What extracting grammars from treebanks can tell us about linguistic theory

Julia Hockenmaier
University of Illinois
juliahmr@cs.uiuc.edu

Treebanks and Linguistic Theories 2007

Grammar extraction

In this talk:

- Source treebank: Penn Treebank, Tiger
- Target treebank: Combinatory Categorial Grammar

The Penn Treebank

```
(NP (NP the shares)
    (SBAR (WHNP-1 (WDT which))
          (S (NP-SBJ IBM)
             (VP (VBZ has)
                 (VP (VBN bought)
                     (NP (-NONE- *T*-1))
                     (NP-TMP last year))))))
```

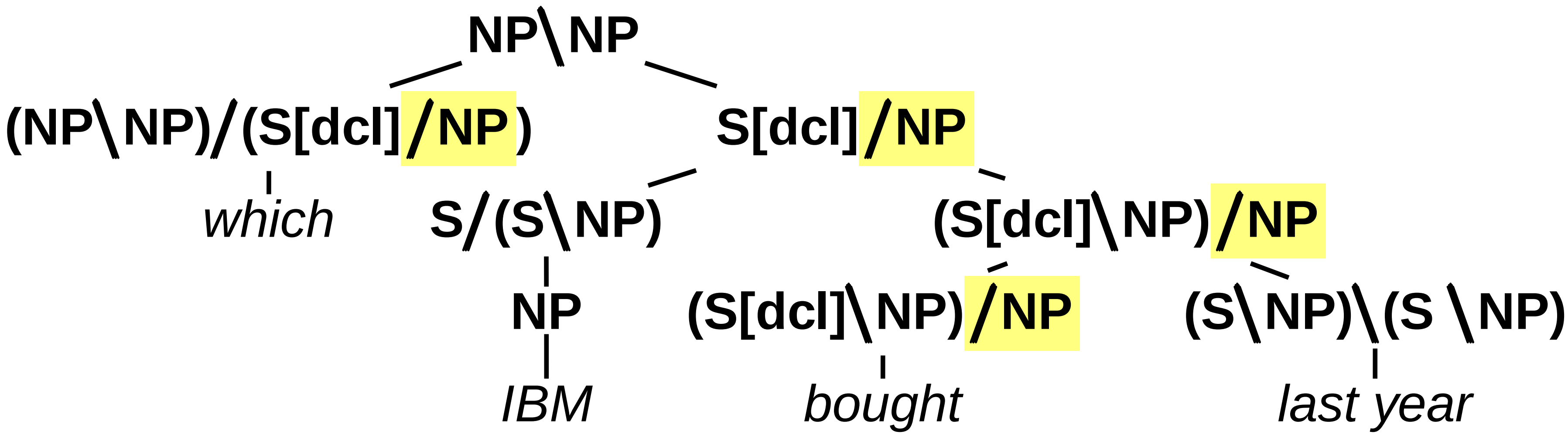
- **Function tags**: complement-adjunct distinction
- **Co-indexed null elements**: long-range dependencies

Penn Treebank Parsing

```
(NP (NP the shares)
    (SBAR (WHNP-1 (WDT which))
          (S (NP-SBJ IBM)
             (VP (VBZ has)
                 (VP (VBN bought)
                     (NP (-NONE- *T*-1))
                     (NP-TMP last year))))))
```

- Function tags: **complement-adjunct** distinction
- **Co-indexed null elements**: long-range dependencies
- **Standard parsers do not reproduce this information**

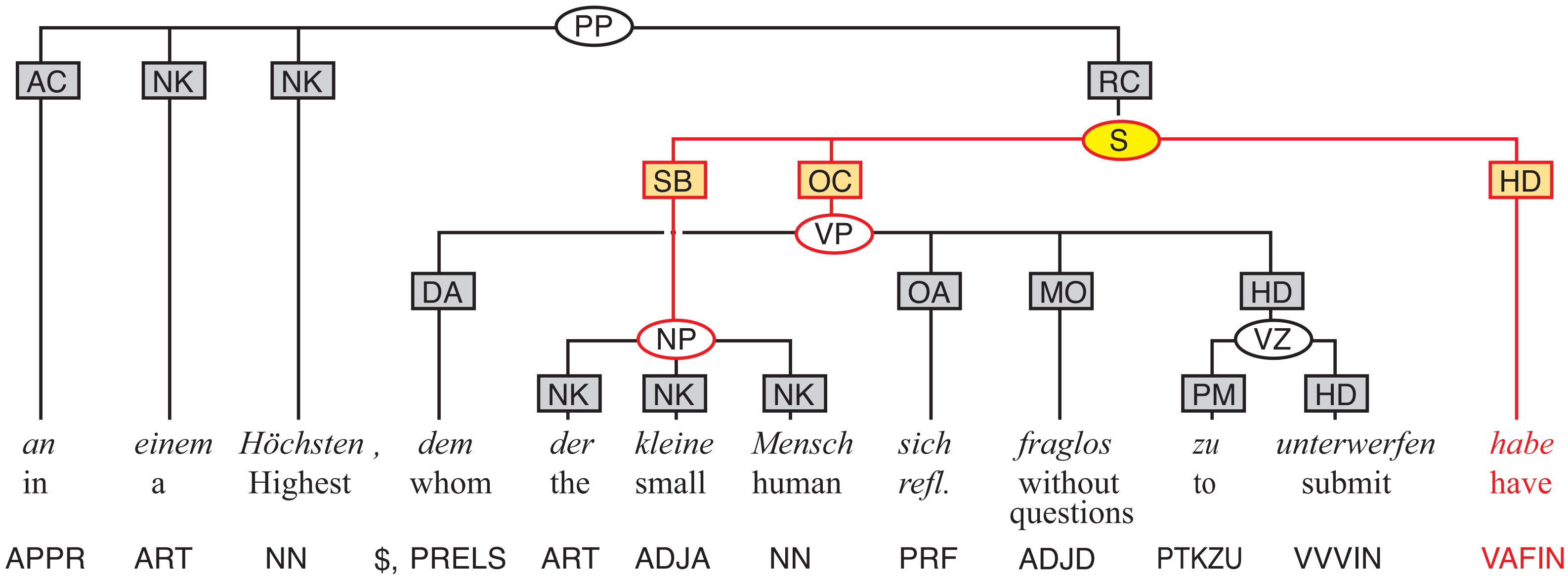
The CCG derivation



- **Long-range dependency** is captured
- **Complement-adjunct** distinction is captured
- **No traces** are needed.

  We can use standard parsing algorithms

The Tiger corpus



- **Edge labels**: heads, complement-adjunct distinction
- **Discontinuous constituents**: extraction, scrambling
- **Secondary edges**: non-constituent coordination

Combinatory Categorial Grammar

... is a *lexicalized* grammar formalism

- Small number of universal combinatory rules
- All language-specific information is in the lexicon

... is a *mildly context-sensitive* grammar formalism

- can be parsed in polynomial time
- can capture Dutch cross-serial dependencies
- is weakly equivalent to Tree-Adjoining Grammar (TAG)

... has a *transparent syntax-semantics interface*

- Fully compositional semantics

Combinatory Categorial Grammar

... had no wide-coverage implementation

Treebanks...

... can contain arbitrary text:

- **arbitrarily *long* sentences:**
  parentheticals, speech repairs, complex coordinations...
- **arbitrarily *short* sentences:**
  fragments, ellipsis, headlines, ...

... can provide arbitrary descriptions:

- **arbitrarily *complex* descriptions:**
  coindexation, null elements, secondary edges, crossing edges,...
- **arbitrarily *simplified/shallow* descriptions:**
  compound nouns, fragments, complement-adjunct distinction, ...

Linguistic theories

... provide analyses for well-studied constructions

- It might be unclear how to analyze
  less well-studied constructions

... may provide limited expressivity

- *Mildly context-sensitive* formalisms can't capture
  arbitrary dependencies

... may require complete analyses

- *Lexicalized* formalisms need lexical entries for every word

Research questions

- **Are the descriptions in the treebank sufficient** to obtain the analyses stipulated by the linguistic theory?
- **Can the linguistic theory account for the descriptions** given in the treebank?

Combinatory Categorial Grammmar

CCG categories and derivations

- **Atomic categories:**

  NP, **S**, **PP**...

- **Complex categories:**

  S\NP, **(S\NP)/NP**, **(NP\NP)/NP**,...

- **Derivations:** spell out process of combining constituents

I	*drink*	*coffee*
NP	**(S\NP)/NP**	**NP**

S\NP (>)

S (<)

CCG's combinatory rules

Function application:

$X/Y \quad Y \Rightarrow X$

$Y \quad X\backslash Y \Rightarrow X$

Function composition:

$X/Y \quad Y/Z \Rightarrow X/Z$

$Y\backslash Z \quad X\backslash Y \Rightarrow X\backslash Z$

Type raising:

$X \Rightarrow T/(T\backslash X)$

$X \Rightarrow T\backslash(T/X)$

Coordination:

$X \text{ conj } X \Rightarrow X$

Features on categories

- We need different types of sentences, VPs, adjectives:
  - **S[dcl]** = main clause
  - **S[b]\NP** = bare infinitival VP
  - **S[to]\NP** = to-VP
  - **S[adj]\NP** = predicative adjective
- In CCGbank, features are atomic.

