

Wide Coverage Parsing with Expressive Grammars

Mark Steedman, Informatics, University of Edinburgh
CLSP Summer Workshop, July 2002

www.cogsci.ed.ac.uk/~steedman/papers.html

Introduction

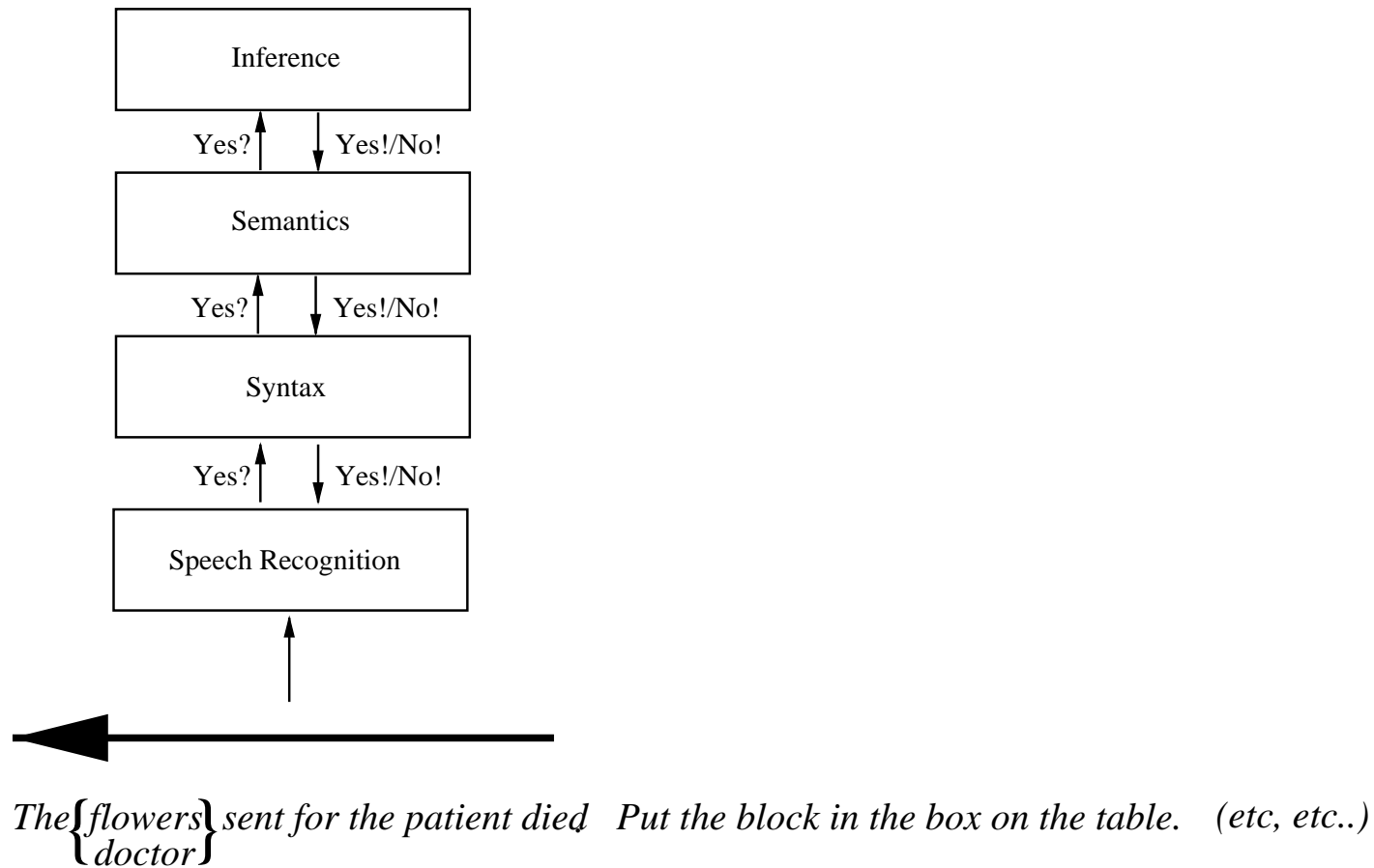
- Great progress in wide-coverage parsing has been made by combining statistical models of headword dependencies with grammar-based parsing—see Charniak lecture in this series.
- However, performance as measured by ParsEval has reached asymptote:
 - Magerman (1995): 84.3% LP / 84.0% LR
 - Collins (1999): 88.3% LP / 88.1% LR
 - Charniak (2000): 89.5% LP / 89.6% LR
- Moreover, the ParsEval measure is very forgiving. Such parsers have until now been based on highly overgenerating context-free covering grammars. Analyses depart in important respects from interpretable structures.
- In particular, they fail to represent the long-range “deep” semantic dependencies that are involved in relative and coordinate constructions.

