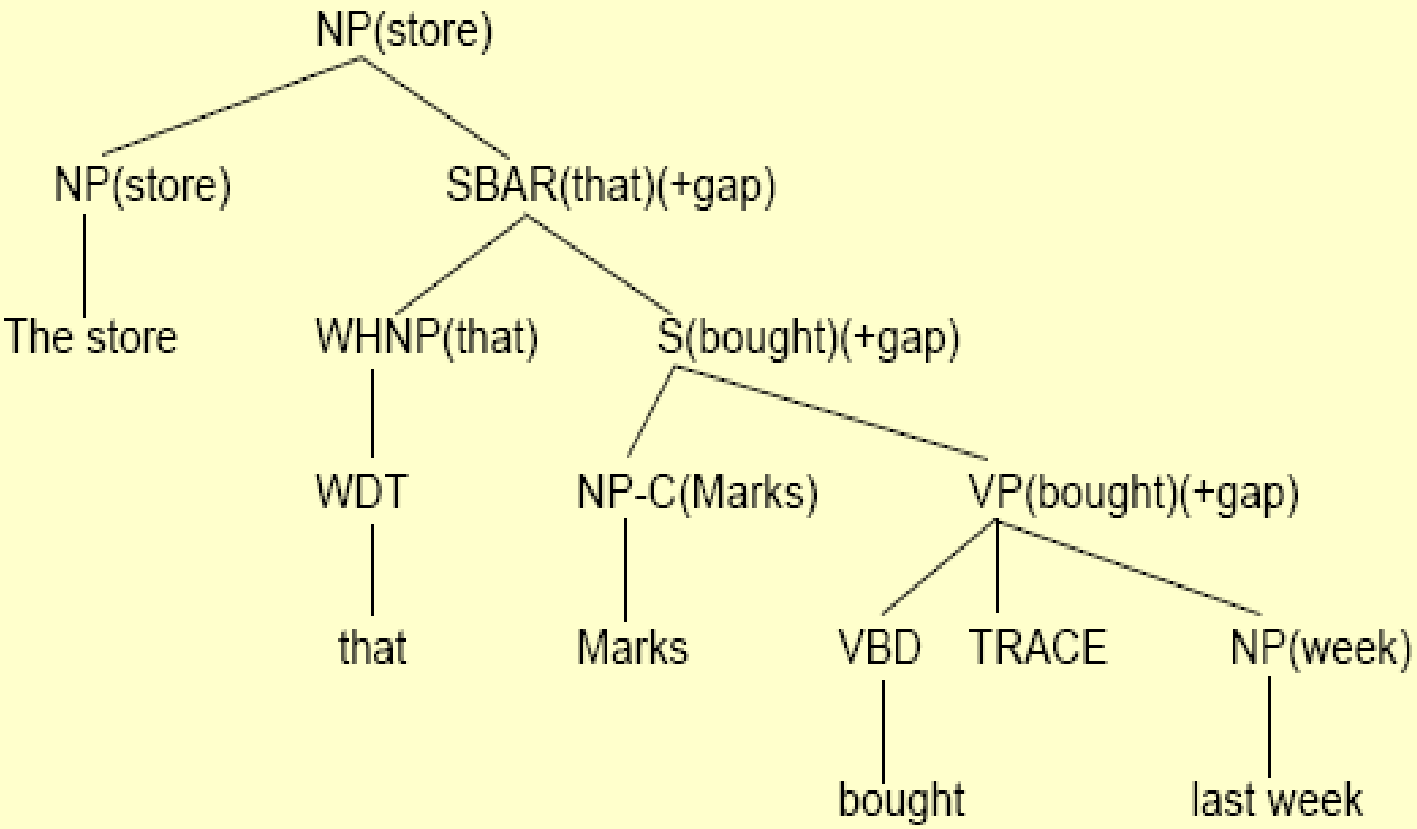


Collins: Drei statistische, generative Modelle (3)

- *Modell 1*
  - generatives Verfahren
  - Lexikalisierung
  - Basis-NP [base NP] und Abhängigkeiten [dependencies]
- *Modell 2*
  - Modell 1 +
  - Subkategorisierungsrahmen (Wahrscheinlichkeiten den möglichen Modifizierern/Argumenten des Kopfes zuordnen)
  - Komplemente [complement] markieren: „-C” (Adjunkte bleiben unmarkiert)
- *Modell 3*
  - Modell 2 +
  - Wh-Bewegungen [wh-movement] kennzeichnen
    - Spur [trace]: die Ausgangsposition des bewegten Elementes
    - (+gap): Markierung für Phrasen, die bewegte Elemente beinhalten