CCG: syntax-semantics interface

Every syntactic category and rule has
a **semantic interpretation:**

Function application	**X/Y**:$\lambda x.f(x)$	**Y**:a	$\Rightarrow$**X**:$f(a)$
Function composition	**X/Y**:$\lambda x.f(x)$	**X/Y**:$\lambda x.g(x)$	$\Rightarrow$**X/Z**:$\lambda x.f(gx)$
Type-raising	**X**:a		$\Rightarrow$ **T/(T\X)**:$\lambda f.f(a)$

I — *drink* — *coffee*

NP:I' — **(S[dcl]\NP)/NP:**$\lambda x.\lambda y.drink'xy$ — **NP:**$coffee'$

$>$ **S[dcl]\NP:**$\lambda y.drink'coffee'y$

$<$ **S[dcl]:**$drink'coffee'I'$

Approximating semantics with dependencies

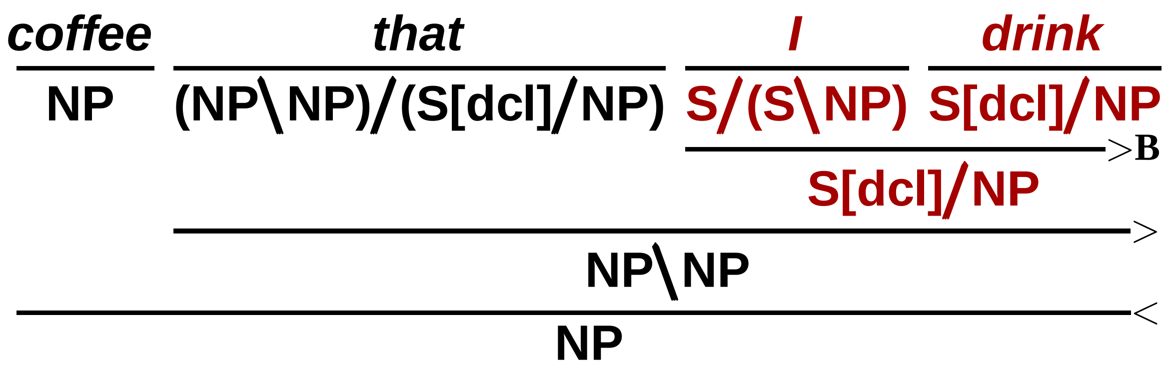
Every argument of a lexical functor category defines a dependency:

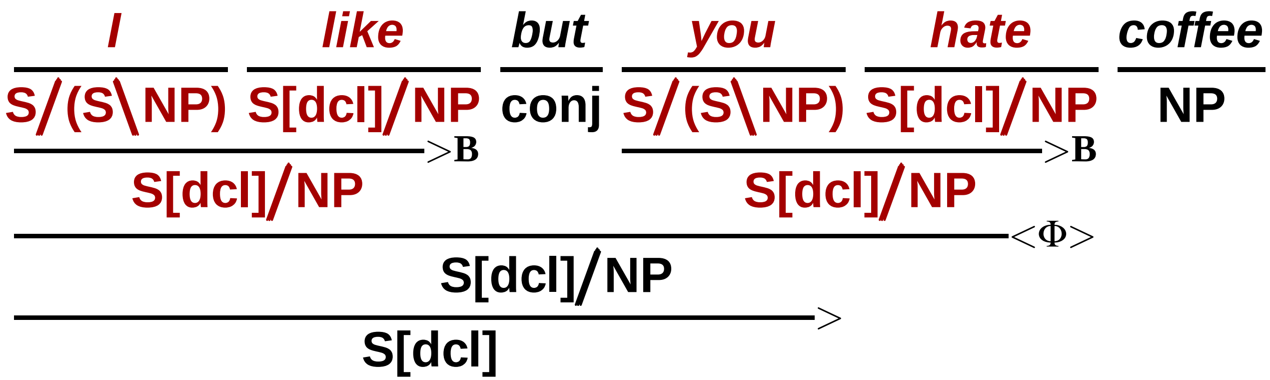
$$\dfrac{\dfrac{\textit{drink}}{(S[dcl]\backslash NP_1)/NP_2} \quad \dfrac{\textit{coffee}}{NP_2}}{S[dcl]\backslash NP_1}>$$

$$\langle \textit{drink}, \mathbf{(S[dcl]\backslash NP_1)/NP_2}, \mathbf{2}, \textit{coffee} \rangle$$

Non-local dependencies in CCG

Unified analysis of wh-extraction and right-node raising

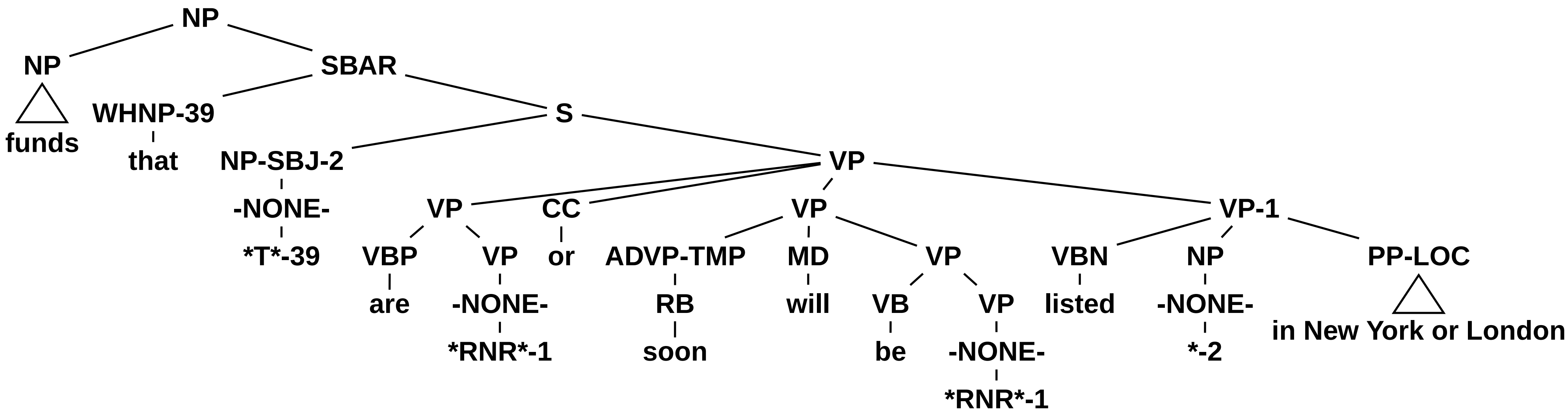




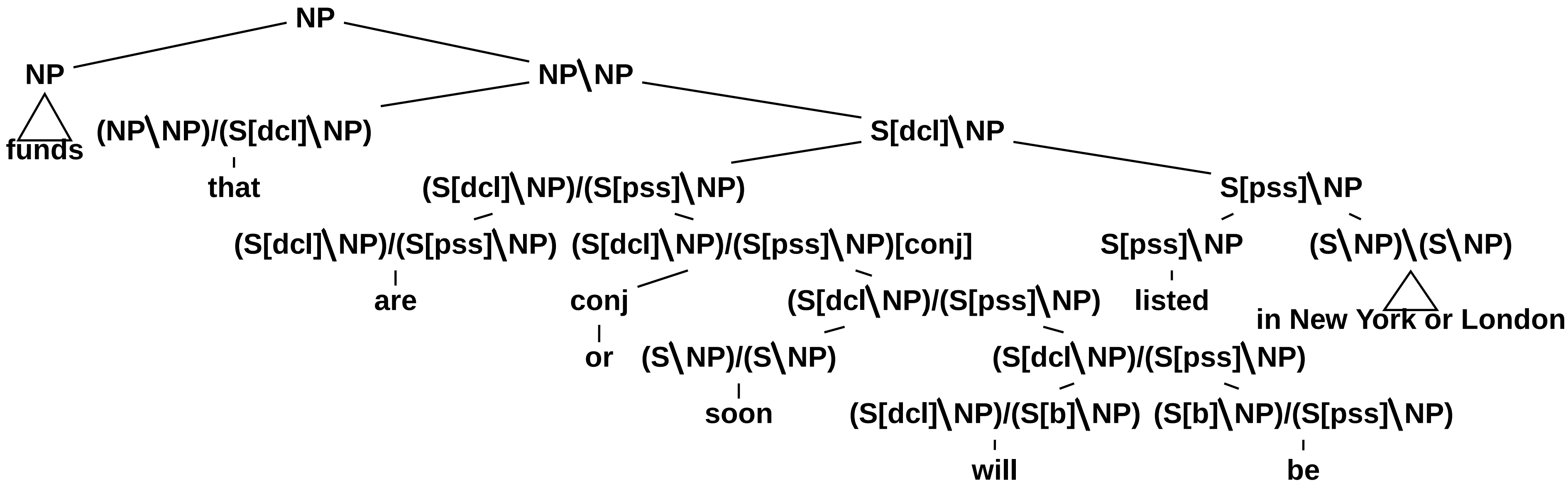
Translating the Penn Treebank to CCG

Input: Penn Treebank tree

funds that are or soon will be listed in New York or London.

