ACL2010

Uppsala, Sweden
July 11–16, 2010

The 48th Annual Meeting of the Association for Computational Linguistics

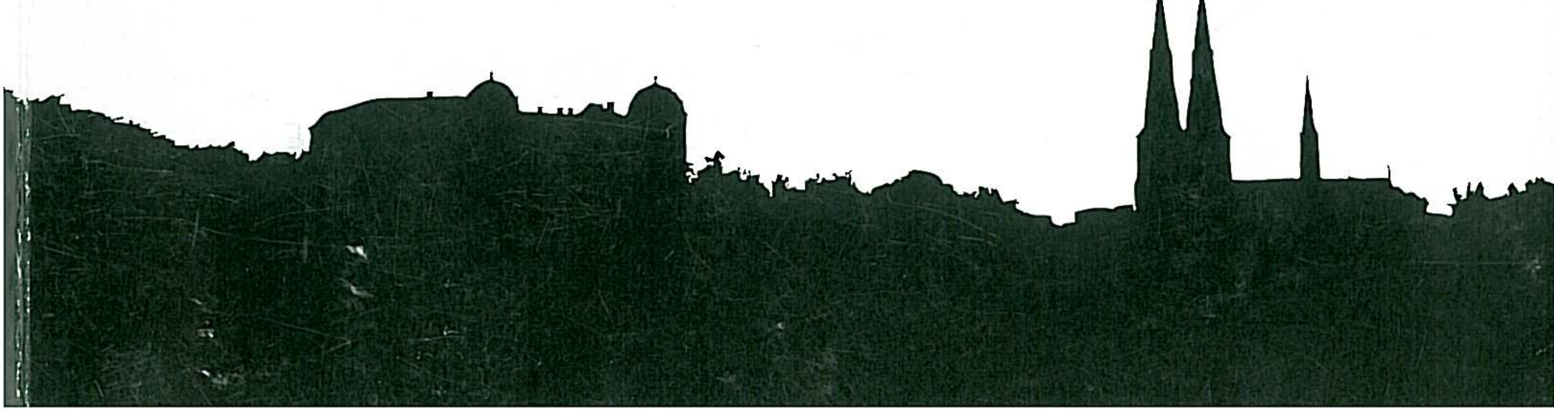
TUTORIAL NOTES T3

Wide-Coverage NLP with Linguistically Expressive Grammars

Josef van Genabith
Julia Hockenmaier
Yusuke Miyao

July 11, 2010
Uppsala, Sweden

UPPSALA
UNIVERSITET



ACL TUTORIAL: WIDE-COVERAGE NLP WITH EXPRESSIVE GRAMMARS

Josef van Genabith (Dublin City University)
Julia Hockenmaier (University of Illinois)
Yusuke Miyao (National Institute of Informatics)

Why this tutorial?

- A lot of progress in robust, wide-coverage NLP with expressive grammars in the last 10 years
  - Expressive grammars and machine learning go together
  - Expressive grammars are needed for semantics
  - Expressive grammars are being used in real applications

But:

- Research is limited to small number of groups
- Each group works within their own formalism
  - What are the commonalities and differences?
  - How can others get started?

Overview

- **Part 1: Introduction to expressive grammars**
  - Why expressive grammars?
  - Tree-Adjoining Grammar
  - Combinatory Categorial Grammar
  - Lexical-Functional Grammar
  - Head-Driven Phrase Structure Grammar
- **Part 2: NLP with expressive grammars**
  - Grammar extraction: obtaining the grammar
  - Wide-coverage parsing: using the grammar
  - Other applications: using the grammar

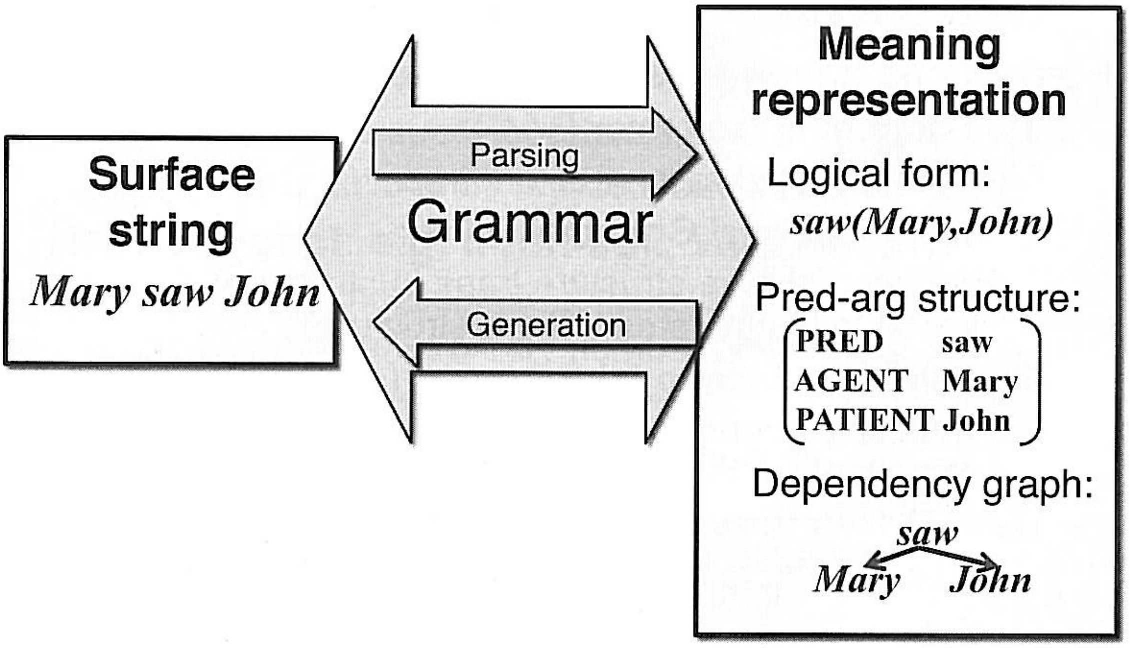
I. Why Expressive Grammars?

Overview

- **Part 1: Introduction to expressive grammars**
  – Why expressive grammars?
  – Tree-Adjoining Grammar
  – Combinatory Categorial Grammar
  – Lexical-Functional Grammar
  – Head-Driven Phrase Structure Grammar
- **Part 2: NLP with expressive grammars**
  – Grammar extraction: obtaining the grammar
  – Wide-coverage parsing: using the grammar
  – Other applications: using the grammar

I. WHY EXPRESSIVE GRAMMARS?

Why grammar?



Grammar formalisms

– Formalisms provide a **language** in which linguistic theories can be expressed and implemented

– Formalisms define **elementary objects** (trees, strings, feature structures) and **recursive operations** which generate complex objects from simple objects.

– Formalisms may impose **constraints** (e.g. on the kinds of dependencies they can capture)

How do grammar formalisms differ?

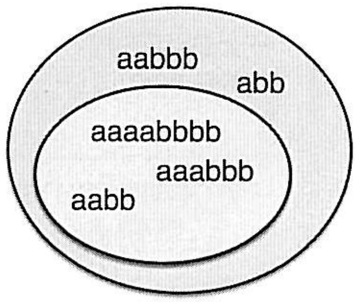
Formalisms define different **representations**

- **Tree-adjoining Grammar (TAG)**:
  Fragments of phrase-structure trees
- **Lexical-functional Grammar (LFG)**:
  Annotated phrase-structure trees (c-structure) linked to feature structures (f-structure)
- **Combinatory Categorial Grammar (CCG)**:
  Syntactic categories paired with meaning representations
- **Head-Driven Phrase Structure Grammar(HPSG):**
  Complex feature structures (Attribute-value matrices)

How do grammar formalisms differ?

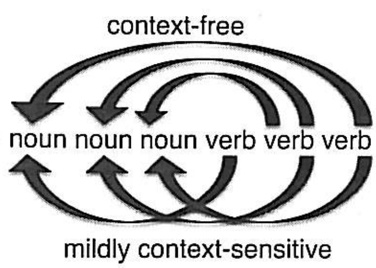
Weak generative capacity:

- What languages (sets of strings) can be defined?
  - $a^n b^m$ is regular, $a^n b^n$ is context-free
- **Expressive grammars can represent more languages**



Strong generative capacity:

- What structures can be defined?
- **Expressive grammars can represent more structures**



Different types of dependencies

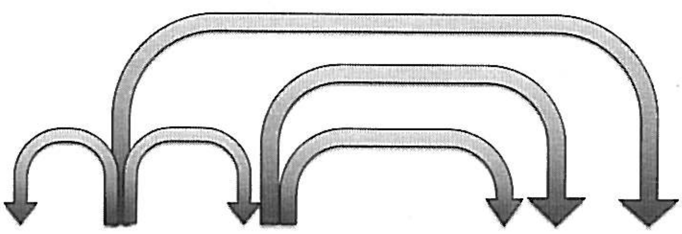
Head-Argument: e.g. verb-subject
- Arguments are subcategorized for
- Arguments have to be realized, but only once

Head-Adjunct: e.g. noun-adj., verb-adverb
- Adjuncts are not subcategorized for
- There can be an arbitrary number of adjuncts

Coordination:
- Conjuncts may be standard constituents
  John and Mary; live or die
- Conjuncts may be nonstandard constituents
  ((John will) and (Mary may want)) to go

Context-free grammars

- CFGs capture only **nested** dependencies
  - The dependency graph is a **tree**
  - The dependencies **do not cross**

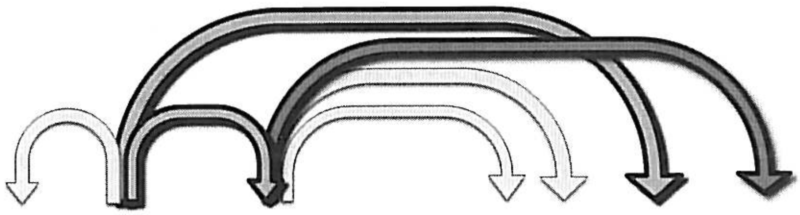


Beyond CFGs: Nonprojective dependencies

Dependencies: **tree with crossing branches**

Arise in the following constructions

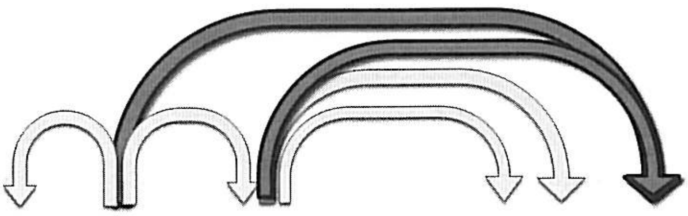
- (Non-local) **scrambling** (free word order languages)
  Die Pizza hat Klaus versprochen zu bringen
- **Extraposition** (*The guy is coming who is wearing a hat*)
- **Topicalization** *(Cheeseburgers, I thought he likes)*



11

Beyond CFGs: Nonlocal dependencies

- Dependencies form a **DAG**
  (a node may have **multiple incoming edges**)
- Arise in the following constructions:
  - **Control** *(He has promised me to go)*, **raising** (He seems to go)
  - **Wh-movement** *(the man who you saw yesterday is here again)*,
  - **Non-constituent** coordination
    (right-node raising, gapping, argument-cluster coordination)



12

Unbounded non-local dependencies

Extraction:

- *Wh*-movement:
  *the **articles** which (you believed he saw that...) I **filed***
- *Tough*-movement:
  *the **articles** are easy to **file***
- Parasitic gaps:
  *the **articles** that I **filed** without **reading***

Non-standard coordination:

- Right-node raising:
  *[[**Mary ordered**] and [**John ate**]] the tapas.*
- Argument cluster coordination:
  *Mary ordered [[**tapas for herself**] and [**wine for John**]].*
- Sentential gapping:
  *[[**Mary** ordered **tapas**] and [**John beer**]].*

Commonalities and differences: Lexicalization

No lexicalization: (CFG)

- The lexicon contains little syntactic information (e.g. just POS-tags)
- Recursion is entirely defined by *language-specific* grammar rules

Weak lexicalization: (LFG)

- The lexicon (and lexical rules) specify *some* language-specific information
  (e.g. subcategorization, semantics, control, binding theory, passivization)
- Recursion is defined by language-specific grammar rules
  (but lexical information may constrain which rules can be used in which context)

Strong lexicalization: (TAG, CCG, HPSG)

- The lexicon (and lexical rules) specifies *all* language-specific information
  (e.g. word order, subcategorization, semantics, control, binding theory)
- The lexicon pairs words with complex elementary objects
  These objects may have an *extended domain of locality*
  (i.e. capture structure beyond a single CFG rule)
- Recursion is defined by completely universal operations

II. Tree-Adjoining Grammar

Tree-Adjoining Grammar

TAG is a **tree-rewriting formalism**:

- TAG's elementary objects are trees (not strings)
- TAG's operations (substitution, adjunction) work on trees.
- TAG requires a linguistic theory which specifies the shape of these elementary trees.

TAG is **mildly context-sensitive**:

- can capture Dutch crossing dependencies
- but is still efficiently parseable

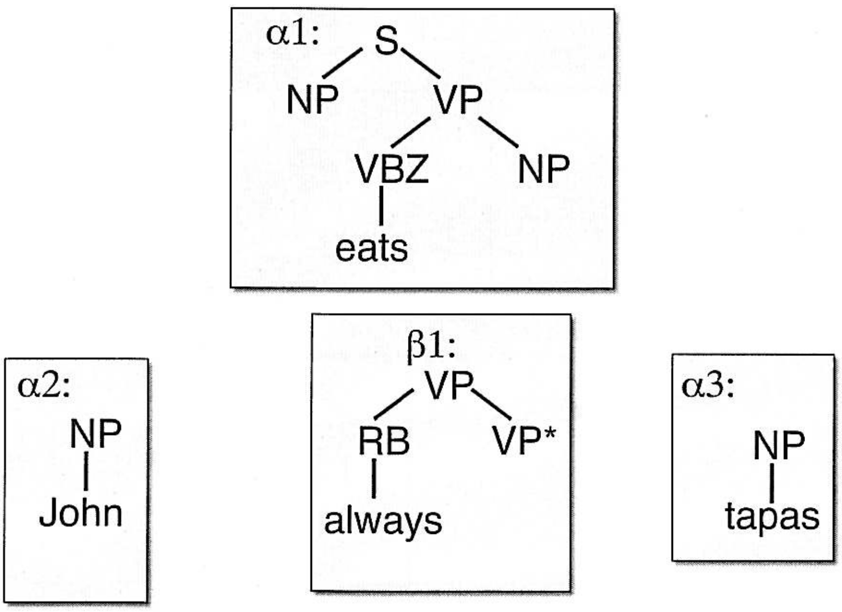
TAG: the machinery

Elementary trees:
- **Initial trees**: combine via substitution
- **Auxiliary trees**: combine via adjunction

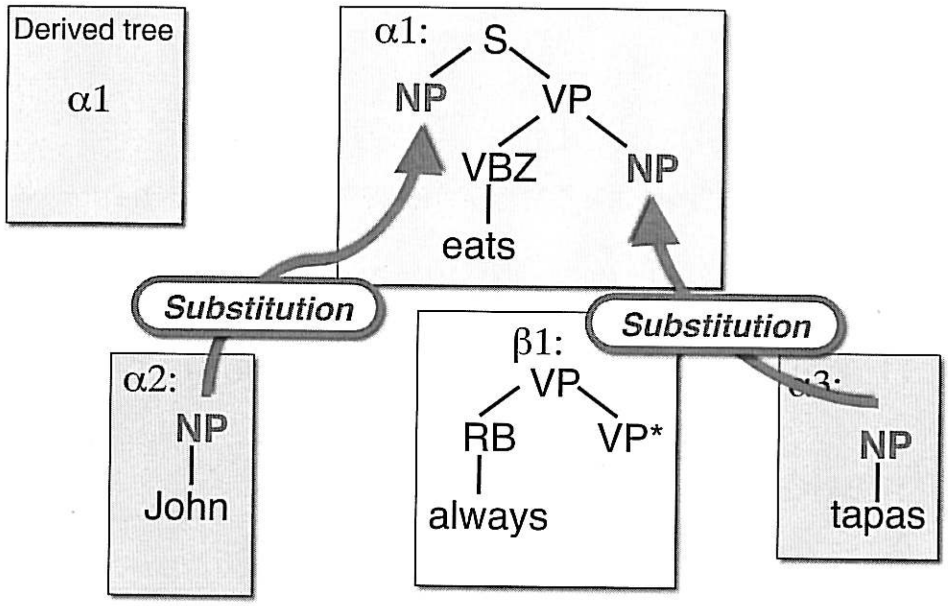
Derived trees:
- The output of substitution and adjunction

Derivation trees:
- A record of the derivation process

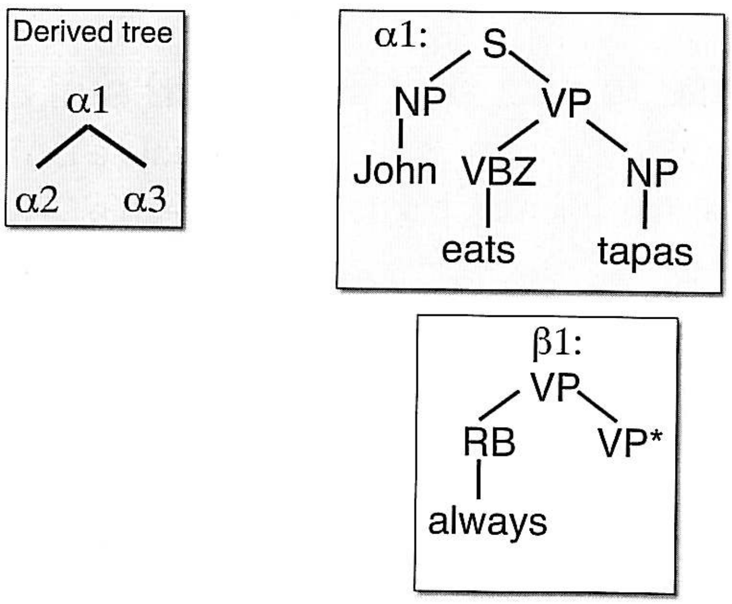
A small TAG lexicon



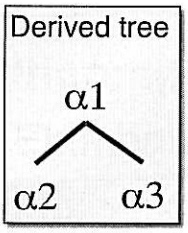
A TAG derivation: arguments

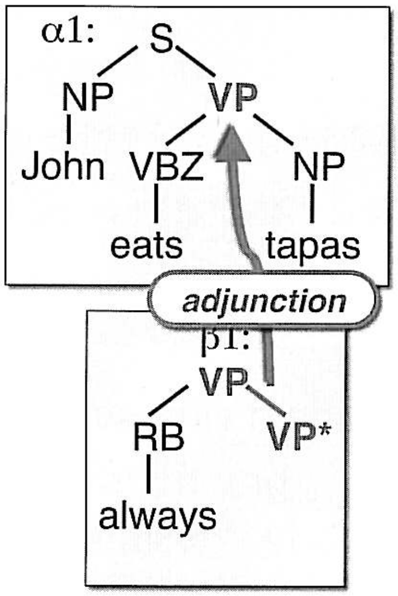


A TAG derivation: arguments

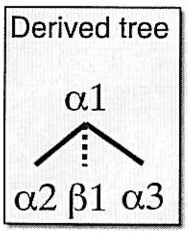


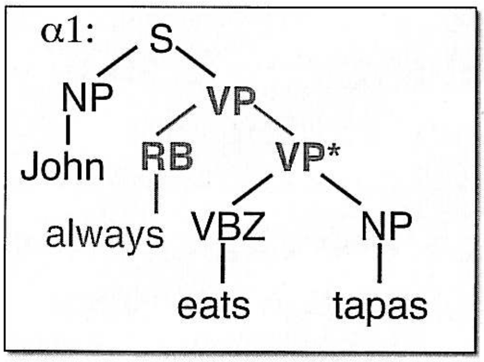
A TAG derivation: adjuncts





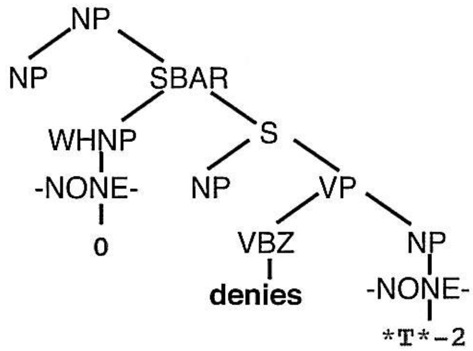
A TAG derivation: adjuncts



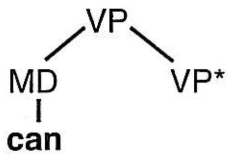


Nonlocal dependencies in TAG

Use different elementary trees

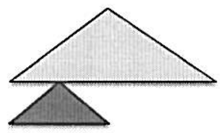


Use obligatory adjunction

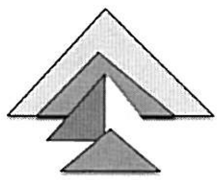


TSG, TIG, and TAG

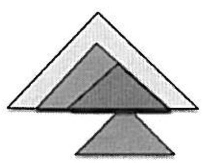
- **Tree Substitution Grammar:**
  - only **substitution**



- **Tree Insertion Grammar:**
  - only substitution and **sister adjunction**



- **Tree Adjoning Grammar:**
  - substitution, sister adjunction and **wrapping adjunction**



Extensions and variants of TAG

Multicomponent TAG

- Elementary trees can be **sets** of trees
- More expressive than standard TAG

Spinal TAG

- Elementary trees have only a spine
- Leaves subcategorization and argument/adjunct distinction underspecified

III. Combinatory Categorial Grammar

Properties of CCG

- CCG rules are **type-driven**, not structure-driven
  - Types = functions
  - Transitive verbs and VPs are indistinguishable
- CCG's **syntax-semantics interface** is transparent
  - Lexicon pairs syntactic categories with interpretations
  - Every syntactic rule has a semantic counterpart
  - CCG rules are monotonic (no movement/traces)
- CCG has a **flexible constituent structure**
  - Simple, unified treatment of extraction and coordination
- CCG is **mildly context-sensitive**

CCG: the machinery

Syntactic categories:

specify subcategorization; define word order

Semantic interpretations:

specify logical forms (pred.-arg. structure)

Combinatory rules:

specify how constituents can combine.

Derivations:

spell out process of combining constituents.