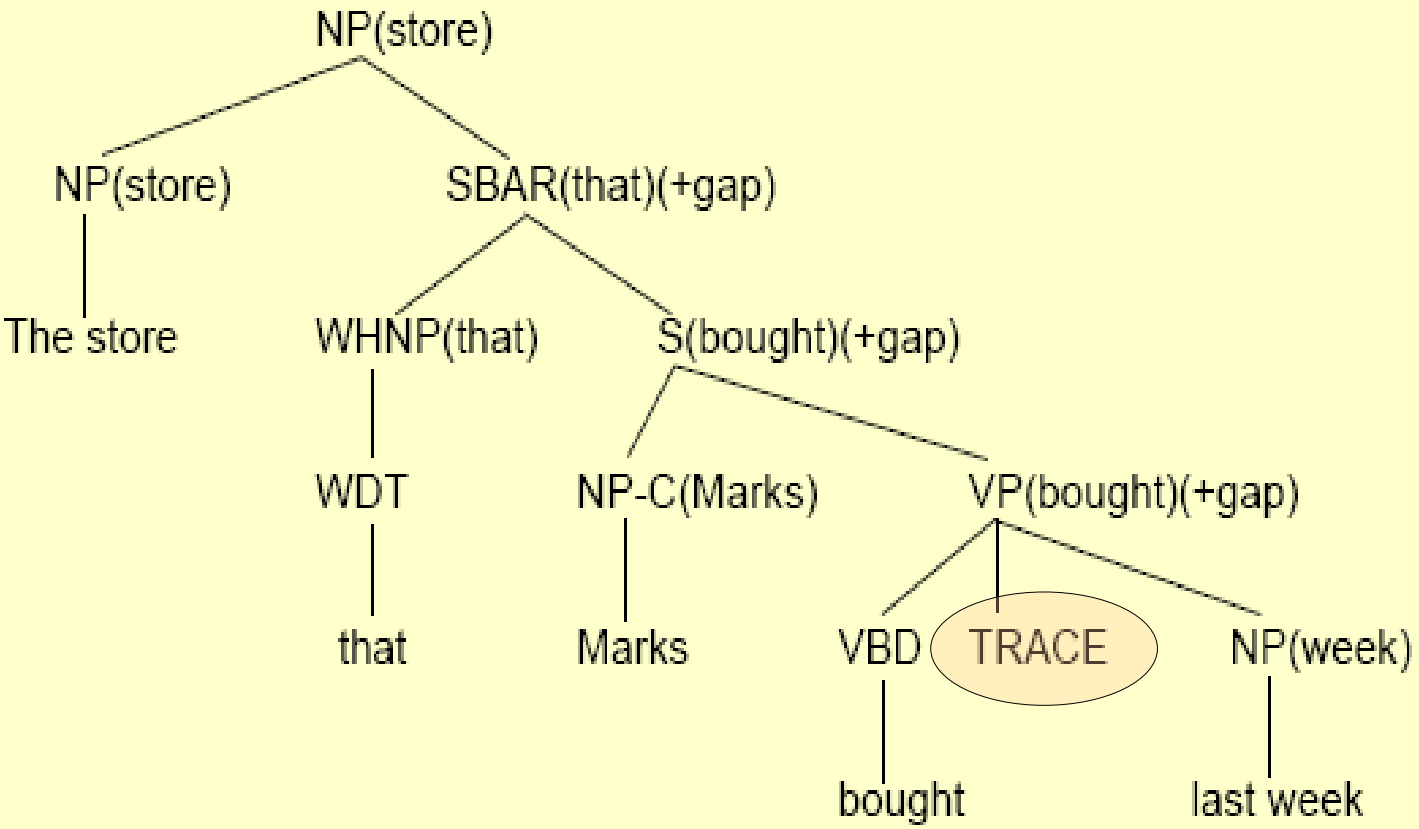
Collins: Modell 3

- Traces and wh-movement
- NP-s werden oft aus der Subjektposition, Objektposition, oder in eine PP bewegt.



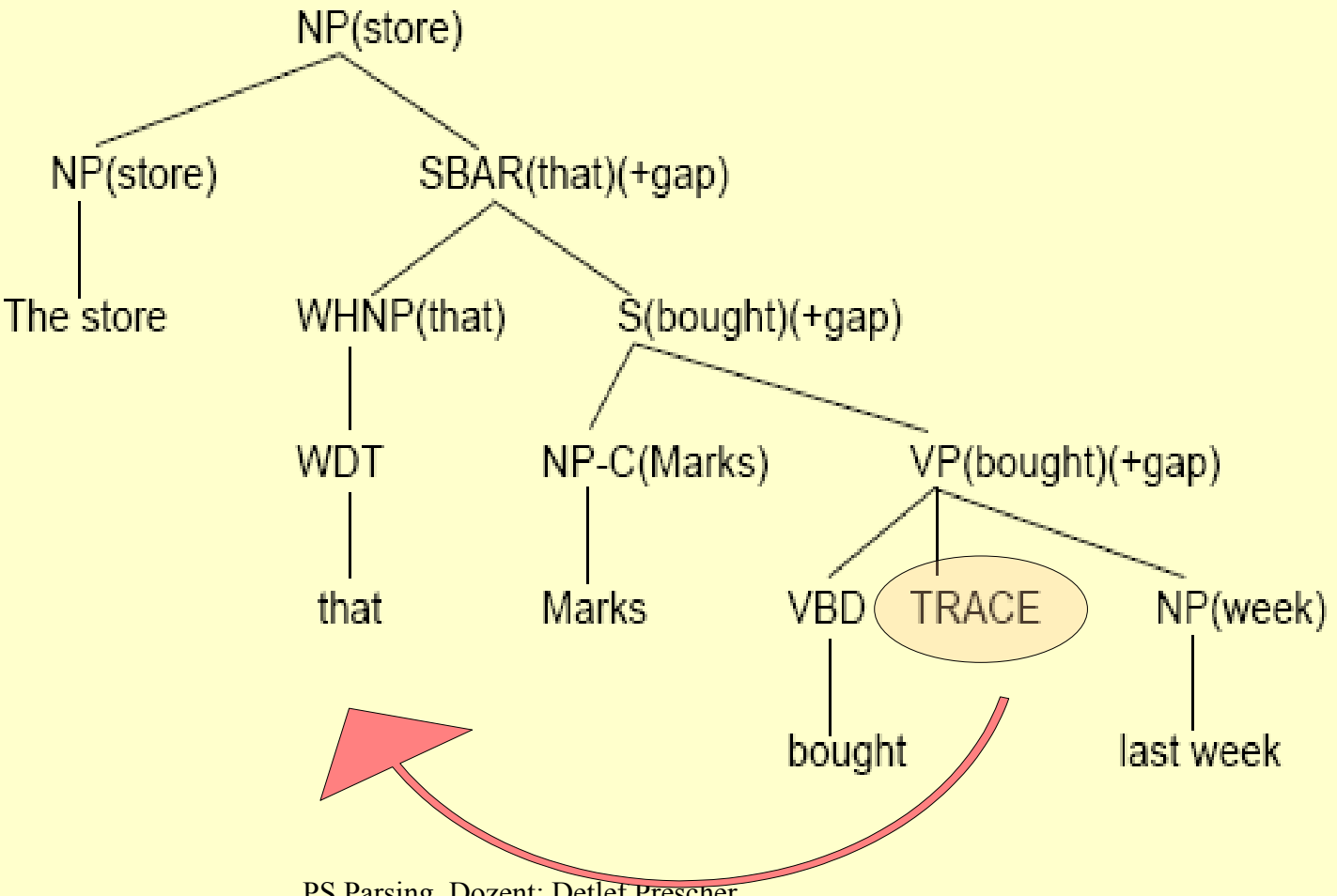
Collins: Modell 3

- Traces and wh-movement
- NP-s werden oft aus der Subjektposition, Objektposition, oder in eine PP bewegt.