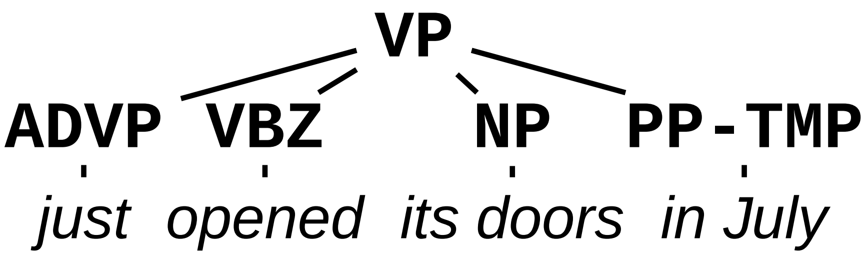


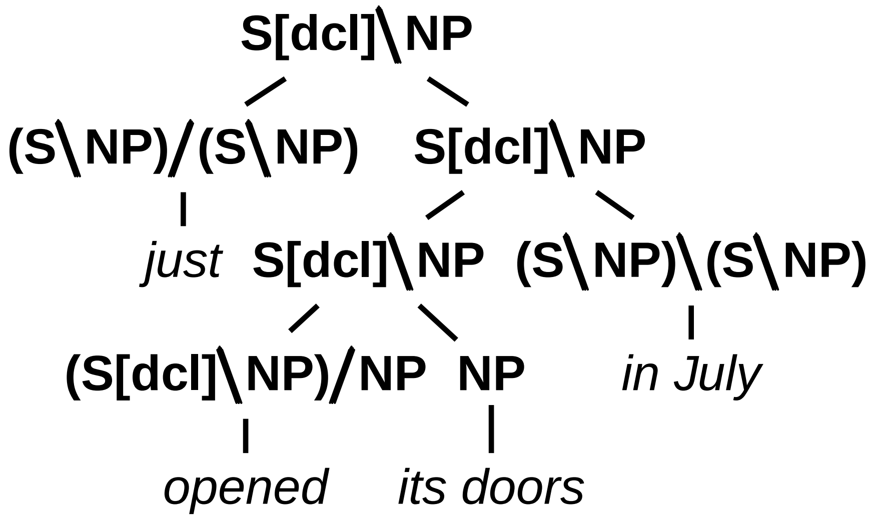
Out: CCG derivation + dependencies



that	((NP\NP)/(S[dcl]\NP))	funds	are, will	
are	((S[dcl]\NP)/(S[pss]\NP))	*funds*	*listed*	
soon	((S\NP)/(S\NP))		will	
will	((S[dcl]\NP)/(S[b]\NP))	*funds*	be	
be	((S[b]\NP)/(S[pss]\NP))		listed	
listed	(S[pss]\NP)	*funds*		
in	(((S\NP)\(S\NP))/NP)		listed	York, London

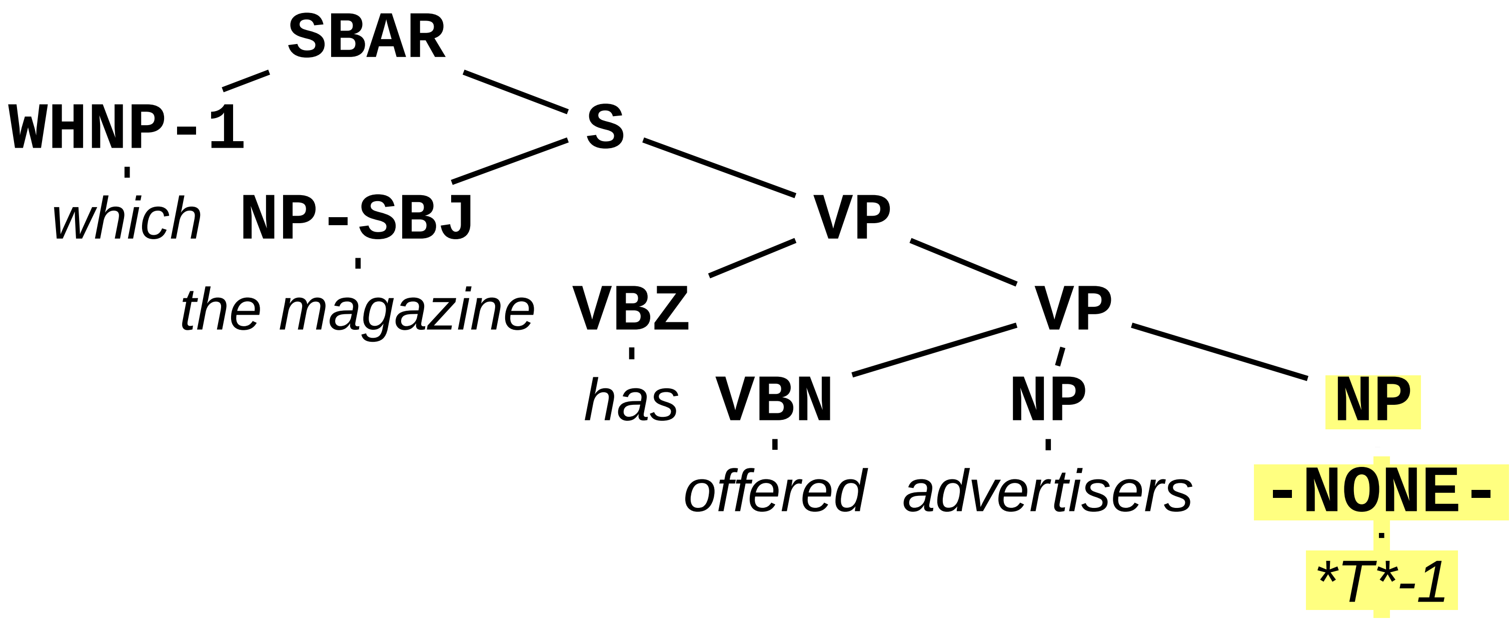
The basic translation algorithm



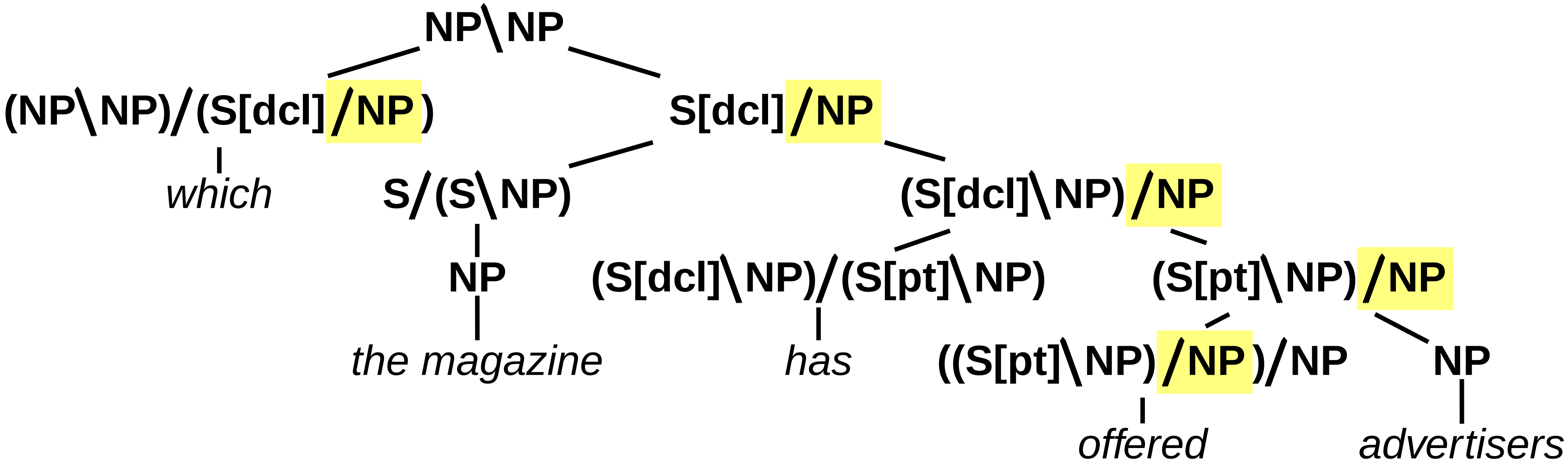


1. Identify heads, arguments, adjuncts
2. Binarize tree
3. Read off CCG categories
4. Get dependency structure

Dealing with extraction



Dealing with extraction – the final tree



- The **relative pronoun** subcategorizes for an incomplete sentence.
- Use **type-raising** and **composition** where necessary
- The **trace** is cut out, but appears in the derivation

