4

Processing

> '*Take complexity-based accounts. . . . Here, the idea is to establish a nonarbitrary metric for complexity, one that makes reference to structure. These metrics are rarely spelled out explicitly or motivated theoretically.*'
>
> Grodzinsky 2000: 56

> '*. . . the complexity of the component processes in sentence processing does not lend itself well to developing [computational] models that make close contact with empirical data without making numerous ancillary assumptions.*'
>
> Tanenhaus 2003: 1145

Accounts of linguistic competence rest on abstractions and idealizations which, however fruitful, must eventually be integrated in a full account of language with human computational performance in language use. Here we advocate the modelling of language processing on the basis of an incremental synthesis of categorial proof nets, therein obtaining a wide-ranging complexity metric which is nonarbitrary, simple, explicit, free of ancillary assumptions, and theoretically motivated.[1]

Centre embedding unacceptability is illustrated by the fact that while the nested subject relativizations of (1) exhibit little variation in acceptability,

[1] This chapter is a reworking of material from Morrill (2000) Incremental Processing and Acceptability. *Computational Linguistics*, **26**(3), 319–338, used with permission of the Association for Computational Linguistics.

978–0–19–958985–2 (Hbk) 978–0–19–958986–9 (Pbk) 04-Morrill-c04 Morrill (Typeset by SPi, Chennai) 52 of 80 July 1, 2010 21:24
OUP UNCORRECTED PROOF – REVISES, 1/7/2010, SPi

the increasingly nested object relativizations (2) are increasingly unacceptable (Chomsky 1965, ch. 1).

(1) a. The dog that chased the cat barked.
b. The dog that chased the cat that saw the rat barked.
c. The dog that chased the cat that saw the rat that ate the cheese barked.

(2) a. The cheese that the rat ate stank.
b. ?The cheese that the rat that the cat saw ate stank.
c. ??The cheese that the rat that the cat that the dog chased saw ate stank.

Gibson (1998) analyses such phenomena in terms of a dependency locality theory according to which the resources required for storing a partially processed structure are proportional to the number of incomplete syntactic dependencies at that point in processing the structure. Taking inspiration from Gibson, Johnson (1998) analyses centre embedding in terms of categorial proof nets, relating the maximal nesting of axiom links to the degree of unacceptability. Morrill (2000), on which this chapter is based, implements and rationalizes these ideas in terms of syntactic structures as proof nets and a complexity metric derived from the load on memory of an incremental algorithm of language processing.

4.1 Introduction

Girard (1987) introduced linear logic. Linear logic preserves from standard logic the freely applying structural rule of permutation, but not freely applying structural rules of contraction and weakening. The Lambek calculus lacks all three structural rules. Thus the Lambek calculus and linear logic are instances of what has come to be known as *substructural logic* (Restall, 2000). Just as the Lambek calculus is a *sequence logic*, linear logic is an *occurrence* logic.

Occurrence logics had been studied before under the rubric of BCI-logic.[2] However, Girard's linear logic established in a clear and unprecedented way the concepts of additive, multiplicative, and exponential connective.[3] We consider additives in Chapter 7. Examples of multiplicative connectives are the

[2] It is called 'BCI' after the names of the combinators (pure lambda terms) the type schemata of which are the axiom schemata of a Hilbert-style presentation of the logic:

(i)

Combinator	Lambda term	Type/axiom schema
B	$\lambda x\lambda y\lambda z(x\ (y\ z))$	$(B \to C) \to ((A \to B) \to (A \to C))$
C	$\lambda x\lambda y\lambda z((x\ z)\ y)$	$(A \to (B \to C)) \to (B \to (A \to C))$
I	$\lambda x x$	$A \to A$

[3] The arithmetic terms refer to the complexity of Cut-elimination of the connective classes.

978–0–19–958985–2 (Hbk) 978–0–19–958986–9 (Pbk) 04-Morrill-c04 Morrill (Typeset by SPi, Chennai) 53 of 80 July 1, 2010 21:24
OUP UNCORRECTED PROOF – REVISES, 1/7/2010, SPi

times and divisions of the Lambek calculus, which is a non-commutative intuitionistic linear logic. Exponential connectives were entirely new kinds of modalities licensing controlled use of structural rules. We consider a variation in Chapter 5.

In addition, Girard (1987) introduced a new proof format, *proof nets* (for multiplicative classical linear logic). Proof nets are a deeper representation than both sequent calculus and natural deduction.[4] They form a graphical syntax which, as Girard *et al.* (1989) observe, captures the *essence même* of a proof. Roorda (1991) adapted proof nets to multiplicative non-commutative intuitionistic linear logic, that is, to the Lambek calculus. Proof nets are an ideal proof format for categorial logic: they are the syntactic structures of logical categorial grammar. The format of proof nets, unlike the format of sequent calculus, exhibits no spurious ambiguity and proof nets play the role in logical categorial grammar that parse trees play in phrase structure grammar. In this chapter we present, in terms of proof net synthesis, a nondeterministic incremental parsing algorithm for Lambek categorial grammar which consists purely in lexical choice and the complementization of syntactic valencies by shift/reduce syntactic choice. Furthermore, we associate with it a simple metric of complexity, the maximum number of unresolved valencies at any point in an analysis, that is the working memory stack-depth required for analysis according to the algorithm. We discuss how this complexity metric correctly predicts a wide variety of performance phenomena.