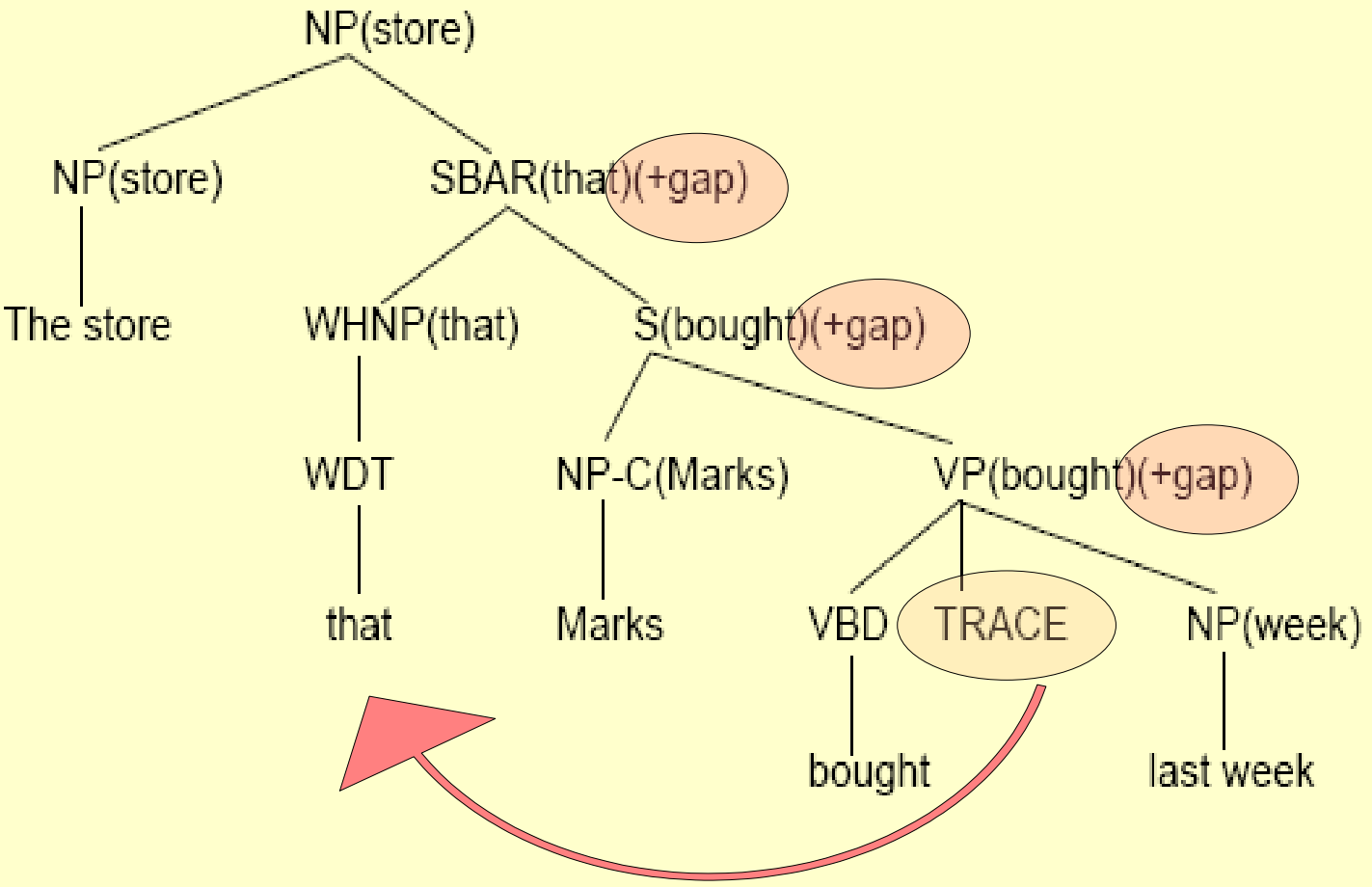
Collins: Modell 3

- Traces and wh-movement
- NP-s werden oft aus der Subjektposition, Objektposition, oder in eine PP bewegt.



Collins: Modell 3

- Spuren [trace]: TRACE und Wh-Bewegungen [wh-movement]: (+gap)
- NP-s werden oft aus der Subjektposition, Objektposition, oder in eine PP bewegt.



Collins: Hauptmerkmale (3)

- Statistisches, generatives, lexikalistisches Modell (|| Charniak)
- Markovisierung der Regeln (↔ Charniak)
    - Die Regeln werden aufgebrochen („zerhackt”) und während des Generierungsprozesses nach bestimmten Wahrscheinlichkeiten erzeugt!
        - 1. Kopf + 2. rechte Argumente + 3. linke Argumente
    - (↔ Charniak: explizite Grammatik ~ Die Regeln werden von der Baumbank abgelesen und als Ganzes verwendet.)
- Intensive, zusätzliche linguistische Annotation der Grammatik
    - Basis-NP: nicht-rekursive NP → *Modell 1*
        - NP hat keine NP als unmittelbare Konstituente
    - Subkategorisierung *und*
    - Komplement/Adjunkt → *Modell 2*
    - Wh-Bewegungen → *Modell 3*
- ...