The Anatomy of a Parser

- Every parser can be identified by three elements:
 - A Grammar (Regular, Context Free, Linear Indexed, etc.) and an associated automaton (Finite state, Push-Down, Extended Push-Down, etc.);
 - A search Algorithm characterized as left-to-right (etc.), bottom-up (etc.), and the associated working memories (etc.);
 - An Oracle, to resolve ambiguity.
- The oracle can be used in two ways, either to actively limit the search space, or in the case of an “all paths” parser, to rank the results.
- In wide coverage parsing, we need the former.

The Architecture of the Human Sentence Processor

- “Garden path” effects are sensitive to semantic content (Bever 1970) and context (Winograd 1972; Crain and Steedman 1985) requiring a “cascade”:



The Architecture of the Human Sentence Processor (Contd.)

- Head-dependency-Based Statistical Parser Optimization works because it approximates an oracle using semantics and real world inference.
- Its probably as close as we will get to the real thing for the foreseeable future.
- In fact, if suitable enhanced by associative Knowledge Representations and Contextual Model management. backing-off word dependencies to ontology-based hypernyms, etc., it may *be* the real thing.
- **There is a strong case for generalizing the method to more expressive grammars that will allow your open domain question answerer to stay on its feet when you ask “what subjects has Fred published articles on this year?”.**
- The “mildly context sensitive” grammars such as LTAG and CCG—the least expressive generalization of CFG known—are an interesting candidate.

Combinatory Categorical Grammar

$$(1) S \rightarrow NP VP$$

$$VP \rightarrow TV NP$$

$$TV \rightarrow \{proved, finds, \dots\}$$

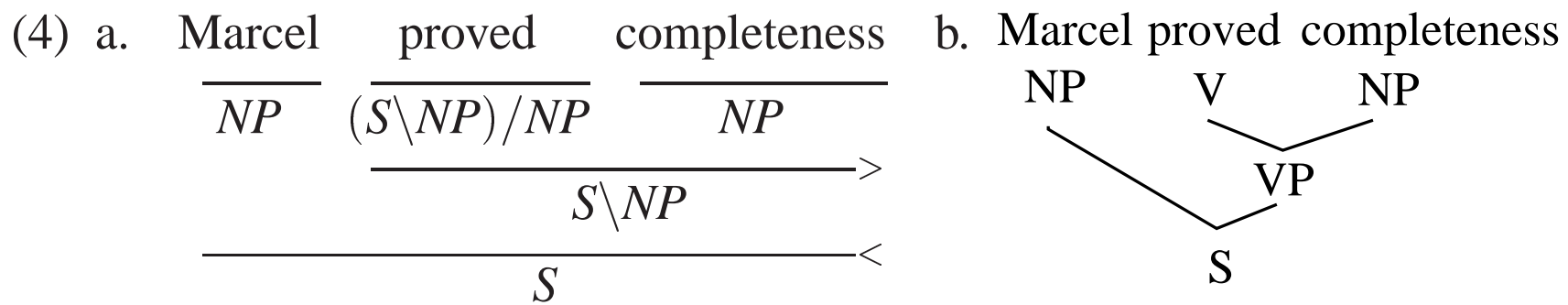
$$(2) proved := (S \setminus NP) / NP$$

(3) *The functional application rules*

$$a. X/Y \ Y \Rightarrow X \quad (>)$$

$$b. Y \ X \setminus Y \Rightarrow X \quad (<)$$

A Derivation



Semantics

(5) $\text{proved} := (S \setminus NP_{3s}) / NP : \lambda x. \lambda y. \text{prove}' xy$

(6) *Functional application*

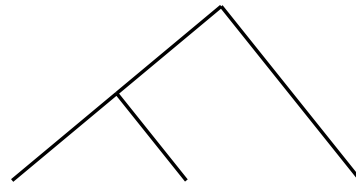
a. $X / Y : f \quad Y : a \Rightarrow X : fa \quad (>)$

b. $Y : a \quad X \setminus Y : f \Rightarrow X : fa \quad (<)$

(7)

<u>Marcel</u>	<u>proved</u>	<u>completeness</u>
$NP_{3sf} : \text{marcel}'$	$(S \setminus NP_{3s}) / NP : \lambda x. \lambda y. \text{prove}' xy$	$NP : \text{completeness}'$
$\xrightarrow{\hspace{10em}}$		
$S \setminus NP_{3s} : \lambda y. \text{prove}' \text{completeness}' y$		
$\xleftarrow{\hspace{10em}}$		
$S : \text{prove}' \text{completeness}' \text{marcel}'$		