CCG categories

Simple (atomic) categories: NP, S, PP

Complex categories (functions):

Return a **result** when combined with an **argument**

VP, intransitive verb	**S\NP**
Transitive verb	**(S\NP)/NP**
Adverb	**(S\NP)\(S\NP)**
Prepositions	**((S\NP)\(S\NP))/NP** **(NP\NP)/NP** **PP/NP**

Function application

Forward application (>):

(S\NP)/NP	**NP**	$\Rightarrow_>$	**S\NP**
eats	tapas		eats tapas

Backward application (<):

NP	**S\NP**	$\Rightarrow_<$	**S**
John	eats tapas		John eats tapas

Used in all variants of categorial grammar

A (C)CG derivation

$$
\begin{array}{ccc}
\text{John} & \text{eats} & \text{tapas} \\
\overline{\mathsf{NP}} & \overline{(\mathsf{S}\backslash\mathsf{NP})/\mathsf{NP}} & \overline{\mathsf{NP}} \\
 & \multicolumn{2}{c}{\underline{\hphantom{(\mathsf{S}\backslash\mathsf{NP})/\mathsf{NP}\quad\mathsf{NP}}}\,>} \\
 & \multicolumn{2}{c}{\mathsf{S}\backslash\mathsf{NP}} \\
\multicolumn{3}{c}{\underline{\hphantom{\mathsf{NP}\quad(\mathsf{S}\backslash\mathsf{NP})/\mathsf{NP}\quad\mathsf{NP}}}\,<} \\
\multicolumn{3}{c}{\mathsf{S}}
\end{array}
$$

CCG: semantics

- Every syntactic category and rule has a semantic interpretation
- Semantic interpretations are functions of the **same arity** as the syntactic category
- Semantics often written as **λ-expressions**

$$
\begin{array}{ccc}
\text{John} & \text{eats} & \text{tapas} \\
\overline{\mathsf{NP} : John'} & \overline{(\mathsf{S}\backslash\mathsf{NP})/\mathsf{NP} : \lambda x.\lambda y.eats'xy} & \overline{\mathsf{NP} : tapas'} \\
 & \multicolumn{2}{c}{\underline{\hphantom{(\mathsf{S}\backslash\mathsf{NP})/\mathsf{NP} : \lambda x.\lambda y.eats'xy\quad\mathsf{NP} : tapas'}}\,>} \\
 & \multicolumn{2}{c}{\mathsf{S}\backslash\mathsf{NP} : \lambda y.eats'tapas'y} \\
\multicolumn{3}{c}{\underline{\hphantom{\mathsf{NP} : John'\quad(\mathsf{S}\backslash\mathsf{NP})/\mathsf{NP} : \lambda x.\lambda y.eats'xy\quad\mathsf{NP} : tapas'}}\,<} \\
\multicolumn{3}{c}{\mathsf{S} : eats'tapas'John'}
\end{array}
$$

Function composition

Harmonic forward composition (>B):

$X / Y \quad Y / Z \quad \Rightarrow_{>B} \quad X / Z$

$\lambda x.f(x) \quad \lambda y.g(y) \quad \lambda z.f(g(z))$

Harmonic backward composition (<B):

$Y \backslash Z \quad X \backslash Y \quad \Rightarrow_{<B} \quad X \backslash Y$

$\lambda y.g(y) \quad \lambda x.f(x) \quad \lambda z.f(g(z))$

Forward crossing composition (>B^x):

$X / Y \quad Y \backslash Z \quad \Rightarrow_{>Bx} \quad X \backslash Z$

$\lambda x.f(x) \quad \lambda y.g(y) \quad \lambda z.f(g(z))$

Backward crossing composition (<B^x):

$Y / Z \quad X \backslash Y \quad \Rightarrow_{>Bx} \quad X / Y$

$\lambda y.g(y) \quad \lambda x.f(x) \quad \lambda z.f(g(z))$

Type-raising

Forward typeraising (>T):

$X \quad \Rightarrow_{>T} \quad T / (T \backslash X)$

$a \quad \lambda f.f(a)$

Backward typeraising (<T):

$X \quad \Rightarrow_{<T} \quad T \backslash (T / X)$

$a \quad \lambda f.f(a)$

The CCG lexicon

Pairs words with their syntactic categories (and semantic interpretation):

eats	**(S\NP)/NP**	$\lambda x \lambda y.eats'xy$
	S\NP	$\lambda x.eats'x$

The main bottleneck for wide-coverage CCG parsing

The CCG lexicon: bounded dependencies

Bounded dependencies are captured in the lexicon through **coindexation** in the **syntactic** category and **copied variables** in the **semantic** interpretation

Auxiliaries

may: $(S\backslash NP_i)/(S\backslash NP_i)$: $\lambda P \lambda x.may'(x,P(x))$

Subject control

promise: $((S\backslash NP_i)/(S\backslash NP_i))/NP$: $\lambda y \lambda P \lambda x.promise'(x,y,P(x))$

Object control

persuade: $((S\backslash NP)/(S\backslash NP_i))/NP_i$: $\lambda y \lambda P \lambda x.promise'(x,y,P(x))$

Another CCG derivation

$$
\begin{array}{lll}
\text{John} & \text{eats} & \text{tapas} \\
\hline
NP & (S\backslash NP)/NP & NP \\
\overline{S/(S\backslash NP)}^{>T} & & \\
\multicolumn{2}{c}{\overline{\qquad S/NP \qquad}^{>B}} & \\
\multicolumn{3}{c}{\overline{\qquad\qquad S \qquad\qquad}^{>}}
\end{array}
$$

- Function composition and type-raising create "spurious ambiguity".
- **Normal form derivations** use composition and type-raising when only necessary

Non-local dependencices: Type-raising and composition

$$
\begin{array}{llll}
\textit{coffee} & \textit{that} & \textit{I} & \textit{drink} \\
\hline
NP & (NP\backslash NP)/(S[dcl]/NP) & S/(S\backslash NP) & (S[dcl]\backslash NP)/NP \\
 & & \multicolumn{2}{c}{\overline{\qquad S[dcl]/NP \qquad}^{>B}} \\
 & \multicolumn{3}{c}{\overline{\qquad\qquad NP\backslash NP \qquad\qquad}^{>}} \\
\multicolumn{4}{c}{\overline{\qquad\qquad\qquad NP \qquad\qquad\qquad}^{<}}
\end{array}
$$

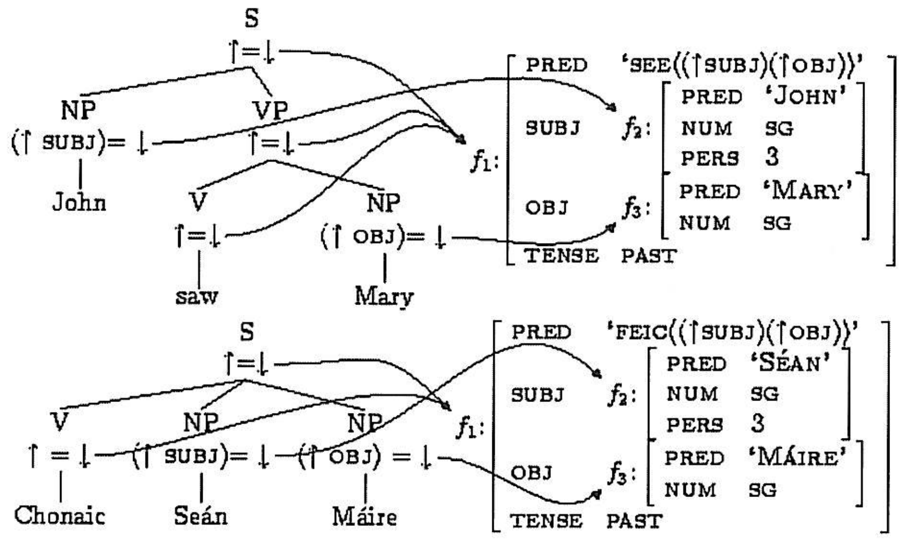
$$
\begin{array}{llllll}
\textit{I} & \textit{like} & \textit{but} & \textit{you} & \textit{hate} & \textit{coffee} \\
\hline
S/(S\backslash NP) & (S[dcl]\backslash NP)/NP & conj & S/(S\backslash NP) & (S[dcl]\backslash NP)/NP & NP \\
\multicolumn{2}{c}{\overline{\qquad S[dcl]/NP \qquad}^{>B}} & & \multicolumn{2}{c}{\overline{\qquad S[dcl]/NP \qquad}^{>B}} & \\
\multicolumn{5}{c}{\overline{\qquad\qquad S[dcl]/NP \qquad\qquad}^{<\Phi>}} & \\
\multicolumn{6}{c}{\overline{\qquad\qquad\qquad S[dcl] \qquad\qquad\qquad}^{>}}
\end{array}
$$

IV. Lexical Functional Grammar

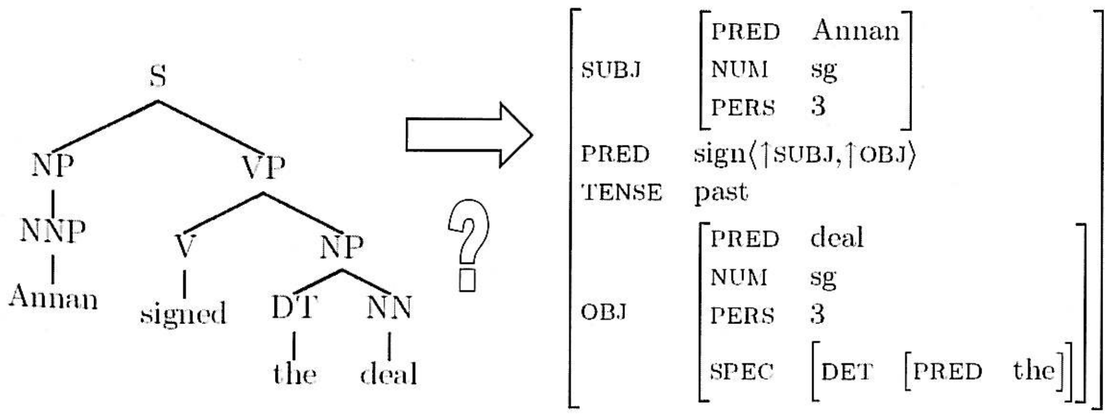
Lexical-Functional Grammar

- LFG is **constraint-based** (Bresnan & Kaplan '81, Bresnan '01, Dalrymple '01)
- Two (basic) levels of representation:
  - **C-structure**:
    - Represents **surface grammatical configurations**: constituency, word order
    - Represented as annotated CFG trees
  - **F-structure:**
    - Represents **abstract syntactic functions, morphological + semantic information**
      - SUBJ(ject), OBJ(ect), OBL(ique), PRED(icate), COMP(lement), ADJ(unct)...
      - TENSE, ASPECT, NUM(ber), PERS(on), ...
    - F-structure ≈ **basic predicate-argument structure**, dependency representation, logical form, ... (van Genabith and Crouch, '96;'97)
    - Represented as attribute-value matrices (AVMs; DAGs)

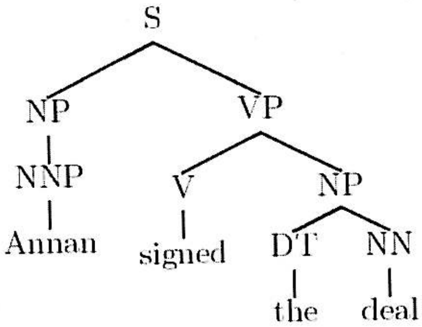
Lexical-Functional Grammar LFG



Lexical-Functional Grammar LFG



Lexical-Functional Grammar LFG



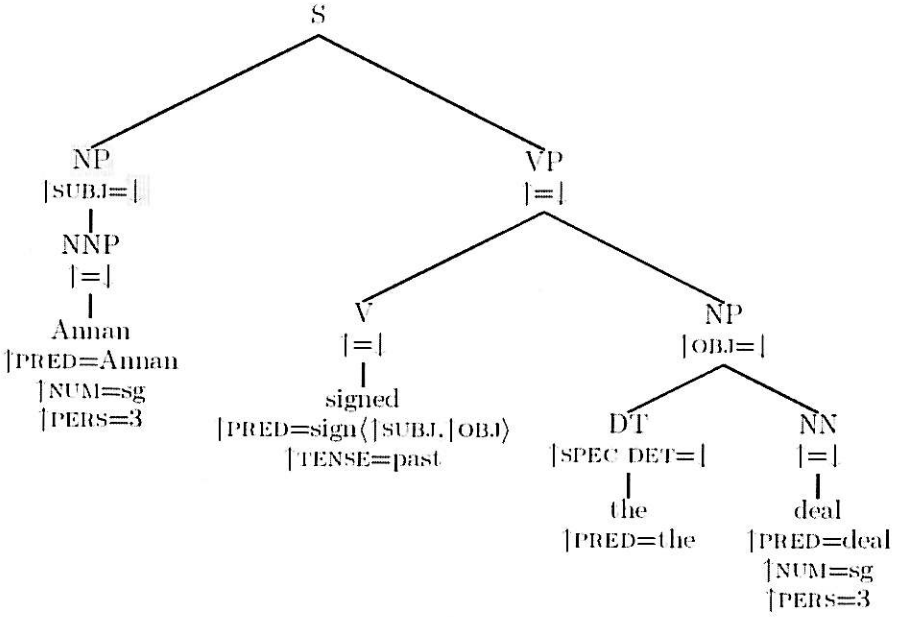
S	$\rightarrow$	NP	VP
VP	$\rightarrow$	V	NP
NP	$\rightarrow$	DT	NN
NP	$\rightarrow$	NNP	
NNP	$\rightarrow$	Annan	
NN	$\rightarrow$	deal	
V	$\rightarrow$	signed	
DT	$\rightarrow$	the	

LFG Grammar Rules and Lexical Entries

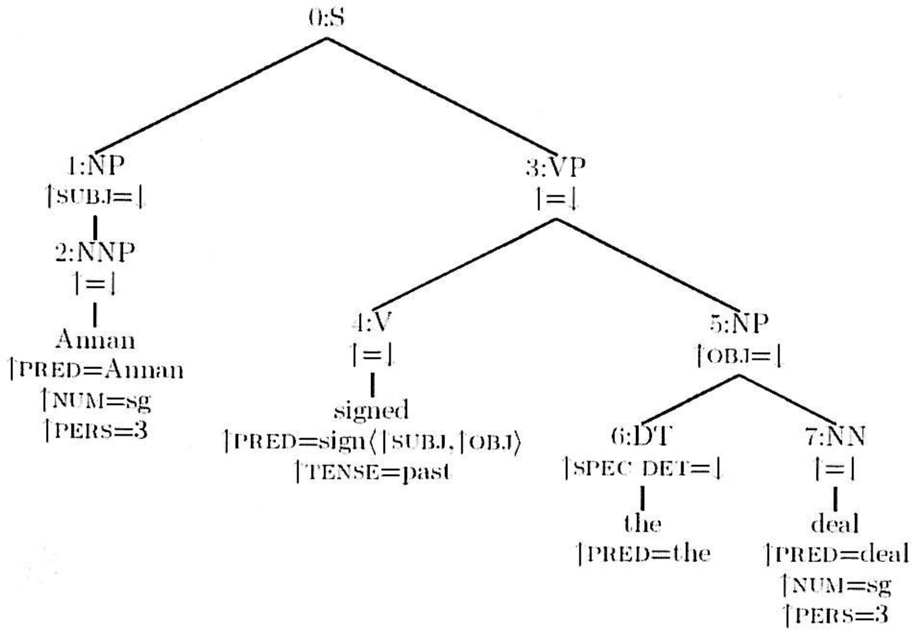
S	$\rightarrow$	NP $\uparrow$SUBJ=$\downarrow$	VP $\uparrow=\downarrow$
VP	$\rightarrow$	V $\uparrow=\downarrow$	NP $\uparrow$OBJ=$\downarrow$
NP	$\rightarrow$	DT $\uparrow$SPEC DET=$\downarrow$	NN $\uparrow=\downarrow$
NP	$\rightarrow$	NNP $\uparrow=\downarrow$	

NNP	$\rightarrow$	Annan $\uparrow$PRED=Annan $\uparrow$NUM=sg $\uparrow$PERS=3
NN	$\rightarrow$	deal $\uparrow$PRED=deal $\uparrow$NUM=sg $\uparrow$PERS=3
V	$\rightarrow$	signed $\uparrow$PRED=sign$\langle\uparrow$SUBJ,$\uparrow$OBJ$\rangle$ $\uparrow$TENSE=past
DT	$\rightarrow$	the

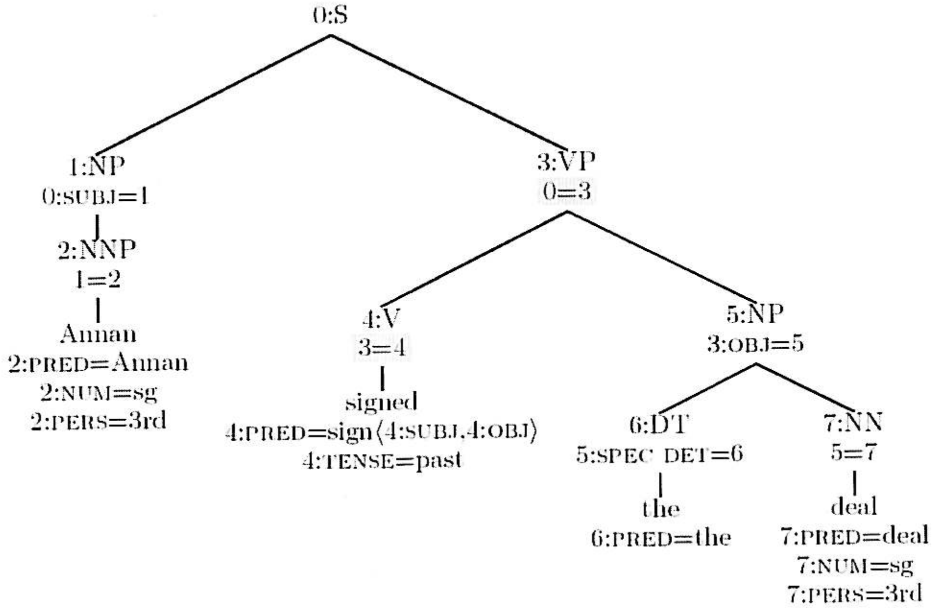
LFG Parse Tree (with Equations/ Constraints)



LFG Constraint Resolution (1/3)



LFG Constraint Resolution (2/3)



47

LFG Constraint Resolution (3/3)

$$\text{F-Str} \models \bigwedge \left\{ \begin{array}{c} 2{:}\text{PRED}{=}\text{Annan},\ 2{:}\text{NUM}{=}\text{sg},\ 2{:}\text{PERS}{=}\text{3rd},\ 1{=}2,\ 0{:}\text{SUBJ}{=}1, \\ 4{:}\text{PRED}{=}\text{sign}\langle 4{:}\text{SUBJ},4{:}\text{OBJ}\rangle,\ 4{:}\text{TENSE}{=}\text{past},\ 3{=}4, \\ 6{:}\text{PRED}{=}\text{the},\ 5{:}\text{SPEC DET}{=}6,\ 7{:}\text{PRED}{=}\text{deal},\ 7{:}\text{NUM}{=}\text{sg},\ 7{:}\text{PERS}{=}\text{3rd}, \\ 5{=}7,\ 3{:}\text{OBJ}{=}5,\ 0{=}3 \end{array} \right\}$$

$$\{\ 0{=}3,\ 5{=}7,\ 1{=}2,\ 3{=}4\ \} \Rightarrow \{3,4\} \rightarrow 0,\ 7 \rightarrow 5 \text{ and } 2 \rightarrow 1$$

$$\text{F-Str} \models \bigwedge \left\{ \begin{array}{l} 0{:}\text{SUBJ}{=}1,\ 1{:}\text{PRED}{=}\text{Annan},\ 1{:}\text{NUM}{=}\text{sg},\ 1{:}\text{PERS}{=}\text{3rd}, \\ 0{:}\text{PRED}{=}\text{sign}\langle 0{:}\text{SUBJ},0{:}\text{OBJ}\rangle,\ 0{:}\text{TENSE}{=}\text{past} \\ 0{:}\text{OBJ}{=}5,\ 5{:}\text{PRED}{=}\text{deal},\ 5{:}\text{NUM}{=}\text{sg},\ 5{:}\text{PERS}{=}\text{3rd}, \\ 5{:}\text{SPEC DET}{=}6,\ 6{:}\text{PRED}{=}\text{the} \end{array} \right\}$$

$$\boxed{0} \left[\begin{array}{ll} \text{SUBJ}\ \boxed{1} & \left[\begin{array}{ll} \text{PRED} & \text{Annan} \\ \text{NUM} & \text{sg} \\ \text{PERS} & 3 \end{array} \right] \\ \text{PRED} & \text{sign}\langle \uparrow\text{SUBJ}, \uparrow\text{OBJ}\rangle \\ \text{TENSE} & \text{past} \\ \text{OBJ}\ \boxed{5} & \left[\begin{array}{ll} \text{PRED} & \text{deal} \\ \text{NUM} & \text{sg} \\ \text{PERS} & 3 \\ \text{SPEC} & \left[\text{DET} \left[\text{PRED}\ \ \text{the} \right] \right] \end{array} \right] \end{array} \right]$$

$\boxed{\wedge,\ \neg,\ \vee}$

$\boxed{\downarrow \in \uparrow\text{GF},\ /}$

48

LFG Subcategorisation & Long Distance Dependencies

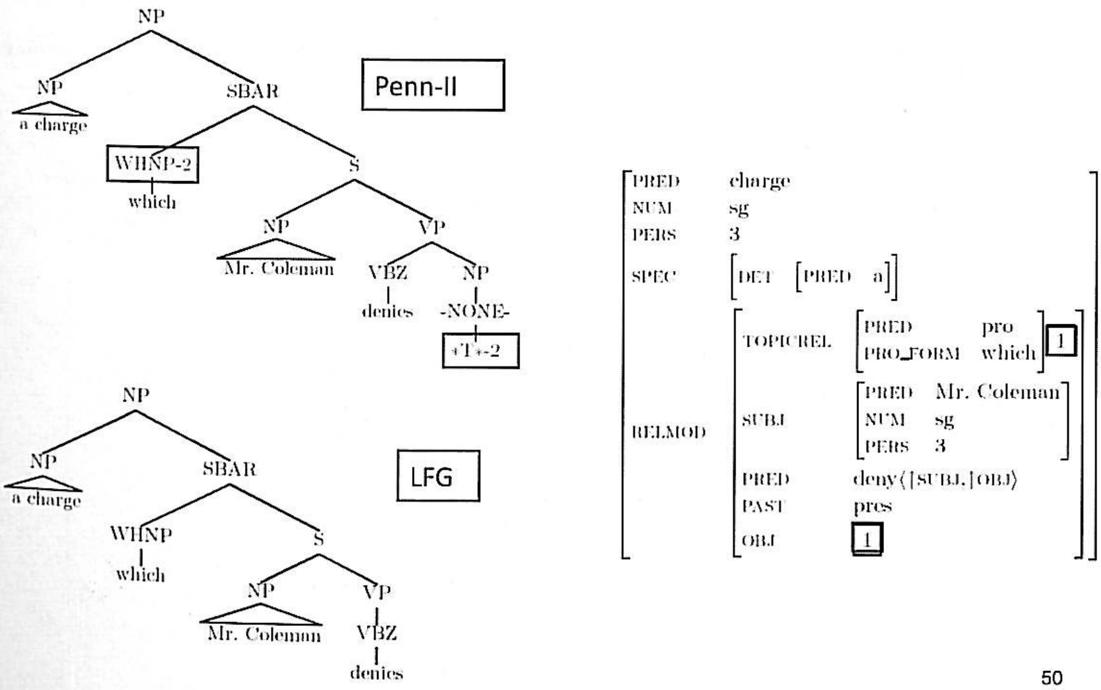
Subcategorisation:

- **Semantic forms** (subcat frames): sign<SUBJ, OBJ>
- **Completeness:**
  all GFs in semantic form present at local f-structure
- **Coherence:**
  only GFs in semantic form present at local f-structure

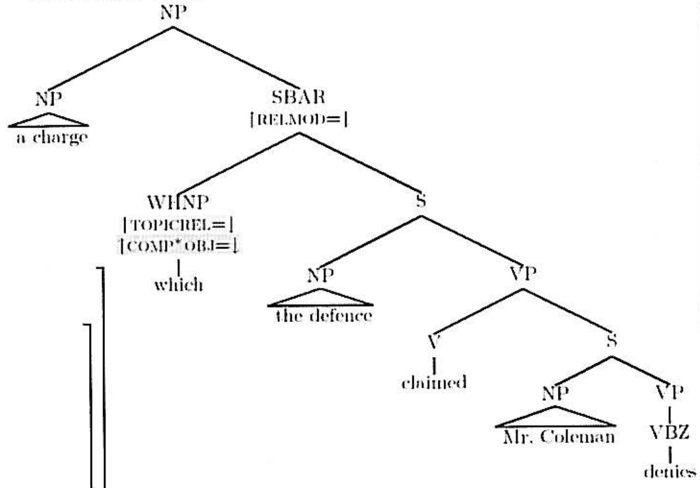
Long Distance Dependencies (LDDs):

- Resolved at f-structure with **Functional Uncertainty Equations** (regular expressions specifying paths in f-structure).

LFG LDDs: Complement Relative Clause



LFG LDDs: Complement Relative Clause



$$\begin{bmatrix} \text{PRED} & \text{charge} \\ \ldots & \ldots \\ \text{RELMOD} & \begin{bmatrix} \text{TOPICREL} & \begin{bmatrix}\text{PRED} & \text{pro}\\ \text{PRO\_FORM} & \text{which}\end{bmatrix}\boxed{1} \\ \text{SUBJ} & \begin{bmatrix}\text{PRED} & \text{defence}\\ \ldots & \ldots\end{bmatrix} \\ \text{PRED} & \text{claim}\langle\uparrow\text{SUBJ},\uparrow\text{COMP}\rangle \\ \text{PAST} & \text{past} \\ \text{COMP} & \begin{bmatrix} \text{SUBJ} & \begin{bmatrix}\text{PRED} & \text{Mr. Coleman}\\ \ldots & \ldots\end{bmatrix} \\ \text{PRED} & \text{deny}\langle\uparrow\text{SUBJ},\uparrow\text{OBJ}\rangle \\ \text{PAST} & \text{pres} \\ \text{OBJ} & \boxed{1}\end{bmatrix}\end{bmatrix}\end{bmatrix}$$

Functional Uncertainty Equations
$\uparrow\text{COMP}^{*}\text{OBJ}=\downarrow$

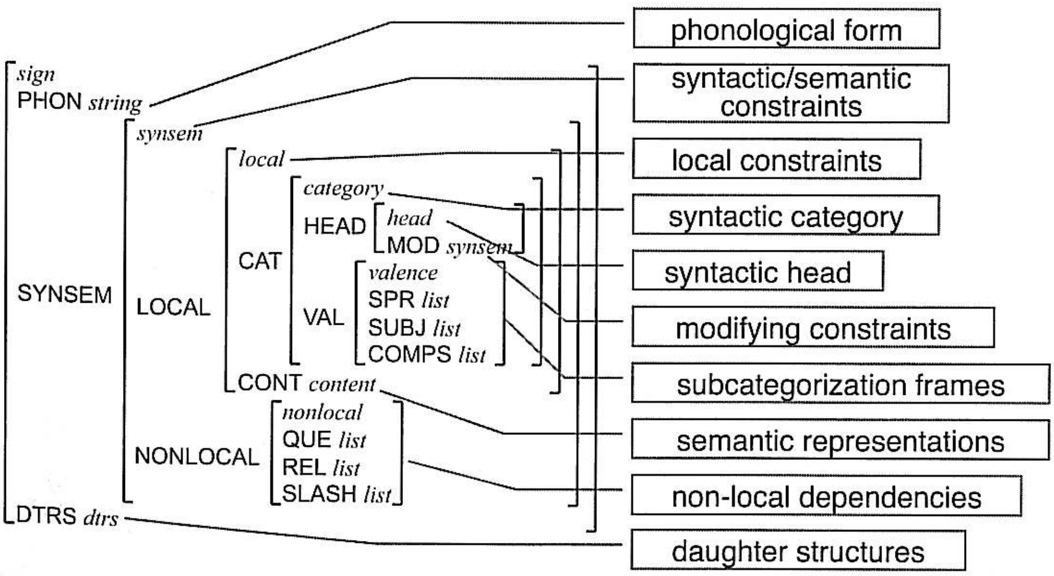
V. Head-Driven Phrase Structure Grammar

Head-Driven Phrase Structure Grammar

- HPSG (Pollard and Sag 1994, Sag et al. 2003) is a **unification-/constraint-based** theory of grammar
- HPSG is a **lexicalized** grammar formalism
- HPSG aims to explain generic regularities that underlie phrase structures, lexicons, and semantics, as well as language-specific/-independent constraints
- Syntactic/semantic constraints are uniformly denoted by **signs**, which are represented with feature structures
- Two components of HPSG
  - **Lexical entries** represent word-specific constraints (corresponding to elementary objects)
  - **Principles** express generic grammatical regularities (corresponding to grammatical operations)

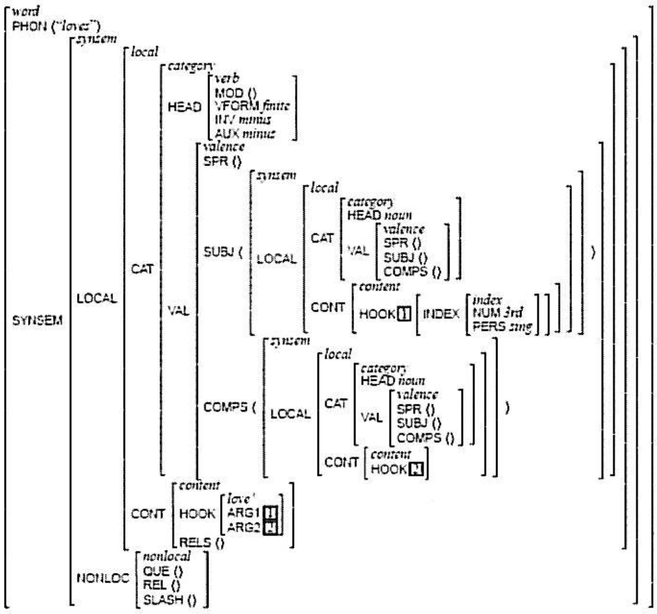
Sign

- *Sign* is a formal representation of combinations of phonological forms, syntactic and semantic constraints



Lexical entries

Lexical entries express word-specific constraints



We use simplified notations in this tutorial

PHON *"loves"*
HEAD *verb*
SUBJ <NP$\boxed{1}$ >
COMPS <NP$\boxed{2}$ >
CONT *love*($\boxed{1}$,$\boxed{2}$)

HEAD *noun*
SUBJ < >
COMPS < >
CONT $\boxed{1}$

Lexical entries

Lexical entries represent **word-specific constraints**

→ Difference in lexical entry
= difference in grammatical characteristics

Subcategorization frames
e.g. sentential complement

PHON *"think"*
HEAD *verb*
SUBJ <NP$\boxed{1}$ >
COMPS <S $\boxed{2}$>
CONT *think*($\boxed{1}$, $\boxed{2}$)

Syntactic alternation
e.g. passive

PHON *"loved"*
HEAD *verb*
SUBJ <NP $\boxed{1}$ >
COMPS <PP$_{by}$ $\boxed{2}$ >
CONT *love*($\boxed{2}$, $\boxed{1}$)

Principles

Principles describe **generic regularities** of grammar

Do not correspond to construction rules

- **Head Feature Principle**

  The value of HEAD must be percolated from the head daughter

  [HEAD $\boxed{1}$] → ····· [HEAD $\boxed{1}$] ·····

  head daughter

- **Valence Principle**

  Subcats not consumed are percolated to the mother

- **Immediate Dominance (ID) Principle**

  A mother and her immediate daughters must satisfy one of *immediate dominance schemas*

Many other principles: percolation of NONLOCAL features, semantics construction, etc.