Right-node raising

```
(S (NP-SBJ She)
   (VP (VP (VBD applied)
           (PP-CLR (IN for)
                   (NP (-NONE- *RNR*-1))))
       (CC and)
       (VP (VBD won)
           (NP (-NONE- *RNR*-1)))
       (NP-1  bonus pay)))
```

Right-node raising in CCG

She	*applied*	*for*	*and*	*won*	*bonus pay*
NP	(S[dcl]\NP)/PP	PP/NP	conj	(S[dcl]\NP)/NP	NP

applied for: (S[dcl]\NP)/PP + PP/NP $\Rightarrow$ (S[dcl]\NP)/NP (>B)

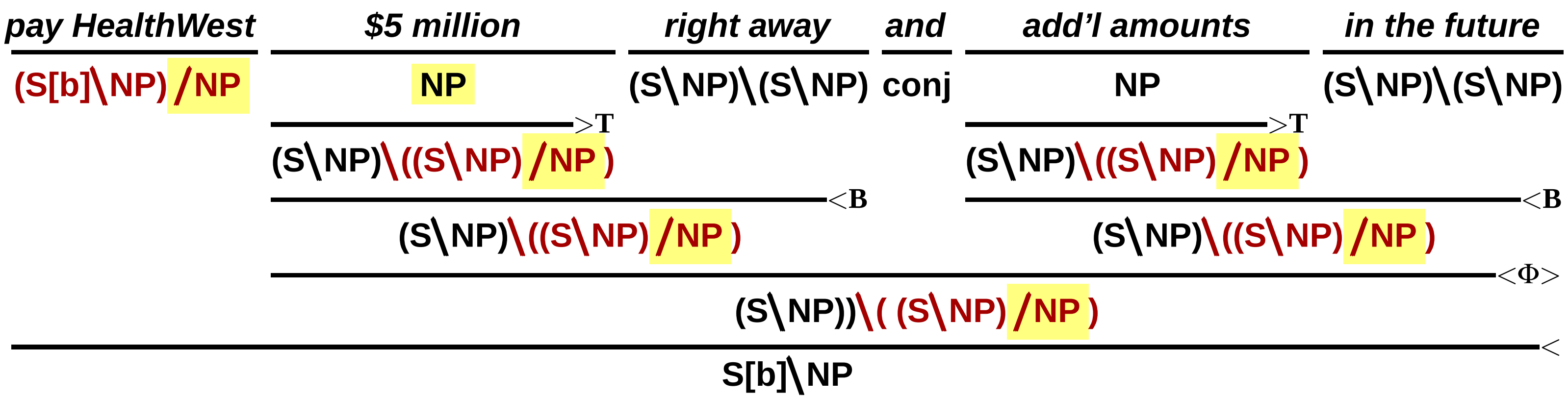
applied for and won: (S[dcl]\NP)/NP conj (S[dcl]\NP)/NP $\Rightarrow$ (S[dcl]\NP)/NP (<Φ>)

applied for and won bonus pay: (S[dcl]\NP)/NP + NP $\Rightarrow$ S[dcl]\NP (>)

She applied for and won bonus pay: NP + S[dcl]\NP $\Rightarrow$ S[dcl] (<)

Non-constituent coordination

```
(VP (VP (VB pay)
        (NP HealthVest)
        (NP-2 $ 5 million)
        (ADVP-TMP-3 right away))
    (CC and)
    (VP (NP=2 additional amounts)
        (PP-TMP=3 in the future))
```

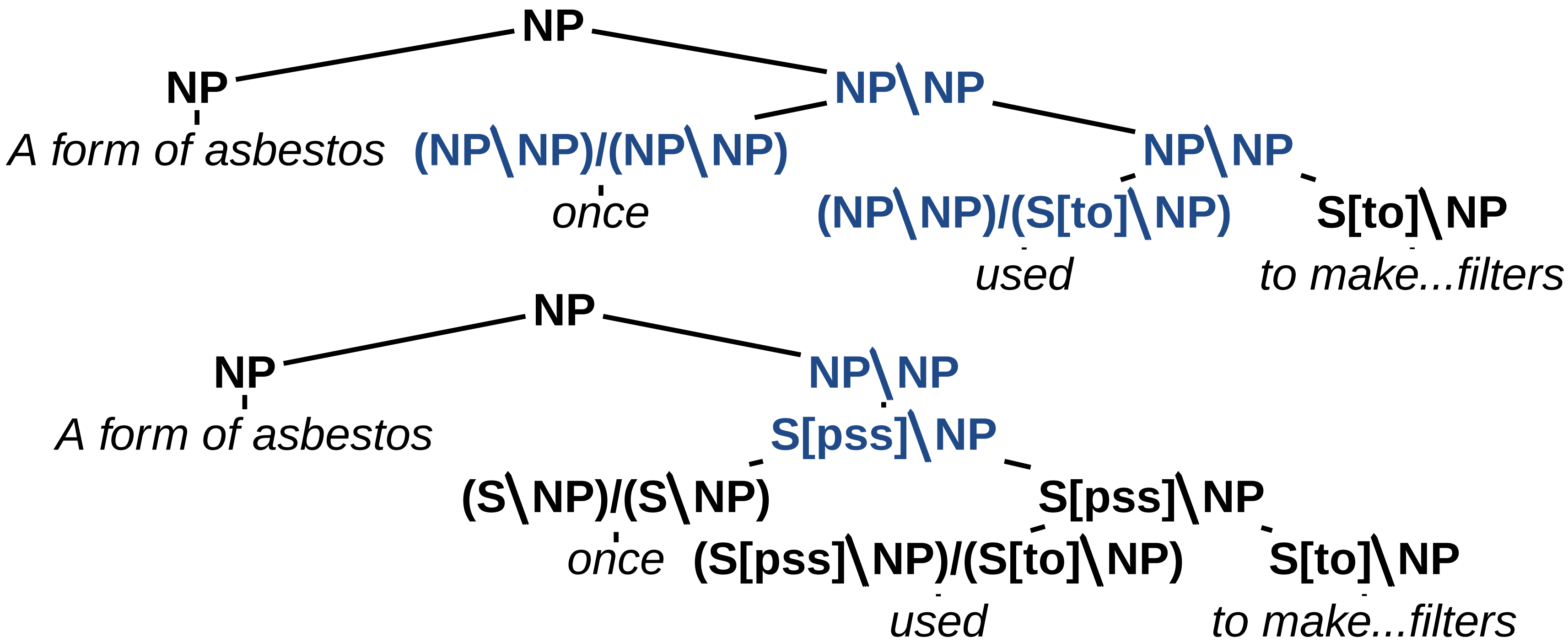


Proliferation of adjunct categories

Standard CCG leads to a proliferation of categories

used	**S[pss]\NP**
used	**NP\NP**
used	**(NP\NP)\(NP\NP)**
used	**(S\NP)\(S\NP)**
used	**((S\NP)\(S\NP))\((S\NP)\(S\NP))**

Type-changing rules for adjuncts



- **Type-changing rules** capture syntactic regularities
- Cf. lexical rules (Carpenter), zero morphemes (Aone/Wittenburg)

Preprocessing the Treebank

- **Clean up Treebank**:
  - POS-tagging errors: wrong categories/verb features
  - Bracketing errors: eg. coordination
- **Change Treebank analysis**
  where it doesn’t conform to CCG analysis, eg.:
  - Insert noun level
  - “Non-constituent” coordination
  - Small clauses

CCGbank: the resulting corpus

- **Translation coverage:** 99.44% of all sentences
- **Size of lexicon and grammar:**
  - Lexicon: 74,669 entries for 44,210 word types
    1286 lexical category types, 439 appear only once
  - Grammar: 3262 rule instantiations, 1146 appear once.
- **Linguistically interesting observations:**
  - At least three parasitic gaps
  - More than twice as many RNR instances as Treebank suggests

Lexical coverage on unseen data

- **How well does this CCG cover unseen text?**
  - Split the corpus into two parts (section 02-21; section 00)
  - Translate both parts.
  - How well does the lexicon from the large part cover the small part?

- **For all word-category pairs in section 00:**
  - The word-category pair is known: **94.0%**
  - The word is known, but not with this category: **2.2%**
  - The word is unknown: **3.8%**

CCGbank parsing

- **Hockenmaier (2003) generative parser:**
  CCGbank dependencies:
  84.4 F-score (labeled) 92.0 F-score (unlabeled)

- **Clark & Curran (2007) loglinear parser:**
  CCGbank dependencies:
  87.6 F-score (labeled) 93.0 F-score (unlabeled)
  Parses section 23 in $\leq$ 7 minutes

Summary: CCGbank

- We can translate the Penn Treebank into CCG derivations and dependency structures.

- The resulting corpus is publicly available from the Linguistic Data Consortium.

- This allows us (you!) to create statistical CCG parsers.

