
Advanced Natural Language Processing

Lecture 8

GPSG, LIG, TAG, and CCG

Mark Steedman

20th October 2009



The Return to Base Generation

We might attempt to extend the notions of phrase structure to account for discontinuities. . . . Similarly, one might seek to remedy some of the other deficiencies of $[\Sigma, F]$ grammars by a more complex account of phrase structure. I think that such an approach is ill-advised, and that it can only lead to the development of ad hoc and fruitless elaborations.

Chomsky 1957:41, note 6

I: Generalized Phrase-Structure Grammar (GPSG)

- Information about a “moved” or “missing” constituent can be passed up and down the tree by introducing
 - Additional category symbols (“slashed” categories) that must dominate a “gap” at some arbitrary depth beneath them
 - A rule introducing a “gap” that must be licensed by a slashed category

The CF-PSG Fragment of English Again

S → *NP VP*

VP → *TV NP*

VP → *DTV NP PP*

VP → *CV S'*

NP → *Det N*

NP → *PN*

N → *N Rel*

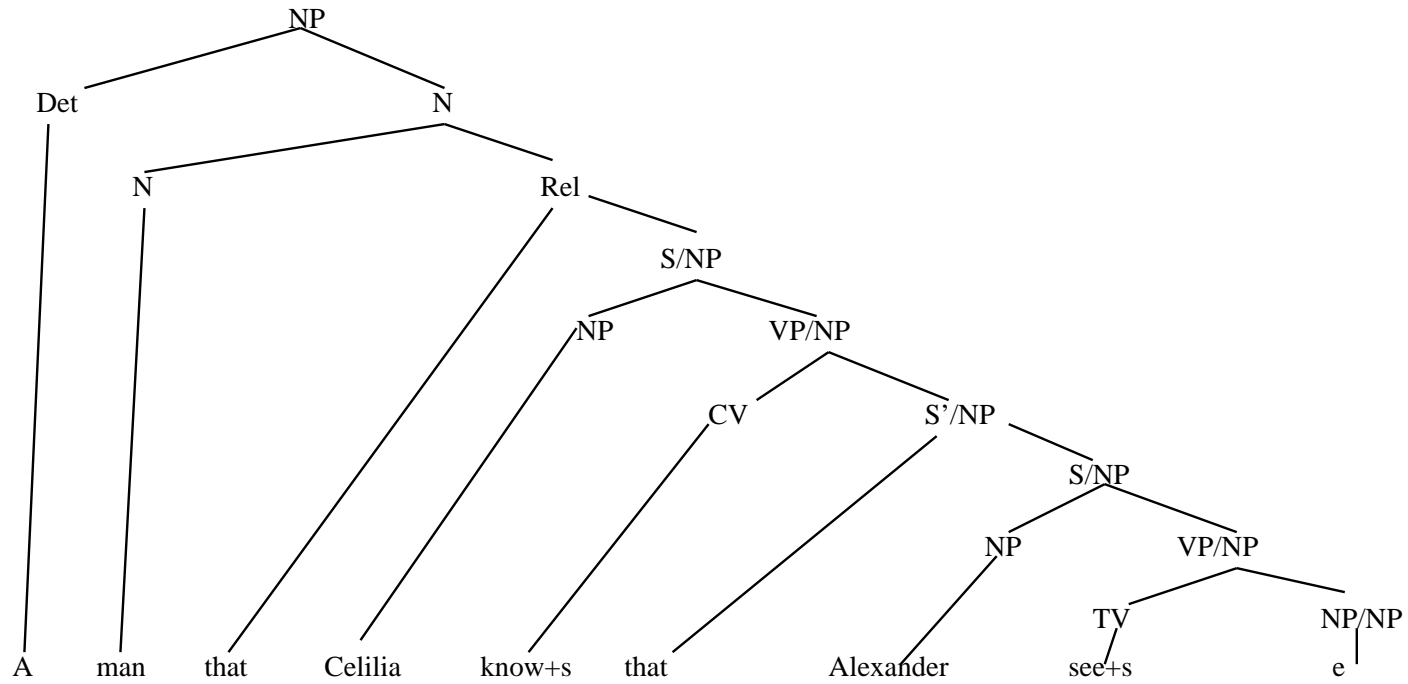
S' → *that S*

Rel → *that VP*

A Third View of Relativisation: CFG Generalized

<i>S</i>	→	<i>NP VP</i>
<i>S/NP</i>	→	<i>NP VP/NP</i>
<i>VP</i>	→	<i>TV NP</i>
<i>VP/NP</i>	→	<i>TV NP/NP</i>
<i>VP</i>	→	<i>DTV NP PP</i>
<i>VP/NP</i>	→	<i>DTV NP/NP PP</i>
<i>VP/NP</i>	→	<i>DTV NP PP/NP</i>
<i>VP</i>	→	<i>CV S'</i>
<i>VP/NP</i>	→	<i>CV S'/NP</i>
<i>NP</i>	→	<i>Det N</i>
<i>NP</i>	→	<i>PN</i>
<i>NP/NP</i>	→	ϵ
<i>N</i>	→	<i>N Rel</i>
<i>S'</i>	→	<i>that S</i>
<i>S'/NP</i>	→	<i>that S/NP</i>
<i>Rel</i>	→	<i>that VP</i>
<i>Rel</i>	→	<i>that S/NP</i>

A GPSG Derivation



Coordinate Structure Constraint Revisited

- General assumption: only “like” categories can be coordinated...
 - (1) a. [_{NP} NP and NP]
b. [_{VP} VP and VP]
c. * [_{?P} NP and VP]
- ... but a “slashed” category is different from an unslashed one
 - (2) * [_{?P} VP/NP and VP]
- So the “Coordinate Structure Constraint” follows directly
 - (3) *a man that_i we [_{?P} [_{VP/NP} met *t_i* yesterday] and [_{VP} saw him today]]

The Across-The-Board Exception to CSC Predicted

- Two unslashed categories can be identical:

(4) I [VP [VP met him yesterday] and [VP saw him again today]].

- A slashed category cannot be identical to an unslashed one:

(5) *a man that I [$?P$ [VP/NP met t yesterday] and [VP saw him again today]].

- But two slashed categories can also be identical so ATB is allowed:

(6) a man that I [VP/NP [VP/NP met t yesterday] and [VP/NP saw t again today]].

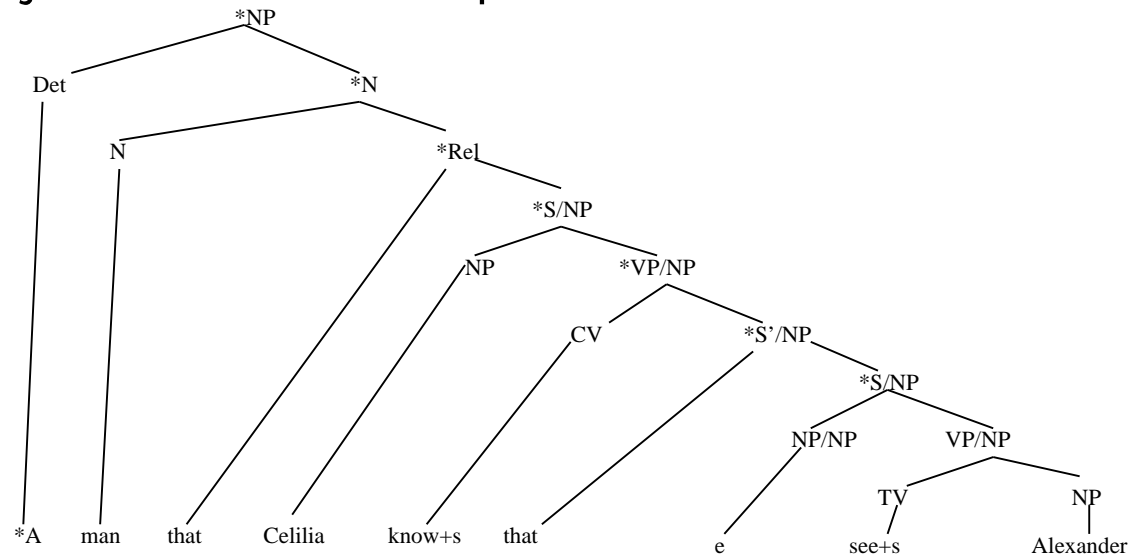
Same Case Condition Captured

- The “same case” condition on the ATB exception to the Coordinate Structure Constraint follows for free:

(7) *a man that [_{?P} [_{S/NP} I met *t* yesterday] and [_{VP} saw me first today]].