Notation: Left-Associativity Convention over Logical Forms



(8) a. *(prove' completeness') marcel'* b. *prove' completeness' marcel'*

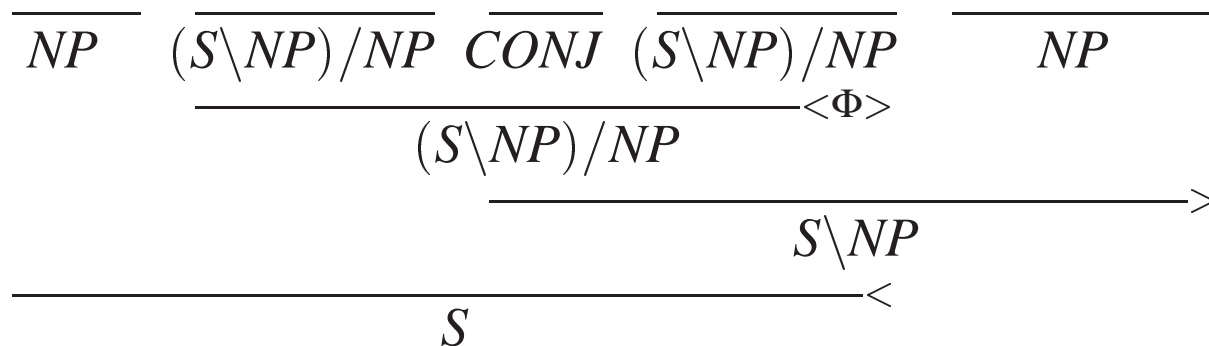
- A nonordered form of the traditional VP is reflected at the level of propositional logical form.
- Such logical forms therefore preserve traditional c-command.

Coordination

(9) *Simplified coordination rule* ($\langle \Phi \rangle$)

$$X \text{ CONJ } X' \Rightarrow X''$$

(10) Marcel conjectured and proved completeness



Composition

(11) *Forward composition* ($> \mathbf{B}$)

$$X/Y : f \quad Y/Z : g \quad \Rightarrow_{\mathbf{B}} \quad X/Z : \lambda x.f(gx)$$

(12)

\overline{NP} <i>: marcel'</i>	$\overline{(S \setminus NP) / NP}$ <i>: conjecture'</i>	\overline{CONJ} <i>: and'</i>	$\overline{(S \setminus NP) / VP}$ <i>: might'</i>	$\overline{VP / NP}$ <i>: prove'</i>	\overline{NP} <i>: completeness'</i>
			$\overline{(S \setminus NP) / NP} > \mathbf{B}$ <i>: $\lambda x.\lambda y.might'(prove' x)y$</i>		
			$\overline{(S \setminus NP) / NP} < \Phi >$ <i>: $\lambda x.\lambda y.and'(might'(prove' x)y)(conjecture' xy)$</i>		
					$\overline{S \setminus NP} >$ <i>: $\lambda y.and'(might'(prove' completeness')y)(conjecture' completeness'y)$</i>
$\overline{S : and'(might'(prove' completeness')marcel')(conjecture' completeness'marcel')} <$					

Type-Raising

(13) *Forward type-raising* ($> \mathbf{T}$)

$$NP : a \Rightarrow_{\mathbf{T}} \mathbf{T}/(\mathbf{T} \setminus NP) : \lambda f.fa$$

(14) Marcel proved and I disproved completeness

\overline{NP}	$\overline{(S \setminus NP)/NP}$	\overline{CONJ}	\overline{NP}	$\overline{(S \setminus NP)/NP}$	\overline{NP}
$\overline{S/(S \setminus NP)}^{>\mathbf{T}}$			$\overline{S/(S \setminus NP)}^{>\mathbf{T}}$		
$\overline{S/NP} \rightarrow \mathbf{B}$			$\overline{S/NP} \rightarrow \mathbf{B}$		
$\overline{S/NP} \langle \Phi \rangle$					
$\overline{S} \rightarrow$					

- Type raising is restricted to primitive argument categories, NP, PP etc., and over primitive functors like verbs, resembling the traditional notion of *case*.

Linguistic Predictions: Unbounded “Movement”

- We correctly predict the fact that right-node raising is unbounded, as in a, below, and also provide the basis for an analysis of the similarly unbounded character of leftward extraction, as in b (see Steedman (1996, 2000) for details):

(15) a. [I conjectured]_{S/NP} and [you think Marcel proved]_{S/NP} completeness.
b. The result [which]_{(N\N)/(S/NP)} [you think Marcel proved]_{S/NP}.
- We also predict the following asymmetry without stipulating the ECP, since we can allow a without allowing b, but can't allow b without also getting c:

(16) a. a man who(m) [I think that]_{S/S} [Keats likes]_{S/NP}
b. *a man who(m) [I think that]_{S/S} [likes Keats]_{S\NP}
c. *[I think]_{S/S} Chapman [that]_{S/S} [likes Keats]_{S\NP}.

Linguistic Predictions: Intonation

- If substrings like *Marcel proved* can behave like constituents for purposes of coordination in *Marcel proved and I disproved completeness* then they can do so in noncoordinate sentences, offering a way to bring prosody within the grammar:

(17) Q: I know who proved soundness. But who proved COMPLETENESS?

A: (MARCEL) (proved COMPLETENESS).