Translating the Tiger corpus to CCG

Statistical parsing for German

- **German has a much freer word order than English:**
  - German corpora (Negra, Tiger) contain discontinuous constituents
  - Context-free representations are not appropriate
    (although still commonly used)

- **This creates problems for surface-dependency models:**
  - Dependency models do not help. (Dubey & Keller, 2003)
  - Context-free representations can only *approximate*
    the underlying dependencies (Levy & Manning, 2004)

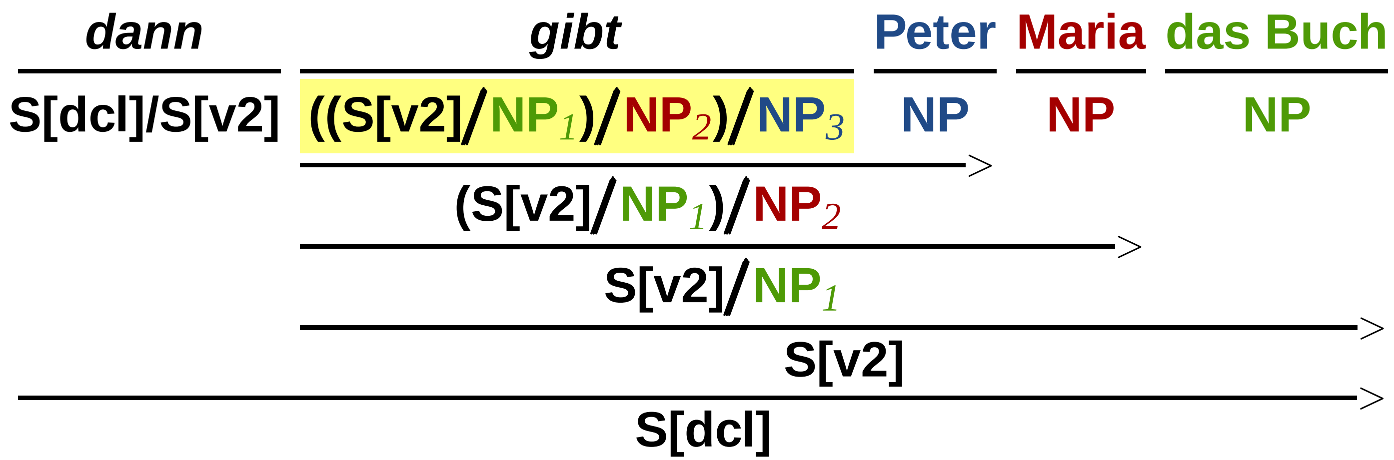
Topological fields

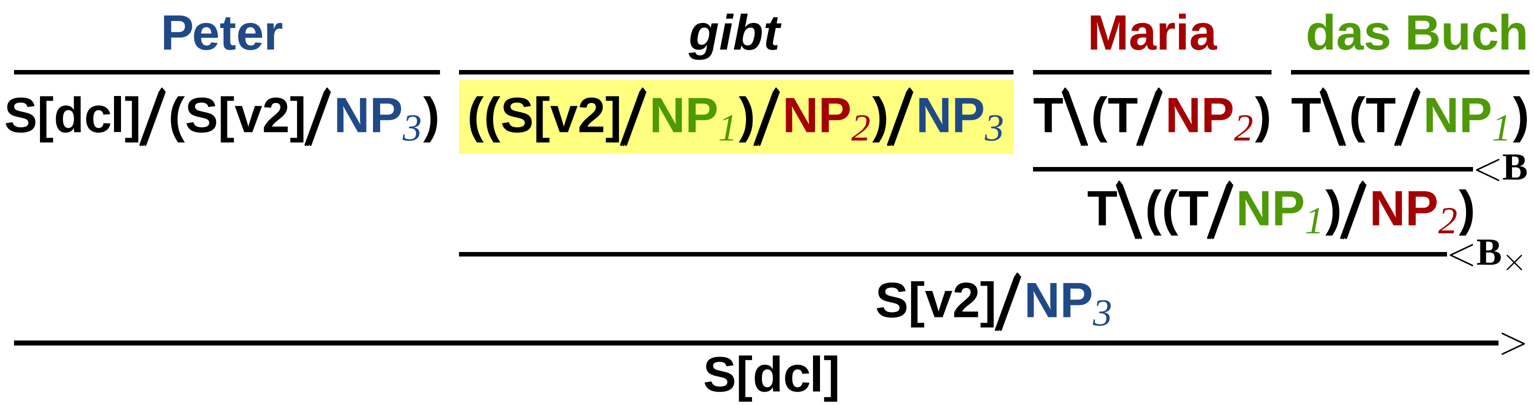
Vorfeld	Left Bracket	Mittelfeld	Right Bracket	Nachfeld
Maria	liest	das Buch morgen		das Peter ihr gegeben hat
Das Buch		Maria morgen		
Morgen		Maria das Buch		
Maria	wird	das Buch morgen	gelesen haben	
Das Buch		Maria morgen		
Morgen		Maria das Buch		
	dass	Maria das Buch morgen	gelesen haben wird	

Translation:

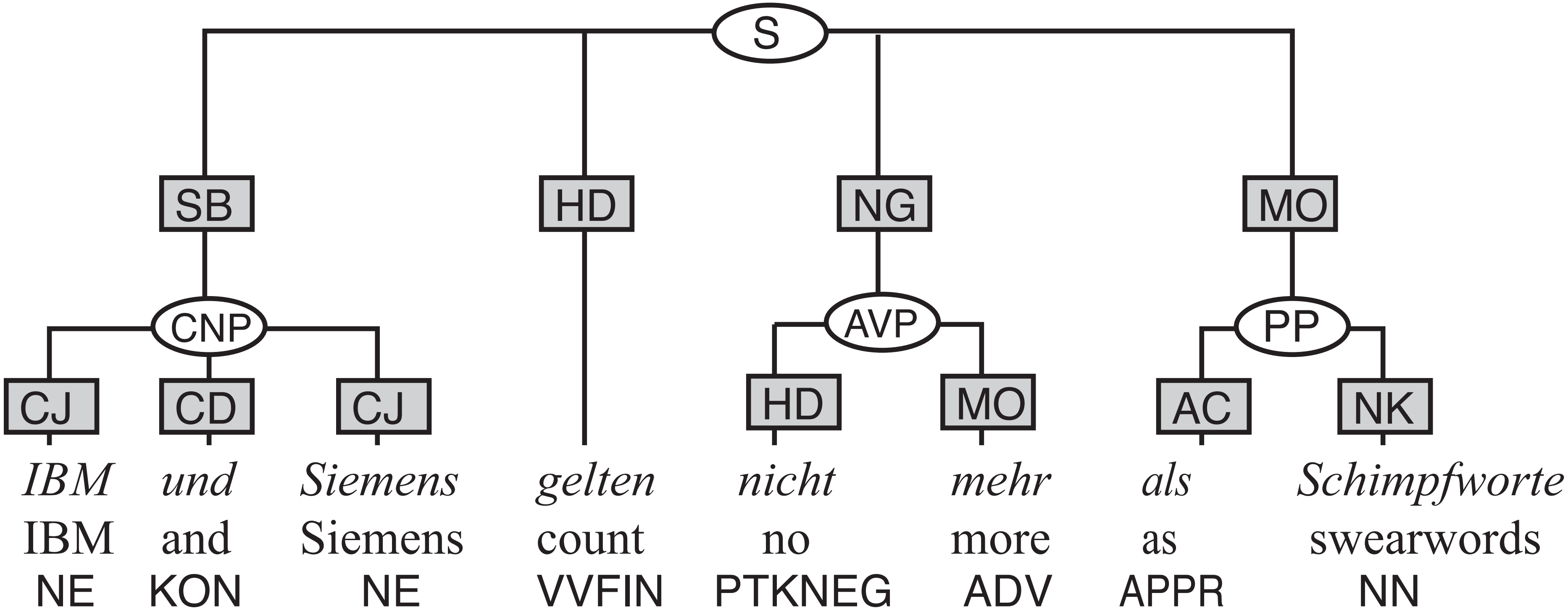
1. Mary reads the book that Peter has given her tomorrow.
2. Mary will have read the book that Peter has given her tomorrow.
3. ...that Mary will have read the book that Peter has given her tomorrow

