What extracting grammars from treebanks can tell us about linguistic theory

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

Treebanks and Linguistic Theories 2007
In this talk:

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

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

- Function tags: complement-adjunct distinction
- Co-indexed null elements: long-range dependencies
Penn Treebank Parsing

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

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

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

We can use standard parsing algorithms
The Tiger corpus

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

... is a *lexicalized* grammar formalism

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

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

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

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

- Fully compositional semantics
Combinatory Categorial Grammar

... had no wide-coverage implementation
Treebanks...

... can contain arbitrary text:

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

... can provide arbitrary descriptions:

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

... provide analyses for well-studied constructions

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

... may provide limited expressivity

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

... may require complete analyses

- *Lexicalized* formalisms need lexical entries for every word
Research questions

- Are the descriptions in the treebank sufficient to obtain the analyses stipulated by the linguistic theory?

- Can the linguistic theory account for the descriptions given in the treebank?
Combinatory Categorial Grammar
CCG categories and derivations

- **Atomic categories:**
  NP, S, PP...

- **Complex categories:**
  S\NP, (S\NP)/NP, (NP\NP)/NP, ...

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

\[
\begin{array}{c}
I & \text{drink} & \text{coffee} \\
\hline
\text{NP} & (S\backslash\text{NP})/\text{NP} & \text{NP} \\
\hline
S\backslash\text{NP} & \text{S} \\
\hline
\end{array}
\]
CCG’s combinatory rules

Function application:
\[ X/Y \quad Y \quad \Rightarrow \quad X \]
\[ Y \quad X\backslash Y \quad \Rightarrow \quad X \]

Function composition:
\[ X/Y \quad Y/Z \quad \Rightarrow \quad X/Z \]
\[ Y\backslash Z \quad X\backslash Y \quad \Rightarrow \quad X\backslash Z \]

Type raising:
\[ X \quad \Rightarrow \quad T/(T\backslash X) \]
\[ X \quad \Rightarrow \quad T\backslash(T/X) \]

Coordination:
\[ X \quad \text{conj} \quad X \quad \Rightarrow \quad X \]
Features on categories

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

- In CCGbank, features are atomic.
CCG: syntax-semantics interface

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

- **Function application**
  \[ \frac{X/Y: \lambda x. f(x)}{Y:a} \Rightarrow X:f(a) \]

- **Function composition**
  \[ \frac{X/Y: \lambda x. f(x) \quad X/Y: \lambda x. g(x)}{X/Z: \lambda x. f(gx)} \Rightarrow X/Z: \lambda x. f(gx) \]

- **Type-raising**
  \[ \frac{X: a}{\Gamma: (\Gamma \setminus X): \lambda f. f(a)} \]

\[
\begin{array}{c}
\text{I drink coffee} \\
\text{NP: I'} \\
(\text{S[dcl]} \setminus \text{NP}) / \text{NP: } \lambda x. \lambda y. \text{drink'xy} \\
\text{NP: coffee'} \\
\text{S[dcl]} \setminus \text{NP: } \lambda y. \text{drink'coffee'y} \\
\text{S[dcl]: drink'coffee'I'}
\end{array}
\]
Approximating semantics with dependencies

Every argument of a lexical functor category defines a dependency:

\[
\langle \text{drink}, (S[dcl] \backslash NP_1)/NP_2, 2, \text{coffee} \rangle
\]
Non-local dependencies in CCG

Unified analysis of wh-extraction and right-node raising

\[
\begin{array}{cccc}
\text{coffee} & \text{that} & I & \text{drink} \\
\text{NP} & (\text{NP}\backslash\text{NP})/(\text{S[dcl]}/\text{NP}) & \text{S}/(\text{S}\backslash\text{NP}) & \text{S[dcl]}/\text{NP} \\
\text{NP}\backslash\text{NP} & \text{NP} & \text{NP} & \text{NP}
\end{array}
\]

\[
\begin{array}{cccc}
I & \text{like} & \text{but} & \text{you} & \text{hate} & \text{coffee} \\
\text{S}/(\text{S}\backslash\text{NP}) & \text{S[dcl]}/\text{NP} & \text{conj} & \text{S}/(\text{S}\backslash\text{NP}) & \text{S[dcl]}/\text{NP} & \text{NP} \\
\text{S[dcl]}/\text{NP} & \text{S[dcl]}/\text{NP} & \text{S[dcl]}/\text{NP} & \text{S[dcl]}/\text{NP} & \text{S[dcl]}/\text{NP} & \text{S[dcl]}/\text{NP}
\end{array}
\]
Translating the Penn Treebank to CCG
funds that are or soon will be listed in New York or London.
Out: CCG derivation + dependencies