H*L L+H* LH%

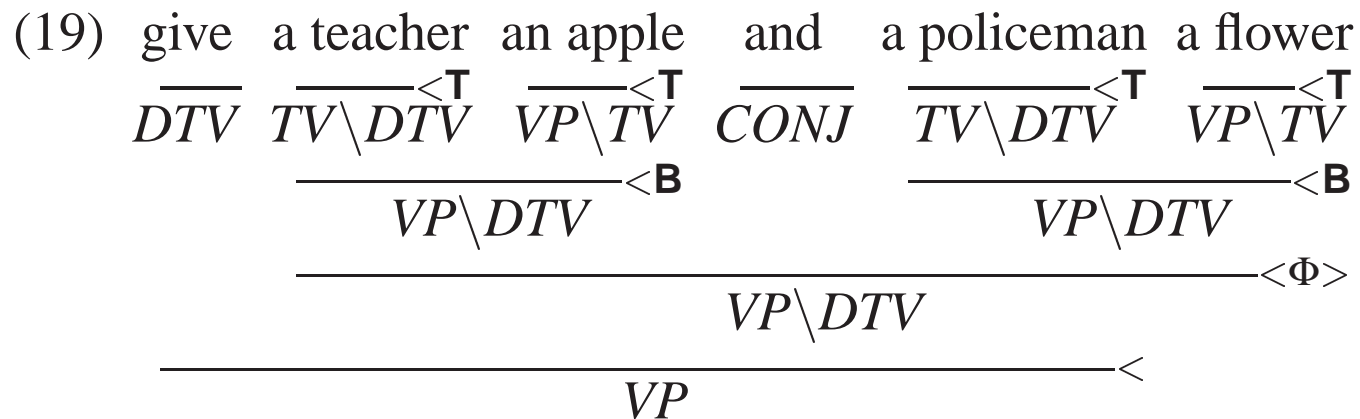
(18) Q: I know which result Marcel PREDICTED. But which result did Marcel PROVE?

A: (Marcel PROVED) (COMPLETENESS).

L+H* LH% H* LL%

Linguistic Predictions: “Non-Constituent” Coordination

- The following construction is predicted on arguments of symmetry.



- $DTV = (VP/NP)/NP$ $TV = VP/NP$
- In accord with observations of Ross (1970) concerning the relation of “verb gapping” and word order, CCG examples like the following cannot occur in an SVO language like English:

(20) *A policeman a flower and give a teacher an apple.

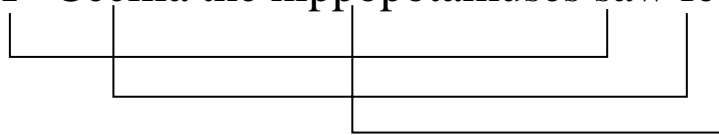
CCG is “Mildly Context Sensitive”

- CCG categories can be viewed as their result category plus a stack-valued feature identifying their arguments and the order of combination.
- Under the principles of inheritance and consistency, combinatory rules then map one-to-one to LIG rules.
- Hence CCG is provably weakly equivalent to Tree-adjoining Grammar (TAG), Head Grammar (HG), and Linear Indexed Grammar, a fact that gives rise to a polynomial time worst-case complexity result (Vijay-Shanker and Weir 1994), and “mildly context sensitive” expressive power.

Linguistic Predictions: Crossed Dependencies in CCG

... omdat ik Cecilia de nijlpaarden zag voeren.

... because I Cecilia the hippopotamuses saw feed



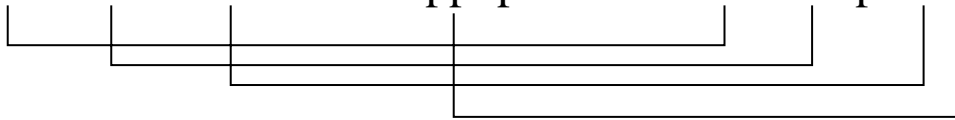
‘... because I saw Cecilia feed the hippopotamuses.’

$$\begin{array}{ccccccc}
 (21) & \text{dat} & \text{ik} & \text{Cecilia} & \text{de} & \text{nijlpaarden} & \text{zag} & \text{voeren} \\
 & \overline{NP_1} & \overline{NP_2} & \overline{NP_3} & & & & \\
 & & & & & & \overline{((S_{+SUB} \setminus NP_1) \setminus NP_2) / VP_{-SUB}} & \overline{VP \setminus NP_3} \\
 & & & & & & \xrightarrow{\mathbf{B}_\times} & \\
 & & & & & & & \overline{((S_{+SUB} \setminus NP_1) \setminus NP_2) \setminus NP_3}
 \end{array}$$

Crossed Dependencies in CCG (Contd.)

... omdat ik Cecilia Henk de nijlpaarden zag helpen voeren.