Scrambling in CCG

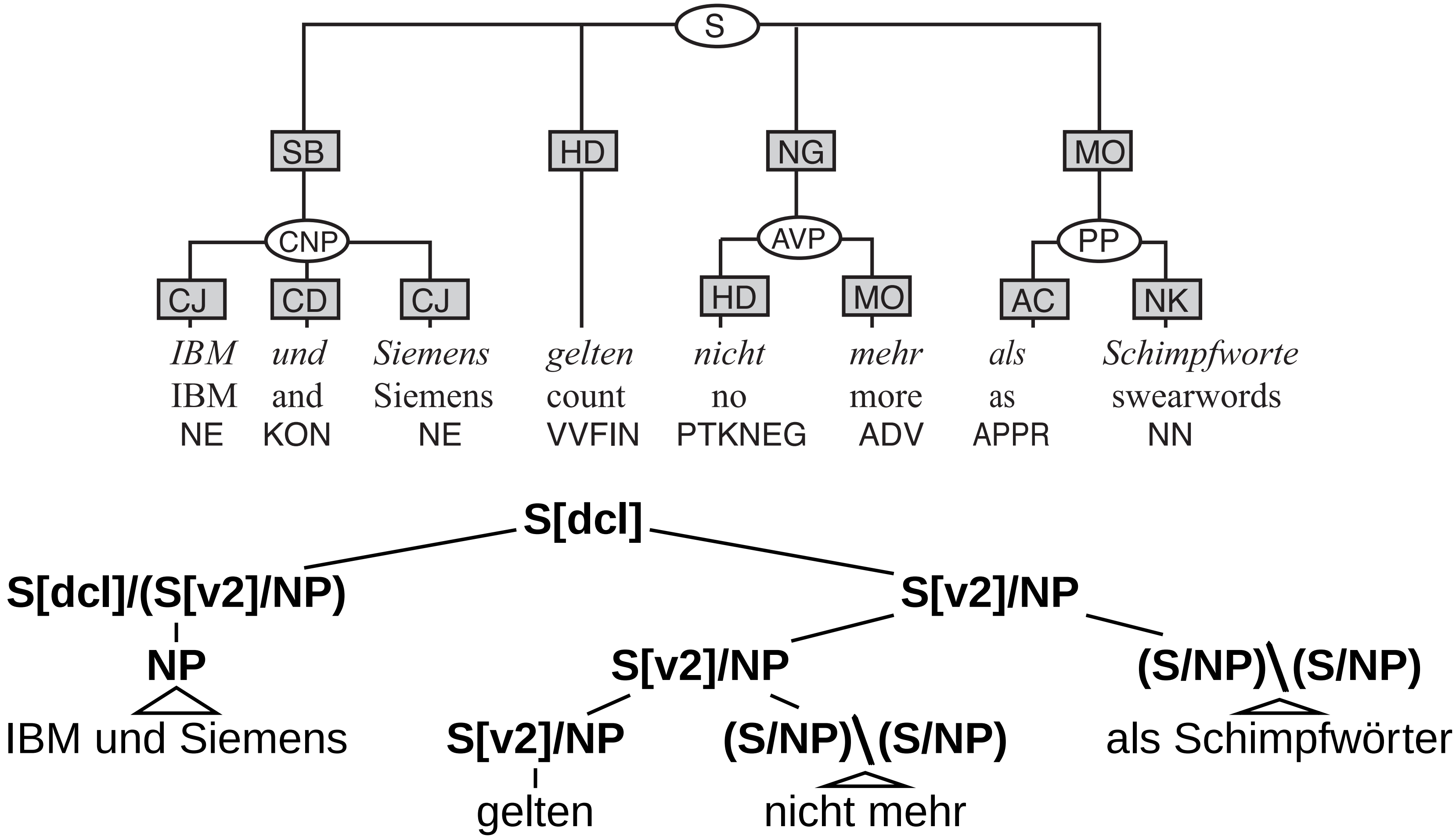




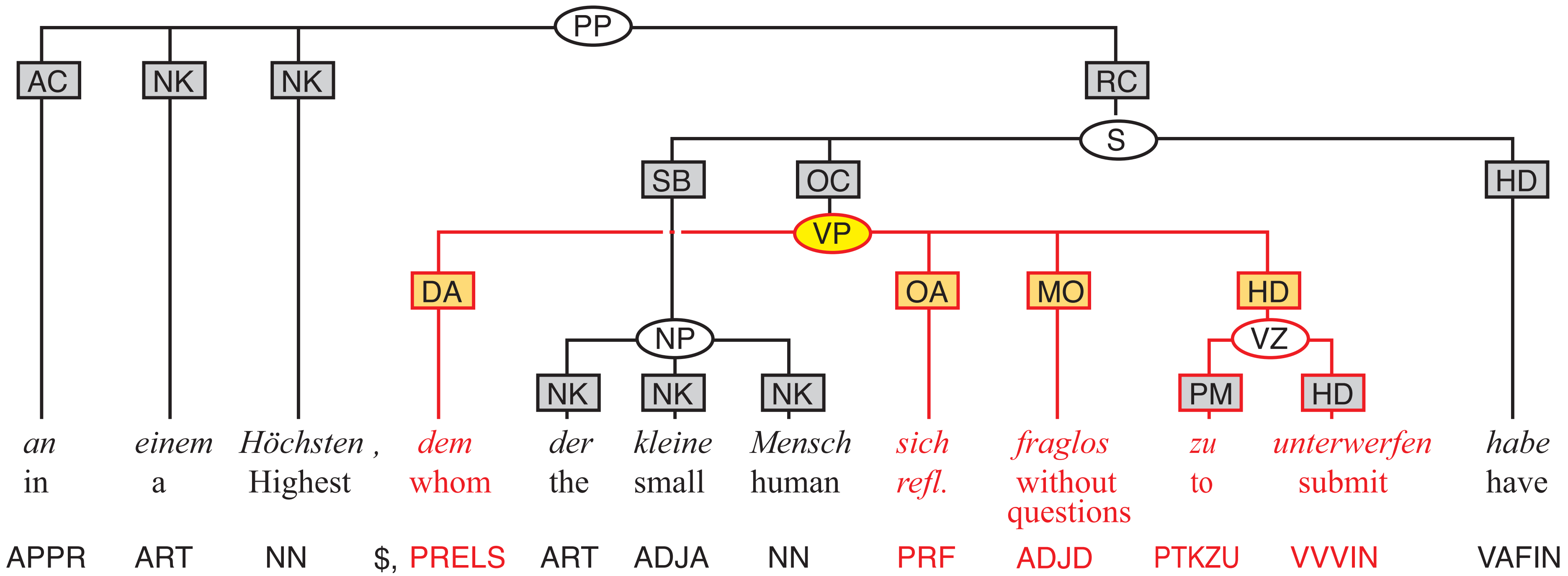
A Tiger graph



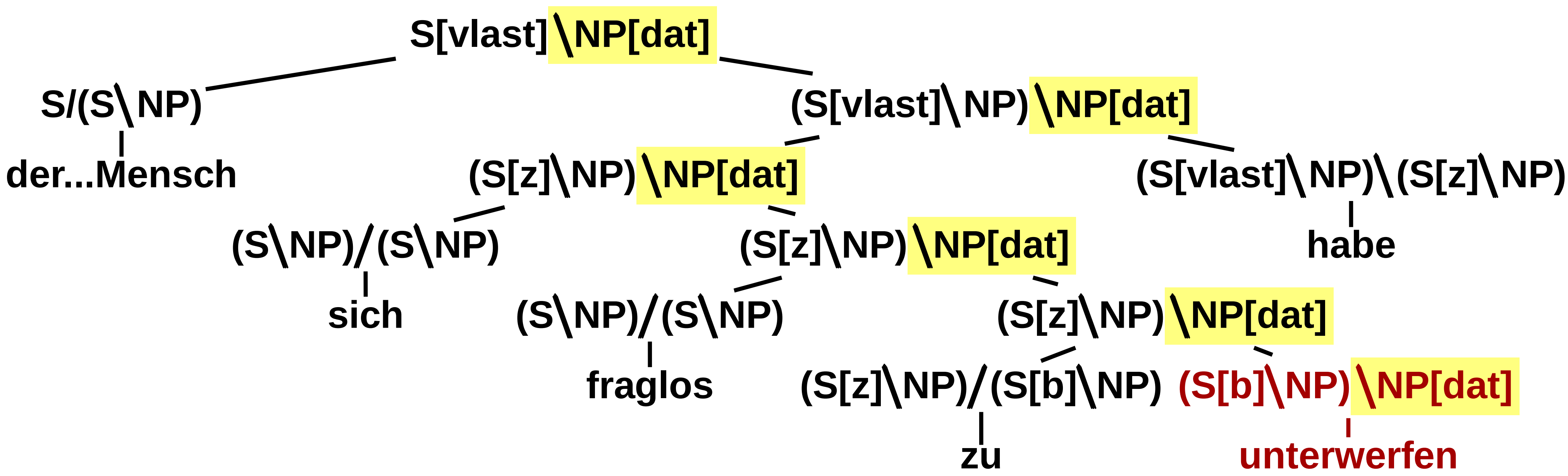
From Tiger to CCG



Discontinuous constituents



The CCG derivation



Translation coverage

- **We translate 46,628 graphs into CCG (92.4% of all graphs, 88.4% of all discontinuous graphs)**
- **Reasons for failure/non-translation:**
  - Graph not a sentence: 1.7%
  - Cannot find head: 1.3%
  - Graph cannot be planarized: 1.3%
  - Translation not a CCG derivation: 1.9%

Lexicon size and coverage

- **2,506 distinct lexical categories**
  (1,018 appear only once, 933 more than 5 times)
- **Lexical coverage** (10-fold CV):
  - overall: 86.7% avg (min: 84.4%, max: 87.6%)
  - Known words: 94.2% avg. (min: 93.5%, max: 92.6%)
- **Comparison with English CCGbank:**
  - 1,300 lexical categories
  - Lexical coverage: 94% (known words: 97.7%)

    But: German CCGbank contains case information

The problem with subjects

"den Kritiker gehasst und Fans geliebt haben"

"whom critics have loved and fans have hated"

den *whom*	*Kritiker* *critics*	*gehasst* *hated*	*und* *and*	*Fans* *fans*	*geliebt* *loved*	*haben* *have*
(NP\NP)/(S[vl]\NP)	NP	(S[pt]\NP)\NP	conj	NP	(S[pt]\NP)\NP	(S[vl]\NP)\(S[pt]\NP)
	S[pt]\NP			S[pt]\NP		

<Φ>

S[pt]\NP

There is no CCG derivation where

the subjects are arguments of the auxiliary!