Collins: Hauptmerkmale (3)

- Statistisches, generatives, lexikalistisches Modell (|| Charniak)
- Markovisierung der Regeln (↔ Charniak)
    - Die Regeln werden aufgebrochen („zerhackt”) und während des Generierungsprozesses nach bestimmten Wahrscheinlichkeiten erzeugt!
        - 1. Kopf + 2. rechte Argumente + 3. linke Argumente
    - (↔ Charniak: explizite Grammatik ~ Die Regeln werden von der Baumbank abgelesen und als Ganzes verwendet.)
- Intensive, zusätzliche linguistische Annotation der Grammatik
    - Basis-NP: nicht-rekursive NP → *Modell 1*
        - NP hat keine NP als unmittelbare Konstituente
    - Subkategorisierung *und*
    - Komplement/Adjunkt → *Modell 2*
    - Wh-Bewegungen → *Modell 3*
- Realisierung als unsichtbare Annotation, erscheint nur im Wahrscheinlichkeitsmodell (STOP, -C, gap, trace)!

Ergebnisse

Ergebnisse: Trainiert auf Penn Treebank

Modell	≤ 40 Wörter					≤ 100 Wörter				
	LR	LP	CB	0 CB	≤ 2 CB	LR	LP	CB	0 CB	≤ 2 CB
Magermann (1995)	84,6%	84,9%	1,26	56,6%	81,4%	84,0%	84,3%	1,46	54,0%	78,8%
Collins (1996)	85,8%	86,3%	1,14	59,9%	83,6%	85,3%	85,7%	1,32	57,2%	80,8%
Charniak (1997)	86,9%	86,8%	1,00	62,1%	86,1%					
Modell 1	87,4%	88,1%	0,96	65,7%	86,3%	86,8%	87,6%	1,11	63,1%	84,1%
Modell 2	88,1%	88,6%	0,91	66,5%	86,9%	87,5%	88,1%	1,07	63,9%	84,6%
Modell 3	88,1%	88,6%	0,91	66,4%	86,9%	87,5%	88,1%	1,07	63,9%	84,6%
Charniak (2000)	90,1%	90,1%	0,74	70,1%	89,6%	89,6%	89,5%	0,88	67,6%	87,7%
Collins (2000)	90,1%	90,4%	0,73	70,7%	89,6%	89,6%	89,9%	0,87	68,3%	87,7%

- ≤ 40 Wörter: 2245 Sätze, ≤ 100 Wörter: 2416 Sätze
- LR (labeled recall): $\frac{\text{korrekt erkannte Konstituenten in einem bestimmten Parse}}{\text{Konstituenten im Treebank-Parse}}$
- LP (labeled precision): $\frac{\text{korrekt erkannte Konstituenten in einem bestimmten Parse}}{\text{Konstituenten in diesem bestimmten Parse}}$
- CB (crossing brackets): Durchschnitt der sich überschneidenden Klammern in einem Baum