Fixed Subject Constraint Sort-of Captured

- The fixed subject constraint is captured:



— but only because the rule $S \rightarrow NP/NP \quad VP$ is excluded by stipulation.

The Multiple Dependency Problem Not Solved

- The Nested Dependency Constraint
 - (8) a. ?a cake which I met a man who ate
 - b. *a man who(m) I bought a cake which ate
- Violins and Sonatas
 - (9) a. a violin which the Moonlight Sonata is easy to play on
 - b. *a sonata which this Stradivarius violin is easy to play on
- Is GPSG SLASH a stack?
- Not clear how that would work for Dutch crossed dependencies.

II: Linear Indexed Grammars (LIG)

- Linear Indexed Grammar (LIG, Gazdar 1988) is a way of integrating the slash-feature-as-stack with the PDA stack.
- LIG is a natural family of languages in the Chomsky hierarchy, in the sense that it is associated with a characteristic automaton, the Embedded PDA (EPDA Joshi *et al.* 1991).
- LIG falls just above the CF class in the hierarchy.
- It is therefore “nearly context free”.

A LIG for $a^n b^n$ Crossed

- LIG can capture crossing dependencies.

$$(10) A_{[\dots]} \rightarrow \alpha B_{[\dots]} \beta$$

$$(11) \begin{array}{l} \text{“pushing”}: A_{[\dots]} \rightarrow \alpha B_{[i,\dots]} \beta \\ \text{“popping”}: A_{[i,\dots]} \rightarrow \alpha B_{[\dots]} \beta \end{array}$$

$$(12) \begin{array}{l} S_{[\dots]} \rightarrow n_i S_{[i,\dots]} \\ S_{[\dots]} \rightarrow S'_{[\dots]} \\ S'_{[i,\dots]} \rightarrow S'_{[\dots]} v_i \\ S'_{[]} \rightarrow \varepsilon \end{array}$$

Example: A LIG Derivation for n^3v^3 Crossed

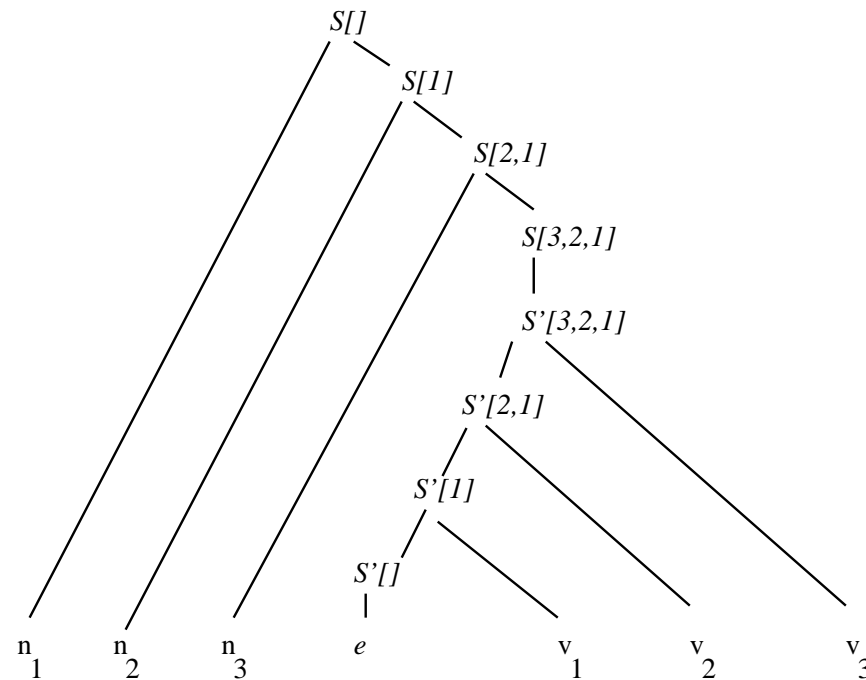


Figure 1: (Weakly adequate) LIG derivation for n^3v^3

An Answer to Chomsky's Implicit Question?

- Later in the course we will see that LIG offers a convincing candidate for a upper bound on the complexity of Natural Grammars.
- However, the above LIG is only weakly adequate. Coordination shows that we need more flexibility in the structural analysis of both NP and VP clusters.

III: Lexicalized Tree Adjoining Grammars (LTAG)

- Lexicalized Tree Adjoining Grammars (LTAG) associate an *elementary tree* with every lexical item.
- Elementary trees are direct descendants of the *Kernel Sentences* of Chomsky (1957).
- Elementary trees are either *initial trees* or *auxiliary trees*.
- They combine by two operations of *substitution* (which is just what it says) and *adjunction* (which is a little harder to explain).
- Substitution and Adjunction are reminiscent of the Double-based or “Generalized” transformations of Chomsky 1957.

LTAG Initial (α) Trees

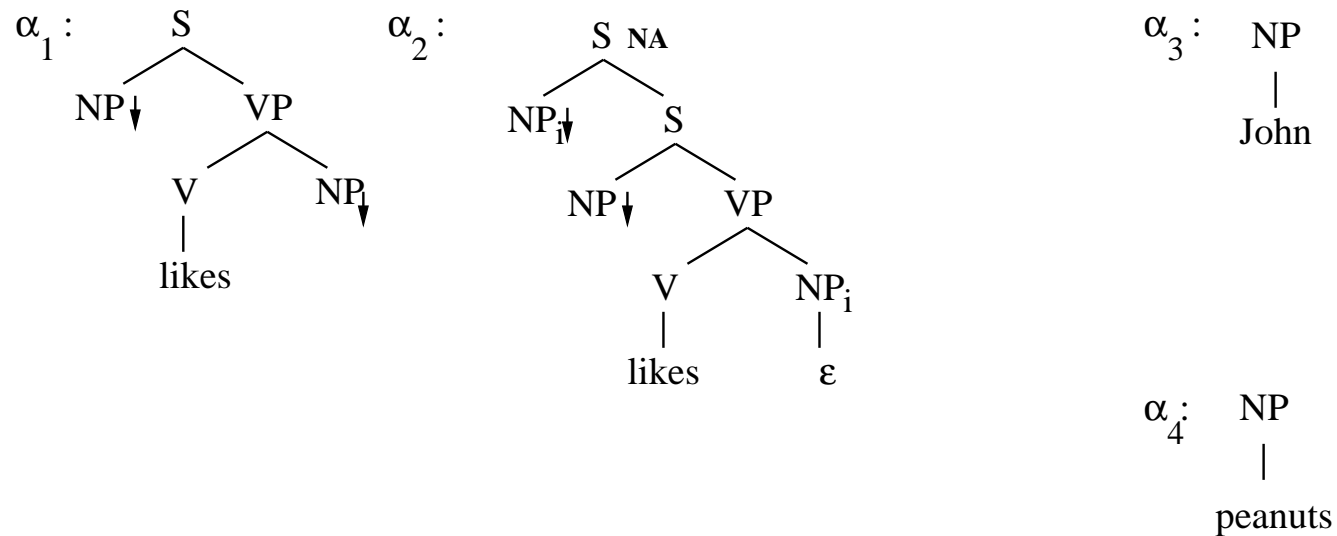


Figure 2: Initial Trees

- Down arrow \downarrow on a category like NP means “an NP must be substituted here.”