Schemas

Schemas correspond to **construction rules** in CFGs and other grammar formalisms

– For **subject-head** constructions (ex. *"John runs"*)

[SUBJ <>] → $\boxed{1}$ [SUBJ < $\boxed{1}$ >]

– For **head-complement** constructions (ex. *"loves Mary"*)

[COMPS $\boxed{2}$] → [COMPS < $\boxed{1}$ | $\boxed{2}$ >] $\boxed{1}$

– For **filler-head** constructions (ex. *"what he bought"*)

[SLASH $\boxed{2}$] → $\boxed{1}$ [SLASH < $\boxed{1}$ | $\boxed{2}$ >]

Example: HPSG parsing

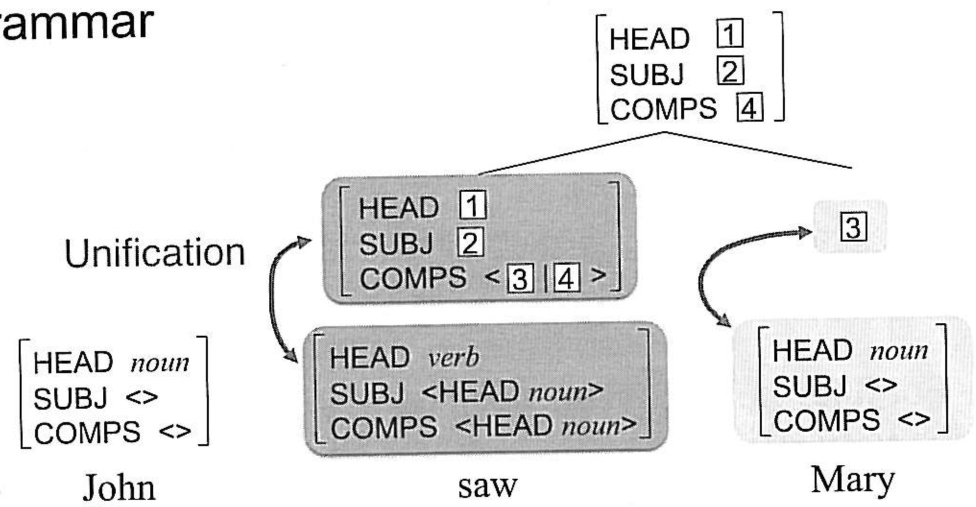
- Lexical entries determine syntactic/semantic constraints of words

Lexical entries

John	saw	Mary
HEAD *noun* SUBJ <> COMPS <>	HEAD *verb* SUBJ <HEAD *noun*> COMPS <HEAD *noun*>	HEAD *noun* SUBJ <> COMPS <>

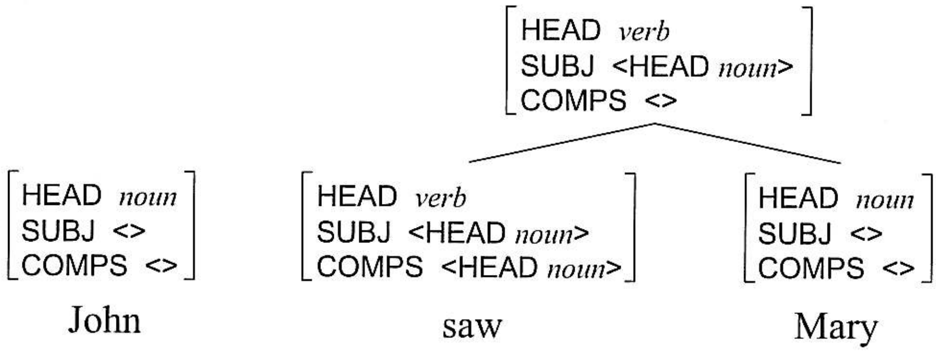
Example: HPSG parsing

Principles determine generic constraints of grammar



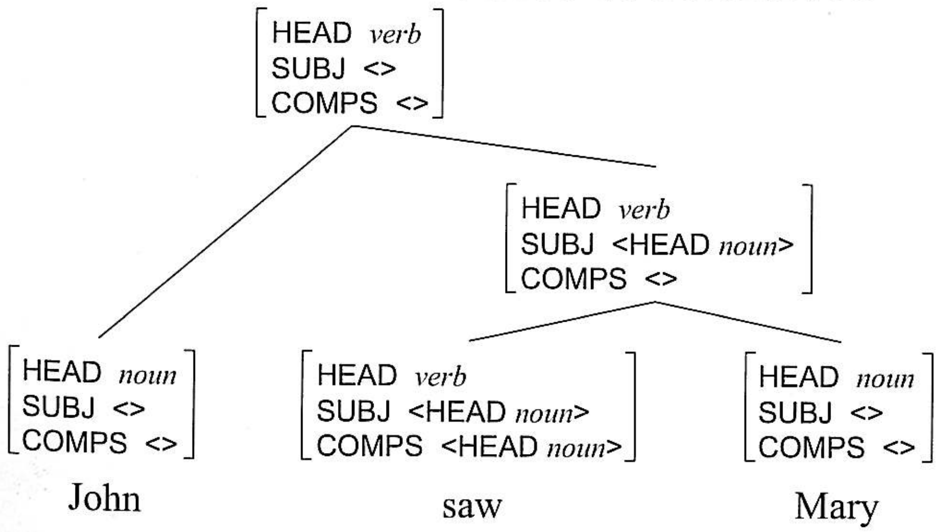
Example: HPSG parsing

Principle application produces phrasal signs



Example: HPSG parsing

Recursive applications of principles produce syntactic/semantic structures of sentences



Example: Control verbs

I ***persuaded*** *him to quit the trip.*
→ He quit the trip **(object control)**

I ***promised*** *him to quit the trip.*
→ I quit the trip **(subject control)**

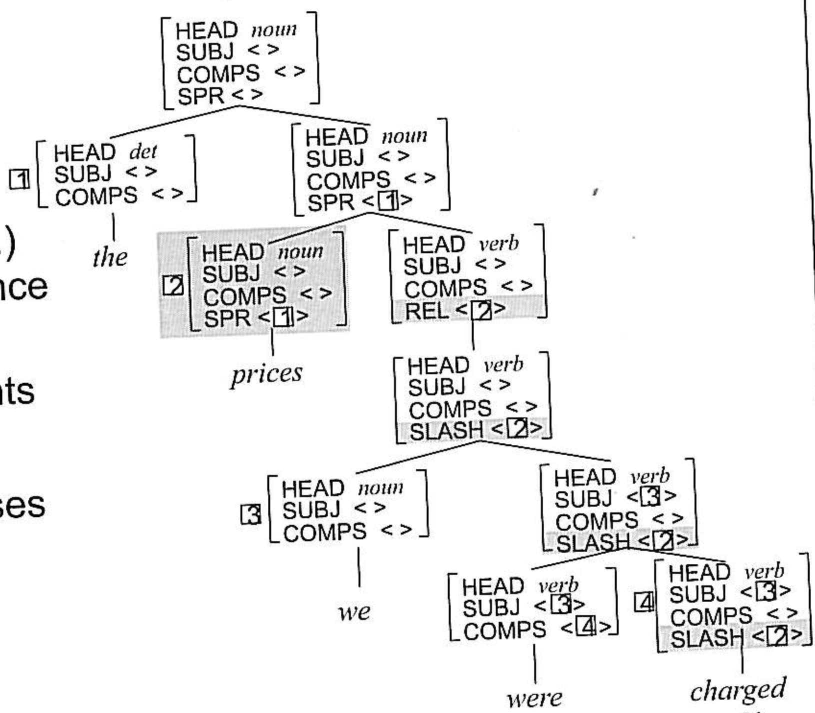
$$\begin{bmatrix} \text{PHON } \textit{"persuade"} \\ \text{HEAD } \textit{verb} \\ \text{SUBJ } \langle \text{NP}\boxed{1} \rangle \\ \text{COMPS } \langle \text{NP}\boxed{2}, \text{VP}\boxed{3} \rangle \\ \text{CONT } \textit{persuade}(\boxed{1}, \boxed{2}, \boxed{3}(\boxed{2}, \ldots)) \end{bmatrix} \qquad \begin{bmatrix} \text{PHON } \textit{"promise"} \\ \text{HEAD } \textit{verb} \\ \text{SUBJ } \langle \text{NP}\boxed{1} \rangle \\ \text{COMPS } \langle \text{NP}\boxed{2}, \text{VP}\boxed{3} \rangle \\ \text{CONT } \textit{promise}(\boxed{1}, \boxed{2}, \boxed{3}(\boxed{1}, \ldots)) \end{bmatrix}$$

persuade(I, he, quit(he, trip))

promise(I, he, quit(I, trip))

Nonlocal dependencies

- NONLOCAL features (SLASH, REL, etc.) explain long-distance dependencies
  - WH movements
  - Topicalization
  - Relative clauses etc...



HPSG resources

- Enju: an English HPSG grammar extracted from Penn Treebank
- Hand-crafted grammars
  - LinGO ERG (English)
  - JaCY (Japanese)
  - GG (German)
  - Alpino (Dutch)
  - Grammars for other languages are underdevelopment in the DELPH-IN community
- Grammar Matrix
  - A framework for the rapid start-up of new grammars
  - The framework provides principles/structures shared among all grammars

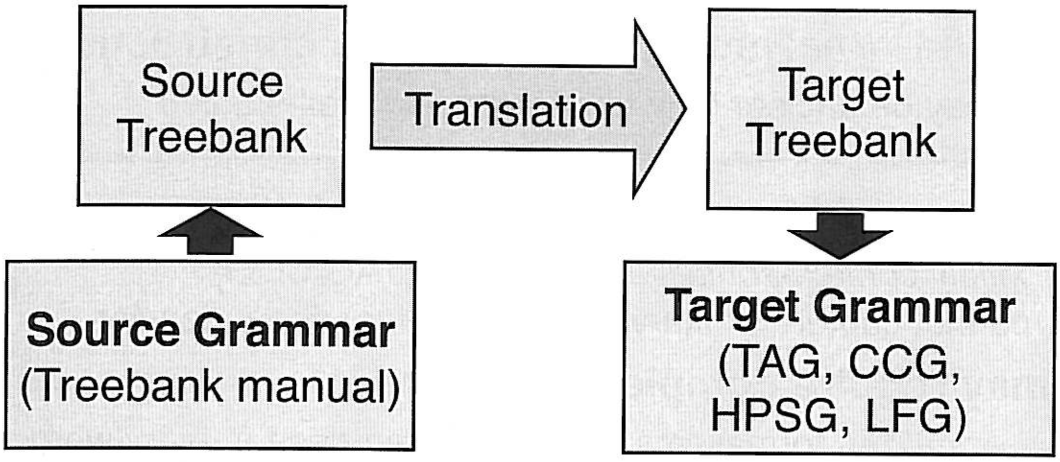
COFFEE BREAK

VI. Inducing Expressive Grammars from Corpora

Obtaining wide-coverage grammars

- **Extracting grammars from treebanks**:
  - Leverage the effort that went into original annotation
  - Requires a formalism (and treebank-)specific algorithm to translate existing treebank into desired target
- **Handwritten grammars**:
  - Require substantial manual effort
  - Difficult to reuse grammars across formalisms
  - Examples: XLE (LFG), ERG (HPSG), XTAG (TAG),...

Grammar extraction



Treebanks...

... contain arbitrary text:

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

... contain arbitrary descriptions:

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

Grammar formalisms...

....provide analyses for well-studied constructions

– It may be unclear how to analyze less well-studied constructions

... may provide constrained expressivity

– *Mildly context-sensitive* formalisms (TAG/CCG) cannot capture arbitary (e.g. anaphoric) dependencies

... may require complete analyses

– *Lexicalized* formalisms need lexical entries for every word

Research questions

- Are the treebank descriptions sufficient to obtain the desired 'deep' analyses?

- Can the grammar formalism account for the descriptions provided in the treebank?

Treebanks

What do we need to extract grammars from treebanks?

Source treebank needs to have
an **explicit representation** of:

- heads
- arguments } core dependencies
- modifiers
- conjuncts } may use different representation
- nonlocal dependencies ⇒ *special treatment!*

Extraction algorithms need to distinguish each dependency type

What do treebanks capture?

Local dependencies and phrase structure
- **Head-argument, head-modifier, simple coordination**
- Core of any annotation;
  but argument/modifier distinction not always clear

Nonprojective dependencies
- **Extraposition, scrambling**
- Captured directly in dependency banks;
  with null elements in treebanks

Nonlocal dependencies
- **Raising, control; wh-extraction, topicalization; non-standard coordination**
- Require other means of representations
  (traces, secondary edges) – often ideosyncratic
- Annotation sometimes missing

75

Challenges for grammar extraction

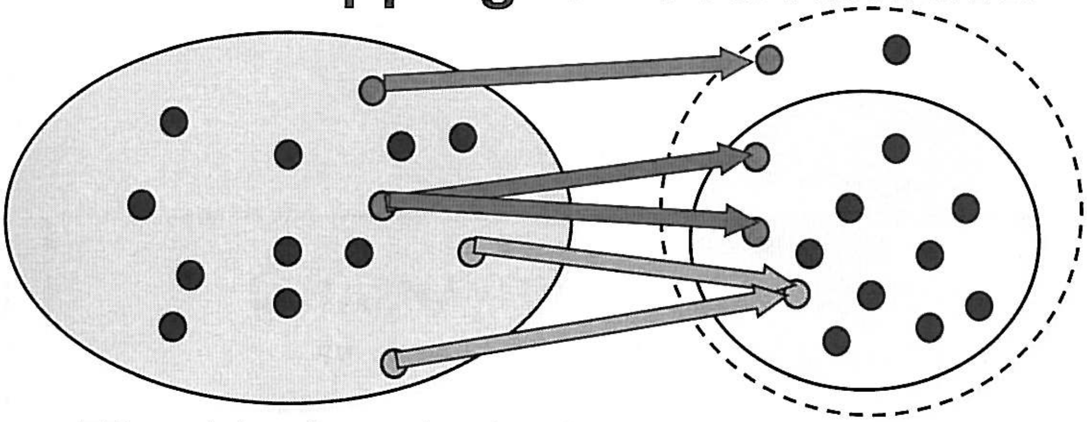
Differences in analysis
- may require **systematic changes** to treebank

Treebank uses underspecified analyses
- may require **additional annotation** or **heuristics**

Noise in treebank analysis
- may require **ad-hoc changes** to treebank

76

The mapping is not a function

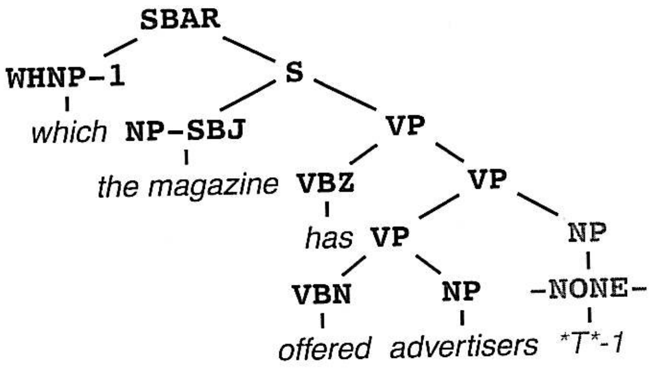


- TB contains dependencies the grammar can't capture.
- TB doesn't contain enough information to define a single target analysis.
- TB makes distinctions which the grammar does not care about (inconsistencies?)

Penn Treebank

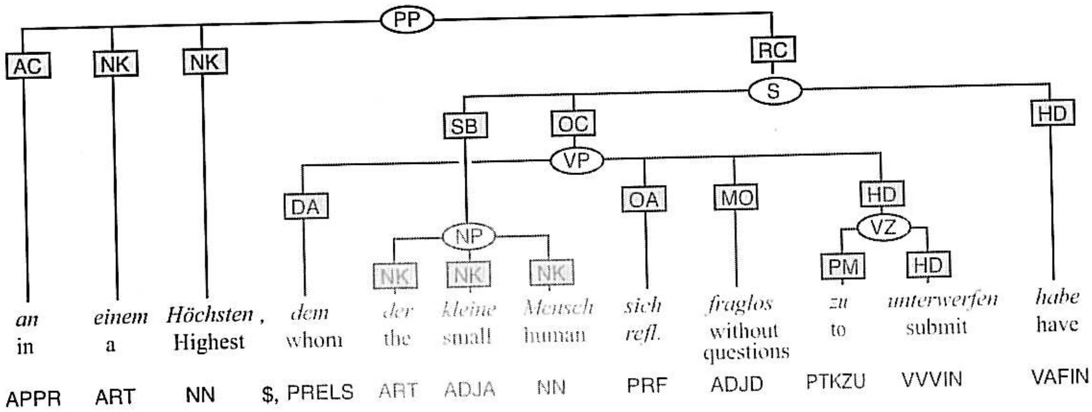
- **Phrase-structure treebank**
  requires **head-finding** and **arg/adjunct distinction** heuristics
- **Non-local dependencies: null elements, traces, and coindexation**
  ***-null elements:** passive, PRO
  ***T*-traces:** wh-movement, tough movement
  ***RNR*-traces:** right-node raising
  Other null elements:
    EXP: expletive,
    ICH ("insert constituent here"): extraposition
    U (units): $ 500 *U*
    PPA (permanent predictable ambiguity)
  =-coindexation: argument cluster coordination and gapping

Wh-extraction in the Penn Treebank



Coindexed traces indicate non-local dependencies

TiGer/NeGra



- Explicit annotation of heads, arguments, modifiers, conjuncts
- Non-local dependencies: discontinuous constituents (or secondary edges)

GRAMMAR EXTRACTION

General procedure

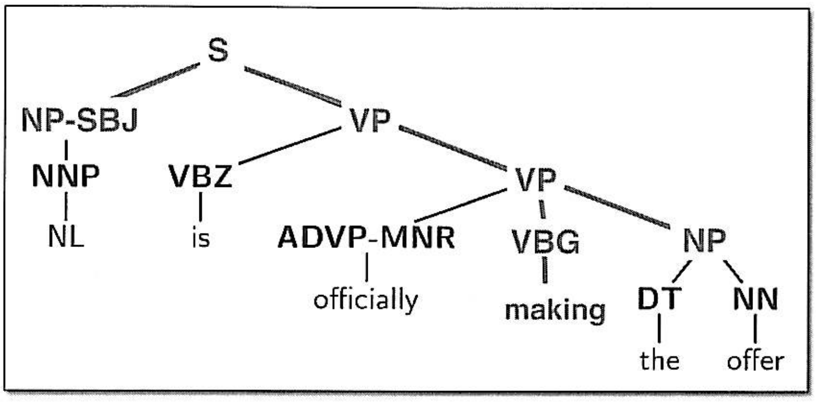
1. **Cleanup/preprocessing** (optional)
   a) **Eliminate noise** and inconsistencies
   b) **Change unwanted analyses**; use heuristics to add information
2. **Parse treebank**
   a) **Identify local dependencies**: heads, args, modifiers, conjuncts
   b) **Identify non-local dependencies**: extraction, non-stand. coordination.
3. **Translate treebank**
   a) **Basic case: local dependencies**
      each type may require different treatment
   b) **Special cases: non-local dependencies**
      each type may require different treatment
4. **Postprocessing** (optional)
   a) **clean-up**
   b) **translate syntactic analysis into semantics**

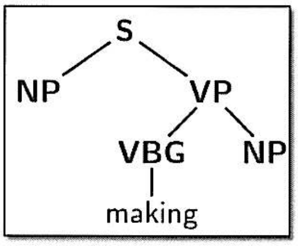
Evaluating extracted grammars or lexicons

- **Grammar/lexicon size**
  - How many entries does each word have?
  - How many kinds of entries (e.g. different categories)?
  - Depends on heuristics used and on granularity of analysis
- **Coverage and convergence**
  - How many lexical entries required to parse unseen data are missing?
- **Distribution of types of lexical entries**
  - How many different kinds of rare categories?
- **Quality?**
  - Inspection, comparison with manual grammar

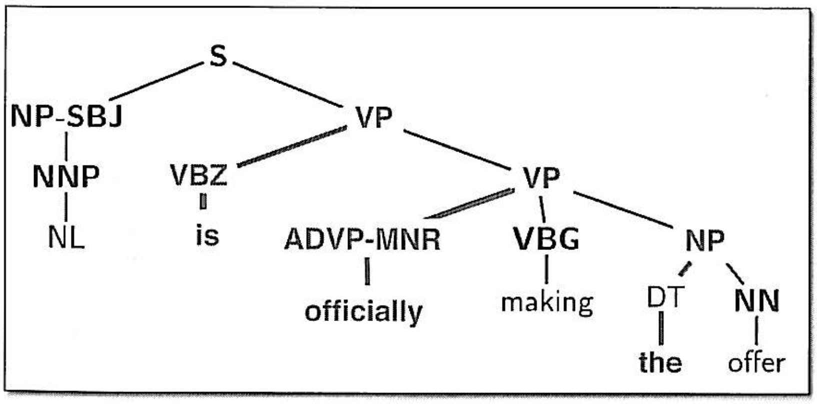
EXTRACTING TAGS

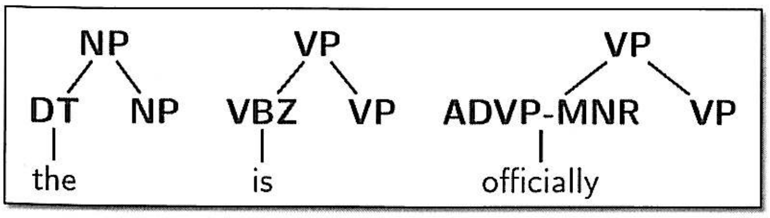
TAG extraction: head + arguments





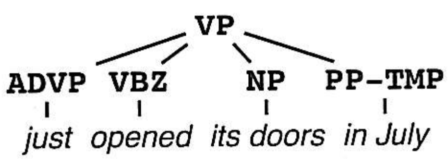
TAG extraction: adjuncts

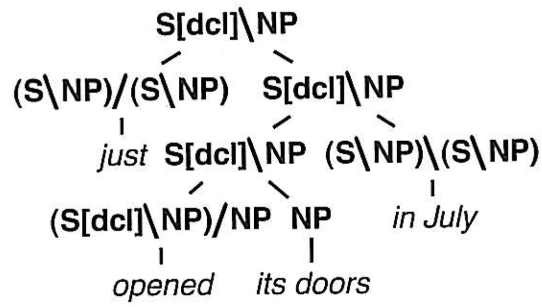




EXTRACTING CCGS

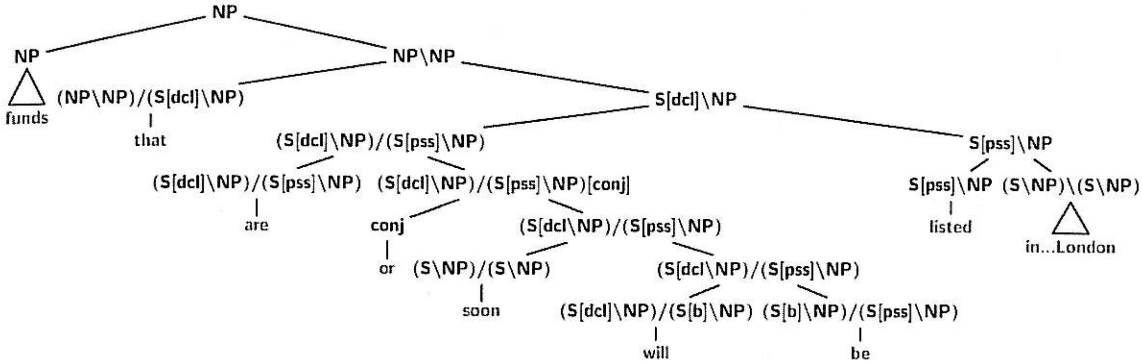
The basic translation algorithm





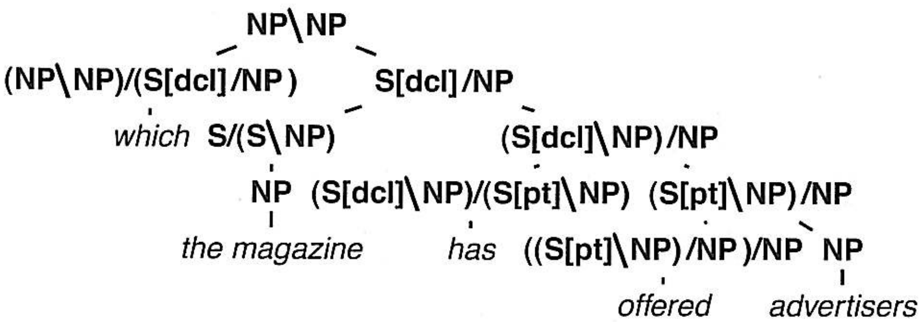
1. Identify heads, arguments, adjuncts
2. Binarize tree
3. Read off CCG categories
4. Get dependency structure