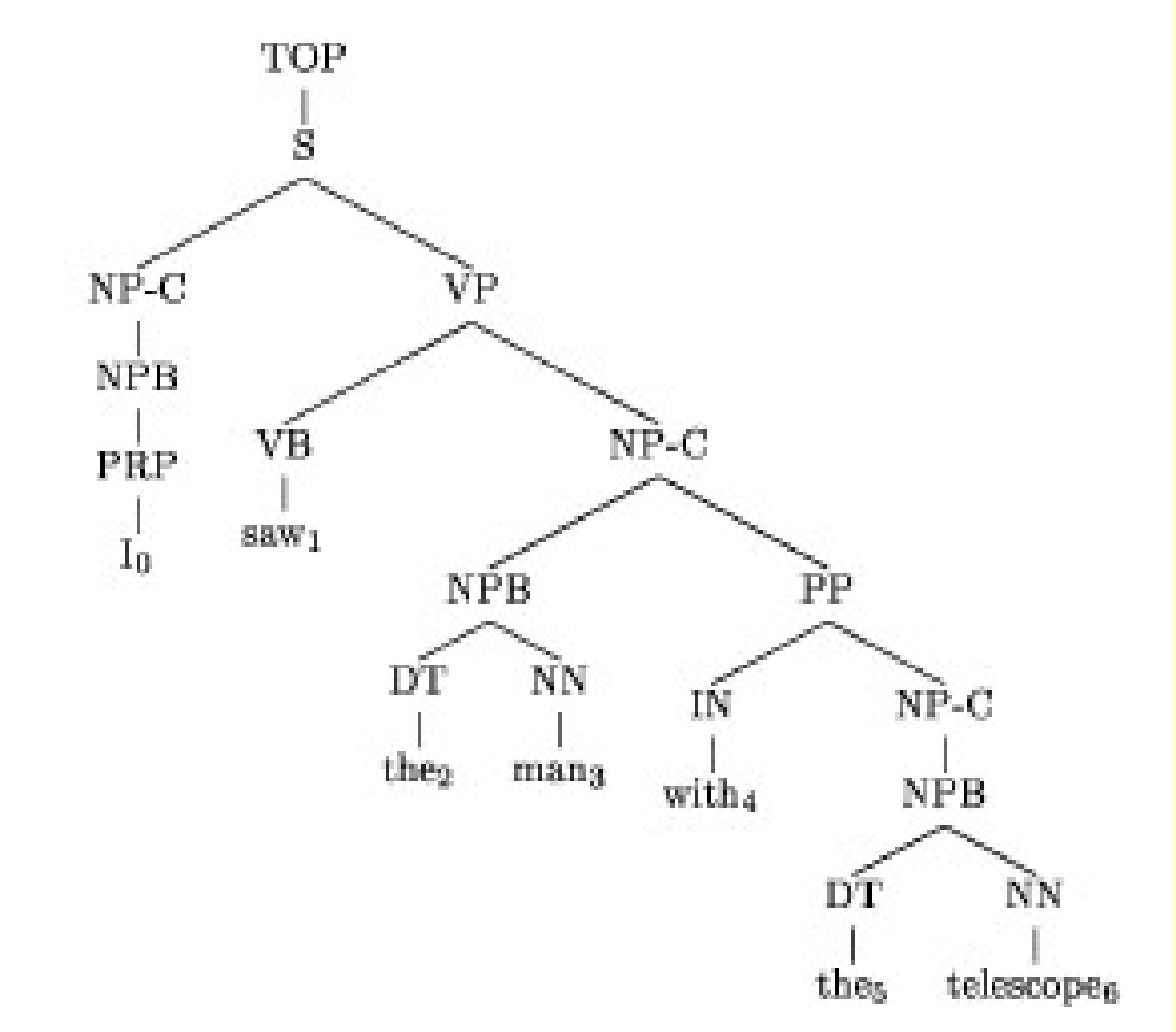
Ergebnisse: Trainiert auf Penn Treebank

Modell	≤ 40 Wörter					≤ 100 Wörter				
	LR	LP	CB	0 CB	≤ 2 CB	LR	LP	CB	0 CB	≤ 2 CB
Magermann (1995)	84,6%	84,9%	1,26	56,6%	81,4%	84,0%	84,3%	1,46	54,0%	78,8%
Collins (1996)	85,8%	86,3%	1,14	59,9%	83,6%	85,3%	85,7%	1,32	57,2%	80,8%
Charniak (1997)	86,9%	86,8%	1,00	62,1%	86,1%					
Modell 1	87,4%	88,1%	0,96	65,7%	86,3%	86,8%	87,6%	1,11	63,1%	84,1%
Modell 2	88,1%	88,6%	0,91	66,5%	86,9%	87,5%	88,1%	1,07	63,9%	84,6%
Modell 3	88,1%	88,6%	0,91	66,4%	86,9%	87,5%	88,1%	1,07	63,9%	84,6%
Charniak (2000)	90,1%	90,1%	0,74	70,1%	89,6%	89,6%	89,5%	0,88	67,6%	87,7%
Collins (2000)	90,1%	90,4%	0,73	70,7%	89,6%	89,6%	89,9%	0,87	68,3%	87,7%

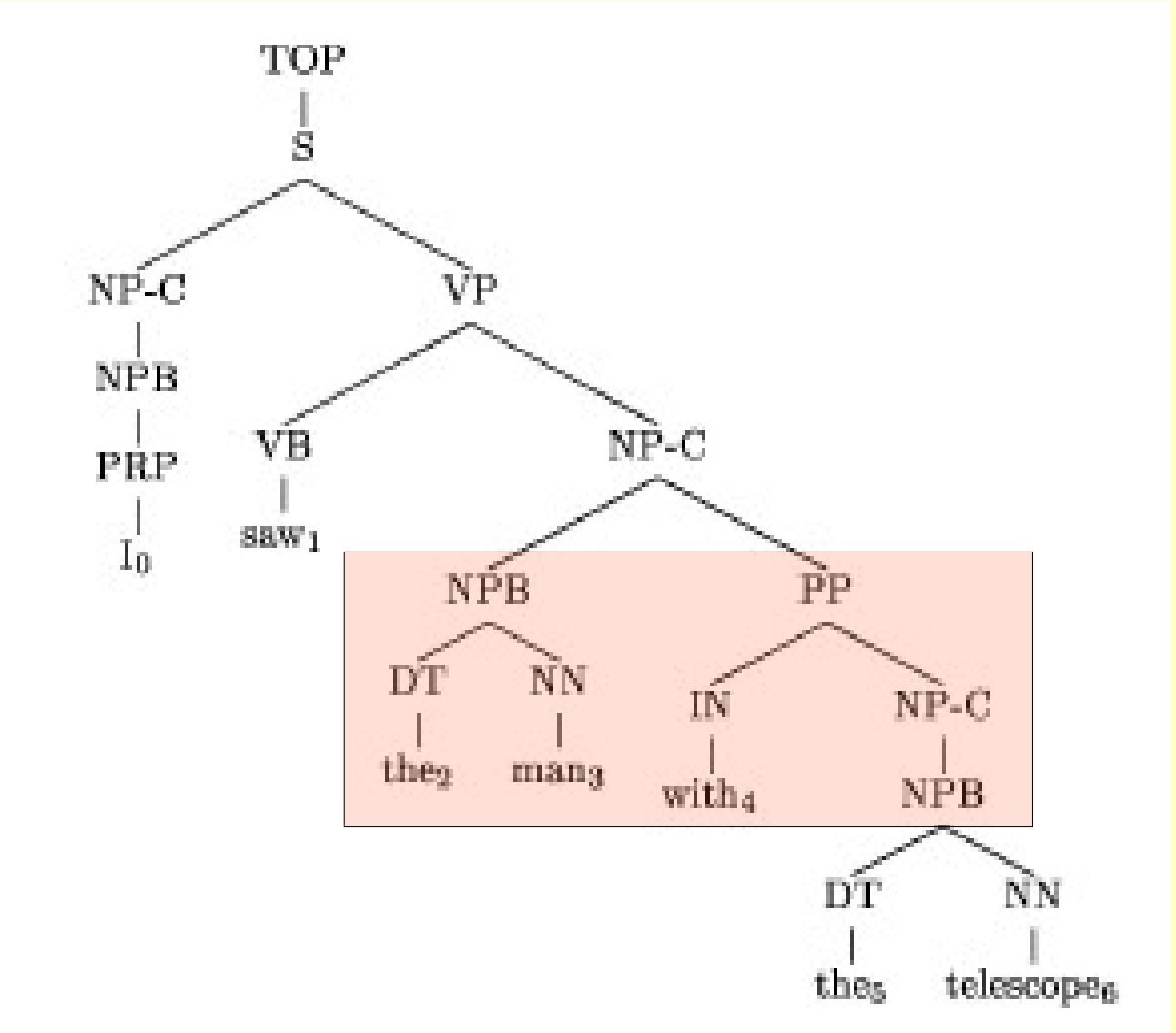
- ≤ 40 Wörter: 2245 Sätze, ≤ 100 Wörter: 2416 Sätze
- LR (labeled recall): $\frac{\text{korrekt erkannte Konstituenten in einem bestimmten Parse}}{\text{Konstituenten im Treebank-Parse}}$
- LP (labeled precision): $\frac{\text{korrekt erkannte Konstituenten in einem bestimmten Parse}}{\text{Konstituenten in diesem bestimmten Parse}}$
- CB (crossing brackets): Durchschnitt der sich überschneidenden Klammern in einem Baum