LTAG Auxiliary (β) Trees

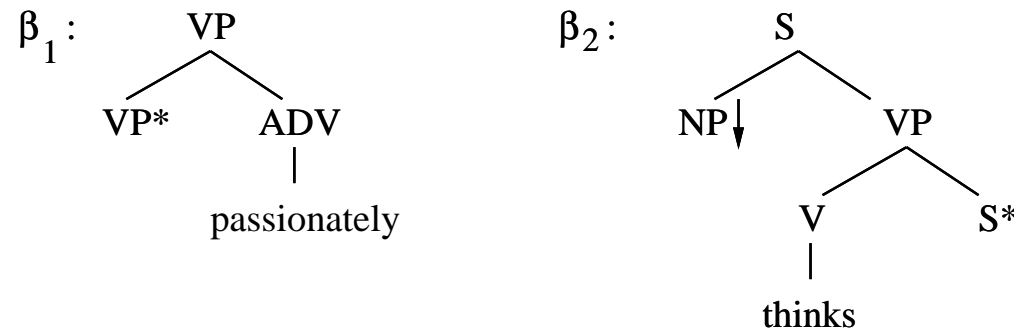


Figure 3: Auxiliary Trees

- An asterisk * on a category like S means “this S must adjoin to S.” **NA** on a category like S means “adjoining at this S is not allowed.” The NA annotation on the initial tree in the last slide will prevent **Bill thinks peanuts, John likes.*

A Derivation involving Substitution and Adjunction

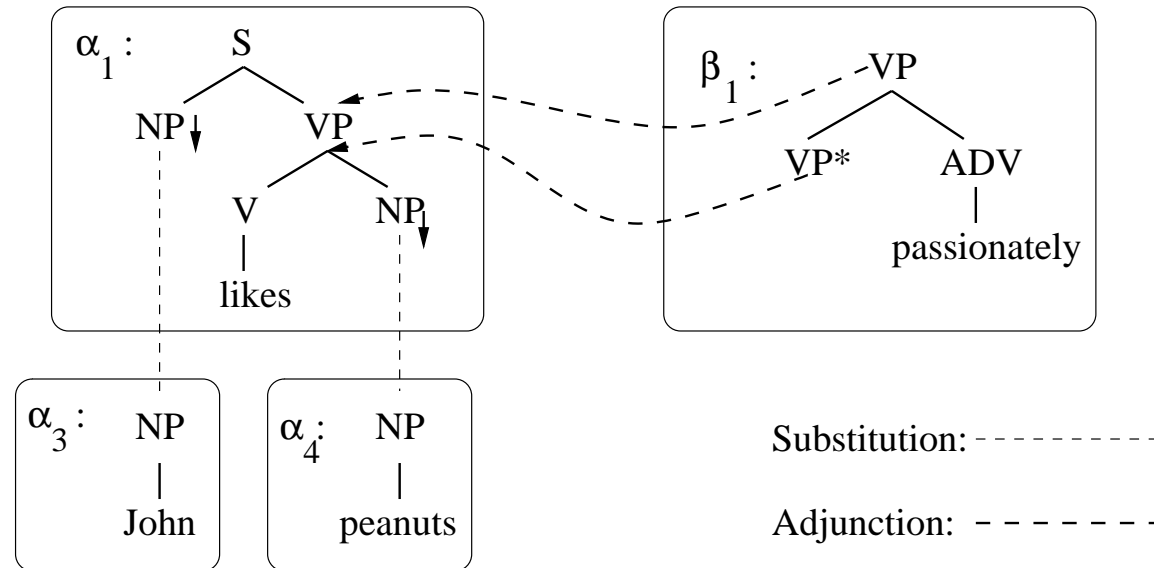
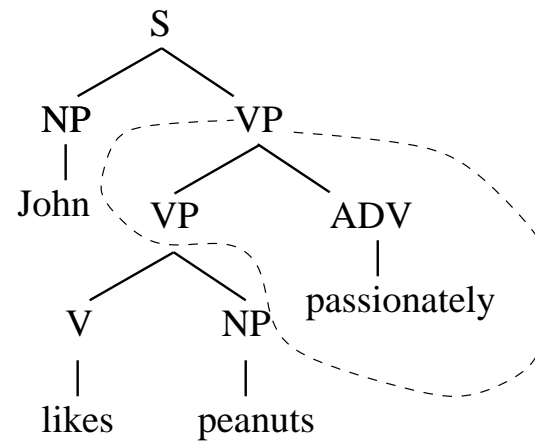


Figure 4: A derivation

- Recall that a asterisk * on a category like S means “this S must adjoin to S.”

A Derivation (Contd.)



Unbounded Dependency: Topicalization

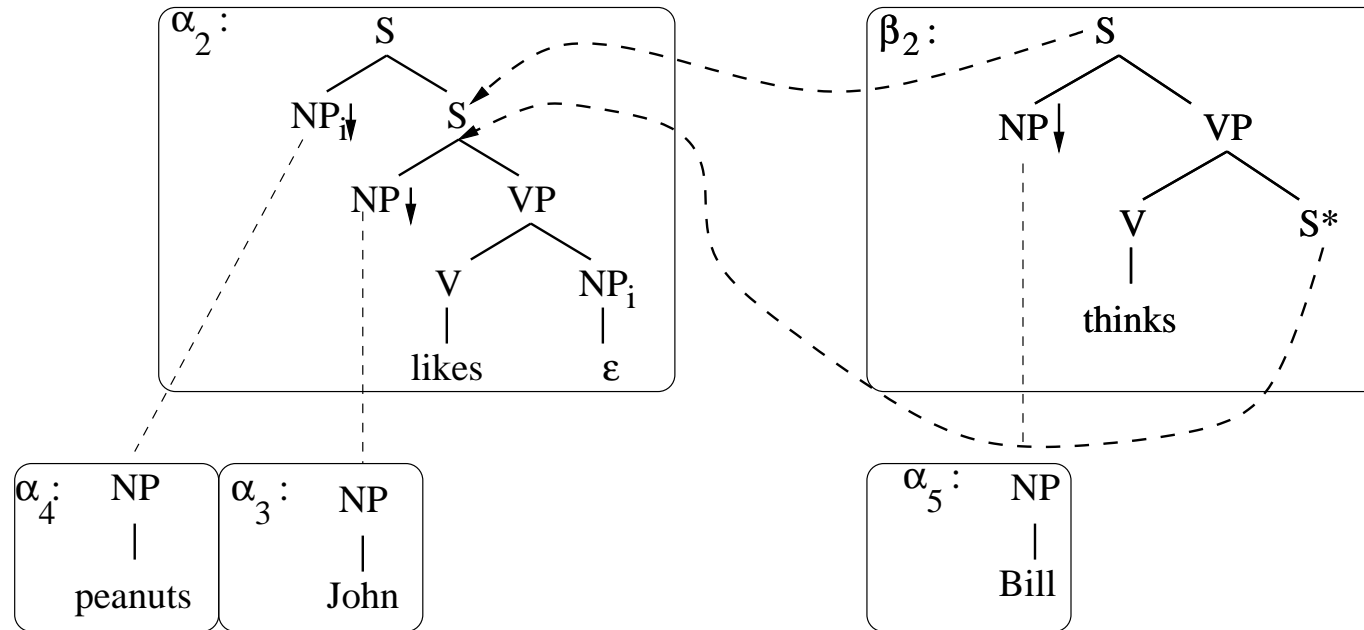


Figure 5: Derivation of long-distance topicalization

- Clearly we could adjoin unboundedly many auxiliary trees like β_2 .