CCGbank derivations



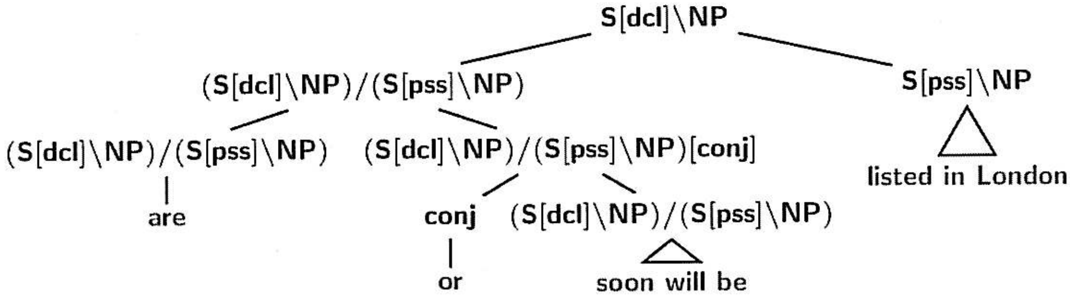
that	**((NP\NP)/(S[dcl]\NP))**	funds	are, will	
are	**((S[dcl]\NP)/(S[pss]\NP))**	*funds*	*listed*	
soon	**((S\NP)/(S\NP))**		will	
will	**((S[dcl]\NP)/(S[b]\NP))**	*funds*	be	
be	**((S[b]\NP)/(S[pss]\NP))**		listed	
listed	**(S[pss]\NP)**	*funds*		
in	**(((S\NP)\(S\NP))/NP)**		listed	York, London

Wh-extraction in CCGbank



- The trace is cut out, but the dependency is captured.
- The relative pronoun subcategorizes for an incomplete sentence.
- This derivation requires type-raising and composition.

Right-node raising



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

CCGbank

- **Coverage of the translation algorithm:**
  99.44% of all sentences in the Treebank
  (main problem: sentential gapping)

- **The lexicon (sec.02-21):**
  – 74,669 entries for 44,210 word types
  – 1286 lexical category types
  (439 appear once, 556 appear 5 times or more)

- **The grammar (sec. 02-21):**
  – 3262 rule instantiations (1146 appear once)

The most ambiguous words

Word	#Cats.	Freq.	Word	#Cats.	Freq.
as	130	4237	*of*	59	22782
is	109	6893	*that*	55	7951
to	98	22056	*-LRB-*	52	1140
than	90	1600	*not*	50	1288
in	79	15085	*are*	48	3662
–	67	2001	*with*	47	4214
's	67	9249	*so*	47	620
for	66	7912	*if*	47	808
at	63	4313	*on*	46	5112
was	61	3875	*from*	46	4437

Many frequent words have *a lot* of categories

Frequency distribution of categories

Lexical categories Category frequency f	#Cats.
$100{,}000 \leq f < 220{,}000$	2
$10{,}000 \leq f < 100{,}000$	13
$1{,}000 \leq f < 10{,}000$	49
$100 \leq f < 1{,}000$	108
$10 \leq f < 100$	253
$5 \leq f < 10$	131
$2 \leq f < 5$	291
$0 < f \leq 1$	440

Boxer: from CCG to DRT

Translates CCG derivations (output of C&C parser) to Discourse Representation Theory

The events of April through June damaged the respect and confidence which most Americans previously had for the leaders of China .

```
( | x0 x1 x2 x3 x4 x5                          | ; | x6 x7 x8                         | )
  |--------------------------------------------|   |----------------------------------|
  | timex(x0) = XXXX06XX                       |   | have(x7)                         |
  | event(x1)                                  |   | agent(x7, x6)                    |
  | timex(x2) = XXXX04XX                       |   | patient(x7, x3)                  |
  | through(x2, x0)                            |   | named(x6, americans, nam)        |
  | of(x1, x2)                                 |   | previously(x7)                   |
  | |              |   |                 |     |   | event(x7)                        |
  | |--------------| v |-----------------|     |   | for(x7, x5)                      |
  | | respect(x3)  |   | confidence(x3)  |     |   | damage(x8)                       |
  | named(x4, china, loc)                      |   | event(x8)                        |
  | leader(x5)                                 |   | agent(x8, x1)                    |
  | of(x5, x4)                                 |   | patient(x8, x3)                  |
```

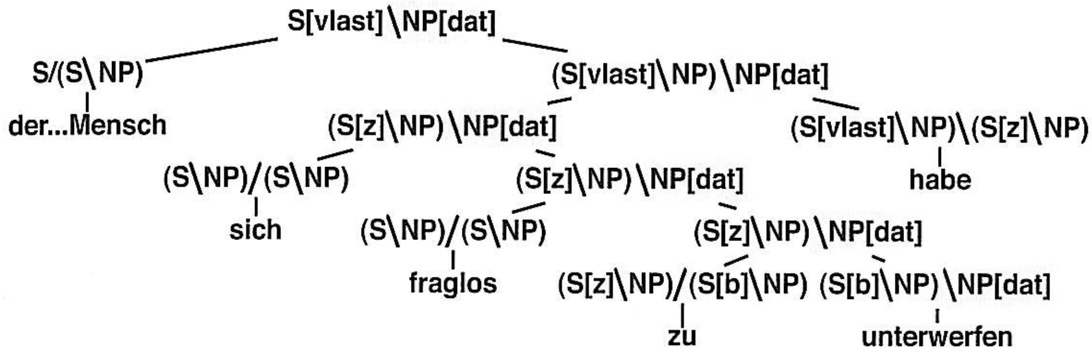

Reanalyzing the Penn Treebank

- Propbank and Nombank add information to the original Penn Treebank
- Vadas & Curran (ACL'08) add internal structure to compound nouns in Penn Treebank
- Honnibal, Curran & Bos (ACL'10) integrate this information into CCGbank

Extracting a CCG from Tiger

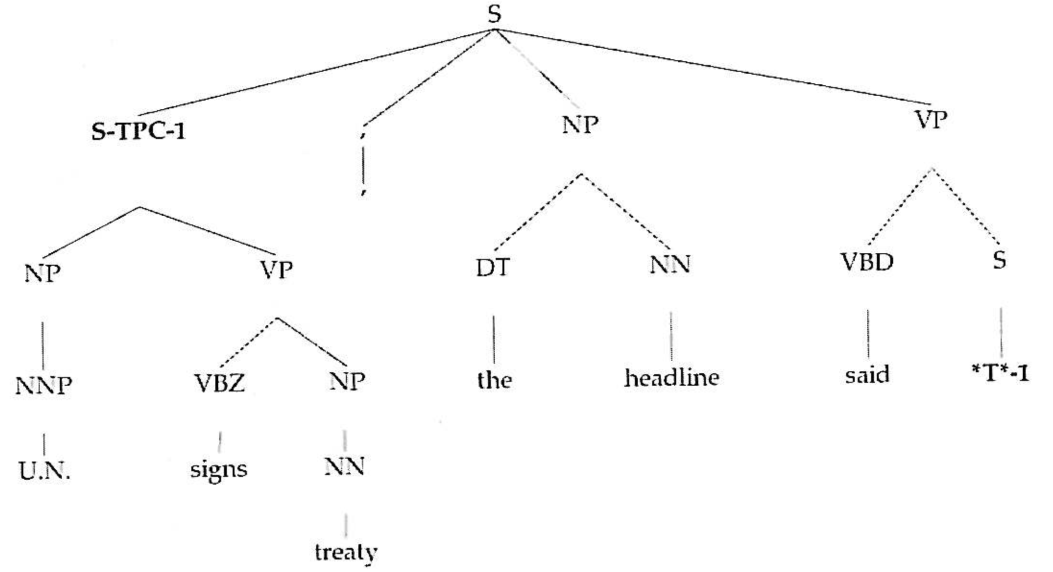
– We translate 92.4% of all trees into CCG (more work required...)

– >2500 lexical categories



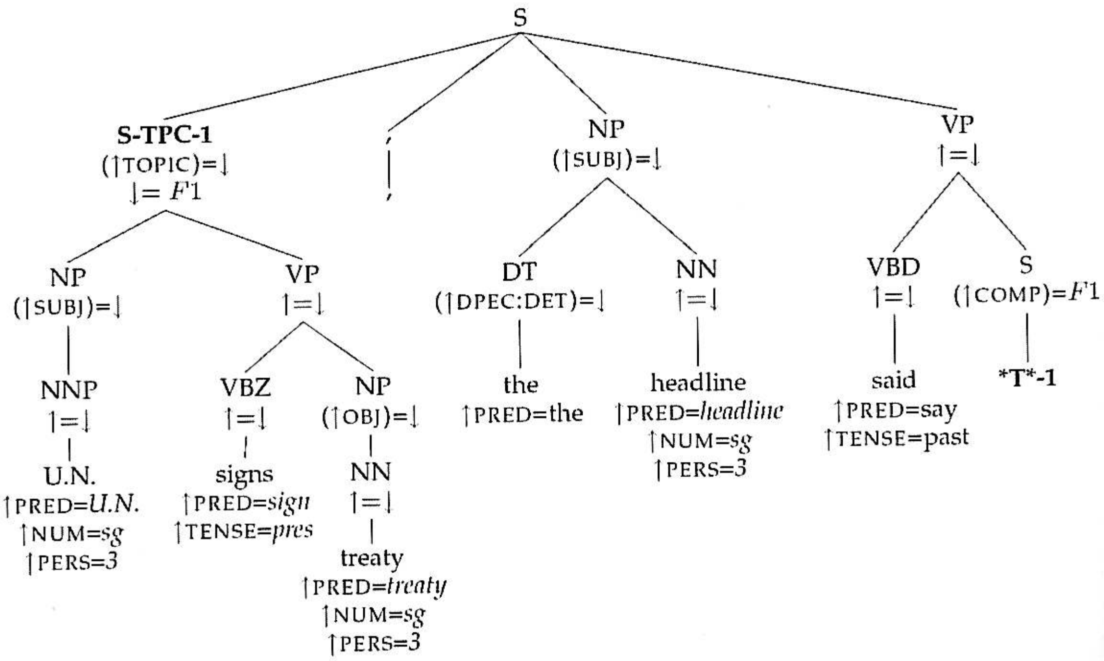
Extracting LFGs

Treebank Annotation: what we have



99

Treebank Annotation: what we want

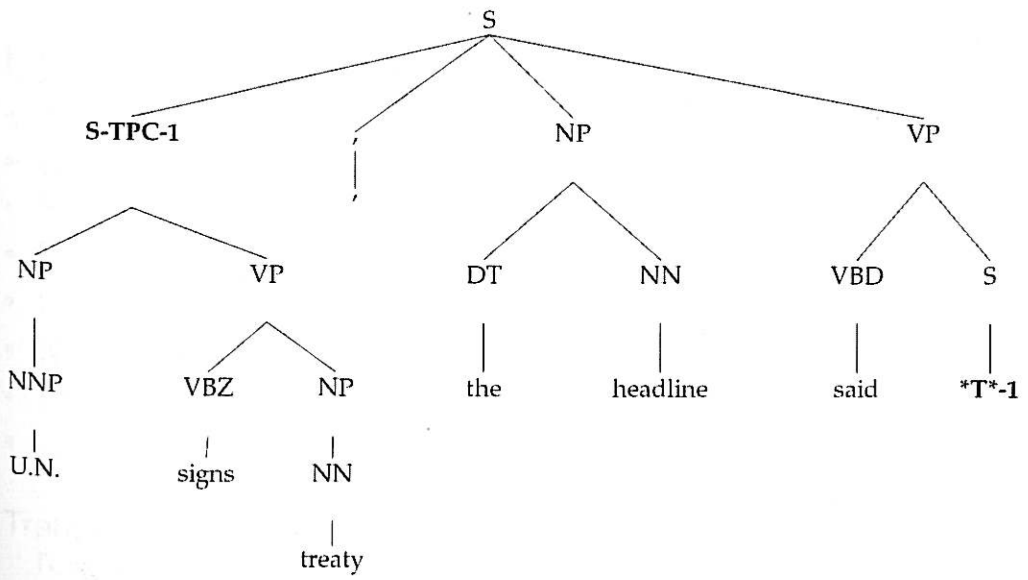


100

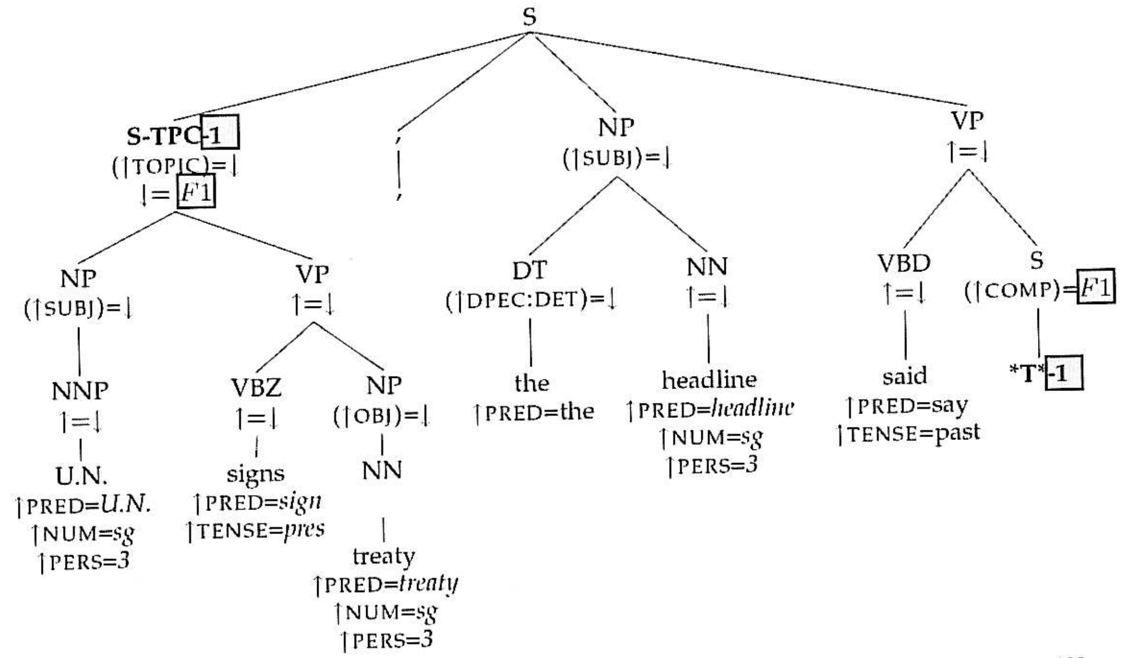
Treebank Annotation: what we want

$$
\begin{bmatrix}
\text{TOPIC} & \begin{bmatrix}
\text{PRED} & \text{sign} \\
\text{TENSE} & \text{pres} \\
\text{SUBJ} & \begin{bmatrix} \text{PRED} & \text{U.N.} \\ \text{NUM} & \text{sg} \\ \text{PERS} & 3 \end{bmatrix} \\
\text{OBJ} & \begin{bmatrix} \text{PRED} & \text{treaty} \\ \text{NUM} & \text{sg} \\ \text{PERS} & 3 \end{bmatrix}
\end{bmatrix} \boxed{1} \\
\text{PRED} & \text{say} \\
\text{TENSE} & \text{past} \\
\text{SUBJ} & \begin{bmatrix}
\text{SPEC} & \begin{bmatrix} \text{DET} & \text{the} \end{bmatrix} \\
\text{PRED} & \text{headline} \\
\text{NUM} & \text{sg} \\
\text{PERS} & 3
\end{bmatrix} \\
\text{COMP} & \boxed{1}
\end{bmatrix}
$$

Treebank Annotation: what we have



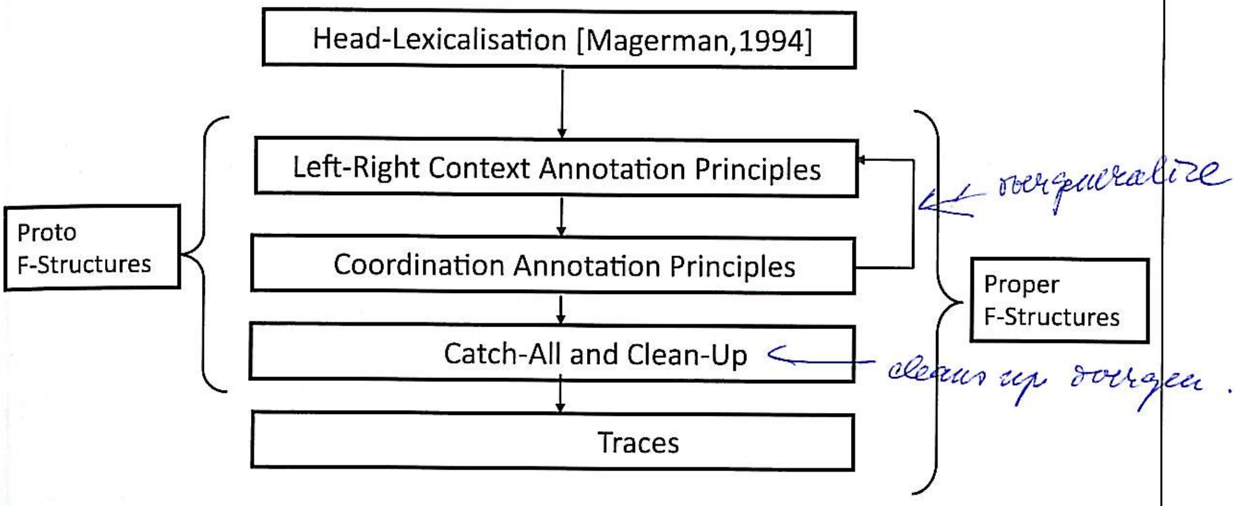
Treebank Annotation: what we have



Treebank Annotation: what we want

$$\begin{bmatrix} \text{TOPIC} & \begin{bmatrix} \text{PRED} & \text{sign} \\ \text{TENSE} & \text{pres} \\ \text{SUBJ} & \begin{bmatrix} \text{PRED} & \text{U.N.} \\ \text{NUM} & \text{sg} \\ \text{PERS} & 3 \end{bmatrix} \\ \text{OBJ} & \begin{bmatrix} \text{PRED} & \text{treaty} \\ \text{NUM} & \text{sg} \\ \text{PERS} & 3 \end{bmatrix} \end{bmatrix} \boxed{1} \\ \text{PRED} & \text{say} \\ \text{TENSE} & \text{past} \\ \text{SUBJ} & \begin{bmatrix} \text{SPEC} & \begin{bmatrix} \text{DET} & \text{the} \end{bmatrix} \\ \text{PRED} & \text{headline} \\ \text{NUM} & \text{sg} \\ \text{PERS} & 3 \end{bmatrix} \\ \text{COMP} & \boxed{1} \end{bmatrix}$$

Treebank Annotation: Penn-II & LFG



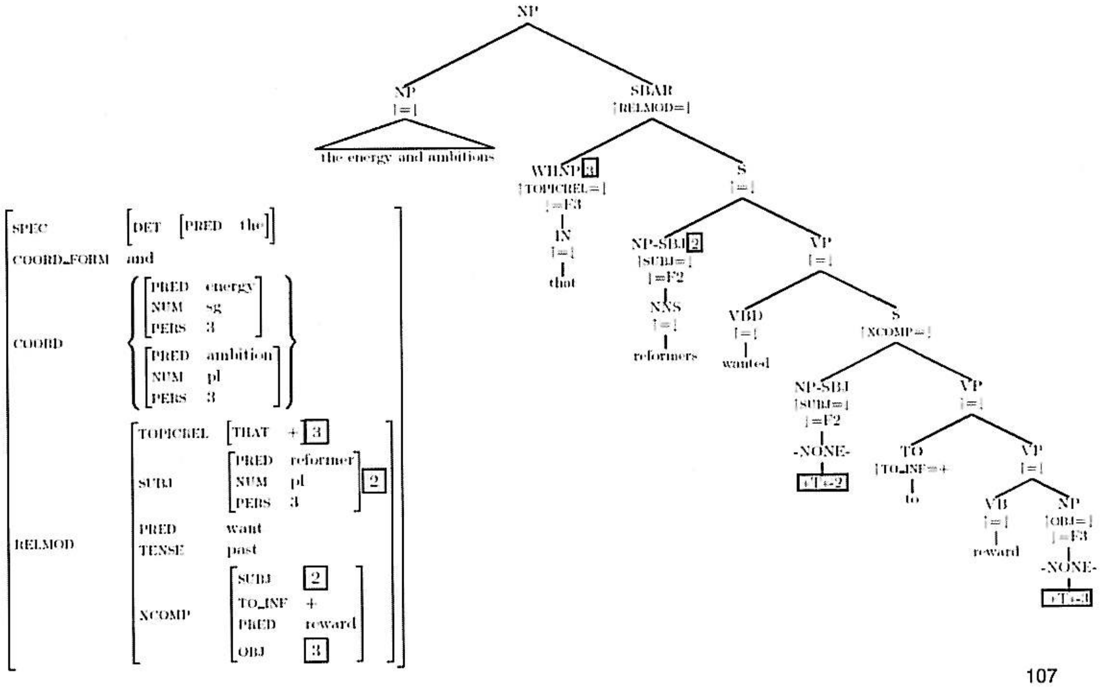
Treebank Annotation: Traces

Long Distance Dependencies:

- Topicalisation
- Questions
- Wh- and wh-less relative clauses
- Passivisation
- Control constructions
- ICH (interpret constituent here)
- RNR (right node raising)
- …

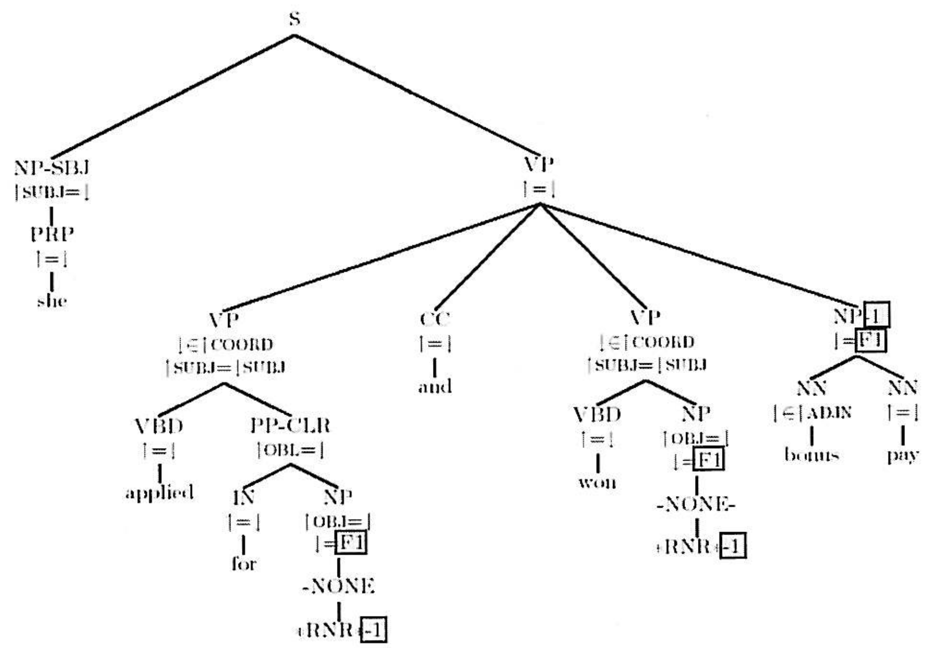
Translate Penn-II traces and coindexation into corresponding reentrancy in f-structure

Treebank Annotation: Control & Wh-Rel. LDD



107

Treebank Annotation: Right Node Raising



108

Treebank Annotation: Right Node Raising

$$
\left[\begin{array}{ll}
\textrm{SUBJ} & \left[\begin{array}{ll} \textrm{PRED} & \textrm{pro} \\ \textrm{PRON\_FORM} & \textrm{she} \end{array}\right] \boxed{2} \\
\textrm{COORD} & \left\{ \begin{array}{l}
\left[\begin{array}{ll}
\textrm{SUBJ} & \boxed{2} \\
\textrm{PRED} & \textrm{apply} \\
\textrm{TENSE} & \textrm{past} \\
\textrm{OBL} & \left[\begin{array}{ll} \textrm{PFORM} & \textrm{for} \\ \textrm{OBJ} & \boxed{1} \end{array}\right]
\end{array}\right] \\
\left[\begin{array}{ll}
\textrm{SUBJ} & \boxed{2} \\
\textrm{PRED} & \textrm{win} \\
\textrm{TENSE} & \textrm{past} \\
\textrm{OBJ} & \left[\begin{array}{ll}
\textrm{PRED} & \textrm{pay} \\
\textrm{NUM} & \textrm{sg} \\
\textrm{PERS} & 3 \\
\textrm{ADJN} & \left\{ \left[\begin{array}{ll} \textrm{PRED} & \textrm{bonus} \\ \textrm{NUM} & \textrm{sg} \\ \textrm{PERS} & 3 \end{array}\right] \right\}
\end{array}\right] \boxed{1}
\end{array}\right]
\end{array} \right\} \\
\textrm{COORD\_FORM} & \textrm{and}
\end{array}\right]
$$

apply<subj,obl:for>
win<subj,obj>

Other Treebanks and Dependency Banks

LFG grammar acquisition for parsing and generation

- **Spanish:** Cast3LB (Grzegorz Chrupala)
- **German:** TiGer Treebank (Ines Rehbein)
- **French:** P7T and MFT (Natalie Schluter)
- **Chinese:** CTB6 (Yuqing Guo)
- **Arabic:** ATB (Jafa Al'Raheb, Lamia Tounsi, Mohammed Attia, Hann Bchara)
- **Japanese:** Kyoto Text Corpus (Massanori Oya)

- Typologically very different languages
- Morpologically rich/poor
- Semi-free word order – strongly configurational languages
- Drop: pro, anything …

Other Treebanks and Dependency Banks

As a consequence:

- Very different LFG f-str annotation algorithms
- Original f-str annotation algorithm for English (configurational , not much morphology) and Penn-II ("X-bar, traces ....)
- More recent f-str annotation algorithms:
- Use richer treebank labels
- "Translate" to f-structures
- More machine learning

Chinese

(1) [illegible]
not want look-for train have potential DE new writer
'(People) don't want to look for and train the new writers who have potential.'