In Section 4.2 we define proof nets for the Lambek calculus. In Section 4.3 we define the extraction of semantics from a proof net analysis. In Section 4.4 we specify the natural nondeterministic incremental parsing algorithm and complexity metric for the proof net syntactic structures. We illustrate by reference to garden-pathing. In Section 4.5 we compare with human performance the predictions of logical categorial grammar and the complexity metric in relation to centre embedding unacceptability, left to right quantifier scope preference, preference for the lower attachment of adverbial phrases and of possessive clitics, heavy noun phrase shift, and preference for the passivization of nested sentential subjects.

4.2 Proof nets for the Lambek calculus

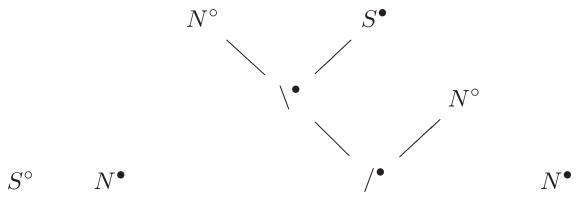
A proof net packs a proof into a minimal graphical structure. In sequent calculus logical inference rules for each connective decompose formulas into subformulas. Consider for example the following sequent proof:

[4] Or rather, it is sometimes said that proof nets are the natural deduction of linear logic.

(3)
$$\dfrac{N \Rightarrow N \qquad \dfrac{N \Rightarrow N \qquad S \Rightarrow S}{N,\, N\backslash S \Rightarrow S}\backslash L}{N,\, (N\backslash S)/N,\, N \Rightarrow S}/L$$

At the leaves, identity axioms relate antecedent and succedent atoms. In the corresponding proof net types and subtypes, marked by • and ○ for antecedent and succedent respectively, are unfolded into formula trees and complementary atomic leaves are connected. First, we obtain a *proof frame* by unfolding the endsequent types:

(4)



Each local tree in a proof frame corresponds to a sequent inference. We obtain a *proof structure* by connecting each atomic leaf with another complementary atomic leaf, corresponding to an identity axiom. Proof structures must satisfy further correctness criteria in order to qualify as well-formed representations of proofs, in which case they are called *proof nets*, so the proof nets are a proper subset of the proof structures. Continuing our example, the proof structure in Fig. 4.1 is also a proof net.

Exercise 4.1. The following two sequent proofs are represented by a single proof net. What do you think it is?

$$\dfrac{CN \Rightarrow CN \qquad \dfrac{N \Rightarrow N \qquad S \Rightarrow S}{N,\, N\backslash S \Rightarrow S}\backslash L}{N/CN,\, CN,\, N\backslash S \Rightarrow S}/L \qquad \dfrac{\dfrac{CN \Rightarrow CN \qquad N \Rightarrow N}{N/CN,\, CN \Rightarrow N}/L \qquad S \Rightarrow S}{N/CN,\, CN,\, N\backslash S \Rightarrow S}\backslash L$$

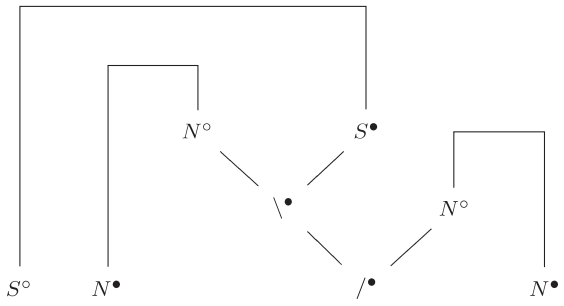
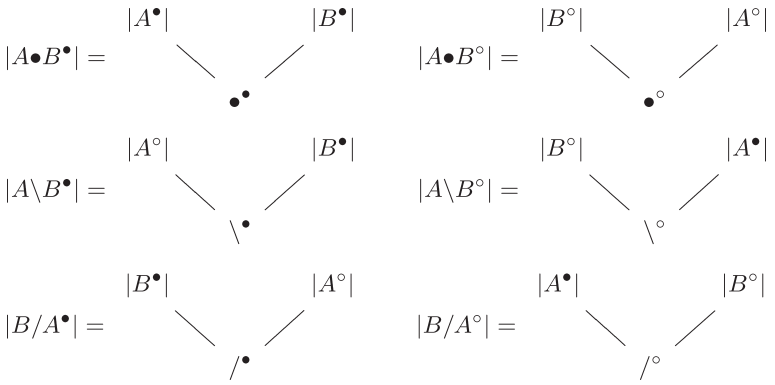


FIGURE 4.1. Proof net for $N,\, (N\backslash S)/N,\, N \Rightarrow S$

A *polar type* A^p comprises a type A together with a *polarity* $p = \bullet$ (*input*) or $\circ$ (*output*). In relating proof nets to sequent calculus and natural deduction, polarities indicate sequent sidedness: a type of input polarity corresponds to an antecedent type, and a type of output polarity corresponds to a succedent type.