Unbounded dependency (Contd.)

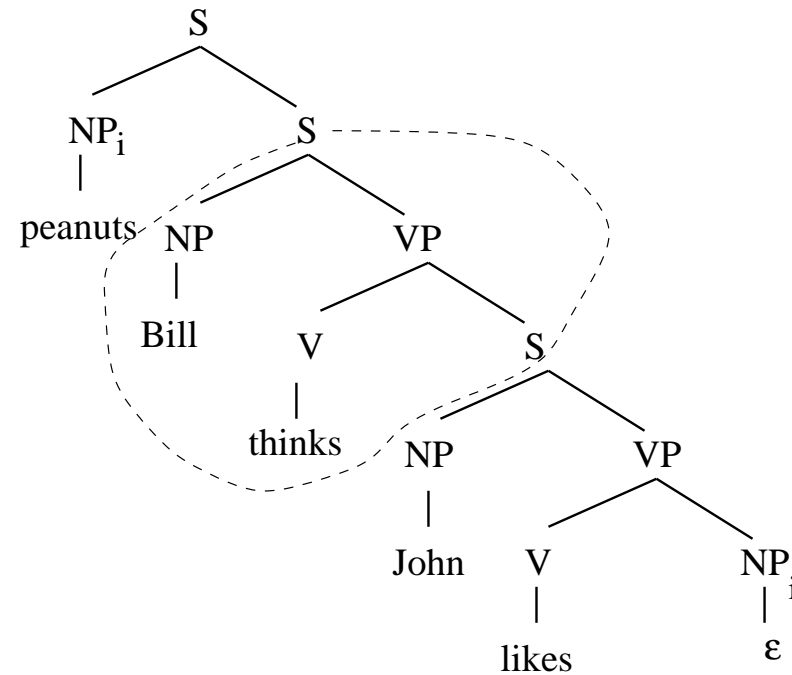


Figure 6: Result of the Derivation

The “Raising” Construction in TAG

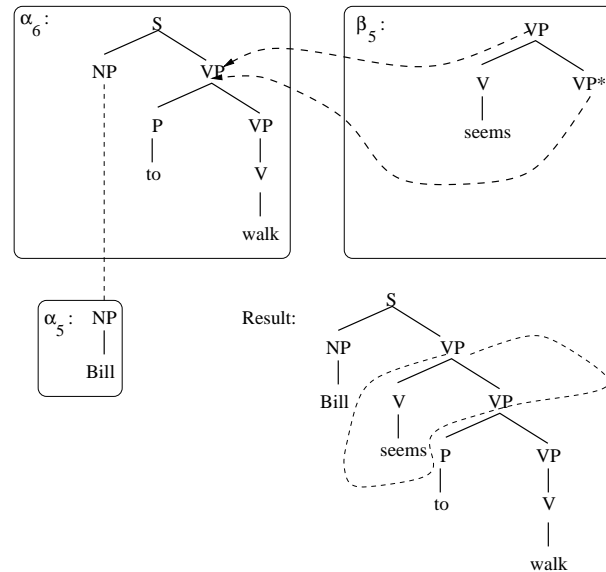


Figure 7: Raising as Adjunction

- Semantically this is function composition.

The “Control” Construction in TAG

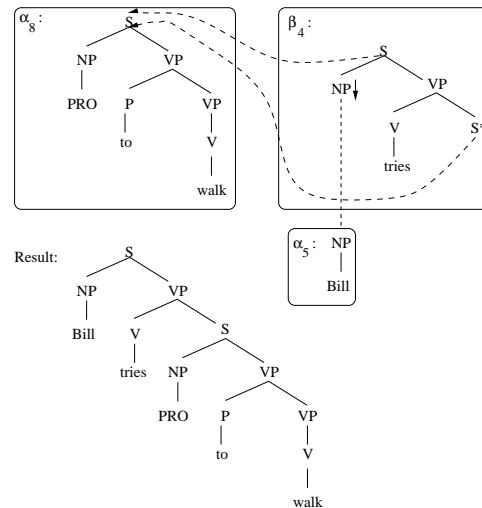


Figure 8: Control as Adjunction

- Since adjunction is at a root node, one might think this could be handled with substitution alone. However by analogy with Figure 5, we need adjunction to generalise to *These men, Bill tries to like*.

LTAG Can be Expressed as an LIG

- LTAG lexical elementary trees like α_1 , α_2 , and β_2 , can be regarded as bearing a stack-valued feature, where the elements on the stack are the \downarrow and $*$ marked leaf nodes, and where their order on the stack is defined by a leftmost-depth-first walk along the arcs of the tree.
- Substitution and adjunction can then be thought of as operations which pass a single such stack-valued feature to their result.
- This intuition underpins Weir's 1988 identification of the equivalence between LTAG and LIG (Joshi *et al.* (1991)).
- Seen in that light its not too surprising that we can capture the trans-CF grammar for $a^n b^n$ crossing.

LTAG for $a^n b^n$ crossing

- Consider the $G = (I, A)$ where I, A are the following sets of Initial and Auxiliary trees:

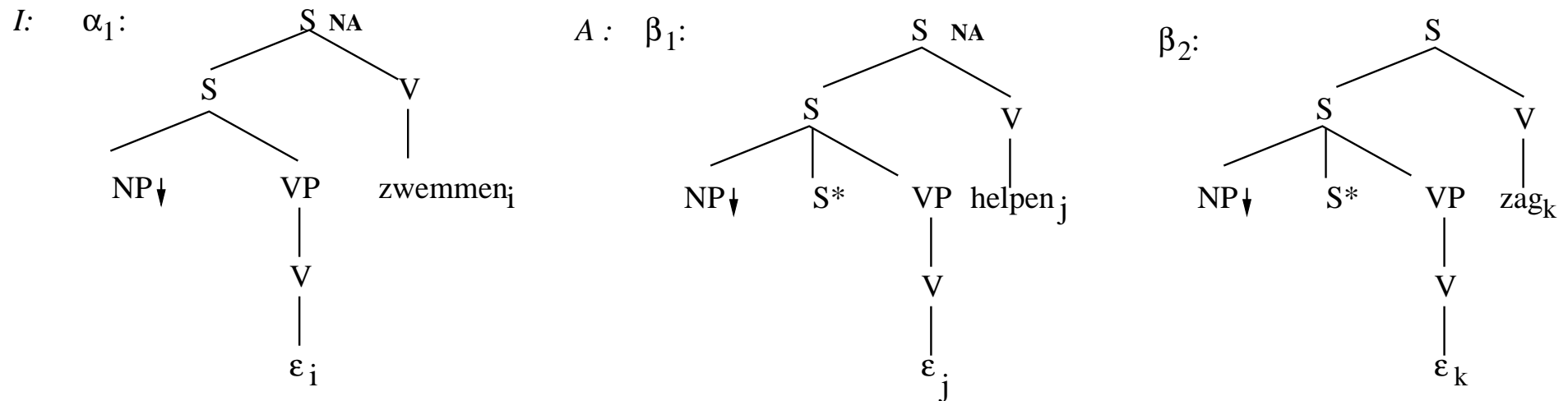


Figure 9: LTAG for $a^n b^n$ crossing.