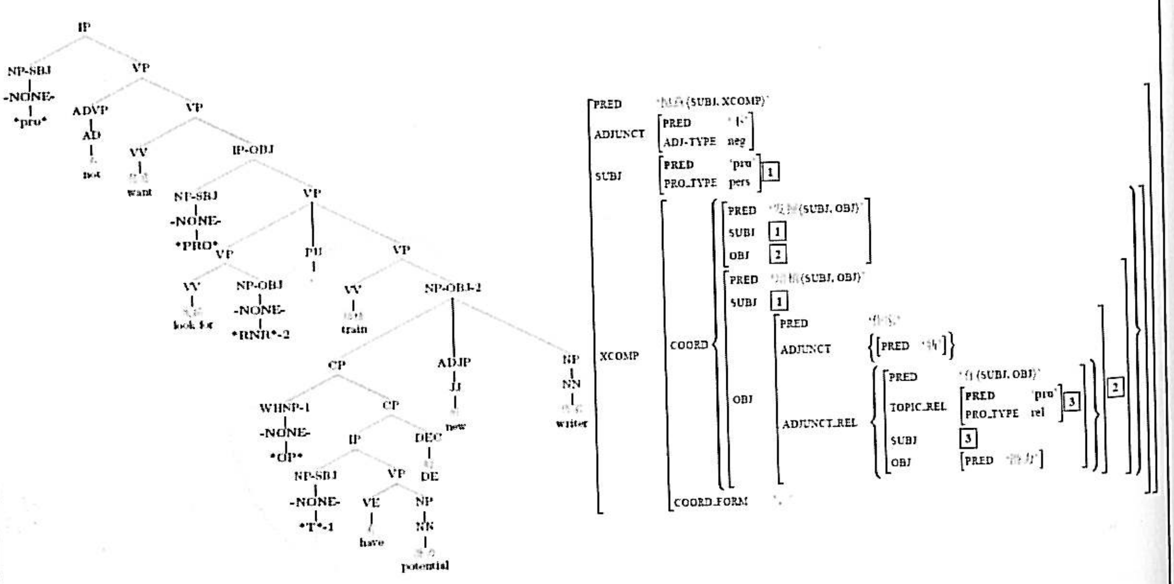


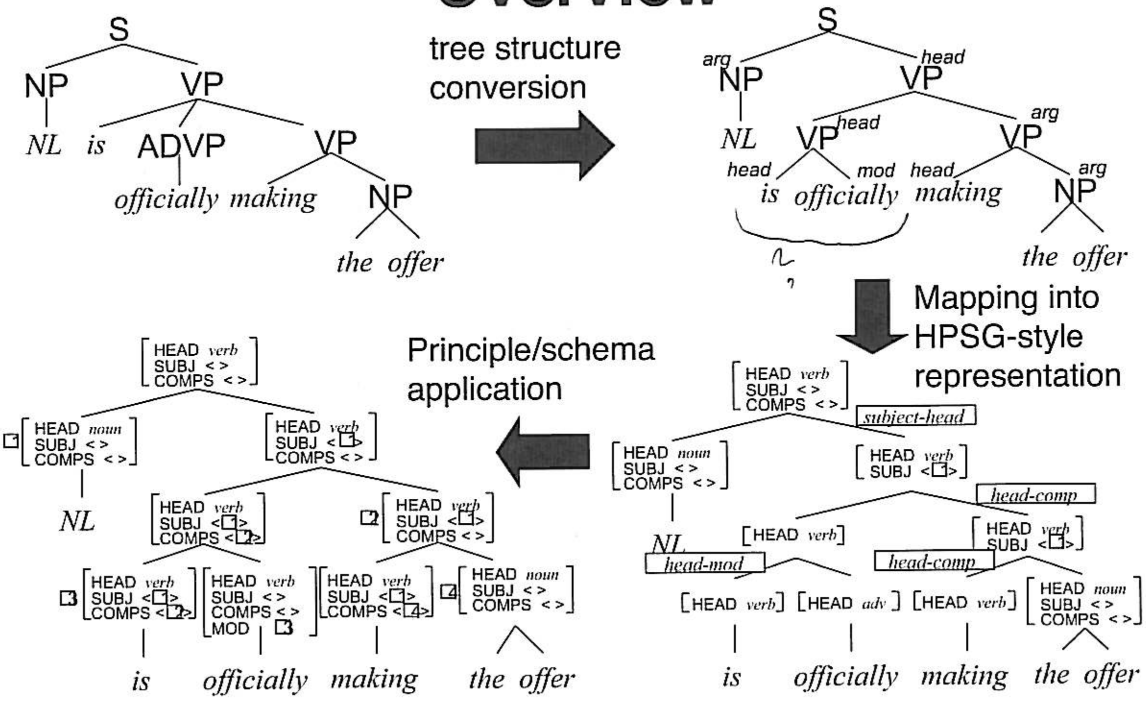
Figure 3.1: Example of NLDs represented in CTB, including dropped subject (*pro*), control subject (*PRO*), WH-trace in relativisation (*T*), and right node raising in coordination (*RNR*)

EXTRACTING HPSG

Translating Penn Treebank into HPSG

- Convert Penn-style phrase structure trees into HPSG-style structures
  - Converting tree structures
    - Small clauses, passives, NP structures, auxiliary/control verbs, LDDs, etc.
  - Mapping into HPSG-style representations
    - Head/argument/modifier distinction, schema name assignment
    - Mapping into HPSG signs
  - Applying HPSG principles/schemas
    - Fully specified HPSG structures are obtained

Overview

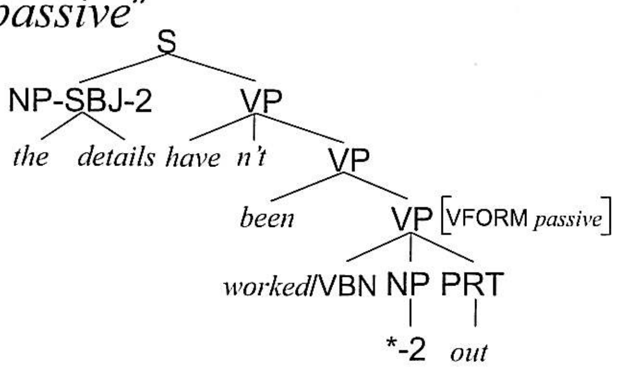


Tree structure conversion

- Coordination, quotation, insertion, and apposition
- Small clauses, “than” phrases, quantifier phrases, complementizers, etc.
- Disambiguation of non-/pre-terminal symbols (TO, etc.)
- HEAD features (CASE, INV, VFORM, etc.)
- Noun phrase structures
- Auxiliary/control verbs
- Subject extraction
- Long distance dependencies
- Relative clauses, reduced relatives

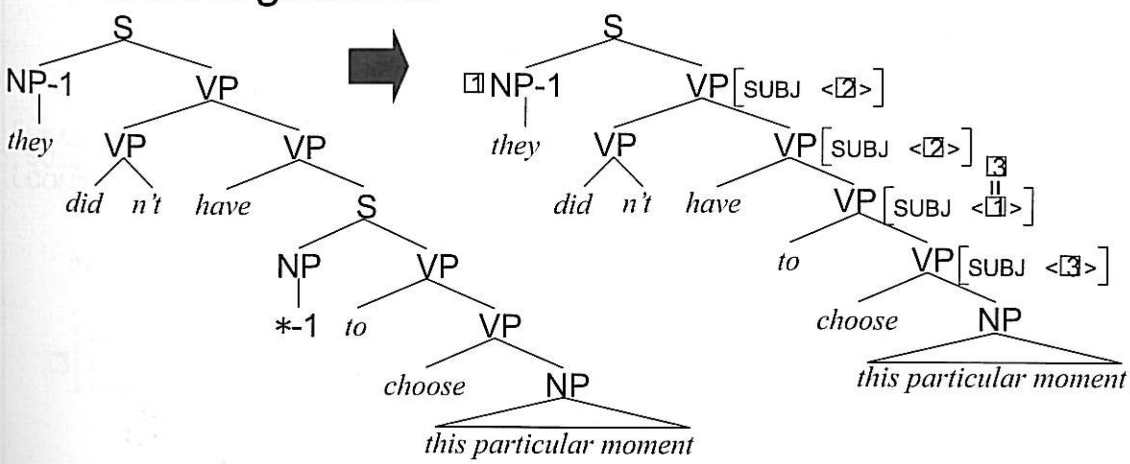
Passive

• "be + VBN" constructions are assigned "VFORM *passive*"



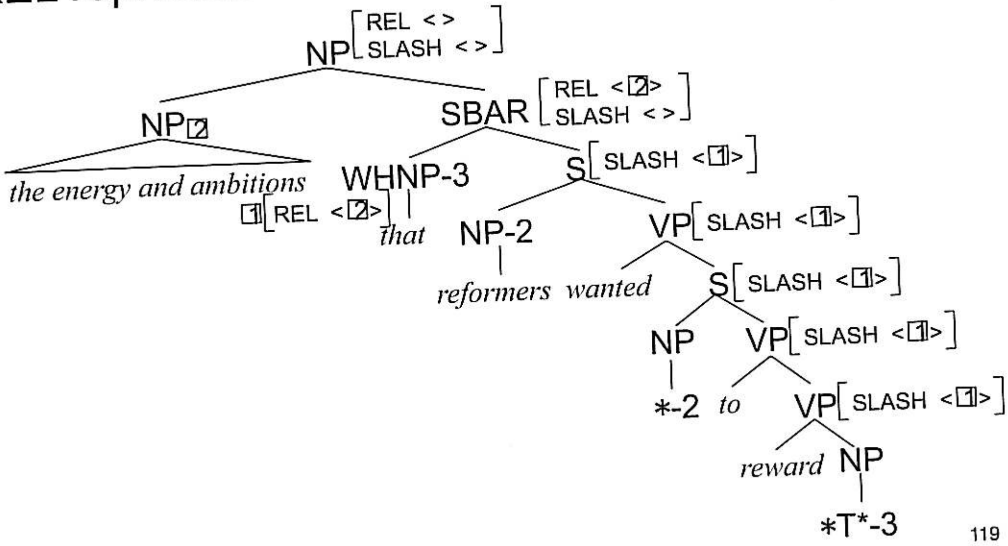
Auxiliary/control verbs

• Reentrancies are annotated for representing shared arguments



LDDs: Object relative

- SLASH represents moved arguments
- REL represents relative-antecedent relations



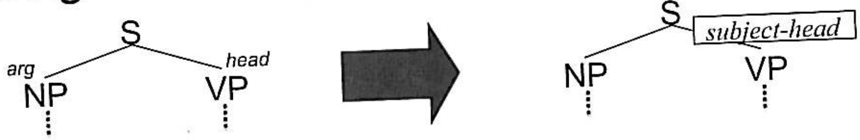
Mapping into HPSG-style representations

- Convert pre-/non-terminal symbols into HPSG-style categories

NN → [HEAD *noun*, AGR *3sg*]

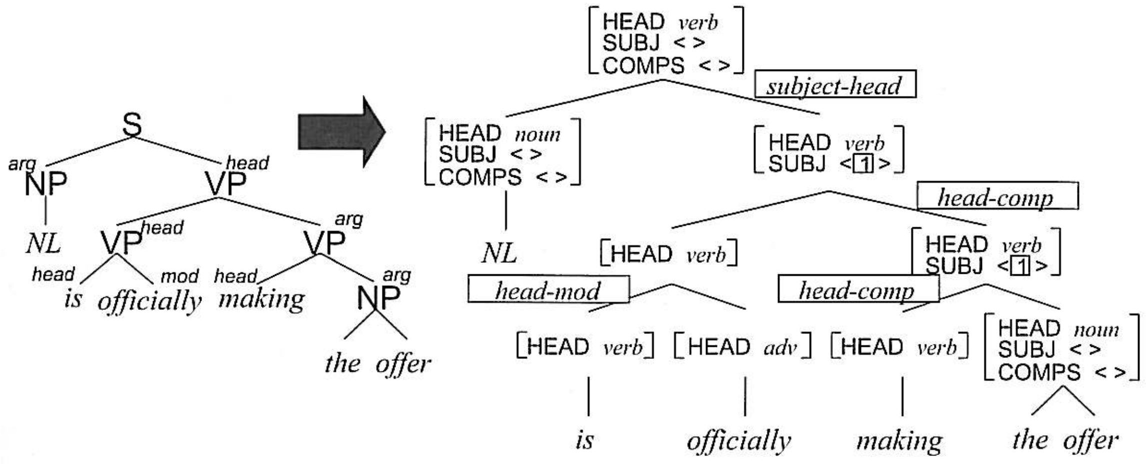
VBD → [HEAD *verb*, VFORM *finite*, TENSE *past*]

- Assign schema names to internal nodes

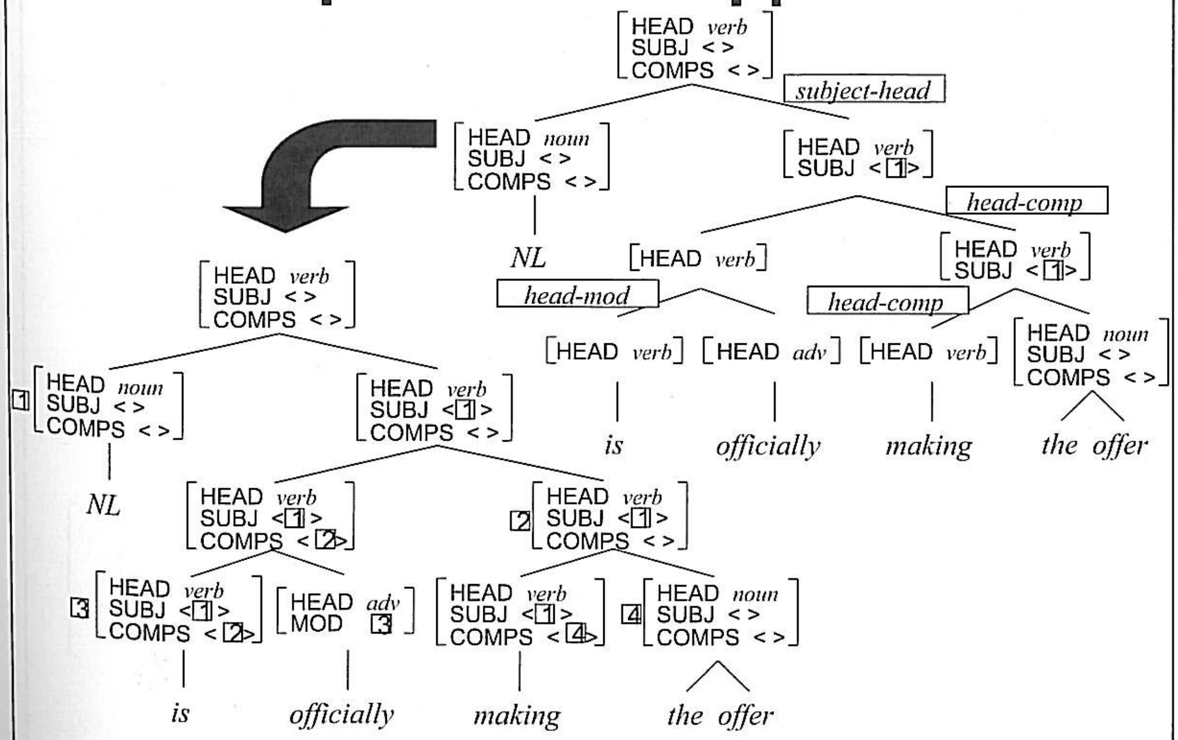


Category mapping & schema name assignment

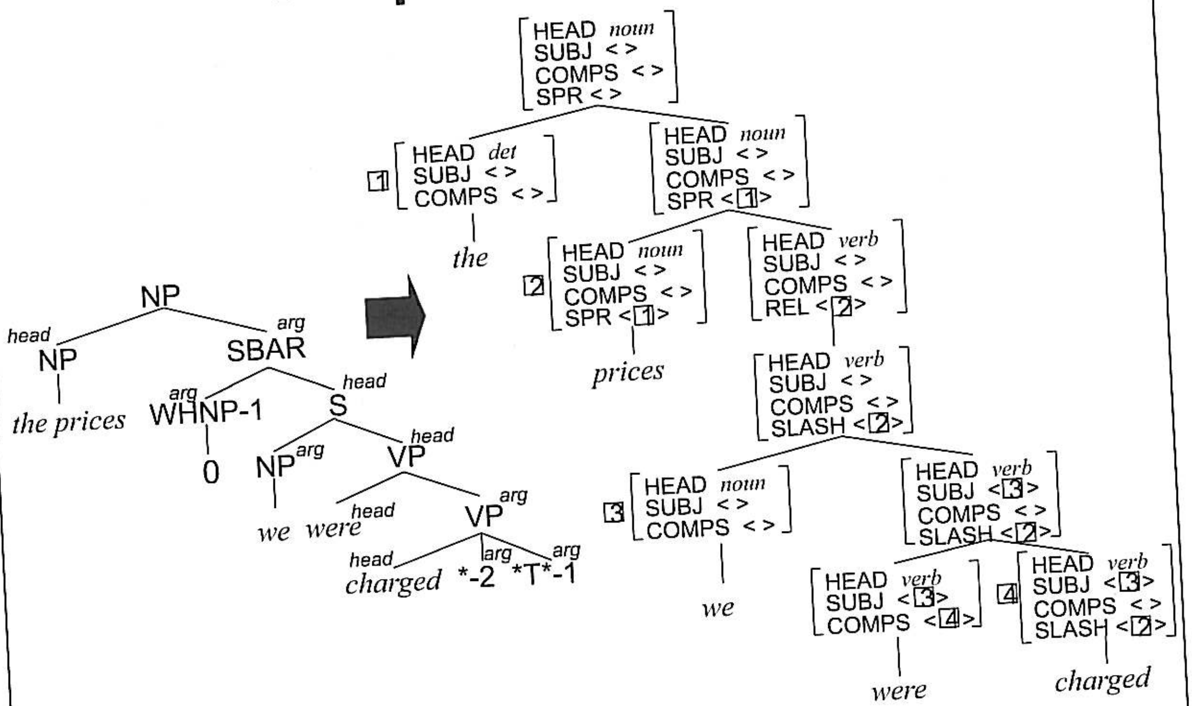
- Example: *"NL is officially making the offer"*



Principle/schema application

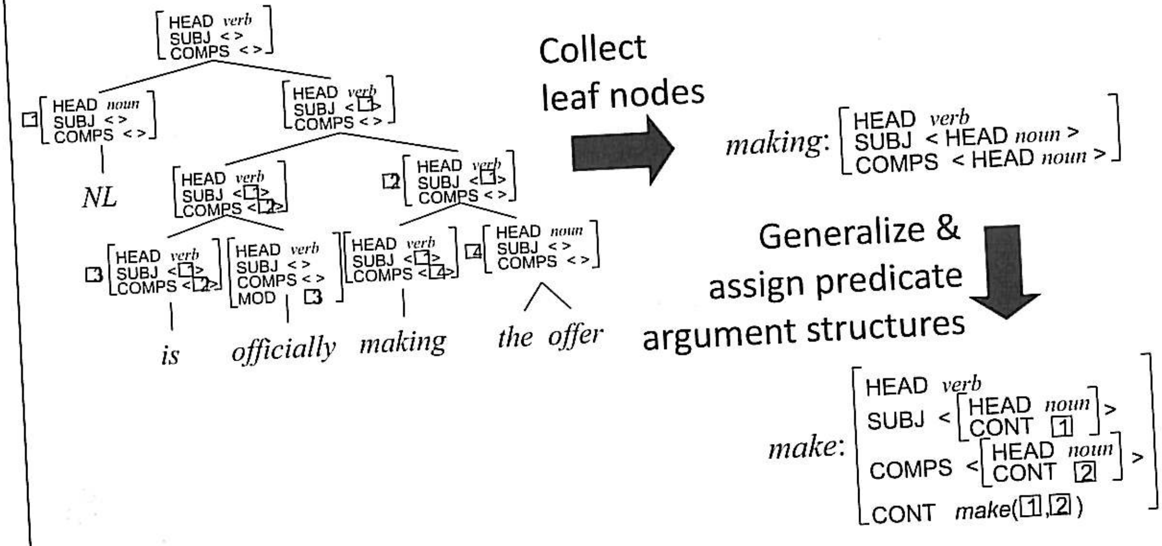


Complicated example



123

Extracting lexical entries



124

Generalization

- Remove unnecessary feature values
- Convert lexical entries of inflected words into lexical entries of lexemes using *inverse* lexical rules
  - Derivational rules: Ex. passive rule

$$\begin{bmatrix} \text{HEAD } \textit{verb} \\ \text{SUBJ <HEAD: } \textit{noun}\text{>} \\ \text{COMPS <HEAD: } \textit{prep\_by}\text{>} \end{bmatrix} \Rightarrow \begin{bmatrix} \cancel{\text{HEAD}} \textit{ verb} \\ \text{SUBJ <}\cancel{\text{HEAD}}\text{: } \textit{noun}\text{>} \\ \text{COMPS <}\cancel{\text{HEAD}}\text{: } \textit{noun}\text{>} \end{bmatrix}$$

  - Inflectional rules: Ex. past-tense rule

$$\begin{bmatrix} \text{HEAD} \begin{bmatrix} \textit{verb} \\ \text{VFORM } \textit{finite} \\ \text{TENSE } \textit{past} \end{bmatrix} \end{bmatrix} \Rightarrow \begin{bmatrix} \text{HEAD} \begin{bmatrix} \textit{verb} \\ \text{VFORM } \textit{base} \end{bmatrix} \end{bmatrix}$$

Predicate argument structures

- Create mappings from syntactic arguments into semantic arguments

Ex. lexical entry for *"make"*

$$\begin{bmatrix} \text{CAT} \begin{bmatrix} \text{HEAD } \textit{verb} \\ \text{VAL} \begin{bmatrix} \text{SUBJ <} \begin{bmatrix} \text{CAT|HEAD } \textit{noun} \\ \text{CONT } \boxed{1} \end{bmatrix} \text{>} \\ \text{COMPS <} \begin{bmatrix} \text{CAT|HEAD } \textit{noun} \\ \text{CONT } \boxed{2} \end{bmatrix} \text{>} \end{bmatrix} \end{bmatrix} \\ \text{CONT } \textit{make}(\boxed{1}, \boxed{2}) \end{bmatrix}$$

Results

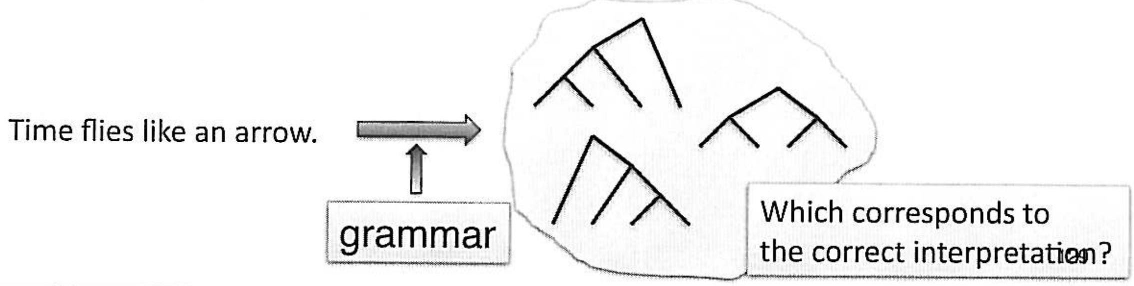
- Conversion coverage: 96% of sentences from Penn Treebank 02-21 were converted
- Lexicon:
  - Lexical entries are extracted for 45,236 word types
  - 1136 lexical entry types for base forms, 2289 types for expanded forms
- Parsing coverage: >99%

127

VII. WIDE-COVERAGE PARSING WITH EXPRESSIVE GRAMMARS

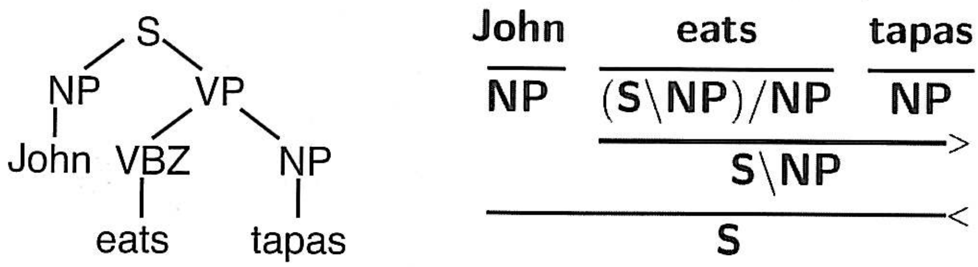
Wide-coverage parsing with expressive grammars

- Wide-coverage grammars are necessary for wide-coverage parsing ← solved!
- Wide-coverage grammars are a halfway to wide-coverage parsing
  - All grammatical structures do not necessarily correspond to "natural interpretation"
  - High parsing accuracy = accurate selection of the "correct" one from possible grammatical structures



LFG/CCG/HPSG parsing ≈ CFG parsing

- LFG/CCG/HPSG parsing is essentially phrase structure parsing
- Conventional methods for CFG parsing can be applied
  - Chart parsing
  - Statistical models for disambiguation (PCFG, machine learning, etc.)
  - Search techniques (Viterbi, beam search, etc.)