The *polar type tree* $|A^p|$ of a polar type A^p is the ordered tree defined by the following polar translation function:

(5) $|P^{\,p}|{=}P^{\,p}$ if P is atomic



We see that the polar type tree of a polar type is basically its parse tree/formula tree decorated with polarities. Observe that the polar translation function transmits polarity from a mother node to its daughters according to the distribution in the corresponding sequent rule of the corresponding active type and its subtypes in the conclusion and the premise(s). For example, the translation of $A{\bullet}B^{\bullet}$ corresponds to the $\bullet L$ sequent rule. In that rule, both the subtypes of the active type have antecedent occurrences in the premise. Accordingly, the daughters in the image of the translation function are both of input polarity.

Exercise 4.2. Check that in the other five clauses of the translation function, the distributions of polarities also follow the sequent rules.

Let us recall that the sequents of the Lambek calculus are intuitionistic, having always a single succedent type, and always have non-empty antecedents, so that, for example, there is no inference from $A \Rightarrow A$ to $\Rightarrow A\backslash A$ (the latter is not even well-formed as a sequent). We define the *proof frame* of a sequent $A_0, \ldots, A_n \Rightarrow A$ as the sequence of polar type trees $\langle |A^{\circ}|, |A_0{}^{\bullet}|, \ldots, |A_n{}^{\bullet}| \rangle$. For example, the proof frame of the sequent (6) is given in Fig. 4.2, where we have additionally numbered the leaves.

(6) $S/(N\backslash S), (N\backslash S)/N, (S/N)\backslash S \Rightarrow S$

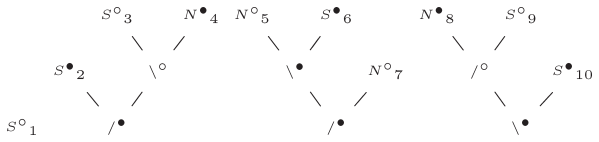


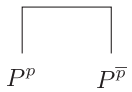
FIGURE 4.2. Proof frame of $S/(N\backslash S)$, $(N\backslash S)/N$, $(S/N)\backslash S \Rightarrow S$

As we shall see, two proof nets can be built on this proof frame.

The local tree unfolded in a clause of the definition of the translation function (2) is called a *logical link*. A logical link in a proof net corresponds to a sequent inference rule, but it represents only the active types themselves. In a non-commutative system we also need to take order into account. The daughters of the logical links are ordered, but observe that in the unfoldings of output types, the order of the subtypes is commuted in the daughters. This is because polarities are read as affirmative or negative and in a non-commutative logic negation commutes the subformulas in the de Morgan laws. Philippe de Groote (p.c.) provides the following intuition. Consider going from a to c via b: first from a to b and then from b to c. The reverse (negation) of this is not to go from b to a and then from c to b, but to go first from c to b (the negation of the second operand) and then from b to a (the negation of the first operand). That is, the subformulas/operands are commuted.[5]

We define the *complement* $\overline{X}$ of a polar type X by $\overline{A^\bullet} = A^\circ$ and $\overline{A^\circ} = A^\bullet$. Two polar types are *complementary* if and only if they are the complements of each other. An *axiom link* on a proof frame is a pair of complementary leaves. We draw one thus as a connecting line, corresponding to an id axiom instance $P \Rightarrow P$ in sequent calculus:

(7)



An *axiom linking* for a proof frame is a set of axiom links with at most one axiom link per leaf and which is *planar*, that is there are no two axiom links (i, k) and (j, l) such that $i < j < k < l$. Geometrically, planarity means that where the polar type trees are ordered on a line, the axiom links can be drawn in the half-plane (on top) without crossing lines. Planarity corresponds to non-commutative order.

A *partial proof structure* (PPS) is a proof frame with zero or more axiom links. A *proof structure* is a frame together with an axiom linking that links

[5] By symmetry we could have chosen to commute the input unfoldings instead, but then our word order would have appeared from right to left on the page.

every leaf. Thus an *axiom linking* of a proof structure is a partitioning of its leaves into complementary pairs where the connections of paired leaves do not cross. Not every proof structure represents a proof. To do so a proof structure must further satisfy certain proof net conditions, but these conditions are not very transparent.