Crossing Dependencies

- These categories engender crossing dependencies between lexical sisters:

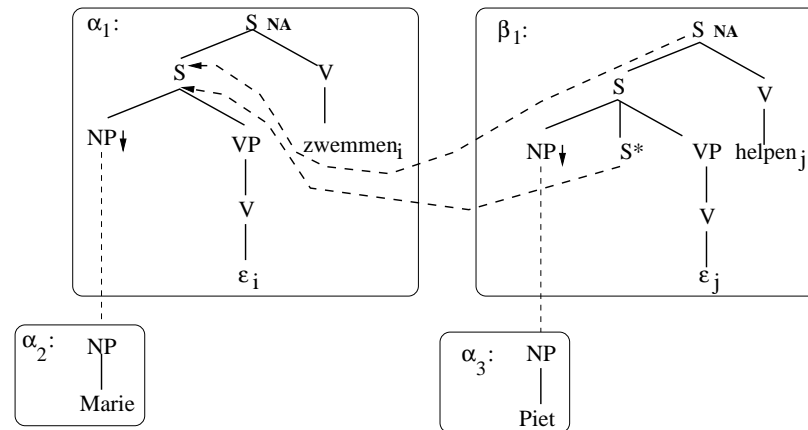


Figure 10: The Crucial Adjunction

Crossing Dependencies (Contd.)

- Piet Marie helpen zwemmen

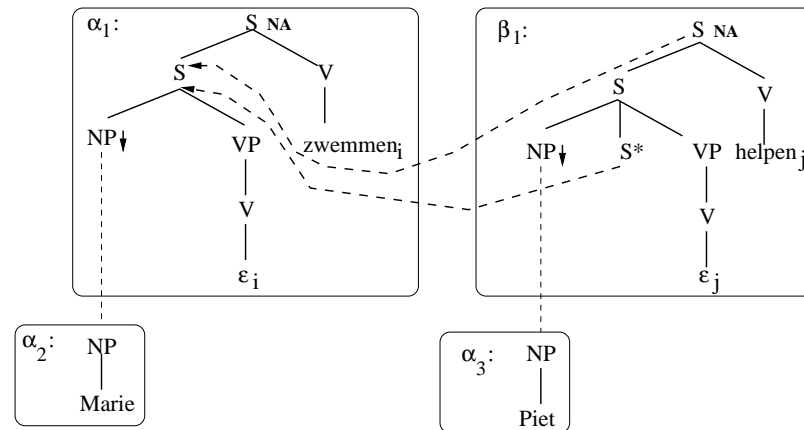


Figure 11: The Crucial Adjunction

Crossing Dependencies (Contd.)

- Piet Marie helpen zwemmen

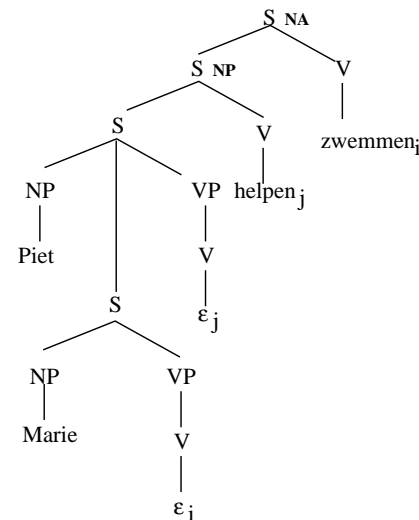


Figure 12: The Result of One Adjunction

Crossing Dependencies (Contd.)

- Jan Piet Marie zag helpen zwemmen

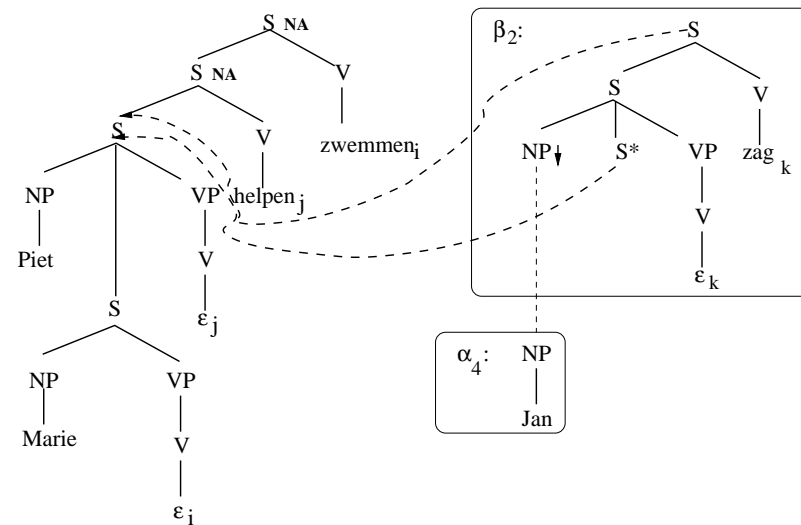


Figure 13: The Crucial Adjunction

Crossing Dependencies (Contd.)

- (dat) Jan Piet Marie zag helpen zwemmen

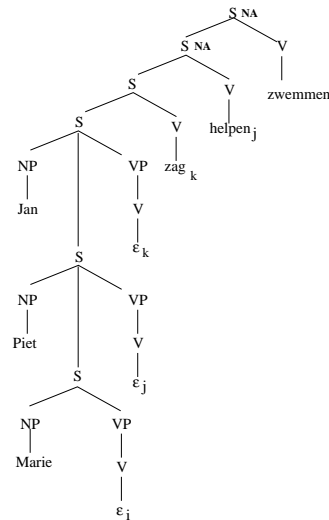


Figure 14: The Result of Two Adjunctions

Discussion

- This does not yet explain coordinations like the following.
 - (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 are all instances of (a generalization of) Ross's Generalization concerning gapping and the order of constituency.

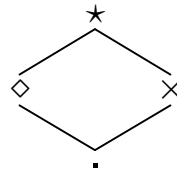
IV: Combinatory Categorical Grammar

- ~~(13) $S \rightarrow NP VP$
 $VP \rightarrow TV NP$
 $IV = \{proved, finds, \dots\}$~~
- (14) $proved := (S \setminus NP) / NP$

Slash Types

- Rules and function categories are typed, as indicated by a subscript on slashes.
- Function categories may be restricted as to the rules that allow them to combine with other categories, via slashes typed with four basic modalities: \star , \times , \diamond , and \cdot . The \star modality is the most restricted and allows only the most basic applicative rules; \diamond permits order-preserving associativity in derivations; \times allows limited permutation; and \cdot is the most permissive, allowing all rules to apply. enditemize

Slash Types



- By convention, the most permissive slash-types $/.$ and $\backslash.$ are abbreviated as categories like (14) as $(S\backslash NP)/NP$, rather than $(S\backslash.NP)/NP$

The Application Rules

– (15) *The functional application rules*

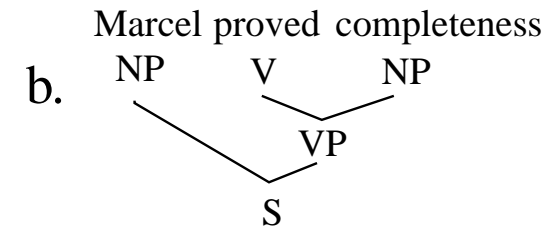
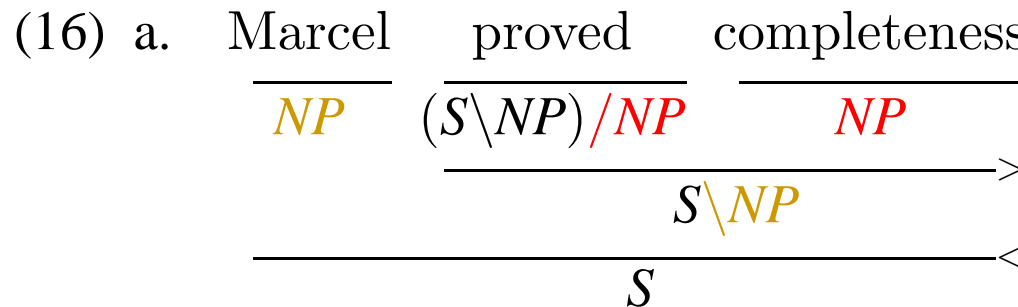
a. $X/_*Y \quad Y \Rightarrow X$

b. $Y \quad X \backslash_* Y \Rightarrow X$

(>)
(<)

– * modality means “this rule applies to all categories, even *”.