Added benefits

- Semantic structures are output as a result of parsing
- Expressive grammars restrict search space
  - Ungrammatical structures are excluded by hard constraints
- Expressive grammars provide additional information for statistical disambiguation
  - Lexical categories, lexical entries → supertagging
  - Predicate argument structures → semantic features
  - f-structures

Available wide-coverage parsers and basic architectures

- **CCG**
  - C&C parser: supertagging + discriminative model for phrase structure parsing
  - StatCCG: generative parser
- **HPSG**
  - Enju parser: supertagging + discriminative model for phrase structure parsing
- **LFG**
  - DCU-LFG: pipeline architecture, integrated architecture

PARSING WITH CCG/HPSG

Basic architecture

- **Supertagging + phrase structure parsing**

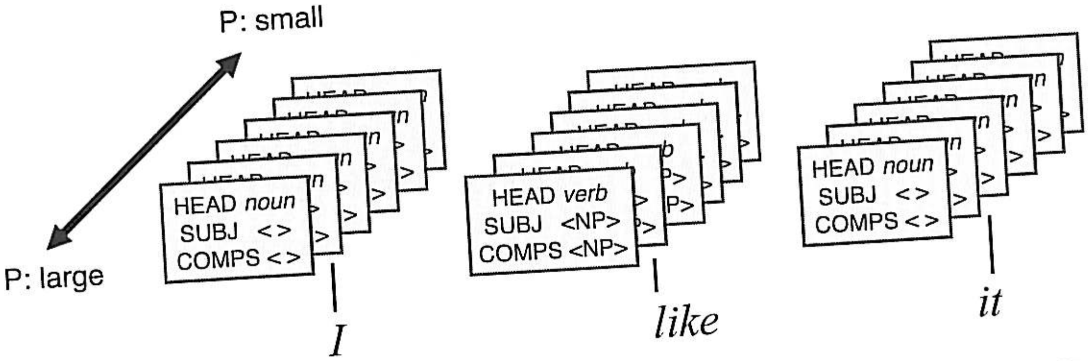
  Looks like chart parsing

  - **Terminal symbol**: lexical category (CCG), lexical entry (HPSG)
  - **Production rule**: combinatory rule (CCG), principle/schema (HPSG)

- **TAG, CCG and HPSG are lexicalized**
  - Lexical categories/entries encode rich grammatical constraints
  - Terminal symbol selection (=supertagging) plays a crucial role

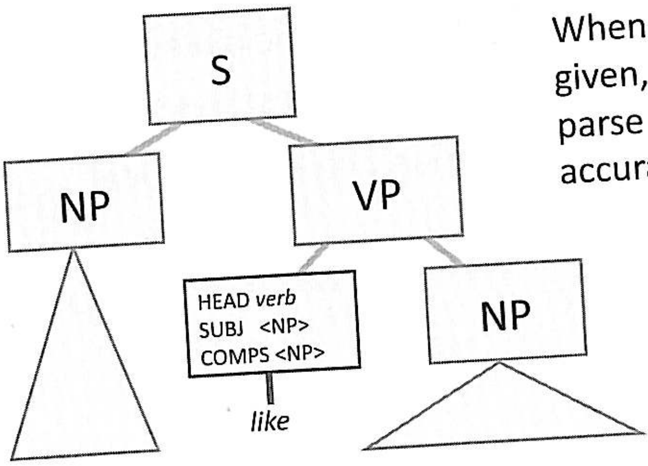
Supertagging

- Supertag = lexical category, lexical entry
- Supertagging = assign supertags to each word without parsing



Supertagging is "almost parsing"

- When a supertag is determined, the structure that will be constructed is almost determined
- Supertagging greatly reduces the search space → boosts parsing speed and accuracy



When gold supertags are given, random choice from a parse forest achieves >95% accuracy

Machine learning for supertagging

- Supertagging is a sequence labeling task

→ Machine learning methods can be applied
  - Log-linear models, perceptron, etc.

- Simple machine learning works: in many cases, supertags can be determined by local contexts

Likely to be an object-control verb

...	man	forced	his	friend	to ...
...	NN	VBD	PRP$	NN	TO ...

Effect of supertagging

- Experiments on HPSG parsing
- Evaluation metrics:
  - Labeled accuracy of predicate argument relations
  - Average parsing time per sentence

prec recall

	LP(%)	LR(%)	F1(%)	Avg. time
Chart parsing w/o supertagging	84.96	84.25	84.60	674ms/sent.
Chart parsing w/ supertagging	87.35	86.29	86.81	183ms/sent.
Supertagging + CFG filtering	86.90	86.71	86.80	19ms/sent.

Probabilistic grammars

Generative models: $P(w, T)$

- Joint distribution over all strings w and trees T
- Use Bayes Rule: $\text{argmax}_T P(T \mid w) \propto argmax_T\ P(w,T)$
- Advantage: easy to estimate (rel. frequencies)
- Disadvantages: difficult to capture complex features

Discriminative models: $P(T|w)$

- Use loglinear models to define distributions $P(T|w)$
- Advantage: can use complex features
- Disadvantage: more difficult to train

139

Generative models for expressive grammars

TAG/CCG:

- Very similar to (lexicalized) probabilistic CFGs
- Lexical entries are treated as atomic units.
  Since coindexation/reentrancies are properties of lexical elements (TAG: trees with traces; CCG: categories with coindexation), this does not cause any problems for generative models

LFG/HPSG:

- Reentrancies in feature structures cannot be modeled with generative models (Abney 2000)
- LFG: can use any (P)CFG parser for c-structure alone

140

Discriminative models

Probability $p(T)$ of parse tree T given sentence $\mathbf{w}$

parameter vector (feature weights)

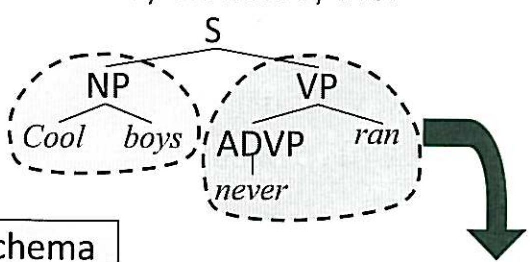
$$p(T \mid \mathbf{w}) \propto \exp\big(\boldsymbol{\lambda} \cdot \mathbf{f}(T)\big)$$

feature vector

Non-probabilistic models can also been applied
- SVM, averaged perceptron, etc.
- Sufficient for choosing the best parse

Design of features

- Feature engineering is essential for high accuracy
- Features should capture syntactic/semantic characteristics of structures
  - Syntactic categories, lexical heads, POSs, constituent size, distance, etc.



lexical entry

schema　　category　　POS

<subj-head, 2, VP, ran, VBD, V_intrans-past, 2, NP, boys, NNS, N_plural, 2>

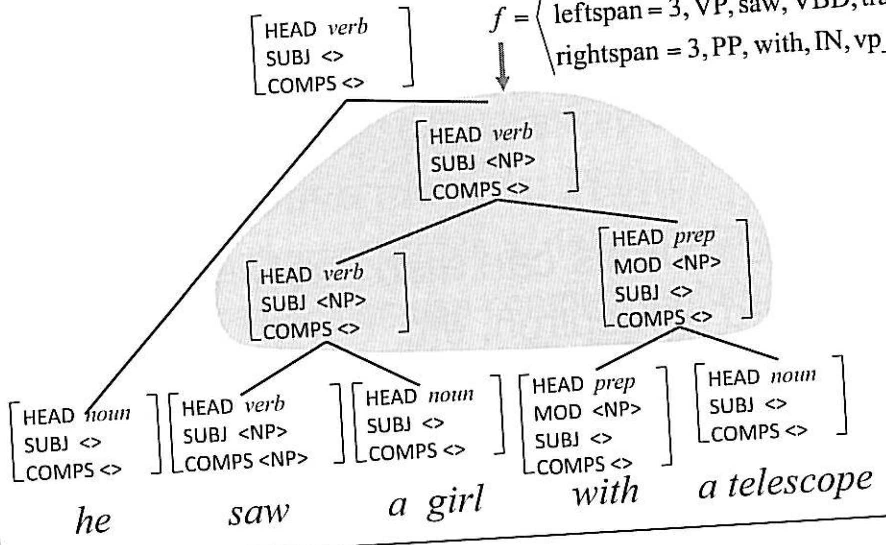
distance　　word　　span

between head words

Example: syntactic features

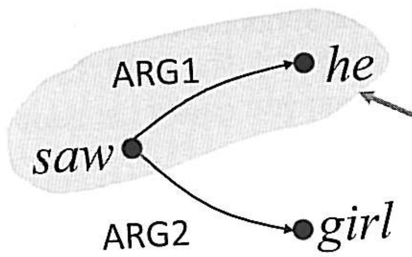
Features for the Head-Modifier construction for *"saw a girl"* and *"with a telescope"*

$$f = \left\langle \begin{array}{l} \text{head\_modifier\_schema, distance} = 3, \\ \text{leftspan} = 3, \text{VP, saw, VBD, transitive\_verb}, \\ \text{rightspan} = 3, \text{PP, with, IN, vp\_modifying\_prep} \end{array} \right\rangle$$



Example: semantic features

Features for the predicate argument relation between *"he"* and *"saw"*



$$f = \left\langle \begin{array}{l} \text{label} = \text{ARG1}, \text{distance} = 1, \\ \text{saw, VBD, transitive\_verb}, \\ \text{he, PRP, pronoun} \end{array} \right\rangle$$

Long distance dependencies

TAG, CCG, HPSG:
The lexicon captures long-distance dependencies

– TAG, HPSG: LDDs require different lexical entries
⇒ Supertagging is crucial

... do you like ...

HEAD *verb*	HEAD *verb*
SUBJ <NP>	SUBJ <NP>
COMPS <NP>	COMPS < >
SLASH < >	SLASH <NP>

Which lexical entry should be assigned?

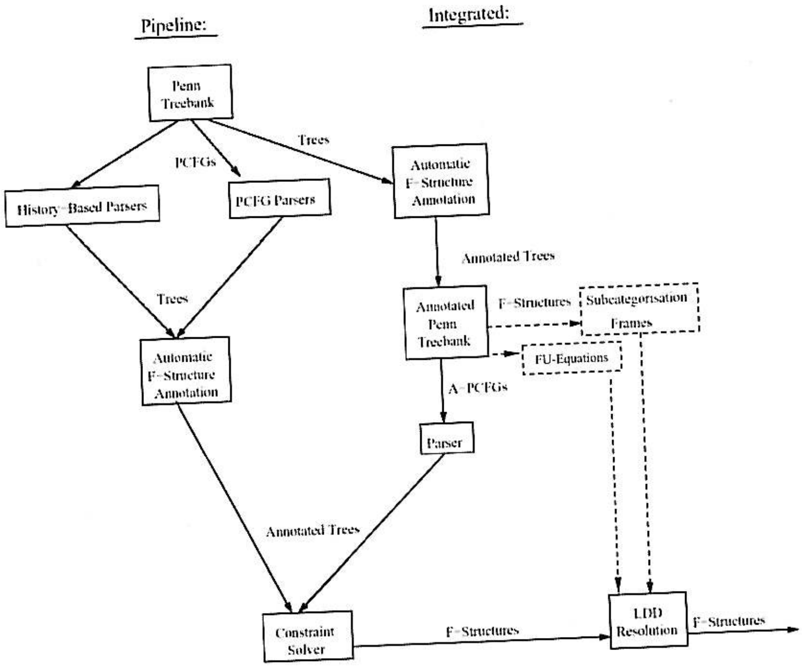
PARSING WITH LFGS

Basic architectures

LFG has **two levels of representation**
- c-structure
- f-structure

- **Pipeline architecture**:
  - Strategy: c-structure first, then f-structure
  - Advantage: existing PCFG parsers can be used
- **Integrated architecture**:
  - Strategy: compute both structures at the same time
  - Advantage: c-/f-structures may effectively constrain ungrammatical structures during parsing

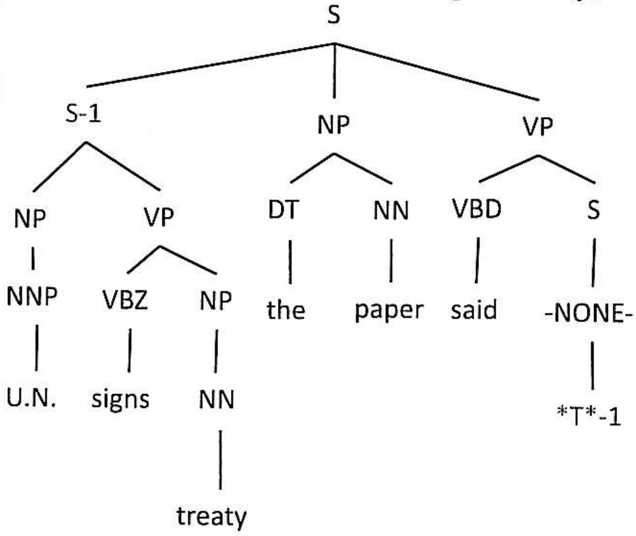
LFG Parsing Architectures



Parsing: LFG and LDD Resolution

- Penn-II tree: traces and co-indexation for LDDs

"U.N. signs treaty, the paper said"



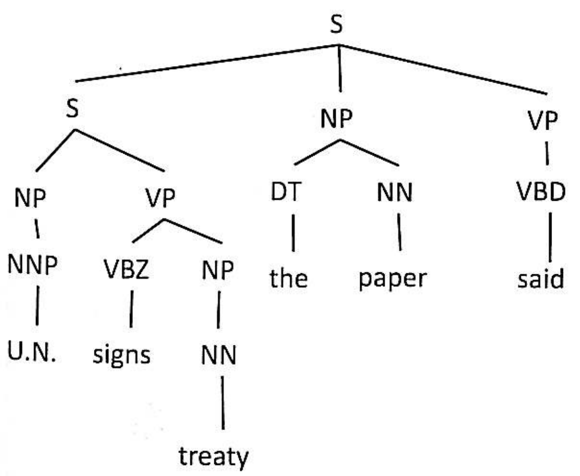
$$\begin{bmatrix} \text{TOPIC} & \begin{bmatrix} \text{SUBJ} & [\text{PRED} \ \text{U.N.}] \\ \text{PRED} & \text{sign} \\ \text{OBJ} & [\text{PRED} \ \text{treaty}] \end{bmatrix} \boxed{1} \\ \text{SUBJ} & \begin{bmatrix} \text{SPEC} & \text{the} \\ \text{PRED} & \text{headline} \end{bmatrix} \\ \text{PRED} & \text{say} \\ \text{COMP} & \boxed{1} \end{bmatrix}$$

Proper f-structure

Parsing: LFG and LDD Resolution

- "PCFG" Parse tree without traces:

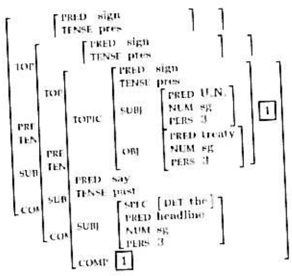
"U.N. signs treaty, the paper said"



$$\begin{bmatrix} \text{TOPIC} & \begin{bmatrix} \text{SUBJ} & [\text{PRED} \ \text{U.N.}] \\ \text{PRED} & \text{sign} \\ \text{OBJ} & [\text{PRED} \ \text{treaty}] \end{bmatrix} \\ \text{SUBJ} & \begin{bmatrix} \text{SPEC} & \text{the} \\ \text{PRED} & \text{headline} \end{bmatrix} \\ \text{PRED} & \text{say} \end{bmatrix}$$

Proto f-structure

Parsing: LFG and LDD Resolution



- Require:
  - functional uncertainty equations
  - subcat frames
- How? From f-str annotated Penn-II …
- Previous Example:
  - ↑TOPIC = ↑COMP*COMP (search along a path of 0 or more comps)
  - say<SUBJ,COMP>

$$\begin{bmatrix} \text{TOPIC} & \begin{bmatrix} \text{SUBJ} & \begin{bmatrix} \text{PRED} & \text{U.N.} \end{bmatrix} \\ \text{PRED} & \text{sign} \\ \text{OBJ} & \begin{bmatrix} \text{PRED} & \text{treaty} \end{bmatrix} \end{bmatrix} \\ \text{SUBJ} & \begin{bmatrix} \text{SPEC} & \text{the} \\ \text{PRED} & \text{headline} \end{bmatrix} \\ \text{PRED} & \text{say} \end{bmatrix} \qquad \begin{bmatrix} \text{TOPIC} & \begin{bmatrix} \text{SUBJ} & \begin{bmatrix} \text{PRED} & \text{U.N.} \end{bmatrix} \\ \text{PRED} & \text{sign} \\ \text{OBJ} & \begin{bmatrix} \text{PRED} & \text{treaty} \end{bmatrix} \end{bmatrix} \boxed{1} \\ \text{SUBJ} & \begin{bmatrix} \text{SPEC} & \text{the} \\ \text{PRED} & \text{headline} \end{bmatrix} \\ \text{PRED} & \text{say} \\ \text{COMP} & \boxed{1} \end{bmatrix}$$

Parsing: LFG and LDD Resolution

- Previous Example:
  - ↑TOPIC = ↑COMP*COMP
  - say<SUBJ,COMP>

$$\begin{bmatrix} \text{TOPIC} & \begin{bmatrix} \text{SUBJ} & \begin{bmatrix} \text{PRED} & \text{U.N.} \end{bmatrix} \\ \text{PRED} & \text{sign} \\ \text{OBJ} & \begin{bmatrix} \text{PRED} & \text{treaty} \end{bmatrix} \end{bmatrix} \\ \text{SUBJ} & \begin{bmatrix} \text{SPEC} & \text{the} \\ \text{PRED} & \text{headline} \end{bmatrix} \\ \text{PRED} & \text{say} \\ & \end{bmatrix}$$

EFFICIENCY AND ACCURACY

Efficiency and accuracy

Is parsing with expressive grammars slow?

- It was very slow more than ten years ago
- Various techniques have been proposed (details omitted)
  - Supertagging
  - Beam search techniques: iterative, global thresholding
  - CFG filtering
- Latest systems are faster than shallow parsers

Which parser is more accurate?

- How to compare parsing accuracy of different parsers?

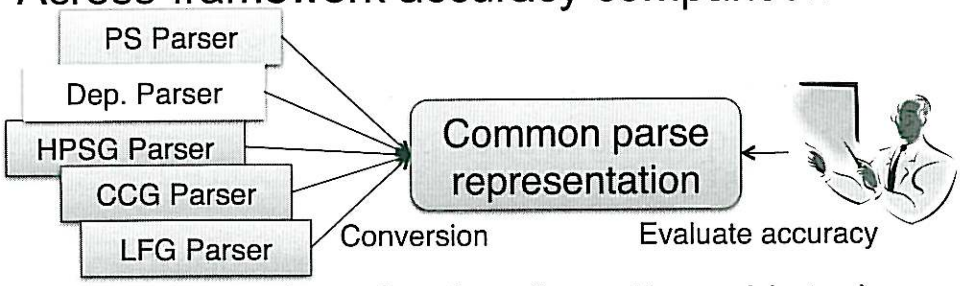
Efficiency comparison

Parser	Framework	Speed
MST parser	dependency	4.5 sent/sec
Sagae's parser	dependency	21.6 sent/sec
Berkeley parser	CFG	4.7 sent/sec
Charniak's parser	CFG	2.2 sent/sec
Charniak's parser + reranker	CFG	1.9 sent/sec
Enju parser	HPSG	2.6 sent/sec
Fast Enju parser	HPSG	18.9 sent/sec

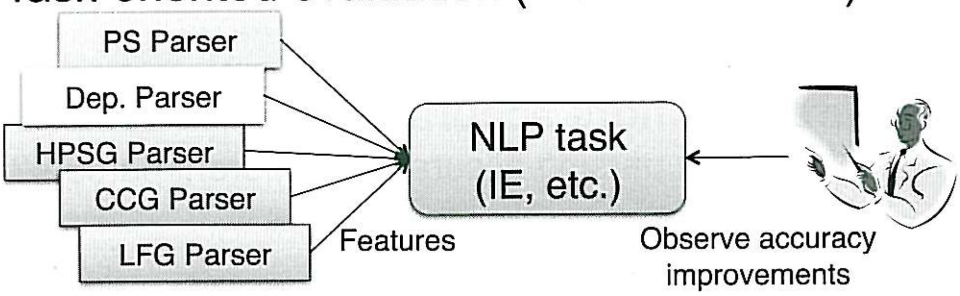
CCG 40 sent/sec

Accuracy evaluation

- Across-framework accuracy comparison



- Task-oriented evaluation (mentioned later)



Across-framework accuracy comparison

- How do treebank-based constraint grammars/ parsers compare to deep hand-crafted grammars/ parsers like XLE and RASP? LFG

- How do treebank–based CCG, LFG and HPSG compare with each other?

(Joint work with Aoife Cahil and Grzegorz Chrupala)

Parsers and data

- **Parsers**
  - Treebank-based LFG, CCG, HPSG parsers
  - RASP (version 2) (Briscoe & Carroll 2006)
  - XLE (Riezler et al. 2002, Kaplan et al. 2004)

- **Data**
  - PARC 700 Dependency Bank gold standard (King et al. 2003), Penn-II Section 23-based
  - DepBank (Briscoe & Carroll 2006) reannotated version of PARC 700 with CBS 500–style GRs
  - CBS 500 Dependency Bank gold standard (Carroll, Briscoe and Sanfillippo 1999), Susanne-based

Cross Comparison

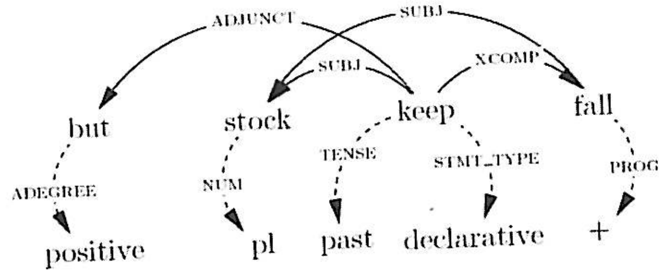


Fig. 7. PARC700 dependencies for *But stocks kept falling*. Non-PRED dependencies are indicated by dashed edges.

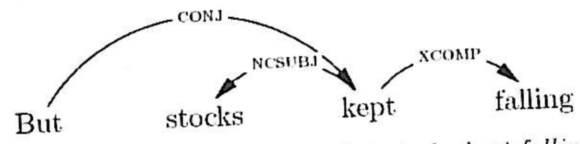


Fig. 8. DepBank dependencies for *But stocks kept falling*

159

Cross Comparison

- Lots of pain points:
  - Different tokenisation Penn-II and PARC700 and DepBank
  - Punctuation changed in DepBank => strings =/= Penn-II
  - Different labels
  - Different analyses
  - Different granularity
  - Lots of fun
- Mapping ....