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

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

SBAR
  WHNP-1
    which

S
  NP-SBJ
    the magazine
  VP
    VBZ
      has
    VBN
    NP
      offered advertisers

NP
  -NONE-
  *T*-1
Dealing with extraction – the final tree

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

(S (NP–SBJ She)
 (VP (VP (VBD applied)
 (PP–CLR (IN for)
 (NP (-NONE- *RNR*–1)))))
 (CC and)
 (VP (VBD won)
 (NP (-NONE- *RNR*–1))
 (NP–1 bonus pay)))
Right-node raising in CCG

She applied for and won bonus pay

NP (S[dcl]\NP)/PP PP/NP conj (S[dcl]\NP)/NP NP

(S[dcl]\NP)/NP

(S[dcl]\NP)/NP

(S[dcl]\NP)/NP

Φ

(S[dcl]\NP)

(S[dcl])
Non-constituent coordination

(VP  (VP  (VB pay)
     (NP  HealthVest)
     (NP-2  $ 5 million)
     (ADV-P-TMP=-3  right away))
(CC  and)
(VP  (NP-2  additional amounts)
     (PP-TMP=-3  in the future))
Proliferation of adjunct categories

Standard CCG leads to a proliferation of categories

\[ \text{used} \quad \text{S[ps]} \backslash \text{NP} \]
\[ \text{used} \quad \text{NP} \backslash \text{NP} \]
\[ \text{used} \quad (\text{NP} \backslash \text{NP}) \backslash (\text{NP} \backslash \text{NP}) \]
\[ \text{used} \quad (\text{S} \backslash \text{NP}) \backslash (\text{S} \backslash \text{NP}) \]
\[ \text{used} \quad (\text{S} \backslash \text{NP}) \backslash (\text{S} \backslash \text{NP})) \backslash (\text{S} \backslash \text{NP}) \backslash (\text{S} \backslash \text{NP})) \]
Type-changing rules for adjuncts

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

- **Clean up Treebank:**
  - POS-tagging errors: wrong categories/verb features
  - Bracketing errors: eg. coordination

- **Change Treebank analysis**
  where it doesn’t conform to CCG analysis, eg.:
  - Insert noun level
  - “Non-constituent” coordination
  - Small clauses
CCGbank: the resulting corpus

- **Translation coverage:** 99.44% of all sentences

- **Size of lexicon and grammar:**
  - Lexicon: 74,669 entries for 44,210 word types
    - 1286 lexical category types, 439 appear only once
  - Grammar: 3262 rule instantiations, 1146 appear once.

- **Linguistically interesting observations:**
  - At least three parasitic gaps
  - More than twice as many RNR instances as Treebank suggests
Lexical coverage on unseen data

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

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

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

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

- We can translate the Penn Treebank into CCG derivations and dependency structures.
- The resulting corpus is publicly available from the Linguistic Data Consortium.
- This allows us (you!) to create statistical CCG parsers.
Translating
the Tiger corpus
to CCG
Statistical parsing for German

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

- This creates problems for surface-dependency models:
  - Dependency models do not help. (Dubey & Keller, 2003)
  - Context-free representations can only approximate
    the underlying dependencies (Levy & Manning, 2004)
### Translation:
1. Mary reads the book that Peter has given her tomorrow.
2. Mary will have read the book that Peter has given her tomorrow.
3. ...that Mary will have read the book that Peter has given her tomorrow.
Scrambling in CCG

dann gibt

Peter Maria das Buch

((S[v2]/NP₁)/NP₂)/NP₃

(S[v2]/NP₁)/NP₂

S[v2]/NP₁

S[v2]

Peter
gibt

Maria
das Buch

((S[v2]/NP₁)/NP₂)/NP₃

T\(T/NP₂)

T\(T/NP₁)

T\((T/NP₁)/NP₂)

S[v2]/NP₃

S[dcl]
A Tiger graph

S

SB
CNP
CJ
IBM
IBM NE

CD
und
and
KON

CJ
Siemens
Siemens NE

gelten
count
VVFIN

HD

NG

AVP
HD

MO

PP

AC

NK

S

gelten
count
VVFIN

nicht
no
PTKNEG

mehr
more
ADV

als
as
APPR

Schimpfworte
swearwords

NN
From Tiger to CCG

S

SB

CNP

IBM
IBM
NE
und
KON

Siemens
Siemens
NE

gelten
count
VVFIN

HD

NG

AVP

MO

PP

und
KON

and

KON

nicht
no
PTKNEG

mehr
more
ADV

als
as
APPR

Schimpfworte
swearwords

NN

S[dcl]/(S[v2]/NP)

S[v2]/NP

S[v2]/NP

(S/NP)(S/NP)

als
Schimpfwörter

gelten

(S/NP)(S/NP)
nicht
mehr

IBM
und
Siemens
Discontinuous constituents
The CCG derivation
Translation coverage