– A Derivation:



Semantics

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

(18) *Functional application*

a. $X /_* Y : f \quad Y : a \Rightarrow X : fa$

b. $Y : a \quad X \setminus_* Y : f \Rightarrow X : fa$

($>$)
($<$)

(19) $\frac{\text{Marcel}}{NP_{3sf} : \text{marcel}'} \quad \frac{\text{proved} \quad \text{completeness}}{(S \setminus NP_{3s}) / NP : \lambda x. \lambda y. \text{prove}'xy} \quad \frac{}{NP : \text{completeness}'}$

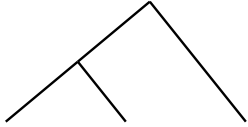
\rightarrow

$S \setminus NP_{3s} : \lambda y. \text{prove}' \text{completeness}'y$

\leftarrow

$S : \text{prove}' \text{completeness}' \text{marcel}'$

Notation: Left-Associativity Convention over Logical Forms

(20) 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

(21) *The Conjunction Category*
 and := $(X \setminus_{*} X) /_{*} X$

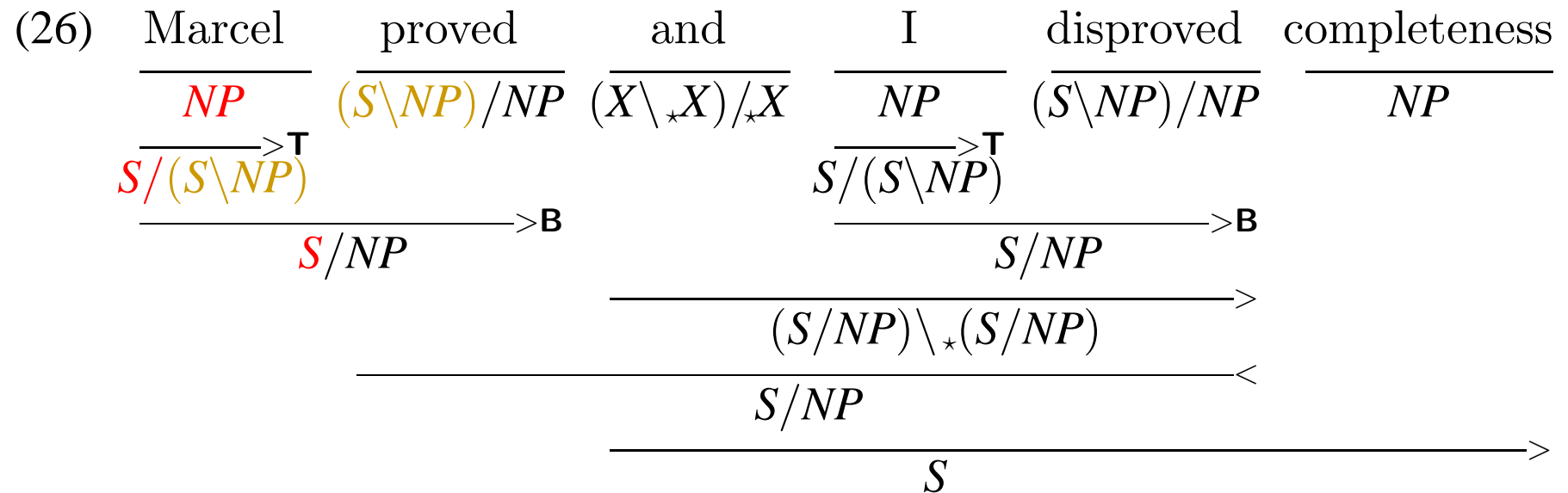
(22) Marcel conjectured and proved completeness

NP	$(S \setminus NP) / NP$	$(X \setminus_{*} X) /_{*} X$	$(S \setminus NP) / NP$	NP
		$((S \setminus NP) / NP) \setminus_{*} ((S \setminus NP) / NP)$		
	$(S \setminus NP) / NP$			
	$S \setminus NP$			
S				

Type-Raising

- (25) *Forward type-raising* ($> \mathbf{T}$)
 $X : a \Rightarrow \mathbf{T}/_i(\mathbf{T}\backslash_i X) : \lambda f.f a$
- The subscript i on the slashes means that they both have the same modality as what ever function $\mathbf{T}\backslash_i X$ the raised category is applied to. \mathbf{T} is a metavariable over categories. If instantiated as S , it allows the following derivation:

Type-Raising



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

Universal Grammar

- Only rules which obey the following principles are allowed:
 - (27) *The Principle of Adjacency*: Combinatory rules may only apply to finitely many phonologically realized and string-adjacent entities.
 - (28) *The Principle of Consistency*: All syntactic combinatory rules must be consistent with the directionality of the principal function.
 - (29) *The Principle of Inheritance*: If the category that results from the application of a combinatory rule is a function category, then the slash defining directionality for a given argument in that category will be the same as the one(s) defining directionality for the corresponding argument(s) in the input function(s).
- The Principal Function is the function into X.

Universal Grammar

- The Principle of Adjacency is merely **the definition of a combinator**.
- The other two principles say that **combinatory rules must “project” the directionality specified in the lexicon**:
- The Principle of Consistency excludes the following kind of rule:

$$(30) X \backslash Y \quad Y \not\Rightarrow X$$

- The Principle of Inheritance excludes rules like the following:

$$(31) X / Y \quad Y / Z \not\Rightarrow X \backslash Z$$

- These principles do allow rules such as the following:

$$(32) \textit{Forward crossing composition} (> \mathbf{B}_\times)$$

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

Many Linguistic Predictions

- 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.

(33) 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}.

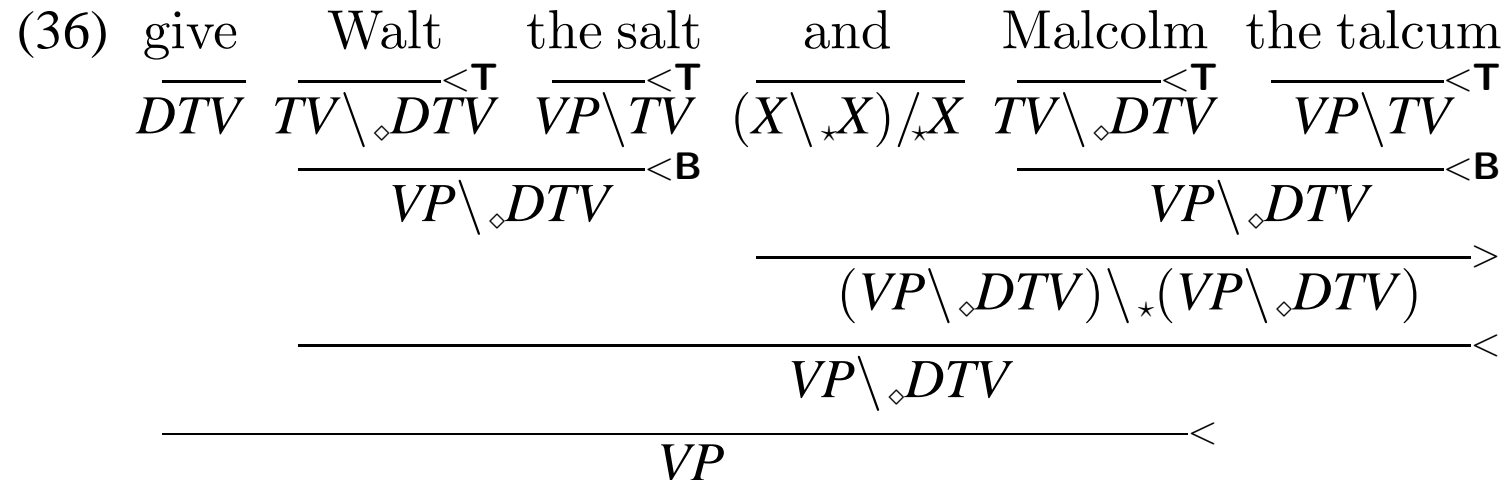
(34) a. a man who(m) [I think that]_{S/◇S} [Keats likes]_{S/NP}
b. *a man who(m) [I think that]_{S/◇S} [likes Keats]_{S\NP}

- The latter example captures the fixed subject constraint without stipulation. Crossed composition would overgenerate examples like the following:

(35) *I Harry think that likes Keats

Linguistic Predictions (contd.)