... because I Cecilia Henk the hippopotamuses saw help feed

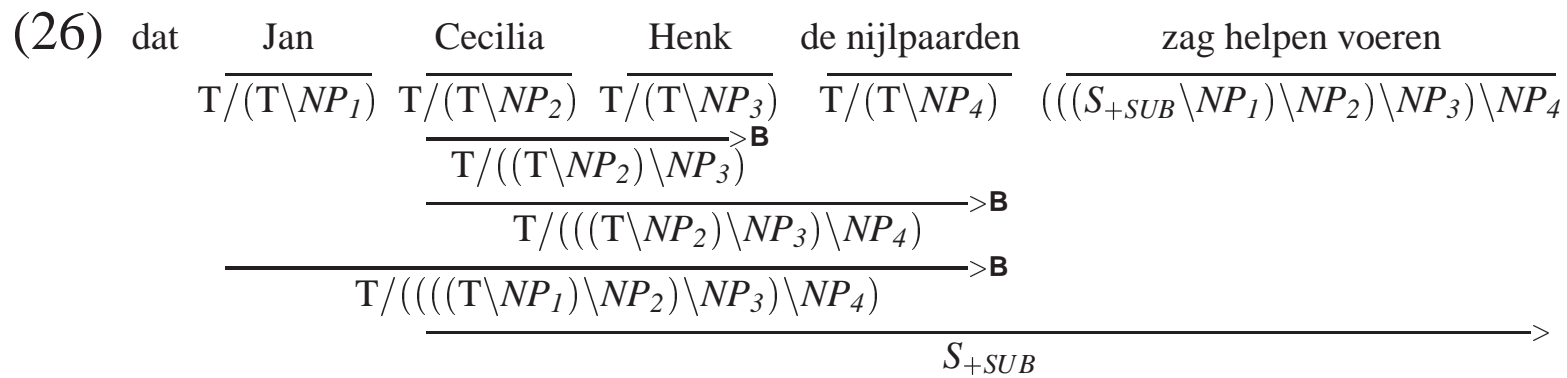


‘... because I saw Cecilia help Henk feed the hippopotamuses.’

$$\begin{array}{ccccccc}
 (22) & \text{dat} & \text{ik} & \text{Cecilia} & \text{Henk} & \text{de nijlpaarden} & \text{zag} & \text{helpen} & \text{voeren} \\
 & \overline{NP_1} & \overline{NP_2} & \overline{NP_3} & \overline{NP_4} & & & & \\
 & & & & & ((S_{+SUB} \backslash NP_1) \backslash NP_2) / VP_{-SUB} & (VP \backslash NP_3) / VP_{-SUB} & VP \backslash NP_4 & \\
 & & & & & & & & > \mathbf{B}_\times \\
 & & & & & & & (VP \backslash NP_3) \backslash NP_4 & \\
 & & & & & & & & > \mathbf{B}_\times^2 \\
 & & & & & (((S_{+SUB} \backslash NP_1) \backslash NP_2) \backslash NP_3) \backslash NP_4 & & &
 \end{array}$$

$$\begin{array}{ccccccc}
 (23) & \text{dat} & \text{ik} & \text{Cecilia} & \text{Henk} & \text{de nijlpaarden} & \text{zag} & \text{helpen} & \text{voeren} \\
 & \overline{NP_1} & \overline{NP_2} & \overline{NP_3} & \overline{NP_4} & & & & \\
 & & & & & ((S_{+SUB} \backslash NP_1) \backslash NP_2) / VP_{-SUB} & (VP \backslash NP_3) / VP_{-SUB} & VP \backslash NP_4 & \\
 & & & & & & & & > \mathbf{B}_\times^2 \\
 & & & & & (((S_{+SUB} \backslash NP_1) \backslash NP_2) \backslash NP_3) / VP_{-SUB} & & & \\
 & & & & & & & & > \mathbf{B}_\times \\
 & & & & & (((S_{+SUB} \backslash NP_1) \backslash NP_2) \backslash NP_3) \backslash NP_4 & & &
 \end{array}$$

Crossed Dependencies in CCG (Contd.)



Ross's Generalization Follows as a Theorem of CCG

- This immediately explain coordinations like the following in Dutch:
 - (dat) Jan Piet Marie en ik Cecilia de nijlpaarden zag helpen zwemmen.
(that) Jan Piet Marie and I Cecilia the hippos saw help swim.
 - (dat) Jan Piet en ik Cecilia de nijlpaarden zag helpen zwemmen.
(that) Jan Piet and I Cecilia the hippos saw help swim.
 - (dat) Jan Piet Marie zag helpen zwemmen en hoorde leren zingen.
(that) Jan Piet Marie saw help swim and heard teach sing.
- These, like (19), are instances of (a generalization of) Ross (1970)'s observation concerning “gapping” and word-order.

How to Parse with a Grammar that Has “Spurious Ambiguity”

- The ambiguity of derivation illustrated in the intonation examples (17) and (18) (which of course applies for written sentences) has been misleadingly referred to as “Spurious” ambiguity.
- **All** grammars exhibit derivational ambiguity—even CFG. Any grammar that captures coordination at all will have the same ambiguity as CCG.
- Use standard table-driven parsing methods such as CKY, checking identity of **underlying** representation of table entries, rather than derivation: Karttunen (1989); Komagata (1997, 1999); Hockenmaier *et al.* (2002).
- Vijay-Shanker and Weir 1994 show how to make this worst-case polynomial, although for realistic grammars exponential parsers seem to be average-case cubic (see Komagata 1999 for English and Japanese).