• We translate 46,628 graphs into CCG
  (92.4% of all graphs, 88.4% of all discontinuous graphs)

• Reasons for failure/non-translation:
  – Graph not a sentence: 1.7%
  – Cannot find head: 1.3%
  – Graph cannot be planarized: 1.3%
  – Translation not a CCG derivation: 1.9%
Lexicon size and coverage

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

“den Kritiker gehasst und Fans geliebt haben”
“whom critics have loved and fans have hated”

There is no CCG derivation where the subjects are arguments of the auxiliary!
The problem with subjects

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

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

Zu reparieren **versucht hat** Hans das Auto

to repair tried has Hans the car

Hans has tried to repair the car.
1. dass sie ihm eine Garantie geben müssen wird.
2. dass sie ihm eine Garantie wird geben müssen.
3. dass sie ihm eine Garantie geben wird müssen.
Hans hat versucht das Auto zu reparieren.
Non-local scrambling

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

Tree-Adjoining Grammar can’t derive this sentence:
(Rambow 1994)
This sentence:  TAG adjunction:

\[
\begin{array}{cccccc}
1 & 2 & 3 & 4 & 5 & 6 \\
\hline
& & & & & \\
\end{array}
\]

\[
\begin{array}{cccccc}
1 & 2 & 3 & 4 & 5 \\
\hline
& & & & & \\
\end{array}
\]
Non-local scrambling in CCG

dieses Buch \( \text{hat} \) den Kindern niemand zu geben versucht

\[
\begin{array}{cccccc}
\text{S/(S[v1]/NP}_a) & \text{S[v1]/S[pt]} & \text{NP}_d & \text{NP}_n & (\text{VP}_{zu}/\text{NP}_a)/\text{NP}_d & (\text{S[pt]}/\text{NP}_n)/\text{VP}_{zu} \\
\text{S/(S/NP}_n) & ((\text{S[pt]/NP}_n)/\text{NP}_a)/\text{NP}_d & <\text{B} & \text{S[pt]/NP}_a & \text{S[v1]/NP}_a & \text{S[dcl]}
\end{array}
\]
The equivalence of TAG and CCG

TAG and CCG are weakly equivalent
(Weir & Joshi 1988, Weir & Vijay-Shanker 1994)

- CCGs can be converted into Linear Indexed Grammars
- TAGs can be converted into Linear Indexed Grammars
Linear Indexed Grammars (LIG)

\[ X[\alpha] \rightarrow \ldots Y[\alpha] \ldots \]

\[ X[\alpha, c] \rightarrow \ldots Y[\alpha] \ldots \]

\[ X[\alpha] \rightarrow \ldots Y[\alpha, c] \ldots \]

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

\[(S\backslash NP) / PP = S_{[\backslash NP, / PP]}\]

**Categories:** \( c = t[\alpha] \)

consist of a *target* \( t \) and a *stack* \( \alpha \)

**Stacks:** \( \alpha_i \in c^i \)

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

**Target categories:** \( t \in \{S, NP, PP, \ldots\} \)
Combinatory rules

Application
\[
\frac{X/Y \ Y}{X} \quad \frac{t[\alpha, u[\gamma]] \ u[\gamma]}{t[\alpha]} \quad B^n
\]

Composition \((n \leq 4)\)
\[
\frac{X/Y \ Y/ Z_1...Z_n}{X/Z_1...Z_n} \quad \frac{t[\alpha, u[\gamma]] \ u[\gamma, \beta_n]}{t[\alpha, \beta_n]} \quad B^n
\]

Type-raising \((m \leq 3)\)
\[
\frac{X}{T/(T\setminus X)} \quad \frac{c}{t[\alpha_m, t[\alpha_m, c]]} \quad T
\]
Typeraising + composition

Standard CCG

\[
\frac{T/X \rightarrow T}{T/(T/X) \rightarrow B^n}
\]

\[
T\{Z_1..Z_n\} \rightarrow B^n
\]

As an IG

\[
\frac{c \rightarrow T}{t[\alpha_m, t[\alpha_m, c]] \rightarrow B^n}
\]

\[
t[\alpha_m, \beta_n] \rightarrow B^n
\]

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

- LIGs can generate
  \[ n_1 \ldots n_{m+n} v_1 \ldots v_{m+n} \text{ and } n_{m+n} \ldots n_1 v_1 \ldots v_{m+n} \]

- CCGs can generate
  \[ N^1 \ldots N^{m+n} v_1 \ldots v_{m+n} \text{ and } N^{m+n} \ldots N^1 v_1 \ldots v_{m+n} \]

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

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

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

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

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

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