To take into account the arities of sequent rules, that is whether the active subtypes go into the same subproof (unary rules) or different subproofs (binary rules) we will need a global correctness condition on proof nets that makes reference to the arities of rules. We define here the *℘-links* as those which are unary, that is those with mother $\bullet^{\bullet}$, $\backslash^{\circ}$or $/^{\circ}$.[6]

A *switching* of a PPS is a graph resulting from removing one of the edges from each ℘-link. A *proof net* is a proof structure in which (i) every switching is a connected and acyclic graph (Danos–Regnier acyclicity and connectedness; see Danos and Regnier 1989), and (ii) no axiom link connects the leftmost and rightmost descendent leaves of an output division (we call this Retoré no subtending; see de Groote and Retoré 2003). Danos–Regnier (DR) connectedness and acyclicity takes care of rule arity. Retoré no subtending prohibits empty antecedents.

(8) Theorem. A sequent is a theorem of (i.e., derivable in) the Lambek sequent calculus if and only if there is an axiom linking which forms a proof net on its frame.

Figs. 4.3 and 4.4 show the two proof nets that can be built on the frame of Fig. 4.2.

Actually, DR connectedness can be omitted because Fadda and Morrill (2005) show that in view of the intuitionistic nature of Lambek sequents (that there is exactly one root of output polarity), every proof structure which satisfies DR acyclicity also satisfies DR connectedness. Therefore we need only check for DR acyclicity (and no subtending). We call a partial proof structure *correct* if and only if it satisfies DR acyclicity and no subtending.

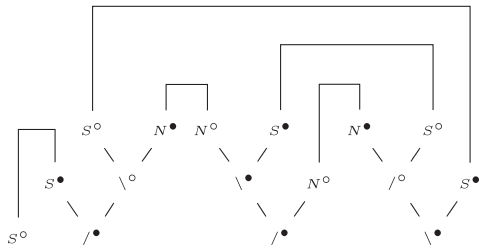


Figure 4.3. A proof net for $S/(N\backslash S)$, $(N\backslash S)/N$, $(S/N)\backslash S \Rightarrow S$

[6] These are the cases which compile into linear multiplicative disjunction ℘ ('par'), which has a unary rule.

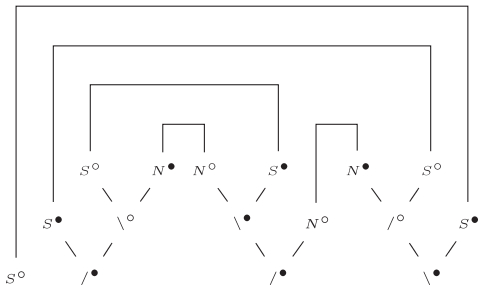


FIGURE 4.4. Another proof net for $S/(N\backslash S)$, $(N\backslash S)/N$, $(S/N)\backslash S \Rightarrow S$

(9) COROLLARY. A sequent is a theorem of the Lambek calculus if and only if there is an axiom linking which forms a correct proof structure on its frame.

Therefore we can carry out Lambek theorem-proving by building up proof nets incrementally, checking for correctness (DR acyclicity and no subtending) at each step. We have a Lambek theorem if and only if we succeed in linking all the leaves while satisfying these criteria.

4.3 The semantic trip and the semantic reading of a proof net

The semantics associated with a categorial proof net, that is the proof as a lambda term (intuitionistic natural deduction proof, under the Curry–Howard correspondence) is extracted by associating a distinct index with each output division node and travelling as follows, starting by going up at the unique output root (De Groote and Retoré, 1996):

(10)
- travelling up at the mother of an output division link, perform the lambda abstraction with respect to the associated index of the result of travelling up at the daughter of output polarity;
- travelling up at the mother of an output product link, form the ordered pair of the result of travelling up at the right daughter (first component) and the left daughter (second component);
- travelling up at one end of an axiom link, continue down at the other end;
- travelling down at an (input) daughter of an input division link, perform the functional application of the result of travelling down at the mother to the result of travelling up at the other (output) daughter;

- travelling down at the left (resp. right) daughter of an input product link, take the first (resp. second) projection of the result of travelling down at the mother;
- travelling down at the (input) daughter of an output division link, return the associated index and bounce;
- travelling down at a root, return the associated lexical semantics and bounce.

We call this special trip around the proof net by which the semantics is extracted the 'semantic trip'. The semantic trip begins at the unique root of polarity output (the 'origin'), starting by travelling upwards. Edges are followed in a uniform direction until we come to logical links. Then the travel instructions are followed, which we illustrate diagrammatically in Fig. 4.5. The labels of edges taken generate the successive symbols of the semantic form. Lambda variables are unique to their link. When we arrive down at an input polarity root, the associated lexical semantics is inserted, and the trip 'bounces' back up. The trip visits each node twice, once travelling upwards and once travelling downwards, and crosses each edge twice, once in each direction. It ends when it arrives back down at the origin.