- Since we have assumed Forward Type Raising and Composition, the following construction is predicted on arguments of symmetry.



- $DTV = (VP/NP) /_{\diamond} NP$ $TV = VP/NP$

Linguistic Predictions (contd.)

- Given the stated limitation on type-raising, we disallow the following:

(37) *Three mathematicians [[in ten]_{PP} [derive a lemma,]_{S\NP}] and [in a hundred prove completeness.]

Gapping and the Order of Constituents

- The phenomenon of “argument cluster coordination” illustrated in (36) is an example of a much broader cross-linguistic generalization due to Ross.
- The position of the “gap” depends on basic constituent order:

(38) a. SOV: *SOV and SO, SO and SOV
b. VSO: VSO and SO, *SO and VSO
c. SVO: SVO and SO, *SO and SVO
- Zapotec (traditionally VSO) and Dutch (traditionally SOV), which allow both leftward and rightward gapping, also have suspiciously mixed word-order.

CCG is “Nearly Context Free”

- 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).
- This LIG equivalent class is not only mildly context sensitive. It is **Nearly Context Free**.

Crossed Dependencies in CCG

- The availability of *crossed* composition under the Principles of Consistency and Inheritance allows crossed dependencies:

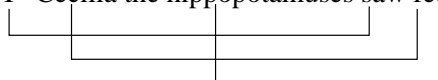
(39) $zag := ((S_{+SUB} \backslash NP) \backslash NP) /_{\times} VP$
zien, helpen: $:= (VP \backslash NP) /_{\times} VP$
voeren: $:= VP \backslash NP$

(40) **Dutch Forward Crossing Composition I** ($> B_{\times}^n$):
 $X /_{\times} Y \quad (Y \backslash_{\times} Z) \$ \Rightarrow_{B^n} X \backslash_{\times} Z \$$

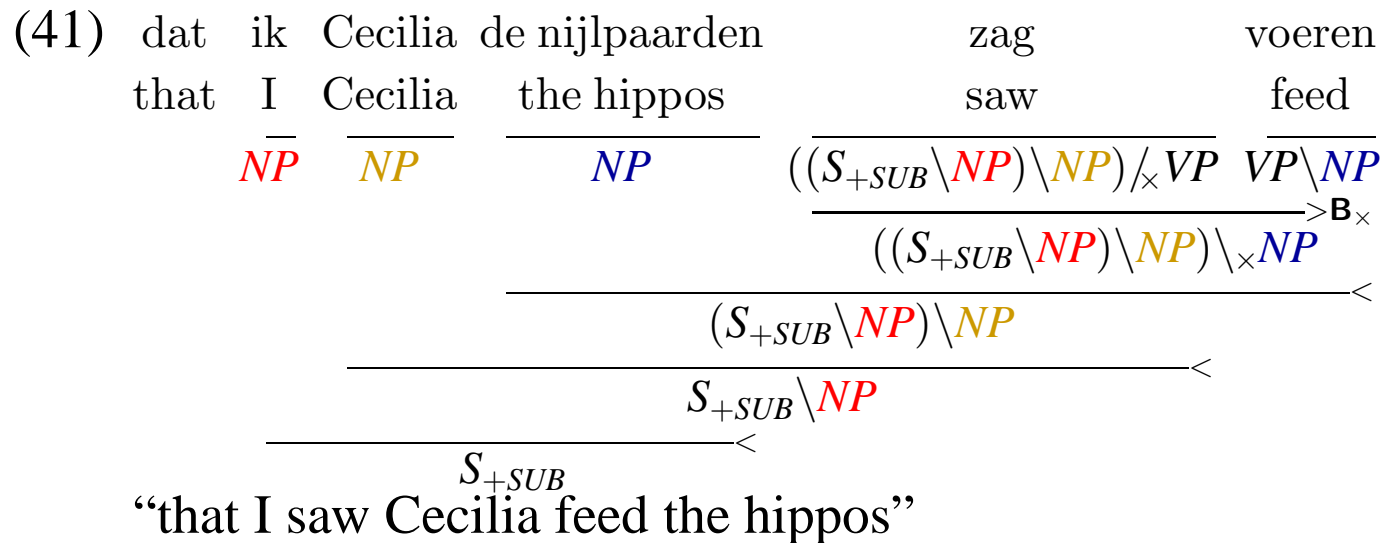
Hippo Sentences

... omdat ik Cecilia de nijlpaarden zag voeren.

... because I Cecilia the hippopotamuses saw feed



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



Hippo Sentences (Contd.)

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

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

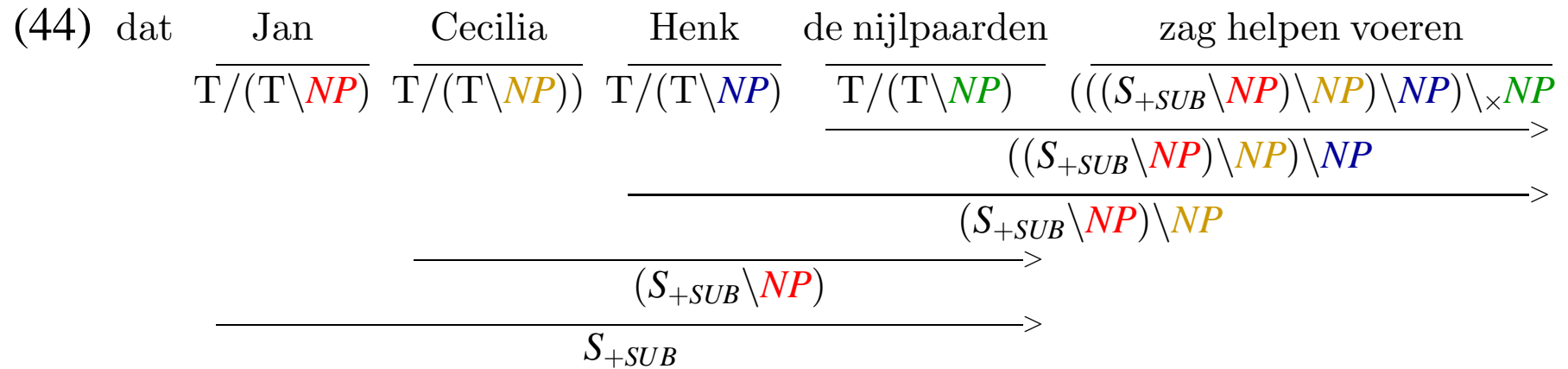


‘... because I saw Cecilia help Henk feed the hippopotamuses.’