Fehleranalyse

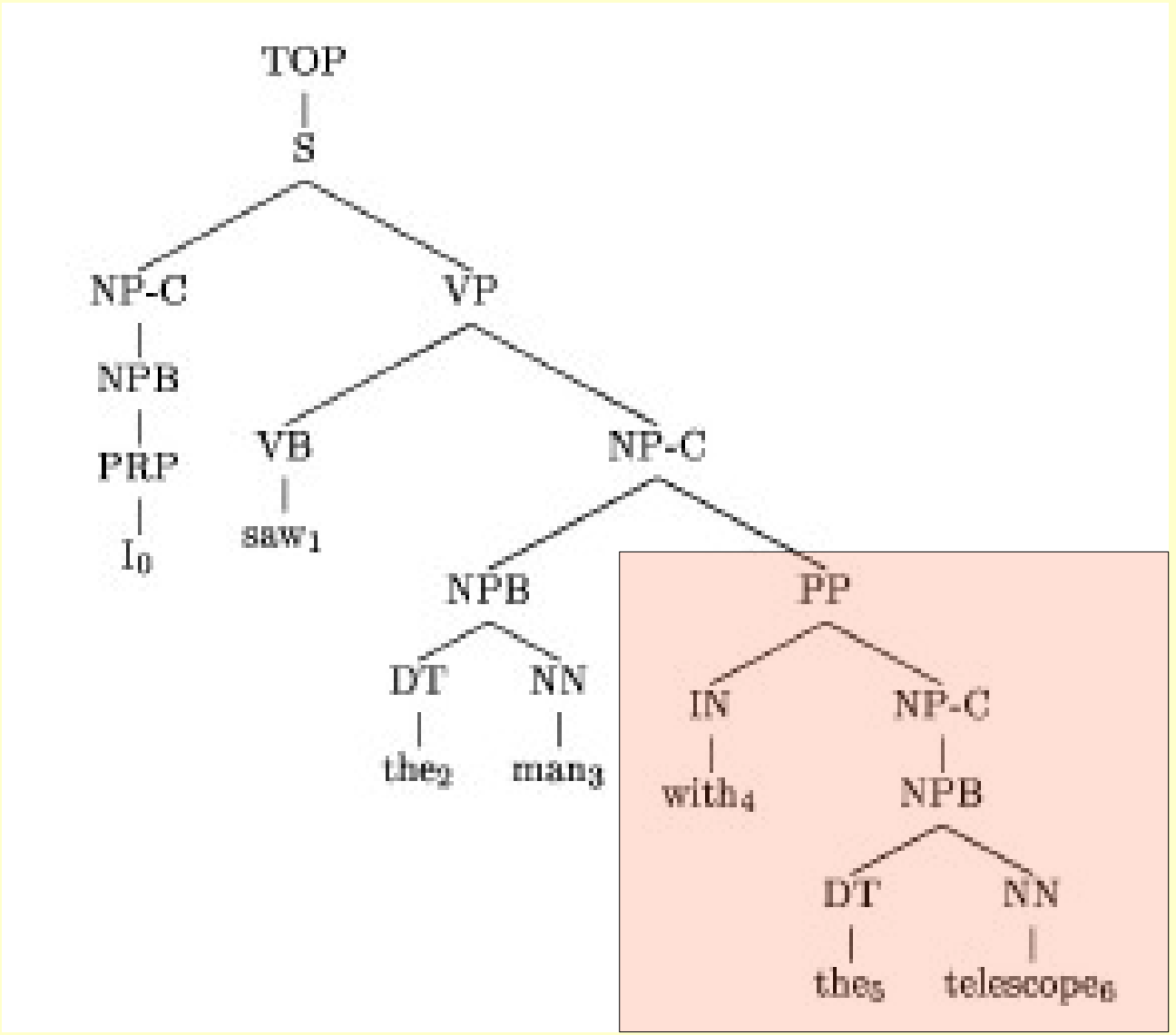
Fehleranalyse



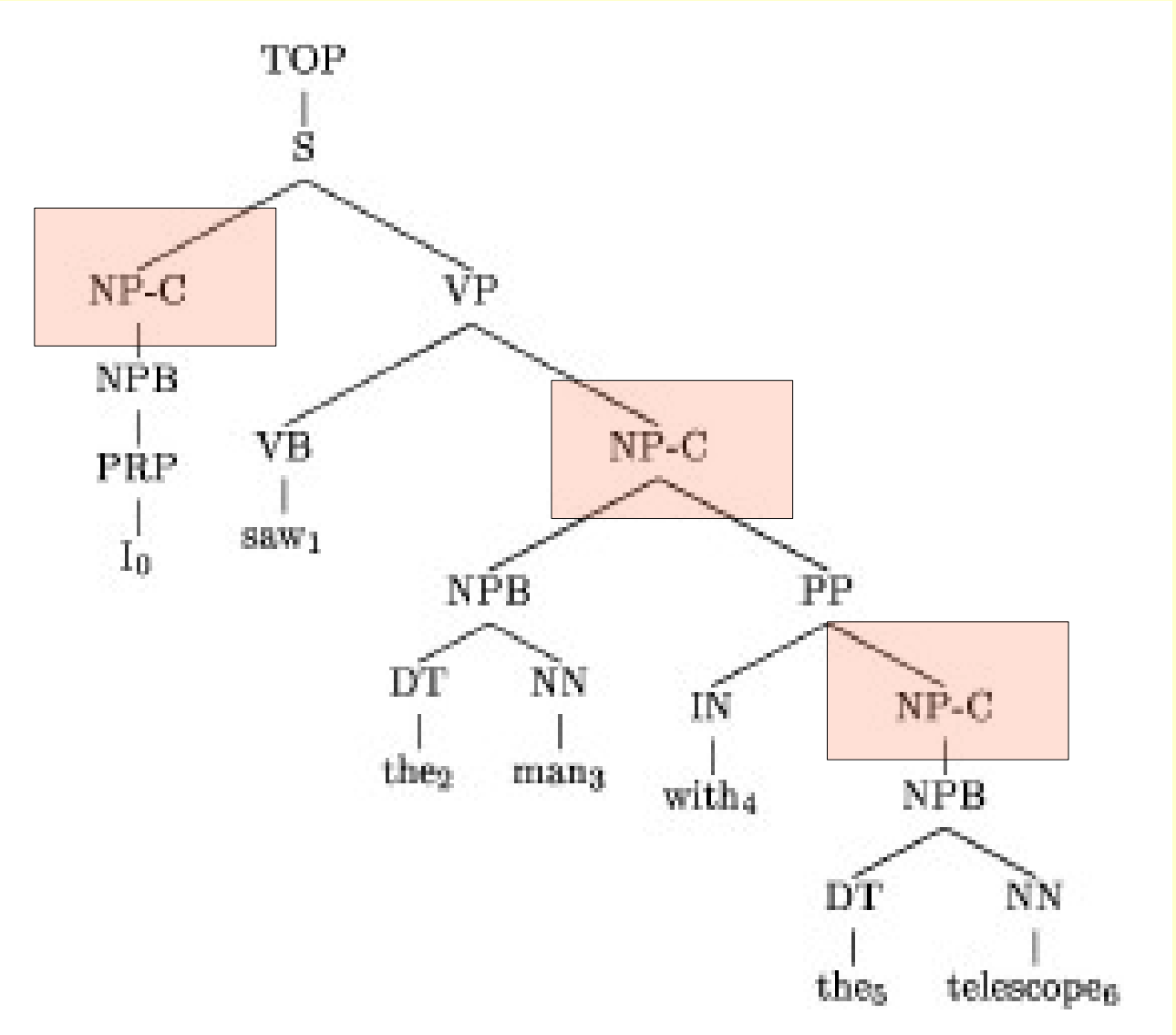
Fehleranalyse



Fehleranalyse



Fehleranalyse



Fehleranalyse

- Fragen zum Analysieren:
  - Wie wirken die einzelnen Unterschiede zwischen den 3 Modellen auf die Ergebnisse aus?
  - Wie oft kommen die verschiedenen Konstruktionen in der Baumbank vor?
  - → Mit Hilfe von linguistisch motivierten Beispielen die unterschiedlichen Möglichkeiten ausprobieren.
- Bei Erkennungsfehler:
  - 1. die Dependencies „normalisieren”: die POS-Tags durch „TAG” ersetzen → POS-Taggingsfehler in Dependencies werden aufgehoben (3% Fehler)
  - 2. alle Komplement-Markierungen tilgen → so können Komplemente nicht fälschlicherweise als Adjunkte oder umgekehrt eingestuft werden.

Fehleranalyse: Modell 2

Recall and precision for different constituent types, for section 0 of the treebank with model 2. *Label* is the nonterminal label; *Proportion* is the percentage of constituents in the treebank section 0 that have this label; *Count* is the number of constituents that have this label.