The problem with subjects

"den die Kritiker gehasst und die Fans geliebt haben"
"which the critics have loved and the fans have hated"

den	Kritiker	gehasst	und	Fans	geliebt	haben
whom	*critics*	*hated*	*and*	*fans*	*loved*	*have*
(N\N)/(S[vl]\NP)	S/(S\NP)	(S[pt]\NP)\NP	conj	S/(S\NP)	(S[pt]\NP)\NP	S[vl]\S[pt]

S/(S\NP) (S[pt]\NP)\NP ⇒ S[pt]\NP

S/(S\NP) (S[pt]\NP)\NP ⇒ S[pt]\NP

S[pt]\NP conj S[pt]\NP ⇒ S[pt]\NP <Φ>

S[pt]\NP S[vl]\S[pt] ⇒ S[vl]\NP <B

If subjects are arguments of the main verb,
this is just standard extraction.

Verb cluster fronting

Zu reparieren versucht *hat* Hans das Auto

to repair tried has Hans the car

Hans has tried to repair the car.

zu reparieren	versucht	*hat*	Hans	das Auto
$\mathrm{VP[zu]}\backslash \mathrm{NP}_a$	$(\mathrm{S[pt]}\backslash \mathrm{NP}_n)\backslash \mathrm{VP[zu]}$	$\mathrm{S[v1]/S[pt]}$	NP_n	NP_a

zu reparieren + versucht ($<B$): $(\mathrm{S[pt]}\backslash \mathrm{NP}_n)\backslash \mathrm{NP}_a$

($>T$): $\mathrm{S[dcl]}/(\mathrm{S[v1]}/(\mathrm{S[pt]}\backslash \mathrm{NP}_n)\backslash \mathrm{NP}_a))$

Hans ($>T$): $\mathrm{S[pt]}/(\mathrm{S[pt]}\backslash \mathrm{NP}_n)$

das Auto ($>T$): $\mathrm{T}/(\mathrm{T}\backslash \mathrm{NP}_a)$

Hans + das Auto ($>B$): $\mathrm{S[pt]}/((\mathrm{S[pt]}\backslash \mathrm{NP}_n)\backslash \mathrm{NP}_a)$

hat + Hans das Auto ($>B$): $\mathrm{S[v1]}/((\mathrm{S[pt]}\backslash \mathrm{NP}_n)\backslash \mathrm{NP}_a)$

($>$): $\mathrm{S[dcl]}$

The verb cluster

1. dass sie ihm eine Garantie geben müssen wird.
2. dass sie ihm eine Garantie wird geben müssen.
3. dass sie ihm eine Garantie geben wird müssen.

1. *geben* *müssen* wird

S[inf]\NP\NP\NP S[inf]\S[inf] S[dcl]\S[inf]

S[dcl]\NP\NP\NP

2. wird *geben* *müssen*

S[dcl]/S[inf] S[inf]\NP\NP\NP S[inf]\S[inf]

S[inf]\NP\NP\NP <B

S[dcl]\NP\NP\NP >B

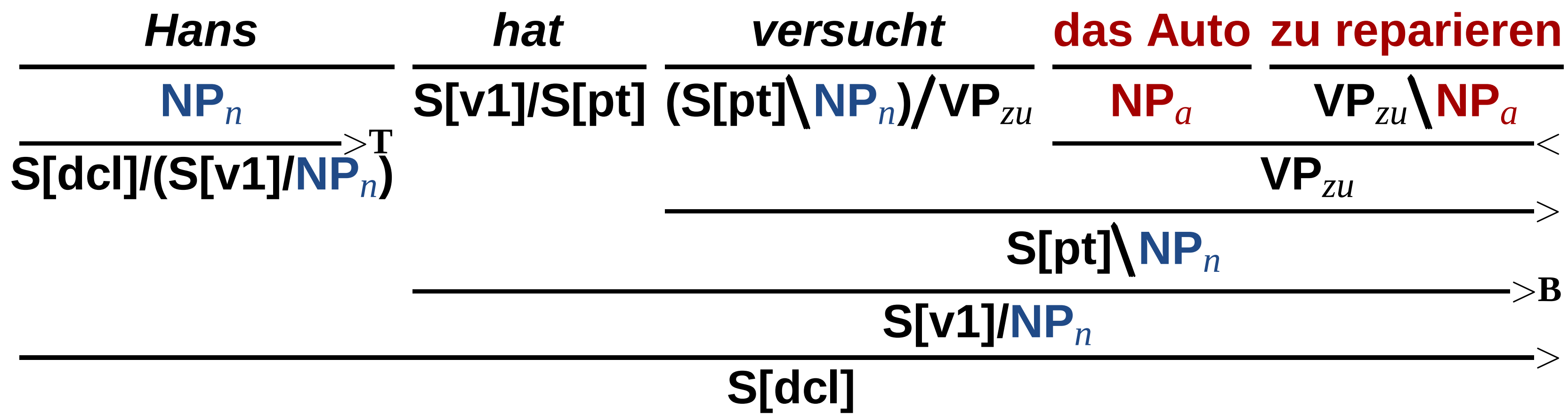
3. *geben* wird *müssen*

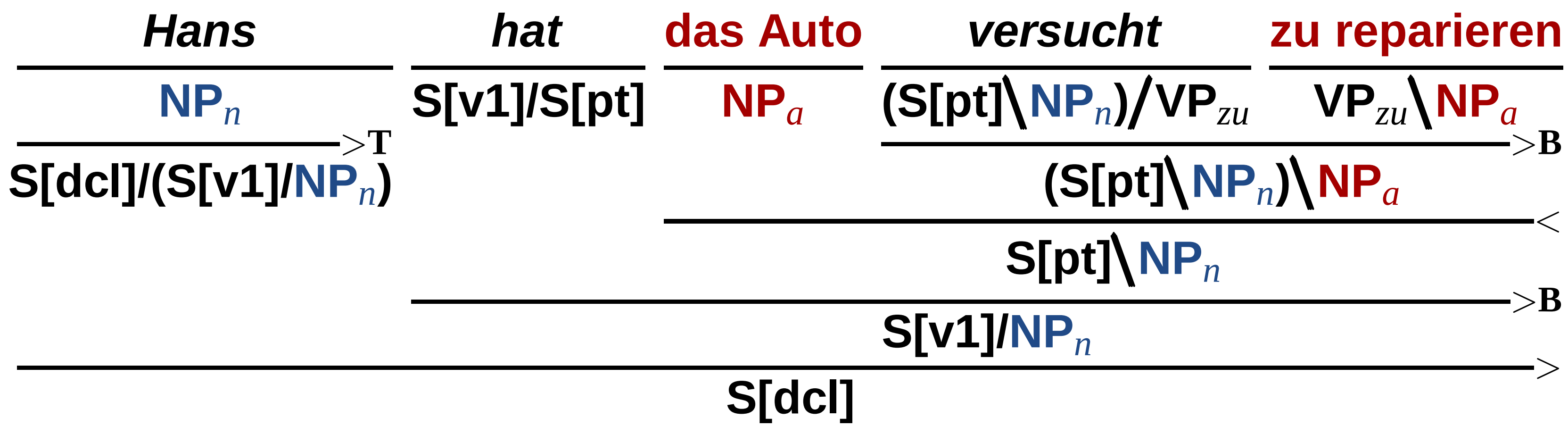
S[inf]\NP\NP\NP S[dcl]/S[inf] $S[inf]\backslash S[inf]$

S[dcl]\S[inf] >B

S[dcl]\NP\NP\NP <B

The Nachfeld



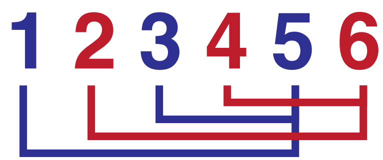


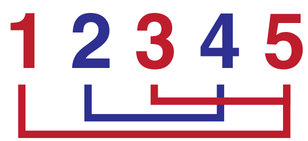
Non-local scrambling

Dieses Buch hat den Kindern niemand zu geben versucht.
this book has to-the-children nobody to give tried
Nobody has tried to give this book to the children.