For example, where we associate with the three left-to-right input roots of Fig. 4.1 the lexical semantics *j*, *love*, and *m* respectively, the semantic trip yields ((*love m*) *j*). And where we associate with the three left-to-right roots of Figs. 4.3 and 4.4 the lexical semantics *someone*, *love*, and *everyone* respectively, the semantic trip yields for Fig. 4.3 (*someone* λx(*everyone* λy((*love y*) *x*))).

Exercise 4.3. What does the semantic trip yield for Fig. 4.4?

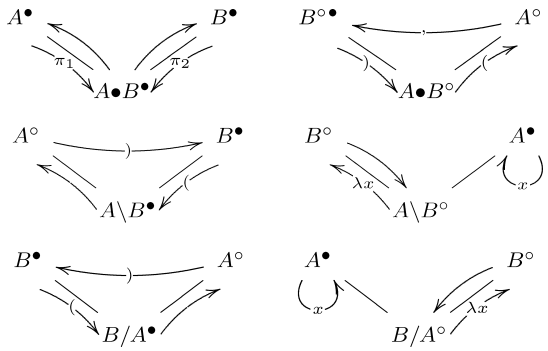


FIGURE 4.5. Semantic trip travel instructions

4.4 Incremental parsing algorithm and complexity metric

4.4.1 Parsing algorithm

We present a stack-based incremental parsing algorithm for Lambek categorial grammar. There is nondeterminism in respect of lexical choice (lexical ambiguity) and in respect of syntactic shift or reduce choice (syntactic ambiguity).

The algorithm works left-to-right through a buffer representing the successive words in the input speech stream. There are two stacks, a global stack and a local stack. When a next segment of the buffer is chosen as a lexical expression, the leaves of the polar tree of its lexical type are put on the local stack. Each such leaf L is then considered in turn, either reducing with a complementary leaf on the top of the global stack (establishing an axiom link ending at L), or pushing it onto the global stack (meaning that an axiom link will have to be established later starting at L).

We present the algorithm as a state transition system. A state comprises five components:

(11)
- A **global stack** G. This is the main stack, containing unresolved syntactic valencies. We define the complexity of a parse as the maximum depth of the global stack.
- A **local stack** Ls. This is an auxiliary stack, containing the remaining valencies of the lexical type most recently inserted by lexical lookup.
- A **buffer** α. The words remaining in the input speech stream.
- A **frame** F. The proof frame built by lexical choice so far.
- A set of **axiom links** X. The set of axiom links (syntactic dependencies) established so far on the proof frame.

We represent a state:

(12) G, Ls, α, F, X

The parsing algorithm is given in Fig. 4.6. We use Prolog notation for lists/stacks, thus [] is the empty list and $[H|T]$ is the list with head H and tail T. $\oplus$ is list concatenation. $\#(F)$ is the total number of type atom occurrences in the types of F; this is for numbering the leaves of the proof frame. $|A^\bullet|_n$ is like $|A^\bullet|$ but in addition numbers the leaves of the polar type tree from left to right starting from n. Where T is a (numbered) polar type tree, fringe(T) is the list of its leaves from left to right (product or yield). Note how the sideconditions on *REDUCE* incrementally ensure no subtending and DR acyclicity, i.e. correctness.

4.4.2 Parsing examples (garden pathing) and complexity metric

By way of illustration of the parsing algorithm and the complexity metric we consider garden pathing (Bever, 1970):

$$\frac{process\ \alpha}{[S^\circ{}_0], [], \alpha, [S^\circ{}_0], \emptyset}\ START$$

$$\frac{G, [], \alpha \oplus \alpha', F, X}{G, \mathrm{fringe}(|A^\bullet|_{\#(F)}), \alpha', F \oplus [|A^\bullet|_{\#(F)}], X}\ LEX, where\ \alpha : A$$

$$\frac{G, [L_i|Ls], \alpha, F, X}{[L_i|G], Ls, \alpha, F, X}\ SHIFT$$

$$\frac{[L_i|G], [L_j|Ls], \alpha, F, X}{G, Ls, \alpha, F, X \cup \{(i,j)\}}\ REDUCE,$$ where L_i and L_j are complementary and do not subtend in frame F, and every path between them in F together with axiom links X crosses both edges of some $\wp$-link.

$$\frac{[], [], [], F, X}{accept}\ END$$

FIGURE 4.6. Parsing algorithm for Lambek categorial grammar

(13) a. The horse raced past the barn.
b. ?The horse raced past the barn fell.

(14) a. The boat floated down the river.
b. ?The boat floated down the river sank.

(15) a. The dog that knew the cat disappeared.
b. ?The dog that knew the cat disappeared was rescued.

Typically, although the (b) sentences are perfectly well-formed they are perceived of as being ungrammatical apparently due to a strong tendency to interpret their initial segments as in the (a) sentences.