Cocke-Younger-Kasami (CKY) Algorithm for Parsing CFGs

INPUT: $a_0 a_1 \dots a_N$

for $i = 0$ **to** N

if $X \rightarrow a_i$ **then** insert X at $[i, i + 1]$

for $j = 2$ **to** $N + 1$

for $i = j - 2$ **downto** 0

for $k = i + 1$ **to** j

if $Z \rightarrow X Y$ **and** $X \in [i, k], Y \in [k, j]$

and $Z \notin [i, j]$

then insert Z at $[i, j]$

CKY Algorithm for Parsing CFGs (Contd.)

- Grammar: $S \rightarrow a; S \rightarrow SS$
- String: aaa

$i \backslash j$	0	1	2	3
0	\	S		
1		\	S	
2			\	S

$i \backslash j$	0	1	2	3
0	\	S	S	
1		\	S	
2			\	S

$i \backslash j$	0	1	2	3
0	\	S	S	S
1		\	S	
2			\	S

$i \backslash j$	0	1	2	3
0	\	S	S	S
1		\	S	S
2			\	S

How to Statistically Optimize the Parser

- Extract a CCG lexicon from the Penn Treebank: Hockenmaier and Steedman (2002a)
- **This trades lexical types (1200 against 48) for rules (3000 instantiations of combinatory rules against 12000) with standard Treebank grammars.**
- Use standard Maximum Entropy techniques to train a FSM “supertagger” Clark (2002), **multitagging at over 98% accuracy**
- **Then either:**
 - Parse using the Komagata technique, building and modeling *deep* or semantic dependency structures: Clark *et al.* (2002).
 - **or** use Normal-form parsing (Wittenburg and Wall (1991)), building and modeling Collins-style *surface* dependencies : Hockenmaier and Steedman (2002b).

A Problem

- Standard generative models for the local dependencies characteristic of CFGs do not immediately generalize to the **reentrant dependencies** generated by these more expressive grammars (Abney 1997—see Johnson lecture in this series).
- The model of Clark et al. 2002 is, technically, unsound. The generative model of Hockenmaier et al. only models local dependencies.
- Log Linear models offer one (rather desperate) kind of solution—see Johnson lecture in this series.
- We conjecture that a sound full generative model is as possible for mildly context sensitive grammars as it is for CFG (Hockenmaier, in preparation).

Performance on Overall Dependency Recovery

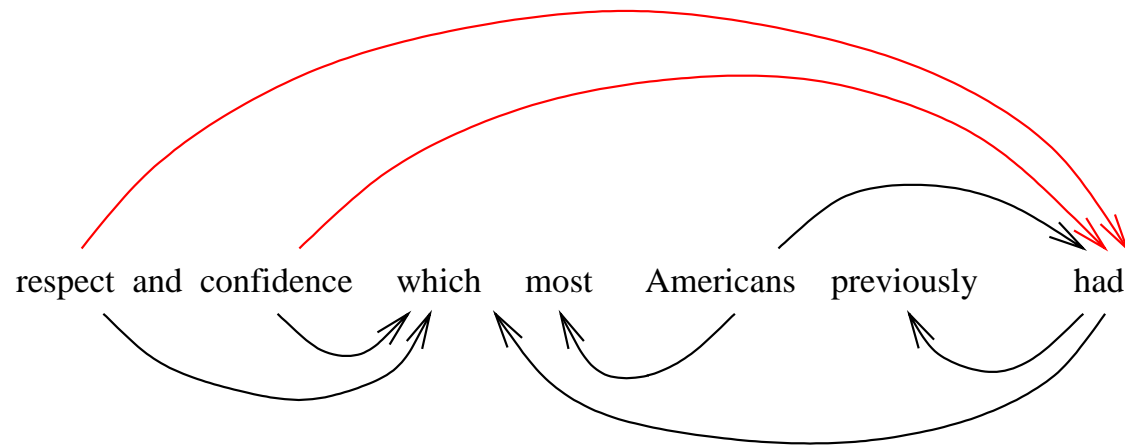
- Hockenmaier and Steedman (2002b)

Model	LexCat	Parseval				Surface dependencies	
		LP	LR	BP	BR	$\langle PHS \rangle$	$\langle \rangle$
Baseline	87.7	72.8	72.4	78.3	77.9	81.1	84.3
HWDep	92.0	81.6	81.9	85.5	85.9	84.0	90.1

- Collins (1999) reports 90.9% for “surface” dependencies.
- **CCG benefits greatly from word-word dependencies.**
(in contrast to Gildea (2001)’s observations for Collins’ Model 1)
- Compare on Clark *et al.* (2002)’s (different) “deep” dependencies:

	LP	LR	UP	UR
Clark et al. '02	81.9%	81.8%	89.1%	90.1%
Hockenmaier	83.7%	84.2%	90.5%	91.1%