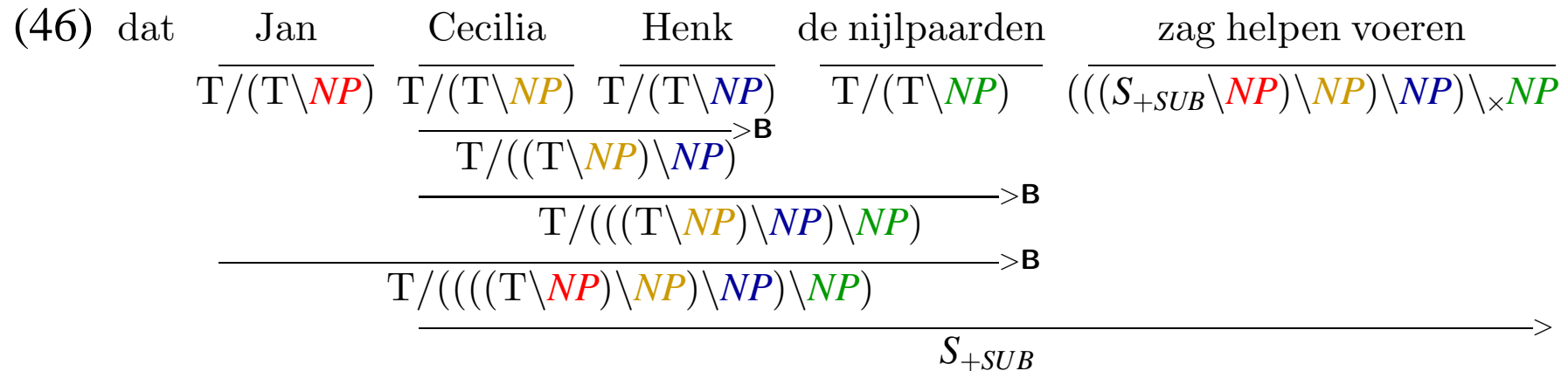
$$\begin{array}{ccccccc}
 (42) & \text{dat} & \text{ik} & \text{Cecilia} & \text{Henk} & \text{de nijlpaarden} & \text{zag} & \text{helpen} & \text{voeren} \\
 & \overline{NP} & \overline{NP} & \overline{NP} & \overline{NP} & & & & \\
 & & & & & & \overline{((S_{+SUB} \backslash NP) \backslash NP) \backslash VP} & \overline{(VP \backslash NP) \backslash VP} & \overline{VP \backslash NP} \\
 & & & & & & & & \xrightarrow{B_x} \\
 & & & & & & & & \overline{(VP \backslash NP) \backslash NP} \\
 & & & & & & & & \xrightarrow{B_x^2} \\
 & & & & & & & & \overline{(((S_{+SUB} \backslash NP) \backslash NP) \backslash NP) \backslash NP}
 \end{array}$$

$$\begin{array}{ccccccc}
 (43) & \text{dat} & \text{ik} & \text{Cecilia} & \text{Henk} & \text{de nijlpaarden} & \text{zag} & \text{helpen} & \text{voeren} \\
 & \overline{NP} & \overline{NP} & \overline{NP} & \overline{NP} & & & & \\
 & & & & & & \overline{((S_{+SUB} \backslash NP) \backslash NP) \backslash VP} & \overline{(VP \backslash NP) \backslash VP} & \overline{VP \backslash NP} \\
 & & & & & & & & \xrightarrow{B_x^2} \\
 & & & & & & & & \overline{(((S_{+SUB} \backslash NP) \backslash NP) \backslash NP) \backslash VP} \\
 & & & & & & & & \xrightarrow{B_x} \\
 & & & & & & & & \overline{(((S_{+SUB} \backslash NP) \backslash NP) \backslash NP) \backslash NP}
 \end{array}$$

Hippo Sentences (Contd.)



Hippo Sentences (Contd.)



- The Dutch “nonconstituent” coordinations in the last slide of the previous set are therefore examples of *constituent coordination* in CCG, just like that in example (36), and are further instances of Ross’s Generalization.

CCG and LIG

- Joshi *et al.* (1991) were the first to observe that there is a close relation between linear indexed rules and the combinatory rules of CCG.
- Function categories like *give* and *zag helpen voeren* can be equated with indexed categories bearing stack-valued features, as follows:

$$(47) \text{ give} := (VP/NP)/NP \equiv VP_{[NP, NP]}$$

$$\text{zag helpen voeren} := (((S \setminus NP) \setminus NP) \setminus NP) \setminus NP \equiv S_{[NP, NP, NP, NP]}$$

- Note that the LIG categories no longer encode directionality—it is up to the LIG rules to do that.

CCG and LIG

- Since LIG categories do not capture directionality, the grammar for a particular language will be made up of more specific instances of these schemata involving just those categories that do in fact combine in the specified order for that language.¹

$$(48) X/Y \ Y \Rightarrow X \equiv X'_{[...]} \rightarrow X'_{[Y,...]} \ Y_{[]}$$

$$(49) X/Y \ Y/Z \Rightarrow X/Z \equiv X'_{[Z,...]} \rightarrow X'_{[Y,...]} \ Y_{[Z]}$$

¹Note also that the category X' in the LIG schemata will not in general be the same as the category X in the corresponding instantiation of the categorial rule. If X is a function category, then X' is its result and the stack \dots is its argument(s). In terms of the earlier $\$$ convention, X is of the form $X'\$$.

CCG and LIG

- Rule (48) is forward application, realized as a binary LIG rule of the “push” variety. Rule (49) is first-order forward composition, \mathbf{B} , and involves both pushing a Y and popping a Z .
- Crucially, the stack, represented as \dots , is passed to only one daughter. Therefore this is a *Linear* Indexed Grammar (LIG).
- Provided higher order composition (\mathbf{B}^n) is kept to bounded n and type-raising is over functions of similarly bounded valency, the same applies.
- Otherwise CCG generalizes to full Indexed Grammar (IG).
- The natural generalization of TAG (set local multicomponent TAG) is equivalent to the general LCFRS, also multiple context free grammars (MCFG) and Stabler (1987)’s version of Minimalist Grammar (MG).

These Things are Out There in the Treebank

- Full Object Relatives (570 in WSJ treebank)
- Reduced Object Relatives (1070 in WSJ treebank)
- Argument Cluster Coordination (230 in WSJ treebank):

```
(S (NP-SBJ It)
  (VP (MD could)
    (VP (VP (VB cost)
      (NP-1 taxpayers)
      (NP-2 $ 15 million))
    (CC and)
    (VP (NP=1 BPC residents)
      (NP=2 $ 1 million))))))
```


These Things are Out There (contd.)

- Parasitic Gaps (at least 6 in WSJ treebank):

```
(S (NP-SBJ Hong Kong's uneasy relationship with China)
  (VP (MD will)
    (VP (VP (VB constrain)
      (NP (-NONE- *RNR*-1)))
      (PRN (: --)
        (IN though)
        (VP (RB not)
          (VB inhibit)
          (NP (-NONE- *RNR*-1)))
          (: --))
        (NP-1 long-term economic growth))))))
```

References

Chomsky, Noam, 1957. *Syntactic Structures*. The Hague: Mouton.

Gazdar, Gerald, 1988. “Applicability of Indexed Grammars to Natural Languages.”
In Uwe Reyle and Christian Rohrer (eds.), *Natural Language Parsing and Linguistic Theories*, Dordrecht: Reidel. 69–94.

Joshi, Aravind, Vijay-Shanker, K., and Weir, David, 1991. “The Convergence of Mildly Context-Sensitive Formalisms.” In Peter Sells, Stuart Shieber, and Tom Wasow (eds.), *Processing of Linguistic Structure*, Cambridge, MA: MIT Press. 31–81.

Stabler, Edward, 1987. “Derivational Minimalism.” In Christian Retoré (ed.), *Logical Aspects of Computational Linguistics (LACL'96)*. New York: Springer, volume 1328 of *Lecture Notes in Computer Science*, 68–95.

Vijay-Shanker, K. and Weir, David, 1994. “The Equivalence of Four Extensions of Context-Free Grammar.” *Mathematical Systems Theory* 27:511–546.

Weir, David, 1988. *Characterizing Mildly Context-sensitive Grammar Formalisms*. Ph.D. thesis, University of Pennsylvania, Philadelphia PA. Technical Report CIS-88-74.