Let us assume the following lexical assignments:

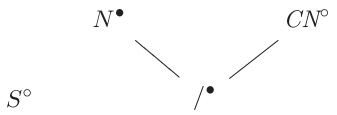
(16)

barn	: CN	: *barn*
horse	: CN	: *horse*
past	: $((N\backslash St)\backslash(N\backslash St))/N$	: $\lambda x\lambda y\lambda z(past\ x\ (y\ z))$
raced	: $N\backslash S+$	: *race*
the	: N/CN	: *the*

The feature + on S marks the projection of a tensed verb form; a verb phrase modified by 'past' need not be tensed and the feature is marked with a variable t in that lexical entry. Let us consider the incremental processing of (13a) as proof net construction. We assume initially that an S is expected; after perception of the word 'the' there is the following partial proof net (for simplicity we omit features, included in lexical entries, from proof nets):

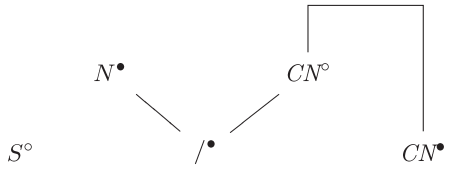
978–0–19–958985–2 (Hbk) 978–0–19–958986–9 (Pbk) 04-Morrill-c04 Morrill (Typeset by SPi, Chennai) 62 of 80 July 1, 2010 21:24

(17)



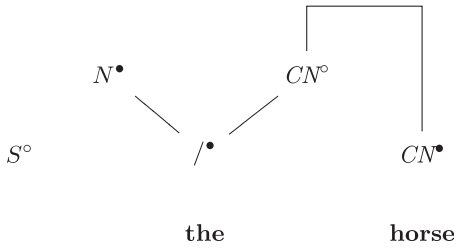
Here there are three unmatched valencies/unresolved dependencies; no axiom links can yet be placed, but after 'horse' we can build:

(18)



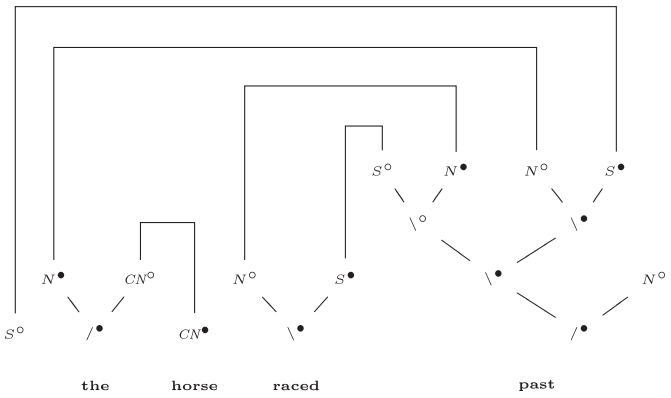
Now there are only two unmatched valencies. After 'raced' we have, on the correct analysis, the following:

(19)



Note that linking the Ns is possible, but we are interested in the history of the analysis which turns out to be correct, and in that analysis the verb valencies are matched by the adverb that follows:

(20)



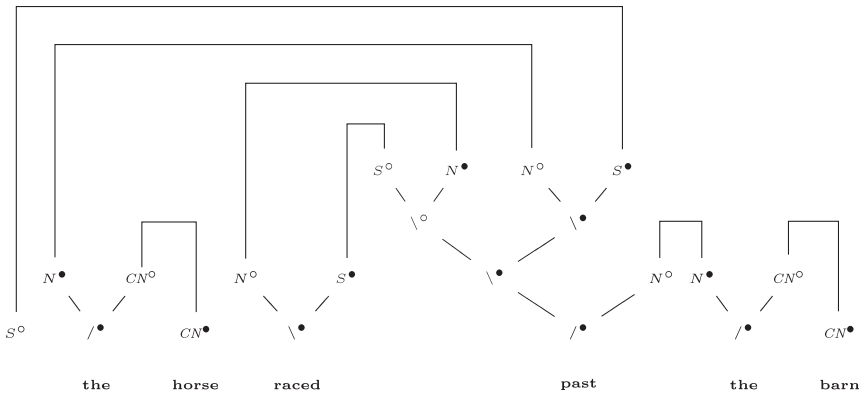
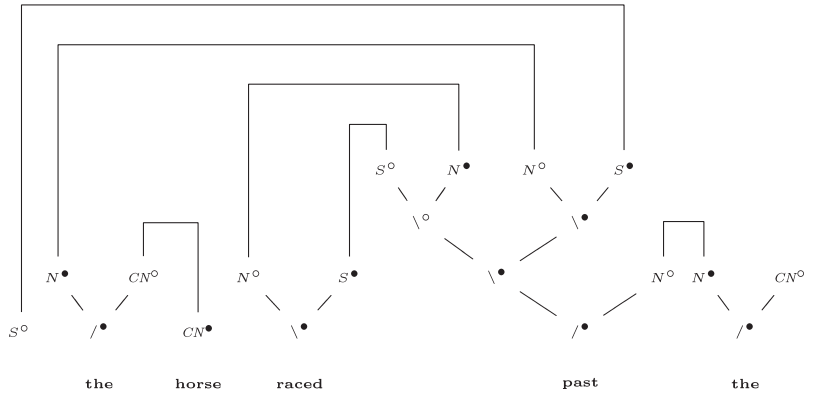


FIGURE 4.7. *the horse raced past the barn*

Observe that a cycle is created, but as required it crosses both edges of a $\wp$-link. At the penultimate step we have:

(21)



The final proof net analysis is given in Fig. 4.7. Carrying out the semantic trip we obtain (22a), which is logically equivalent to (22b).