160

Treebank-Based LFG, CCG and HPSG

Dependency Evaluation Results against DepBank:

	Micro-average P	Micro-average R	Micro-average F	Macro-average P	Macro-average R	Macro-average F
LFG	84.29	80.11	82.15	69.26	62.45	65.68
HPSG-Enju	83.57	81.73	82.64	77.87	71.10	74.33
CCG-C&C	82.44	81.28	81.86	65.61	63.28	64.43
RASP-(v2)	77.66	74.98	76.29	61.12	63.77	62.94

Table 1: Results of LFG parsing resources against DepBank

	Micro-average P	Micro-average R	Micro-average F	Macro-average P	Macro-average R	Macro-average F
LFG	86.06	83.96	85.00	71.42	64.62	67.85
HPSG-Enju	87.49	86.79	87.14	81.19	75.70	78.35
CCG-C&C	86.86	82.75	84.76	71.73	65.85	68.67

Table 2: Upper Bound results of deep parsing resources against DepBank

GOLD TREES

Comparison against XLE and RASP

Labelled dependency f-scores

(Burke et al. 2004, Cahill et al. 2008):

<u>PARC 700</u>

- 80.55% XLE
- 82.73% DCU-LFG (+2.18%)
- 84.00% DCU-LFG now (+3.45%)

<u>CBS 500</u>

- 76.57% RASP
- 80.23% DCU-LFG (+3.66%)

Results statistically significant at ≥ 95% level (Noreen 1989)

VIII. APPLICATIONS

Applications of expressive grammars

- Parsing with expressive grammars is robust, accurate, ready to be applied to real-world problems
- Expressive grammars have shown competitive or state-of-the-art performance in several NLP tasks
  - Sentence realization (generation)
    - Grammars are necessary to bridge semantic representation to its sentence realization
  - Information extraction
    - Predicate argument relations are used like dependencies, with deeper information
  - Machine translation
    - Expressive syntactic/semantic structures are effectively combined with statistical MT

Generation

Sentence Realization

Sentence realization (generation):
Semantic representation → sentence

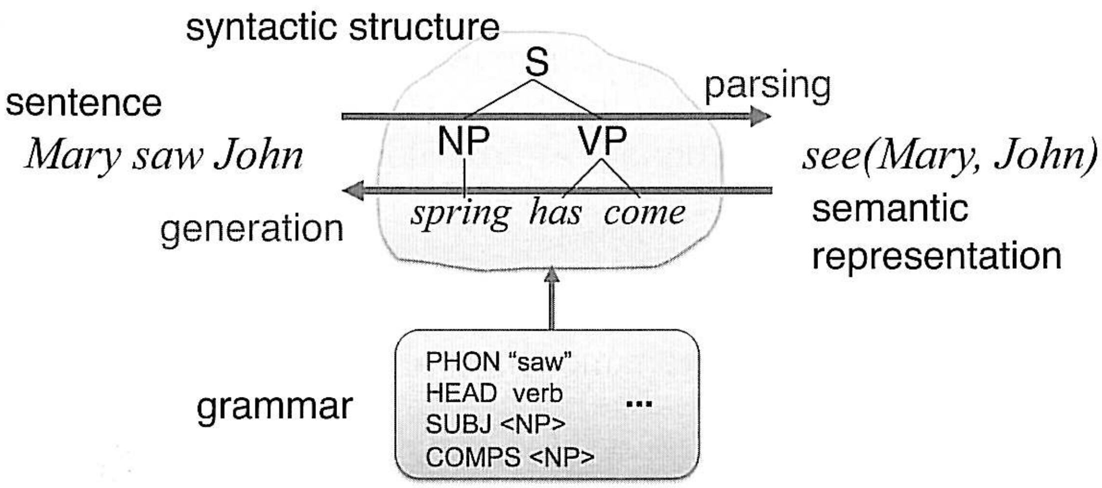


Chart generation

- Chart parsing → chart generation
- Many parsing techniques can be applied to generation
  - Supertagging (*hypertagging*)
  - Beam search

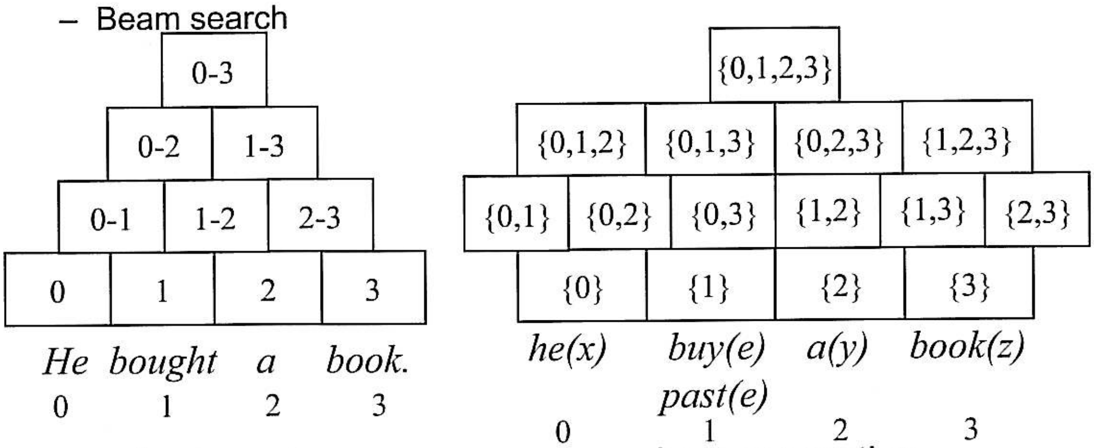


chart parsing

chart generation

LFG Generation

Two architectures for generation from f-structures:

- Chart & Rule-Based Generation: use f-structure annotated CFG rules from Integrated Parsing Architecture + chart generator + probabilities conditioned on input f-structure (!)

- Dependency-Based Generation: linearize dependencies directly by learning n-gram models over dependencies (NOT strings)!

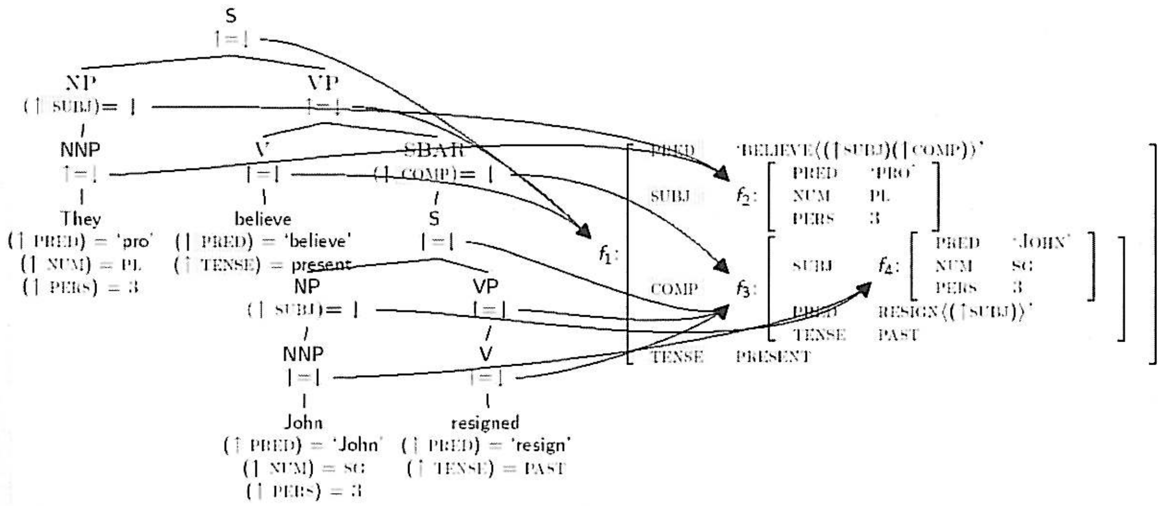
LFG Generation: Chart & F-Str. Annotated Rule-Based

Probability Model

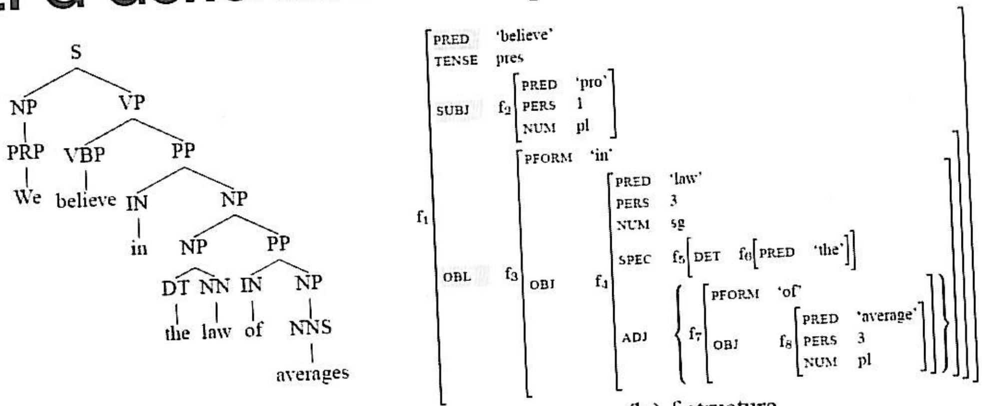
$$\text{argmax}_{Tree} P(Tree|F\text{-}Str)$$

$$P(Tree|F\text{-}Str) := \prod_{\substack{X \to Y \text{ in } Tree \\ \phi(X) = Feats}} P(X \to Y | X.\,Feats) \qquad (1)$$

LFG Generation



LFG Generation: Dependency-Based



(a.) c-structure

(b.) f-structure

string position	We 1	believe 2	in 3	the 4	law 5	of 6	averages 7
f_1	SUBJ	PRED	OBL				
f_3			PFORM		OBJ		
f_4				SPEC	PRED	ADJ	
f_7						PFORM	OBJ

(c.) linearised grammatical functions / bilexical dependencies

Figure 1: C- and f-structures for the sentence *We believe in the law of averages.*

LFG Generation: Dependency-Based

$$P(GF_1^m) = P(GF_1 ... GF_m) = \prod_{k=1}^{m} P(GF_k | GF_{k-n+1}^{k-1}) \quad (1)$$

Model	N-grams			Cond.
basic (P)	SPEC	PRED	ADJ	OBL 'law'
gf (P^g)	SPEC	PRED	ADJ	
pred (P^p)	SPEC	PRED	ADJ	
lex (P^l)	SPEC	PRED['law']	ADJ['of']	

Results

Table 6. *Cross system comparison of results for English WSJ section 23*

System	Coverage	Complete	ExMatch	BLEU	SSA
Callaway (2003)	98.7%		49.0%		0.8884
Langkilde (2002)[a]	82.7%		28.2%	0.757	0.696
Nakanishi et al. (2005)[b]	90.75%			0.7733	
Cahill and van Genabith (2006)	98.05%	89.49%		0.6651	0.6808
Hogan et al. (2007)	99.96%			0.6882	0.7092
Rajkumar et al. (2009)	94.8%	85.04%	33.74%	0.8173	
White and Rajkumar (2009)	97.06%	83.88%	40.45%	0.8506	
Guo et al. (2008)	100%	100%	19.83%	0.7440	0.7534
This article LFG	100%	100%	31.54%	0.8065	0.7871
This article CoNLL	100%	100%	47.76%	0.8820	0.8596

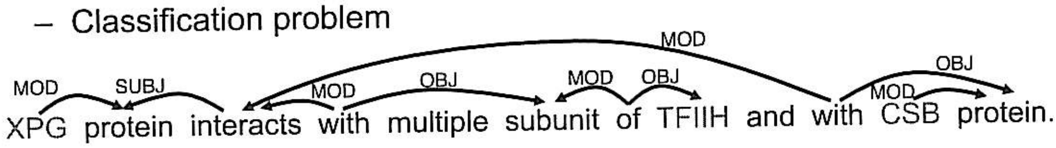
dep. model

[a] The results are for the "permute, no dir" type experiment in Langkilde (2002), where the inputs are most comparable to our f-structures in regard to the level of specification.
[b] The results are for sentences with a length limitation of 20 words.

INFORMATION EXTRACTION

Relation extraction

- Extracting relations expressed in texts
  - Protein-protein interactions
  - Gene-disease associations
  - Network of biological reactions (BioNLP'09 shared task)
- Train a machine learning classifier using parser output as features
  - Classification problem



<XPG, CSB>
<XPG, TFIIH> } positive
<TFIIH, CSB> } negative

175

BioNLP'09 shared task

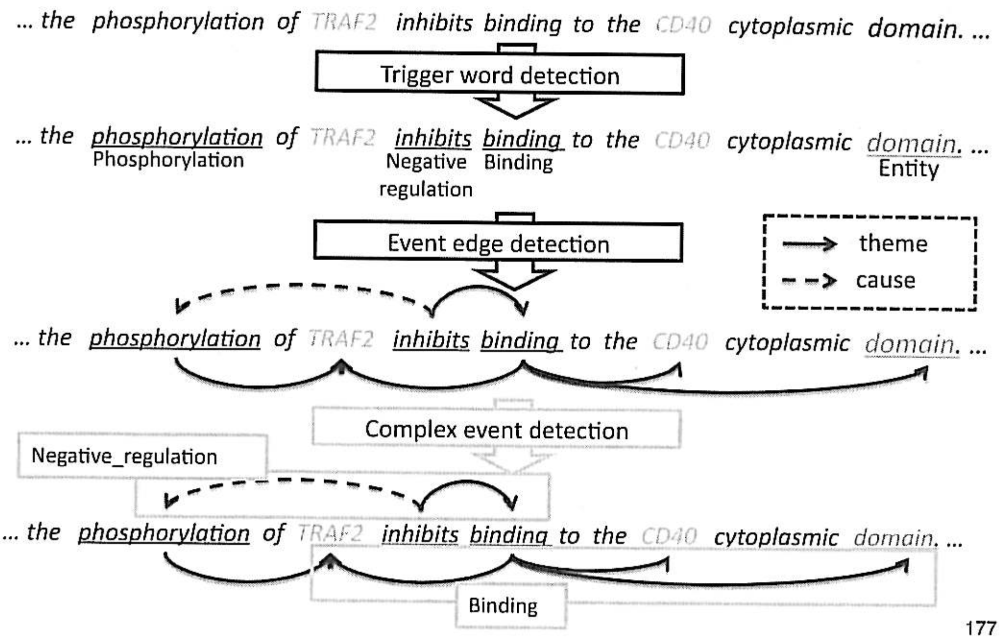
- Finding biological events from abstracts
  - Protein annotations are given

> *... In this study we hypothesized that the phosphorylation of TRAF2 inhibits binding to the CD40 cytoplasmic domain. ...*

negative_regulation
CAUSE:
  phosphorylation
    THEME: *TRAF2*
THEME:
  binding
    THEME: *TRAF2*
    THEME2: *CD40*
    SITE2: *cytoplasmic domain*

176

Event extraction system



177

Event extraction system

- Event extraction by three modules
  - Trigger word detection
  - Event edge detection
  - Complex event detection
- Each module is a linear SVM with features on parsing output
  - Shortest dependency paths
  - Dependent/argument words
- Evaluate contributions from parsers and parse representation formats

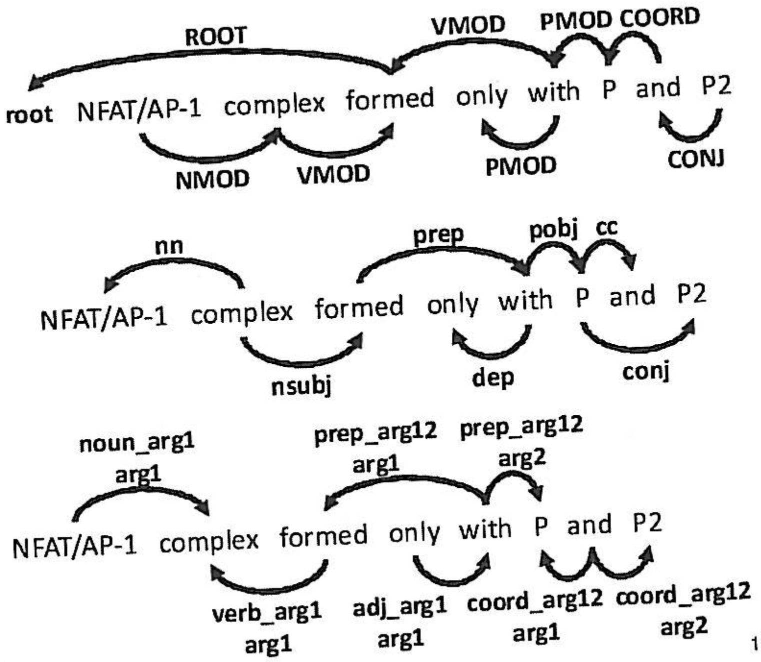
178

Parsers & Formats

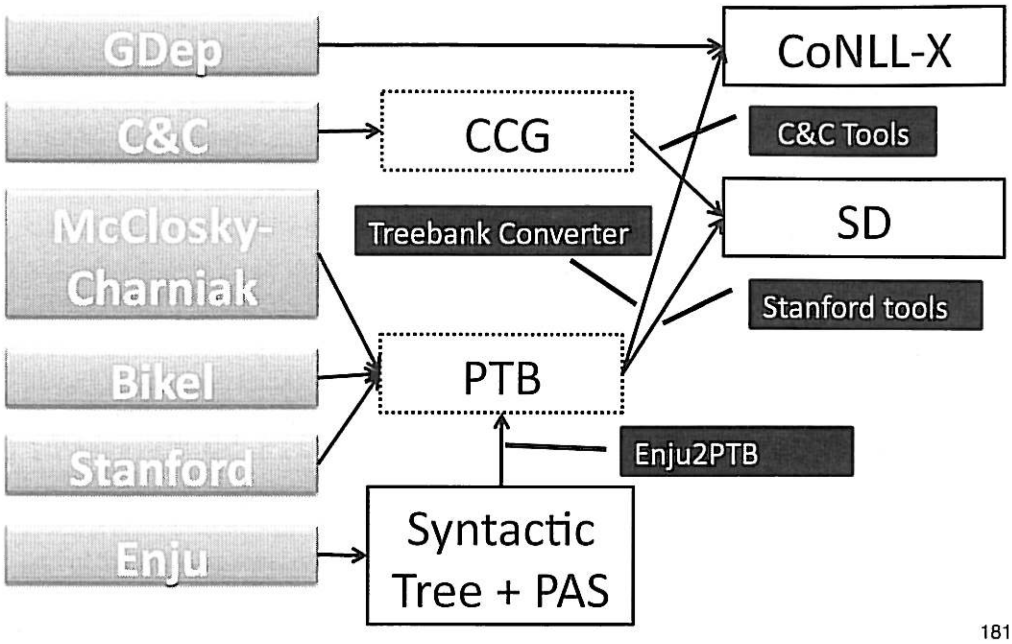
- Dependency parser
  - Gdep
- Phrase structure parsers
  - Stanford parser
  - McClosky's self-trained parser (MC)
- Deep parser
  - C&C parser
  - Enju
- Parse representation formats
  - CoNLL-X
  - Stanford dependency (SD)
  - Predicate Argument Structure (PAS)

Parse representation formats

- CoNLL-X
- Stanford (SD)
- Enju PAS



Format conversion



Results

- Parsers always help
- GDep, MC, C&C, and Enju are comparable
- Best results are close to results with gold parses

	SD	CoNLL	PAS
No parse		51.05	
GDep		55.70	
Stanford	55.02	53.66	
MC	55.60	56.01	
C&C	56.09		
Enju	55.48	55.74	56.57
Gold parse	56.34	56.09	57.94

Parser combination

- Combination helps in most cases
  - Different parsers/formats help a lot

	C&C SD	MC CoNLL	Enju CoNLL
MC CoNLL	57.44 (+1.35)		
Enju CoNLL	56.47 (+0.38)	56.24 (+0.23)	
Enju PAS	57.20 (+0.63)	57.78 (+1.21)	56.59 (+0.02)

For more details, refer to Miwa et al. (COLING 2010)

Search engine for biomedical papers

- NLP tools are applied to 19 million abstracts in MEDLINE
  - HPSG parsing
  - Term recognition (proteins, diseases, etc.)
  - Event expression recognition

- HPSG parsing allows us to search for predicate argument relations rather than cooccurrences
  → improves precision

Search for predicate argument relations

- *"p53 activates something"*

object subject

In this report, we demonstrated that human AMID gene promoter was activated by p53 in reporter gene assays.

The p53 protein integrates multiple upstream signals and functions as a tumor suppressor by activating distinct downstream genes.

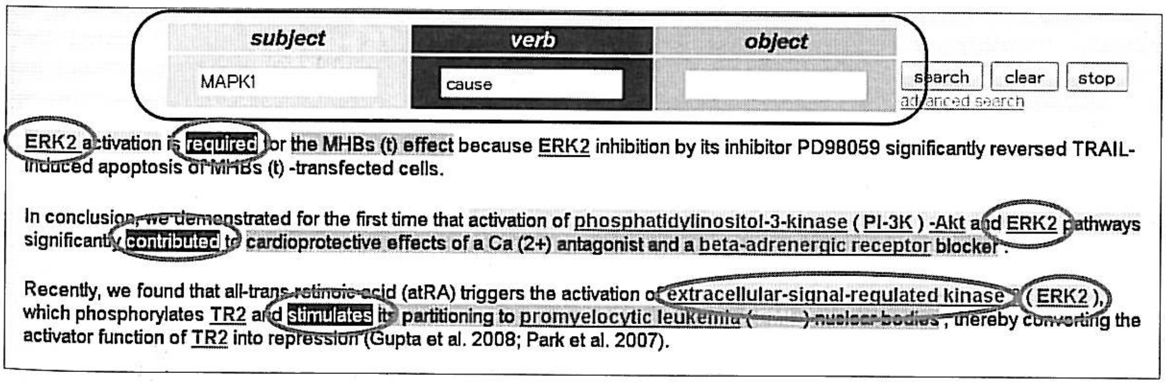
Although p53 has been shown to directly activate transcriptional bax gene and to inhibit expression of bcl-2 gene during radiation-induced apoptosis, it is poorly understood how the Bcl-2 family changes in p53-deficient cells during radiation-induced apoptosis.

Since p21 is known to be transcriptionally activated by p53, these results suggest that TS downregulation of p21 may be occurring through a p53-independent mechanism in this in vitro cell system.

The DDATHF-stabilized p53 bound to the p21 promoter in vitro and in vivo but did not activate histone acetylation over the p53 binding sites in the p21 promoter that is an integral part of the transcriptional response mediated by the DNA damage pathway.

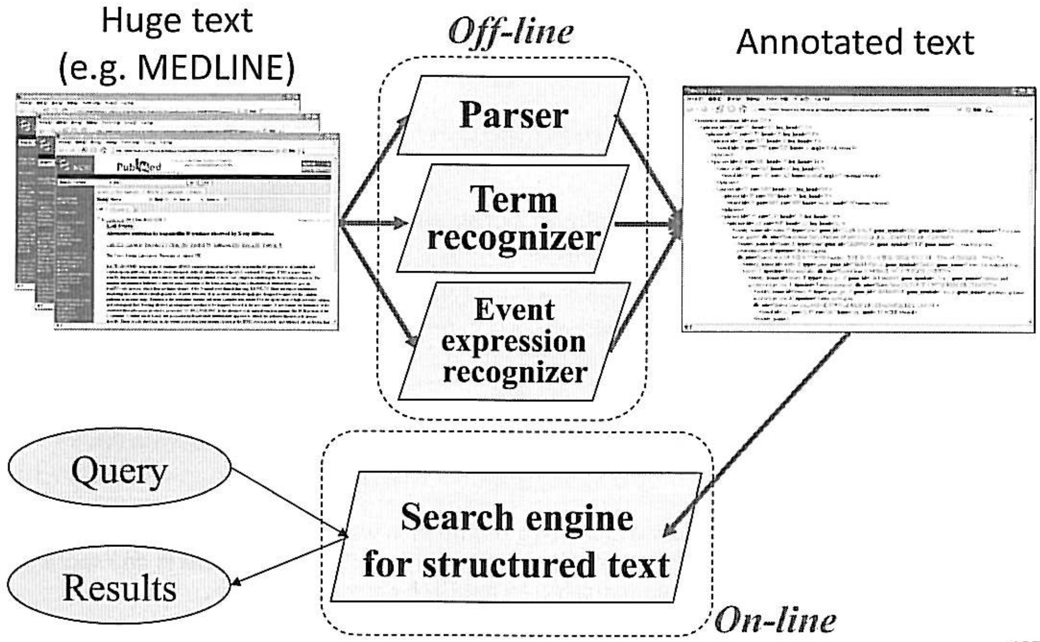
MEDIE

- Subject/predicate/object specification is matched with predicate argument structures
- Synonymous term/event expressions are matched



Publicly available at: http://www-tsujii.is.s.u-tokyo.ac.jp/medie/

System Architecture



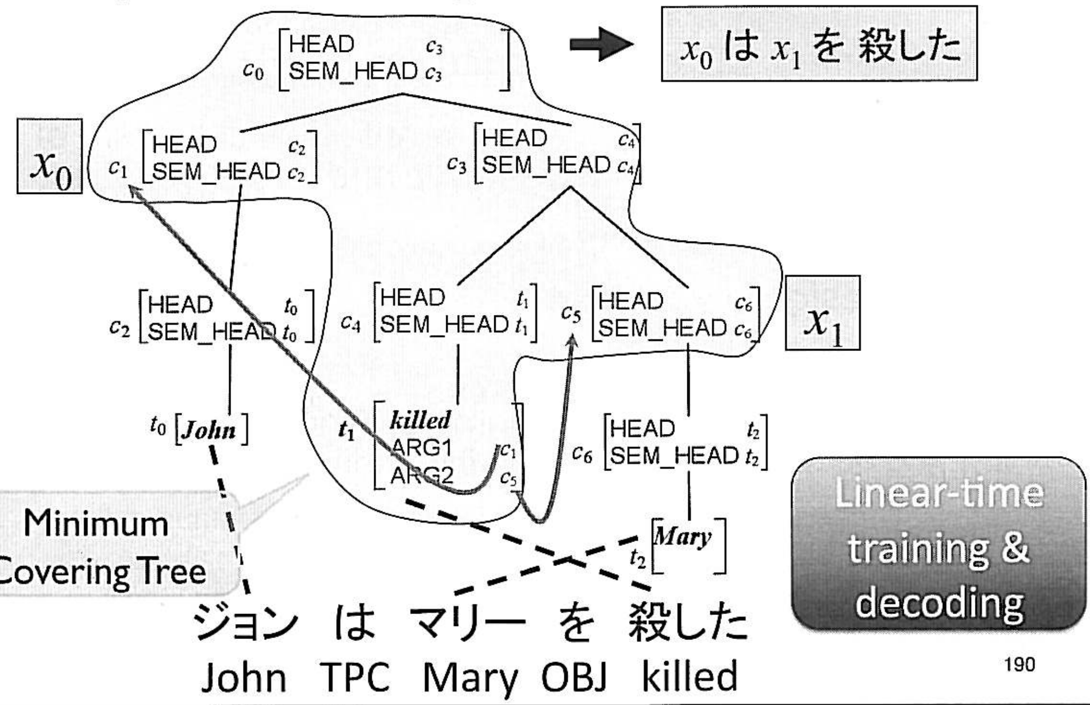
187

MACHINE TRANSLATION

HPSG for syntax-based SMT

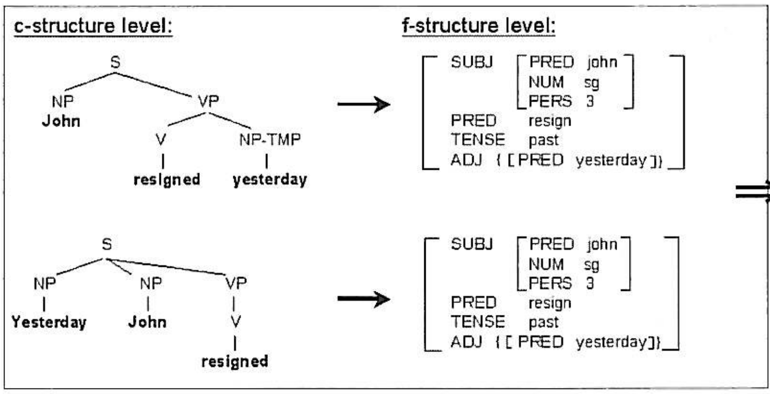
- HPSG works with syntax-aware SMT methods
  - Tree-to-string
  - String-to-tree
  - Forest-to-string
- HPSG structures provide rich syntactic/semantic information as *features*
  - Phrase structure
  - Construction type (i.e. schema name)
  - Syntactic/semantic head
  - Tense, aspect, voice
  - Lexical entry name
  - Predicate argument relations

Extracting translation rules from predicate argument relations



MT evaluation

- Idea: use labelled dependencies for MT evaluation
- Why: dependencies abstract away from some particulars of surface realisation
- Adjunct placement, order of conjuncts in a coordination, topicalisation, ...