Deep Dependencies: Clark *et al.* (2002)



lexical_item	category	slot	head_of_arg
<i>which</i>	$(NP_X \setminus NP_{X,1}) / (S[decl]_2 / cfNP_X)$	2	<i>had</i>
<i>which</i>	$(NP_X \setminus NP_{X,1}) / (S[decl]_2 / cfNP_X)$	1	<i>confidence</i>
<i>which</i>	$(NP_X \setminus NP_{X,1}) / (S[decl]_2 / NP_X)$	1	<i>respect</i>
<i>had</i>	$(S[decl]_{had} \setminus NP_1) / NP_2)$	2	<i>confidence</i>
<i>had</i>	$(S[decl]_{had} \setminus NP_1) / NP_2)$	2	<i>respect</i>

Performance on Full Object Relatives: Clark *et al.* (2002)

- 24 cases of extracted objects in Section 00 associated with object relative pronoun category $(NP_x \setminus NP_x) / (S[dcl] / NP_x)$
- 10 (41.7%) recovered with all dependencies correct
 - so both noun attachment and rel_pronoun-verb dependency correct
 - 10 incorrect because wrong category assigned to relative pronoun
 - * complementizer *that* has a high prior probability
 - 3 incorrect only because relative clause attached to the wrong noun
 - 1 incorrect only because wrong category assigned to predicate
- Much better performance can be expected with a better model.
- Other varieties of deep dependency discussed in Clark *et al.* (2002).

Hockenmaier and Steedman (2002b)

- **Extraction:**

- Lexical cat. for **subject relative pronoun**
(NP\NP)/(S[dcl]\NP): 97.1%P, 94.3%R
- Lexical cat. for **embedded subject extraction** (Steedman '96)
((S[dcl]\NP)/NP)/(S[dcl]\NP): 100.0%P, 83.3%R
- Lexical cat. for **object relative pronoun**
(NP\NP)/(S[dcl]/NP): 84.2%P, 76.2%R

- **Coordination:**

- VP coordination (coordination of S[.] \ NP): 67.3%P, 67.0%R
- Right-node-raising (coordination of (S[.] \ NP) / NP): 73.1%P, 79.2%R

CCG Parsers as Language Models

- Standard technique/baseline is Trigram modeling.
- Strict left-to-right parsing interpolated with trigram model does better: Chelba and Jelinek (1998); Roark (2001).
- Immediate-Head parser modeling alone does even better, even with a non-left-to-right algorithm: Charniak (2001).
- CCG type-raising treats head and complement as **dual**: In some sense, it makes **all** constructions head first.
- Specifically: it tells you how to *invert* the dependency model to obtain a left-to-right prediction.

CCG Parsers as Language Models

- For example, in Dutch the prefix *dat Cecilia een hond een knok ...* (“that Cecilia a dog a bone ...”) has a category $S/(((S\backslash NP)\backslash NP)\backslash NP)$ which predicts a ditransitive verbgroup and tells you all you need to know to estimate its Arg Max from verbs of that class. (For example, the “give” stem is going to come out ahead of the “sell” stem.)
- *dat een hond een knok Cecilia ...* is going to make a quite different predictions.
- CCG similarly offers a direct way to use prosodic information (see (17) and (18), above, and cf. Charniak 2001).

The Need for More Training Data

- Apart from weaknesses in statistical models that can be fixed, the chief limitation on the CCG parsers (and any parser with a comparably large category set and/or constrained grammar) is **known words which have not been seen with the necessary category**.
- This is a problem of the amount of training data: a million words is not enough for effective supervised learning for **any** grammar (Gildea 2001).
- **Either:**
 - Generalize the lexicon by clustering, or sources like WordNet.
 - Used semi supervised techniques to generate labelled data automatically

Weakly Supervised Training for Statistical Models

- Charniak showed that running his parser over 30M words of unlabelled data and retraining slightly improved performance.
- We also know voting among parsers improves performance Hendriks (1998): 92.09% LP / 90.1% LR.
- Co-training (Blum and Mitchell 1998): bootstrapping from small labeled corpus using multiple “views” from different parsers with different models and levels of confidence: Sarkar (2001)
- This is rather like what children do when their partial knowledge of the language tells them that a new word must be a verb, even though they don’t yet know the associated concept.
- We are pursuing this for a number of grammars/parsers at the CSLP Summer Workshop.

- Loop: **Sample** new data from unlabeled corpus.

Label new data using several models.

If parse(s) **reliable**:

Add parse(s) to training data.

Retrain models.

Validate using held-out set.

If performance doesn't degrade:

Accept newly labeled data.

- **Reliability:**

- **Co-training** – one model is very confident on parse.
- **Voting** – most models agree on parse.
- ...

Moral

- You can have your linguistic cake and eat it—with a an automatically induced lexicon and a statistical model

References

- Abney, Steven, 1997. “Stochastic Attribute-Value Grammars.” *Computational Linguistics* 23:597–618.
- Bever, Thomas, 1970. “The Cognitive Basis for Linguistic Structures.” In John Hayes (ed.), *Cognition and the Development of Language*, New York: Wiley. 279–362.
- Blum, Avram and Mitchell, John, 1998. “Combining Labeled and Unlabeled Data with Co-training.” In *COLT: Proceedings of the Workshop on Computational Learning Theory*. Morgan Kaufmann.
- Charniak, Eugene, 2000. “A Maximum-Entropy-Inspired Parser.” In *Proceedings of the 1st Meeting of the North American Chapter of the Association for Computational Linguistics*. Seattle, WA, 132–139.
- Charniak, Eugene, 2001. “Immediate-Head Parsing for Language Models.” In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics, Toulouse*. San Francisco, California: Morgan Kaufmann Publishers, 116–123.