Tree-Adjoining Grammar can't derive this sentence:
(Rambow 1994)

This sentence: TAG adjunction:





Non-local scrambling in CCG

dieses Buch	*hat*	*den Kindern*	*niemand*	*zu geben*	*versucht*
$S/(S[v1]/NP_a)$	$S[v1]/S[pt]$	NP_d	NP_n	$(VP_{zu}\backslash NP_a)\backslash NP_d$	$(S[pt]\backslash NP_n)\backslash VP_{zu}$
			$>T$: $S/(S\backslash NP_n)$	$<B$: $((S[pt]\backslash NP_n)\backslash NP_a)\backslash NP_d$	
			$>B$: $(S[pt]\backslash NP_a)\backslash NP_d$		
		$<$: $S[pt]\backslash NP_a$			
	$>B$: $S[v1]\backslash NP_a$				
$>$: $S[dcl]$					

The equivalence of TAG and CCG

TAG and CCG are weakly equivalent

(Weir& Joshi 1988, Weir & Vijay-Shanker 1994)

- CCGs can be converted into Linear Indexed Grammars
- TAGs can be converted into Linear Indexed Grammars

Linear Indexed Grammars (LIG)

$$
\begin{aligned}
X[\alpha] &\rightarrow \ldots Y[\alpha]\ldots \\
X[\alpha, \mathbf{c}] &\rightarrow \ldots Y[\alpha]\ldots \\
X[\alpha] &\rightarrow \ldots Y[\alpha, \mathbf{c}]\ldots
\end{aligned}
$$

- LIGs are CFGs with a stack of indices.
- The stack can be passed to one daughter.
- The topmost element can be pushed onto or popped off the stack

CCG as an Indexed Grammar

$$(S\backslash NP)/PP = S[\backslash NP, /PP]$$

Categories: $c = t[\alpha]$

consist of a *target* t and a *stack* α

Stacks: $\alpha_i \in c^i$

lists of i categories with $i \geq 0$ and $|\alpha| = i$.

Target categories: $t \in \{S, NP, PP, \ldots\}$

Combinatory rules

Application

$$\frac{X/Y \quad Y}{X} >$$

$$\frac{\mathbf{t}[\alpha, \mathbf{u}[\gamma]] \quad \mathbf{u}[\gamma]}{\mathbf{t}[\alpha]} >\mathbf{B}^n$$

Composition $(n \leq 4)$

$$\frac{X/Y \quad Y/Z_1 \ldots Z_n}{X/Z_1 \ldots Z_n} >\mathbf{B}^n$$

$$\frac{\mathbf{t}[\alpha, \mathbf{u}[\gamma]] \quad \mathbf{u}[\gamma, \beta_n]}{\mathbf{t}[\alpha\, \beta_n]} >\mathbf{B}^n$$

Type-raising $(m \leq 3)$

$$\frac{X}{T/(T\backslash X)} >\mathbf{T}$$

$$\frac{\mathbf{c}}{\mathbf{t}[\alpha_m, \mathbf{t}[\alpha_m, \mathbf{c}]]} >\mathbf{T}$$

Typeraising + composition

Standard CCG

$$\dfrac{\dfrac{X}{T/(T\backslash X)}{>}\mathbf{T} \quad (T\backslash X)\backslash Z_1 \ldots Z_n}{T\backslash Z_1 .. Z_n}{>}\mathbf{B}^n$$

As an IG

$$\dfrac{\dfrac{c}{t[\alpha_m, t[\alpha_m, c]]}{>}\mathbf{T} \quad t[\alpha_m, c\ \beta_n]}{t[\alpha_m\ \beta_n]}{>}\mathbf{B}^n$$

Type-raising + (generalized) composition allow the $n+1$th element to be popped off a stack of $n+m+1$ elements.

LIG vs CCG

- LIGs can generate

  $n_1 \ldots . n_{m+n} v_1 \ldots v_{m+n}$ and $n_{m+n} \ldots . n_1 v_1 \ldots v_{m+n}$

- CCGs can generate

  $N^1 \ldots N^{m+n} v_1 .. v_{m+n}$ and $N^{m+n} \ldots N^1 v_1 .. v_{m+n}$

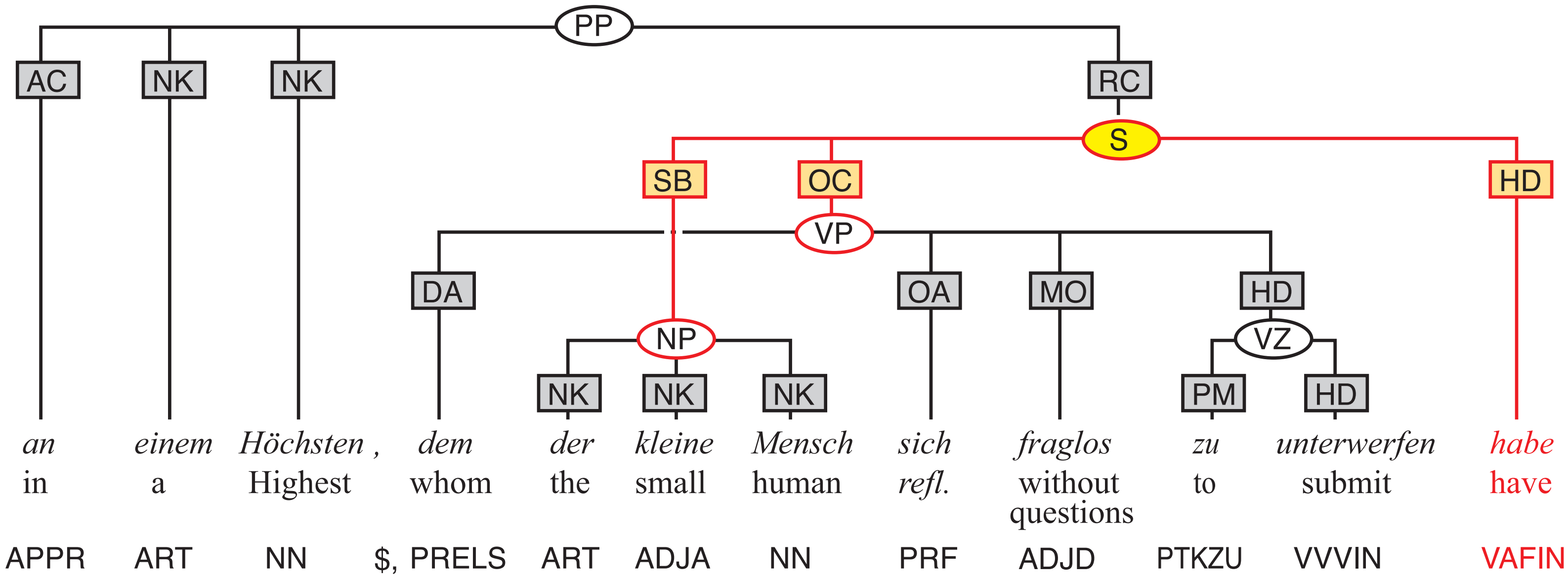
1. $N^1 \in \{n_{m+n} \ldots n_{m+1}\},$
2. $N^2 \in (\{n_{m+n} \ldots n_{m+1}\} \cup \{x_m\}) \setminus \{N^1\}$
3. $N^3 \in (\{n_{m+n} \ldots . n_{m+1}\} \cup \{x_m, x_{m-1}\}) \setminus \{N^1, N^2\}$
4. $\ldots$
5. $N^i \in (\{n_{m+n} \ldots n_{m+1}\} \cup \{x_m \ldots x_{m-1}\}) \setminus \{N^1 \ldots . N^{i-1}\}$

What does this mean?

- TAG and CCG are both *mildly context-sensitive*.
- Standard TAG can be translated into a standard LIG.
- CCG can be translated into a LIG where the nth element in a stack of size $n+m$ can be popped off.
- Conjecture 1:
  There are some scrambling cases for which there is no TAG that gives the same dependencies
- Conjecture 2:
  $n+m$ is very close to the stack size needed to parse real sentences.

What does this have to do with treebanks?

What does this have to do with treebanks?



- Translating discontinuous constituents into CCG requires thinking of categories as a stack

A personal summary

- Good treebanks contain rich linguistic annotations that make it possible to extract expressive wide-coverage grammars if the formalism is expressive enough.
- Grammar extraction requires linguistic knowledge (and stamina). Preprocessing is inevitable.
- Grammmar extraction provides a great empirical test for your linguistic theory.
- Grammar extraction makes efficient wide-coverage parsers for expressive grammars possible.

Thank you!!!

`http://www.cs.uiuc.edu/juliahmr/`
`juliahmr@cs.uiuc.edu`

Thanks to Mark Steedman, Aravind Joshi, Steven Clark, Johan Bos, James Curran, EPSRC and the NSF