⇒

subj(resign, john)
pers(john, 3)
num(john, sg)
tense(resign, past)
adj(resign, yesterday)
pers(yesterday, 3)
num(yesterday, sg)

Dependency-based MT evaluation

- Need a robust parser that can parse MT output ☺
  - Treebank-induced parsers parse (almost) anything ...!
- To make this work, throw in:
  - n-best parsing
  - WordNet synonyms
  - partial matching
  - training weights
- Compare against string-based methods
- Compare (correlation) with human judgement
  - Why: humans not fooled by legitimate syntactic variation

IX. Summary

Conclusions

- Expressive grammars and robust, wide-coverage NLP are not a contradiction:
  - Treebank-based grammar acquisition provides wide coverage
  - Effective statistical parsing methods provide efficient and robust processing
  - These grammars can also be used in other applications, e.g.: IE, generation and MT

X. (PARTIAL) BIBLIOGRAPHY

TAG references

TAG:

A.K. Joshi and Y. Schabes (1996) Tree Adjoining Grammars. In G. Rosenberg and A. Salomaa, Eds., Handbook of Formal Languages

TAG extraction:

F. Xia. *Automatic Grammar Generation From Two Different Perspectives.* PhD thesis, University of Pennsylvania, 2001.

J. Chen, S. Bangalore, K. Vijaj-Shanker. Automated Extraction of Tree-Adjoining Grammars from Treebanks, *Natural Language Engineering*

L. Shen and A.K. Joshi, *Building an LTAG Treebank*, Technical Report MS-CIS-05-15, CIS Department, University of Pennsylvania, 2005

Libin Shen, Lucas Champollion and Aravind K. Joshi *LTAG-spinal and the Treebank: A new resource for incremental, dependency and semantic parsing* Language Resources and Evaluation, Volume 42, Number 1, pp 1-19, March, 2008

TAG references

Supertagging:

S. Bangalore and A.K Joshi. Supertagging: An Approach to almost parsing. Computational Linguistics 1999

Parsing with extracted TAGs:

D. Chiang. Statistical parsing with an automatically extracted tree adjoining grammar. In *Data Oriented Parsing*, CSLI Publications, pages 299–316.

L. Shen and A.K. Joshi. Incremental LTAG parsing, HLT/EMNLP 2005

Libin Shen and Aravind K. Joshi/ TAG Dependency Parsing with Bidirectional Incremental Construction EMNLP 2008

LFG references

Grammar Extraction: English

M. Burke, *Automatic Treebank Annotation for the Acquisition of LFG Resources*, Ph.D. Thesis, School of Computing, Dublin City University, Dublin 9, Ireland. 2005

R. O'Donovan, *Automatic Extraction of Large-Scale Multilingual Lexical Resources*, Ph.D. Thesis, School of Computing, Dublin City University, Dublin 9, Ireland. 2005

R. O'Donovan, M. Burke, A. Cahill, J. van Genabith and A. Way. *Large-Scale Induction and Evaluation of Lexical Resources from the Penn-II and Penn-III Treebanks*, Computational Linguistics, 2005

A. Cahill, M. Forst, M. Burke, M. McCarthy, R. O'Donovan, C. Rohrer, J. van Genabith and A. Way. *Treebank-Based Acquisition of Multilingual Unification Grammar Resources*; Journal of Research on Language and Computation; Special Issue on "Shared Representations in Multilingual Grammar Engineering", (eds.) E. Bender, D. Flickinger, F. Fouvry and M. Siegel, Kluwer Academic Press, 2005

LFG references

Arabic:

Lamia Tounsi, Mohammed Attia and J. van Genabith. 2009. Parsing Arabic Using Treebank-Based LFG Resources. In LFG09, *Proceedings of the 14th International LFG Conference*, Cambridge, UK, 2009, CSLI Publications, Stanford University, USA, pp. 583-586.

Lamia Tounsi, Mohammed Attia and Josef van Genabith. 2009. Automatic Treebank-Based Acquisition of Arabic LFG Dependency Structures. *Proceedings of the EACL-Workshop on Computational Approaches to Semitic Languages*, March 31, 2009, Athens, Greece, pp.45-52.

Yafa Al-Raheb, Amine Akrout, Josef van Genabith and Joseph Dichy, DCU250 Arabic Dependency Bank: An LFG Gold Standard Resource for the Arabic Penn Treebank. In *Proceedings of the Arabic NLP/MT Conference*, British Computer Society Natural Language Translation Specialist Group, London, 2006, pp. 105-116

LFG references

French:

Natalie Schluter and Josef van Genabith, Dependency Parsing Resources for French: Converting Acquired Lexical Functional Grammar F-Structure Annotations and Parsing F-Structures Directly, in *Nodalida 2009 Conference Proceedings*, 2009, (eds.) Kristiina Jokinen and Eckhard Bick, pp.166-173.

Natalie Schluter and Josef van Genabith, Treebank-Based Acquisition of LFG Parsing Resources for French, Proceedings of the Sixth International Language Resources and Evaluation (LREC'08) , pp.2909-2916, Marrakech, Morocco, May 28-30, 2008, ISBN 2-9517408-4-0

Natalie Schluter and Josef van Genabith. Preparing, Restructuring and Augmenting a French Treebank: Lexicalised Parsing or Coherent Treebanks? In *Proceedings of the 10th Conference of the Pacific Association of Computational Linguistics* PACLING 2007, Melbourne Australia

LFG references

Chinese:

Yuqing Guo, Haifeng Wang and Josef van Genabith, 2008. Accurate and Robust LFG-Based Generation for Chinese. In *Proceedings of the 5th International Natural Language Generation Conference (INLG 08)*, pages 86-94. June 12-14, 2008, Ohio, USA.

Yuqing Guo, Josef van Genabith and Haifeng Wang: Acquisition of Wide-Coverage, Robust, Probabilistic Lexical-Functional Grammar Resources for Chinese. In *Proceedings of the 12th International Conference on Lexical Functional Grammar*, July 28-30, 2007, Stanford, CA

Yuqing Guo, Haifeng Wang and Josef van Genabith, Recovering Non-Local Dependencies for Chinese, in *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Natural Language Learning* (EMNLP-CoNLL 2007), Prague, Czeck Republic. pp.257-266

LFG references

Parsing:

Aoife Cahill, Michael Burke, Ruth O'Donovan, Stefan Riezler, Josef van Genabith and Andy Way. 2008. *Wide-Coverage Deep Statistical Parsing using Automatic Dependency Structure Annotation* in Computational Linguistics, Vol. 34, No. 1, pages 81-12

Aoife Cahill, Michael Burke, Ruth O'Donovan, Josef van Genabith, and Andy Way. Long-Distance Dependency Resolution in Automatically Acquired Wide-Coverage PCFG-Based LFG Approximations. In *ACL-04, Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*, July 21-26 2004, Barcelona, Spain, pages 320-327

Aoife Cahill, Mairéad McCarthy, Josef van Genabith and Andy Way. 2002. Parsing with PCFGs and Automatic F-Structure Annotation, In M. Butt and T. Holloway-King (eds.): *Proceedings of the Seventh International Conference on LFG CSLI Publications, Stanford, CA.*, pp.76--95

LFG references

Loglinear models:

Stefan Riezler, Tracy H. King, Ronald M. Kaplan, Richard Crouch, John T. Maxwell, Mark Johnson (2002) *Parsing the Wall Street Journal using a Lexical-Functional Grammar and Discriminative Estimation Techniques.* In ACL'02

Ron Kaplan, Stefan Riezler, Tracy King, John Maxwell, Alexander Vasserman, Richard Crouch (2004) *Speed and Accuracy in Shallow and Deep Stochastic Parsing.* In HLT'04

Geman, S. and M. Johnson (2002) *"Dynamic programming for parsing and estimation of stochastic unification-based grammars".* In ACL'02.

LFG references

Generation:

Yuqing Guo, Josef van Genabith and Haifeng Wang, 2008. Dependency-Based N-Gram Models for General Purpose Sentence Realisation. In *Proceedings of the 22th International Conference on Computational Linguistics (COLING 2008),* pages 297-304. 18-22 August 2008, Manchester, UK.

Deirdre Hogan, Conor Cafferkey, Aoife Cahill and Josef van Genabith, Exploiting Multi-Word Units in History-Based Probabilistic Generation, in *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Natural Language Learning* (EMNLP-CoNLL 2007), Prague, Czeck Republic. pp.267-276

Aoife Cahill and Josef van Genabith. Robust PCFG-Based Generation using Automatically Acquired LFG Approximations, In *COLING-ACL'06, Proceedings of the Joint Conference of the International Committee on Computational Linguistics and the Association for Computational Linguistics 2006*, Sydney, Australia, pages 1033-1040, ,ISBN 1-932432-65-5

LFG references

Question Bank

John Judge, Aoife Cahill and Josef van Genabith. QuestionBank: Creating a Corpus of Parse-Annotated Questions, In *COLING-ACL'06, Proceedings of the Joint Conference of the International Committee on Computational Linguistics and the Association for Computational Linguistics 2006*, Sydney, Australia, pages 497-504, ISBN 1-932432-65-5

MT evaluation

Karolina Owczarzak, Josef van Genabith, and Andy Way. 2008. Evaluating Machine Translation with LFG Dependencies. In Machine Translation, Volume 21, Issue 2, pages 95-119

HPSG references

HPSG theory

C. Pollard and I. A. Sag. (1994) *Head-Driven Phrase Structure Grammar.* University of Chicago Press.

I. A. Sag, T. Wasow, E. Bender. (2003) *Syntactic Theory – A Formal Introduction.* CSLI Lecture Notes no. 152. CSLI Publications.

HPSG grammars

A. Copestake and D. Flickinger. (2000) *An open-source grammar development environment and broad coverage English grammar using HPSG.* LREC 2000.

S. Müller and W. Kasper. (2000) *HPSG analysis of German.* Verbmobil: Foundations of Speech-to-Speech Translation, Artificial Intelligence.

M. Siegel and E. Bender. (2002) *Efficient deep processing of Japanese.* Third Workshop on Asian Language Resources and International Standardization

Y. Miyao, T. Ninomiya, and J. Tsujii. (2004) *Corpus-oriented grammar development for acquiring a Head-Driven Phrase Structure Grammar from the Penn Treebank.* IJCNLP 2004.

H. Nakanishi, Y. Miyao and J. Tsujii. (2004) *An empirical investigation of the effect of lexical rules on parsing with a treebank grammar.* TLT 2004.

R. Malouf, G. van Noord. (2004) *Wide coverage parsing with stochastic attribute value grammars.* IJCNLP-04 Workshop on "Beyond Shallow Analyses"

HPSG references

HPSG treebanks

E. M. Bender and D. Flickinger. (2005) *Rapid prototyping of scalable grammars: Towards modularity in extensions to a language-independent core.* IJCNLP 2005.

F. Bond, S. Fujita, C. Hashimoto, K. Kasahara, S. Nariyama, E. Nichols, A. Ohtani, T. Tanaka, and S. Amano. (2004) *The Hinoki Treebank: A treebank for text understanding.* IJCNLP 2004.

S. Oepen, D. Flickinger, and F. Bond. (2004) *Towards holistic grammar engineering and testing — grafting treebank maintenance into the grammar revision cycle.* IJCNLP-04 Workshop on "Beyond Shallow Analyses"

S. Oepen, K. Toutanova, S. Shieber, C. Manning, D. Flickinger, and T. Brants. (2002) *The LinGO Redwoods treebank: motivation and preliminary applications.* COLING 2002.

HPSG references

Supertagging

T. Ninomiya, T. Matsuzaki, Y. Tsuruoka, Y. Miyao and J. Tsujii. (2006) *Extremely lexicalized models for accurate and fast HPSG parsing.* EMNLP 2006.

R. Dridan, V. Kordoni, J. Nicholson. (2008) *Enhancing performance of lexicalised grammars.* ACL-08:HLT.

Y. Zhang, T. Matsuzaki, J. Tsujii. (2010) *A simple approach for HPSG supertagging using dependency information.* NAACL-HLT'10.

HPSG references

Parsing models for HPSG

M. Osborne. (2000) *Estimation of stochastic attribute-value grammar using an informative sample.* COLING 2000.

Y. Miyao and J. Tsujii. (2002) *Maximum entropy estimation for feature forests.* HLT 2002.

Y. Miyao, and J. Tsujii. (2005) *Probabilistic disambiguation models for wide-coverage HPSG parsing.* ACL 2005.

K. Toutanova, C. Manning, D. Flickinger, and S. Oepen (2005) Stochastic HPSG parse disambiguation using the Redwoods corpus. Journal of Research on Language and Computation. 3(1).

K. Sagae, Y. Miyao and J. Tsujii. (2008) *HPSG parsing with shallow dependency constraints.* ACL 2007.

Y. Miyao and J. Tsujii. (2008) *Feature forest models for probabilistic HPSG parsing.* Computational Linguistics. 34(1).

HPSG references

Efficient parsing techniques for HPSG

B. Kiefer, H.-U. Krieger, J. Carroll, and R. Malouf. (1999) *A bag of useful techniques for efficient and robust parsing.* ACL '99.

B. Kiefer and H.-U. Krieger. (2000) *A context-free approximation of Head-Driven Phrase Structure Grammar.* IWPT '00.

T. Ninomiya, Y. Tsuruoka, Y. Miyao, and J. Tsujii. (2005). *Efficacy of Beam Thresholding, Unification Filtering and Hybrid Parsing in Probabilistic HPSG Parsing.* IWPT 2005.

T. Matsuzaki, Y. Miyao and J. Tsujii. (2007) *Efficient HPSG Parsing with Supertagging and CFG-filtering.* IJCAI 2007.

HPSG references

Parser evaluation

Y. Miyao, K. Sagae and J. Tsujii. (2007) *Towards Framework-Independent Evaluation of Deep Linguistic Parsers.* GEAF 2007.

K. Sagae, Y. Miyao, T. Matsuzaki, and J. Tsujii. (2008) *Challenges in Mapping of Syntactic Representations for Framework-Independent Parser Evaluation.* Workshop on Automated Syntatic Annotations for Interoperable Language Resources.

Y. Miyao, R. Sætre, K. Sagae, T. Matsuzaki and J. Tsujii. (2008) *Task-Oriented Evaluation of Syntactic Parsers and Their Representations.* ACL-08:HLT.

Y. Miyao, K. Sagae, R. Sætre, T. Matsuzaki and J. Tsujii. (2009) *Evaluating Contributions of Natural Language Parsers to Protein-Protein Interaction Extraction.* Bioinformatics. 25(3).

HPSG references

Sentence realization

E. Velldal, S. Oepen and D. Flickinger. (2004) *Paraphrasing treebanks for stochastic realization ranking.* TLT 2004.

E. Velldal and S. Oepen. (2005) *Maximum entropy models for realization ranking.* MT-Summit X.

J. Carroll and S. Oepen. (2005) *High efficiency realization for a wide-coverage unification-based grammar.* IJCNLP 2005.

H. Nakanishi, Y. Miyao, and J. Tsujii. (2005). *Probabilistic models for disambiguation of an HPSG-based chart generator.* In Proc. IWPT 2005.

HPSG references

Applications

Y. Miyao, T. Ohta, K. Masuda, Y. Tsuruoka, K. Yoshida, T. Ninomiya, and J. Tsujii. (2006). *Semantic Retrieval for the Accurate Identification of Relational Concepts.* In Proc. COLING-ACL 2006.

M. Miwa, R. Sætre, Y. Miyao, and J. Tsujii. (2009) *Protein-Protein Interaction Extraction by Leveraging Multiple Kernels and Parsers.* International Journal of Medical Informatics. 78(12).

M. Miwa, R. Sætre, J. Kim, and J. Tsujii. (2010) *Event Extraction with Complex Event Classification Using Rich Features.* Journal of Bioinformatics and Computational Biology. 8(1).

X. Wu, T. Matsuzaki, and J. Tsujii. (2010) Fine-grained tree-to-string translation rule extraction. ACL 2010.

CCG references

CCG:

Mark Steedman (2000). *The Syntactic Process,* MIT Press.

Mark Steedman (1996). *Surface Structure and Interpretation* (Linguistic Inquiry Monograph No. 30), MIT Press.

Mark Steedman and Jason Baldridge (forthcoming) *Combinatory Categorial Grammar* to appear in: R. Borsley and K. Borjars (eds.) *Non-Transformational Syntax*, Blackwell

Cem Bozsahin (2002). *The Combinatory Morphemic Lexicon. Computational Linguistics,* 28(2):145-186.

CCG implementations:

OpenCCG: http://openccg.sourceforge.net/

CCG references

CCG grammar extraction:

Julia Hockenmaier (2003) *Data and Models for Statistical Parsing with Combinatory Categorial Grammar*, PhD thesis, University of Edinburgh

Julia Hockenmaier and Mark Steedman (2007). CCGbank: A Corpus of CCG Derivations and Dependency Structures Extracted from the Penn Treebank. In *Computational Linguistics, 33(3) 355-396.*

Julia Hockenmaier (2006). Creating a CCGbank and a wide-coverage CCG lexicon for German. In COLING/ACL 2006, Sydney.

Ruken Cakici (2005). *Automatic Induction of a CCG Grammar for Turkish.* ACL Student Research Workshop, Ann Arbor, MI.

Julia Hockenmaier and Mark Steedman (2005). *CCGbank: User's Manual.* Technical Report MS-CIS-05-09, Department of Computer and Information Science, University of Pennsylvania.

CCG references

CCGbank modifications:

Matthew Honnibal, James R. Curran and Johan Bos (2010). *Rebanking CCGbank for improved NP interpretation.* ACL 2010.

CCG supertagging:

Stephen Clark and James R. Curran (forthcoming). *Supertagging for Efficient Wide-Coverage CCG Parsing.* Book chapter to appear in Complexity of Lexical Descriptions and its Relevance to Natural Language Processing: A Supertagging Approach, Eds. Srinivas Bangalore and Aravind Joshi

Stephen Clark and James R. Curran (2004). *The Importance of Supertagging for Wide-Coverage CCG Parsing* in Proceedings of the 20th International Conference on Computational Linguistics (COLING '04), Geneva, Switzerland

Stephen Clark (2002). *Supertagging for Combinatory Categorial Grammar.* In Proceedings of the 6th International Workshop on Tree Adjoining Grammars and Related Frameworks (TAG+6), pp.19-24, Venice, Italy

CCG references

Statistical CCG parsing:

Stephen Clark and James R. Curran (2007) *Wide-Coverage Efficient Statistical Parsing with CCG and Log-Linear Models.* Computational Linguistics 33(4).

Stephen Clark and James R. Curran (2004). *Parsing the WSJ using CCG and Log-Linear Models.* ACL '04

Stephen Clark and James R. Curran (2003). *Log-Linear Models for Wide-Coverage CCG Parsing.* EMNLP '03

Julia Hockenmaier (2003). *Parsing with Generative Models of Predicate-Argument Structure.* ACL '03.

Julia Hockenmaier and Mark Steedman (2002). *Generative Models for Statistical Parsing with Combinatory Categorial Grammar,* in ACL '02

Stephen Clark, Julia Hockenmaier and Mark Steedman (2002). *Building Deep Dependency Structures with a Wide-Coverage CCG Parser. ACL'02.*

CCG references

Parser evaluation:

Stephen Clark and Julia Hockenmaier (2002). Evaluating a Wide-Coverage CCG Parser, in Proceedings of the LREC 2002 Beyond Parseval Workshop, pp.60-66, Las Palmas, Spain

Stephen Clark; James R. Curran. *Comparing the Accuracy of CCG and Penn Treebank Parsers. ACL 2009*

CCG generation:

Dominic Espinosa; Michael White; Dennis Mehay (2008). *Hypertagging: Supertagging for Surface Realization with CCG*

Michael White; Rajakrishnan Rajkumar (2009). *Perceptron Reranking for CCG Realization. EMNLP 2009*

CCG references

Boxer

Johan Bos, Stephen Clark, Mark Steedman, James R. Curran and Julia Hockenmaier (2004). Wide-Coverage Semantic Representations from a CCG Parser, in *Proceedings of the 20th International Conference on Computational Linguistics (COLING '04)* , Geneva, Switzerland

Johan Bos (2005): Towards Wide-Coverage Semantic Interpretation. In *Proceedings of Sixth International Workshop on Computational Semantics IWCS-6*. Pages 42-5

James R. Curran, Stephen Clark, and Johan Bos (2007): *Linguistically Motivated Large-Scale NLP with C&C and Boxer*. Proceedings of the ACL 2007 Demonstrations Session (ACL-07 demo), pp.33-36.

Johan Bos (2008): Wide-Coverage Semantic Analysis with Boxer. In: J. Bos, R. Delmonte (eds): Semantics in Text Processing. STEP 2008 Conference Proceedings, pp.277-286, Research in Computational Semantics, College Publications

CCG references

Semantic Role Labeling:

Daniel Gildea and Julia Hockenmaier (2003). *Identifying Semantic Roles using Combinatory Categorial Grammar,* in Proceedings of 2003 Conference on Empirical Methods in Natural Language Processing (EMNLP), Sapporo, Japan

Stephen Boxwell; Dennis Mehay; Chris Brew (2009). *Brutus: A Semantic Role Labeling System Incorporating CCG, CFG, and Dependency Features.* ACL 2009

CCG applications:

Brian Harrington and Stephen Clark (2007). *ASKNet: Automated Semantic Knowledge Network in* Proceedings of the Twenty-Second Conference on Artificial Intelligence (AAAI-07) , pp.889-894, Vancouver, Canada.

Stephen Clark, Mark Steedman and James R. Curran (2004). *Object-Extraction and Question-Parsing using CCG,* Proceedings of the SIGDAT Conference on Empirical Methods in Natural Language Processing (EMNLP '04)

CCG references

Machine translation

Alexandra Birch, Miles Osborne and Philipp Koehn (2007). *CCG Supertags in factored translation models.* In SMT Workshop at ACL

Hany Hassan, Khalil Sima'an and Andy Way. (2009) *A syntactified direct translation model with linear-time decoding.* In EMNLP

Hassan, H., K. Sima'an and A. Way. 2008. *A syntactic language model based on incremental CCG parsing.* Second IEEE Spoken Language Technology Workshop

Treebank references

Penn Treebank:

Mitchell P. Marcus, Mary Ann Marcinkiewicz, Beatrice Santorini (1993) *Building a Large Annotated Corpus of English: The Penn Treebank.* In Computational Linguistics

Mitchell Marcus et al. (1994) *The Penn Treebank: Annotating Predicate Argument Structure.* In ARPA Human Language Technology Workshop

Tiger Treebank:

Sabine Brants, Stefanie Dipper, Silvia Hansen, Wolfgang Lezius, George Smith (2002) *The Tiger Treebank.* Workshop on Treebanks and Linguistic Theories

Parc700 and DepBank:

Stefan Riezler, Tracy H. King, Richard Crouch, Mary Dalrymple, Ronald M. Kaplan (2003) *The PARC 700 Dependency Bank.* Workshop on "Linguistically Interpreted Corpora" (LINC'03)

Ted Briscoe and John Carroll. (2006) *Evaluating the accuracy of an unlexicalized statistical parser on the PARC DepBank.* ACL '06.