(22) a. $(\lambda x \lambda y \lambda z(\mathit{past}\ x\ (y\ z))\ (\mathit{the\ barn})\ \lambda w(\mathit{race}\ w)\ (\mathit{the\ horse}))$
b. $(\mathit{past}\ (\mathit{the\ barn})\ (\mathit{race}\ (\mathit{the\ horse})))$

The analysis of (13b) is less straightforward. Whereas in (13a) 'raced' expresses a one-place predication ('go quickly'), in (13b) it expresses a two-place predication (there was some agent racing the horse); 'horse' is modified by an agentless passive participle, but the adverbial 'past the barn' is modifying 'race'. Within the confines of the Lambek calculus the characterization we offer assumes the lexical assignment to the passive participle given in the following.[7]

[7] In general grammar requires the expressivity of more powerful categorial logics than just Lambek calculus (see Part II); however, so far as we are aware, the characterizations we offer within the Lambek calculus bear the same properties with regard to our processing considerations as their more

(23) **fell** : $N\backslash S+$: *fall*
raced : $((CN\backslash CN)/(N\backslash(N\backslash S-)))\bullet(N\backslash(N\backslash S-))$
: $(\lambda x\lambda y\lambda z[(y\ z) \wedge \exists w(x\ z\ w)], race2)$

Here 'raced' is classified as the product of an untensed transitive verbal type, which can be modified by the adverbial 'past the barn' by composition, and an adnominalizer of this transitive verbal type. According to this, (13b) has the proof net analysis given in Fig. 4.8. The semantics extracted is (24a), equivalent to (24b)

(24) a. $(fall\ (the\ (\pi_1(\lambda x\lambda y\lambda z[(y\ z) \wedge \exists w(x\ z\ w)], race2)\ \lambda n\lambda o(\lambda u\lambda v\lambda w(past\ u\ (v\ w))$
$(the\ barn)\ \lambda r(((\pi_2(\lambda p\lambda s\lambda t[(s\ t) \wedge \exists q(p\ t\ q)], race2)\ n\ r)\ o)horse)))$
b. $(fall\ (the\ \lambda x[(horse\ x) \wedge \exists y(past\ (the\ barn)\ (race2\ x\ y))]))$

Let us assign to each proof net analysis a complexity profile which indicates, before and after each word, the number of unmatched literals, that is unresolved valencies or dependencies, according to the processing up to that point. This is a measure of the course of memory load in optimal incremental processing. We are not concerned here with the resolution of lexical ambiguity or serial backtracking: we are supposing sufficient resources that the non-determinism of selection of lexical entries and their parallel consideration is not the critical burden. Rather, the question is: which among parallel competing analyses places the least load on memory?

The complexity profile is easily read off a completed proof net: the complexity in between two words is the number of axiom links bridging over at that point. Thus for (13a) and (13b) analysed in Figs. 4.7 and 4.8 the complexity profiles are as follows:

(25)

6										b										
5																				
4							b			a										
3			ab										b		b					
2							a						b							
1	ab														a		a			
0																	a		b	
a.		the		horse		raced		past		the		barn								
b.		the		horse		raced		past		the		barn		fell						

We see that after the first words the complexity of the correct analysis of (13b) is consistently higher than that of its garden path (13a), just as would be expected on the assumption that in (13b) the less costly but incorrect analysis is salient.

sophisticated categorial logic refinements, because the latter concern principally generalizations of word order, whereas the semantic dependencies on which our complexity metric depends remain the same.