Proportion	Count	Label	Recall	Precision
42.21	15146	NP	91.15	90.26
19.78	7096	VP	91.02	91.11
13.00	4665	S	91.21	90.96
12.83	4603	PP	86.18	85.51
3.95	1419	SBAR	87.81	88.87
2.59	928	ADVP	82.97	86.52
1.63	584	ADJP	65.41	68.95
1.00	360	WHNP	95.00	98.84
0.92	331	QP	84.29	78.37
0.48	172	PRN	32.56	61.54
0.35	126	PRT	86.51	85.16
0.31	110	SINV	83.64	88.46
0.27	98	NX	12.24	66.67
0.25	88	WHADVP	95.45	97.67
0.08	29	NAC	48.28	63.64
0.08	28	FRAG	21.43	46.15
0.05	19	WHPP	100.00	100.00
0.04	16	UCP	25.00	28.57
0.04	16	CONJP	56.25	69.23
0.04	15	SQ	53.33	66.67
0.03	12	SBARQ	66.67	88.89
0.03	9	RRC	11.11	33.33
0.02	7	LST	57.14	100.00
0.01	3	X	0.00	—
0.01	2	INTJ	0.00	—

Fehleranalyse: Modell 2

Recall and precision for different constituent types, for section 0 of the treebank with model 2. *Label* is the nonterminal label; *Proportion* is the percentage of constituents in the treebank section 0 that have this label; *Count* is the number of constituents that have this label.