Chelba, Ciprian and Jelinek, Frederick, 1998. “Exploiting Syntactic Structure for Language Modeling.” In *Proceedings of the Thirty-Sixth Annual Meeting of the Association for Computational Linguistics and Seventeenth International Conference on Computational Linguistics*. San Francisco, California: Morgan Kaufmann Publishers, 225–231.

Clark, Stephen, 2002. “A Supertagger for Combinatory Categorical Grammar.” In *Proceedings of the TAG+ Workshop*.

Clark, Stephen, Hockenmaier, Julia, and Steedman, Mark, 2002. “Building Deep Dependency Structures with a Wide-Coverage CCG Parser.” In *Proceedings of the 40th Meeting of the ACL (to appear)*. Philadelphia, PA.

Collins, Michael, 1999. *Head-Driven Statistical Models for Natural Language Parsing*. Ph.D. thesis, University of Pennsylvania.

Crain, Stephen and Steedman, Mark, 1985. “On Not Being Led up the Garden Path: the Use of Context by the Psychological Parser.” In Lauri Karttunen David Dowty and Arnold Zwicky (eds.), *Natural Language Parsing: Psychological, Computational and Theoretical Perspectives*, Cambridge: Cambridge University Press. 320–358.

- Gildea, Dan, 2001. “Corpus Variation and Parser Performance.” In *Proceedings of the 2001 Conference on Empirical Methods in Natural Language Processing*. Pittsburgh, PA, 167–202.
- Hendriks, H., 1998. “Links without Locations.” In F. Hamm and E. Zimmermann (eds.), *Linguistische Berichte, Sonderheft 9/1998: SEMANTIK*, Opladen: Westdeutscher Verlag.
- Hockenmaier, Julia, Bierner, Gann, and Baldrige, Jason, 2002. “Extending the Coverage of a CCG System.” *Journal of Logic and Computation* (forthcoming).
- Hockenmaier, Julia and Steedman, Mark, 2002a. “Acquiring Compact Lexicalized Grammars from a Cleaner Treebank.” In *Proceedings of the Third International Conference on Language Resources and Evaluation (to appear)*. Las Palmas, Spain.
- Hockenmaier, Julia and Steedman, Mark, 2002b. “Generative Models for Statistical Parsing with Combinatory Categorical Grammar.” In *Proceedings of the 40th Meeting of the ACL (to appear)*. Philadelphia, PA.

Karttunen, Lauri, 1989. “Radical Lexicalism.” In Mark Baltin and Anthony Kroch (eds.), *Alternative Conceptions of Phrase Structure*, Chicago: University of Chicago Press.

Komagata, Nobo, 1997. “Efficient Parsing for CCGs with Generalized Type-Raised Categories.” In *Proceedings of the 5th International Workshop on Parsing Technologies*, Boston MA. ACL/SIGPARSE, 135–146.

Komagata, Nobo, 1999. *Information Structure in Texts: A Computational Analysis of Contextual Appropriateness in English and Japanese*. Ph.D. thesis, University of Pennsylvania.

Magerman, David, 1995. “Statistical Decision Tree Models for Parsing.” In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*, Cambridge MA. San Francisco, CA: Morgan Kaufmann, 276–283.

Roark, Brian, 2001. “Probabilistic top-down parsing and language modeling.” *Computational Linguistics* 27:249–276.

- Ross, John Robert, 1970. “Gapping and the Order of Constituents.” In Manfred Bierwisch and Karl Heidolph (eds.), *Progress in Linguistics*, The Hague: Mouton. 249–259.
- Sarkar, Anoop, 2001. “Applying Co-Training Methods to Statistical Parsing.” In *Proceedings of NAACL 2001. Pittsburgh, PA, June*. Morgan Kaufmann.
- Steedman, Mark, 1996. *Surface Structure and Interpretation*. Cambridge, MA: MIT Press.
- Steedman, Mark, 2000. *The Syntactic Process*. Cambridge, MA: MIT Press.
- Vijay-Shanker, K. and Weir, David, 1994. “The Equivalence of Four Extensions of Context-Free Grammar.” *Mathematical Systems Theory* 27:511–546.
- Winograd, Terry, 1972. *Understanding Natural Language*. Edinburgh: Edinburgh University Press.
- Wittenburg, Kent and Wall, Robert, 1991. “Parsing with Categorical Grammar in Predictive Normal Form.” In Masaru Tomita (ed.), *Current Issues in Parsing Technology*, Dordrecht: Kluwer. 65–83. Revised selected papers from International Workshop on Parsing Technology (IWPT) 1989, Carnegie Mellon University.