Proportion	Count	Label	Recall	Precision
42.21	15146	NP	91.15	90.26
19.78	7096	VP	91.02	91.11
13.00	4665	S	91.21	90.96
12.83	4603	PP	86.18	85.51
3.95	1419	SBAR	87.81	88.87
2.59	928	ADVP	82.97	86.52
1.63	584	ADJP	65.41	68.95
1.00	360	WHNP	95.00	98.84
0.92	331	QP	84.29	78.37
0.48	172	PRN	32.56	61.54
0.35	126	PRT	86.51	85.16
0.31	110	SINV	83.64	88.46
0.27	98	NX	12.24	66.67
0.25	88	WHADVP	95.45	97.67
0.08	29	NAC	48.28	63.64
0.08	28	FRAG	21.43	46.15
0.05	19	WHPP	100.00	100.00
0.04	16	UCP	25.00	28.57
0.04	16	CONJP	56.25	69.23
0.04	15	SQ	53.33	66.67
0.03	12	SBARQ	66.67	88.89
0.03	9	RRC	11.11	33.33
0.02	7	LST	57.14	100.00
0.01	3	X	0.00	—
0.01	2	INTJ	0.00	—

Fehleranalyse

- Fehlermöglichkeiten (Trainiert auf Sektion 0 der Penn Treebank)
  - Komplement zum Verb

Type	Sub-type	Description	Count	Recall	Precision
Complement to a verb	S VP NP-C L	Subject	3,248	95.75	95.11
	VP TAG NP-C R	Object	2,095	92.41	92.15
6,495 = 16.3% of all cases	VP TAG SBAR-C R		558	94.27	93.93
	VP TAG SG-C R		316	92.41	88.22
	VP TAG S-C R		150	74.67	78.32
	S VP S-C L		104	93.27	78.86
	S VP SG-C L		14	78.57	68.75
	...				
	Total		6,495	93.76	92.96

  - Coordination

Type	Sub-type	Description	Count	Recall	Precision
Coordination	NP NP NP R		289	55.71	53.31
	VP VP VP R		174	74.14	72.47
763 = 1.9% of all cases	S S S R		129	72.09	69.92
	ADJP TAG TAG R		28	71.43	66.67
	VP TAG TAG R		25	60.00	71.43
	NX NX NX R		25	12.00	75.00
	SBAR SBAR SBAR R		19	78.95	83.33
	PP PP PP R		14	85.71	63.16
	...				
	Total		763	61.47	62.20

Fehleranalyse

- NP-Erweiterung

Type	Sub-type	Description	Count	Recall	Precision
Modification to NPs	NP NPB NP R	Appositive	495	74.34	75.72
	NP NPB SBAR R	Relative clause	476	79.20	79.54
1,418 = 3.6% of all cases	NP NPB VP R	Reduced relative	205	77.56	72.60
	NP NPB SG R		63	88.89	81.16
	NP NPB PRN R		53	45.28	60.00
	NP NPB ADVP R		48	35.42	54.84
	NP NPB ADJP R		48	62.50	69.77
	...				
	Total		1,418	73.20	75.49

- PP-Erweiterung

Type	Sub-type	Description	Count	Recall	Precision
PP modification	NP NPB PP R		2,112	84.99	84.35
	VP TAG PP R		1,801	83.62	81.14
4,473 = 11.2% of all cases	S VP PP L		287	90.24	81.96
	ADJP TAG PP R		90	75.56	78.16
	ADVP TAG PP R		35	68.57	52.17
	NP NP PP R		23	0.00	0.00
	PP PP PP L		19	21.05	26.67
	NAC TAG PP R		12	50.00	100.00
	...				
	Total		4,473	82.29	81.51

Fehleranalyse

- Weitere oft vorkommende Fehler:
  - Komplemente von NP, ADJP
  - Basis-NP-Erweiterung
  - Satzwertiger Kopf
  - Adjunkte zum Verb

Es wurde hier nicht ausführlich behandelt ...

- Der erste Schritt beim Generieren: TOP → S(h)
  - Deshalb muss man die Wahrscheinlichkeit von S(h) einbeziehen.
- POS-Tagging – Collins betrachtet als letzte Ebene die POS-Tags, nicht die Wörter selbst
- Subkategorisierungsrahmen [subcategorisation frame]: Festlegung der Anzahl der Argumente rechts und links vom Kopf
- Distanz [distance]: gemeint wird die Distanz zwischen dem Kopfknoten und STOP-Knoten, also die Anzahl der im Subkategorisierungsrahmen festgelegten, potentiellen Argumente
- Glättung [smoothing]: mit Hilfe vom Glättungsparameter λ werden die in der Penn Treebank nicht vorkommenden Daten auch betrachtet und ins Modell einbezogen
  - Methode: Deleted Interpolation → ausgeglichenere Ergebnisse
- „UNKNOWN" Token: alle Wörter, die weniger als 5-mal im Trainingskorpus vorkommen, werden als Unbekannte markiert.
- Einbezug
  - der Interpunktion in die Auswertung
  - der Koordination
  - der Adjazenz

Literatur

- *Michael John Collins (1996): A New Statistical Parser Based on Bigram Lexical Dependencies*
  - http://acl.ldc.upenn.edu/P/P96/
- *Michael John Collins (1997), Three Generative, Lexicalised Models For Statistical Parsing*
  - http://www.dei.unipd.it/~satta/teach/nlp/online/collins.pdf
- *Michael John Collins (1997, Folien): Three Generative, Lexicalised Models For Statistical Parsing*
  - http://people.csail.mit.edu/mcollins/papers/acl97.2.ps
- *Michael John Collins (2003): Head-Driven Statistical Models for Natural Language Parsing*
  - http://people.csail.mit.edu/mcollins/papers/CL2003.ps
- *Manfred Klenner, (WS 2005/2006, Folien): Quantitative Methoden in der Computerlinguistik*
  - http://www.ifi.unizh.ch/cl/klenner/lehre/ws0304/syntax/skript_pcfg.4.pdf
- *Sharon Friedrich, Katharina Wäschle (SS 2007, Folien): Lexikalisiertes Parsing*
  - http://www.cl.uni-heidelberg.de/kurs/ss07/pars/bib/2007parsing.friedrich.waeschle