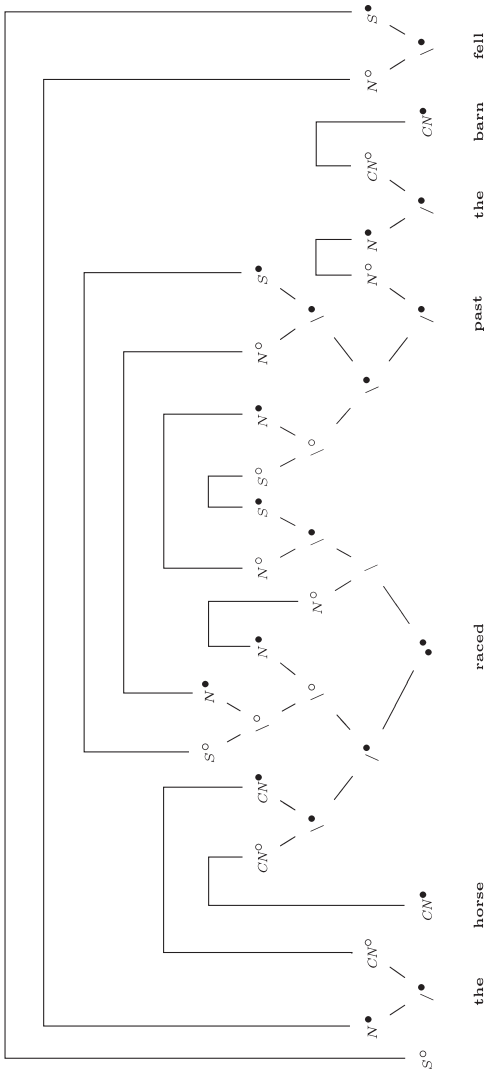


FIGURE 4.8. *the horse raced past the barn fell*

978–0–19–958985–2 (Hbk) 978–0–19–958986–9 (Pbk) 04-Morrill-c04 Morrill (Typeset by SPi, Chennai) 66 of 80 July 1, 2010 21:24
OUP UNCORRECTED PROOF – REVISES, 1/7/2010, SPi

4.5 Predicting performance

4.5.1 Centre embedding unacceptability

Turning now to the performance phenomenon mentioned at the beginning of the chapter, for subject and object relativization we assume the relative pronoun lexical assignments (26).

(26) **that** : $(CN\backslash CN)/(N\backslash S+)$: $\lambda x\lambda y\lambda z[(y\ z) \wedge (x\ z)]$
that : $(CN\backslash CN)/(S+/N)$: $\lambda x\lambda y\lambda z[(y\ z) \wedge (x\ z)]$

Sentence (1b) is analysed in Fig. 4.9. Sentence (2b) is analysed in Fig. 4.10. Let us compare the complexities:

(27)

9								b				
8									b			
7						b	b					
6					b							
5										b		
4			ab	ab			a	a				
3		ab			a	a			a	a		
2											ab	
1	ab											
0												ab
a.	the	dog	that	chased	the	cat	that	saw	the	rat	barked	
b.	the	cheese	that	the	rat	that	the	cat	saw	ate	stank	

Again, the profile of (2b) is higher[8]. We suggest that this is what causes the unacceptability of centre embedding.

4.5.2 Left-to-right quantifier scope preference

Left-to-right quantifier scope preference is illustrated by:

(28) a. Someone loves everyone.
b. Everyone is loved by someone.

Both sentences exhibit both quantifier scopings:

(29) a. $\exists x \forall y(love\ y\ x)$
b. $\forall y \exists x(love\ y\ x)$

However, while the dominant reading of (28a) is (29a), that of (28b) is (29b), that is, the preference is for the first quantifier to have wider scope. Note that the same effect is observed when the quantifiers are swapped:

(30) a. Everyone loves someone.
b. Someone is loved by everyone.

[8] Indeed it rises above 7–8, thus reaching what are usually taken to be the limits of short-term memory.

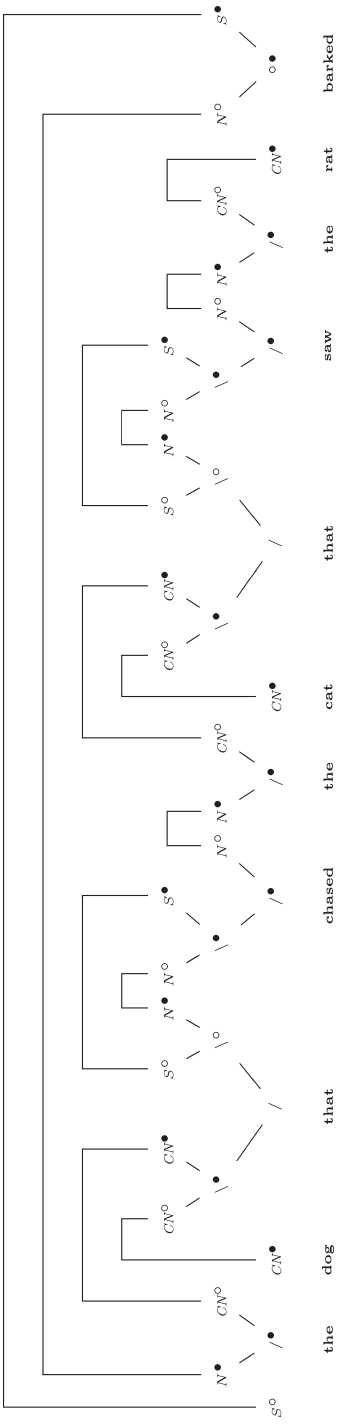


FIGURE 4.9. *the dog that chased the cat that saw the rat barked*

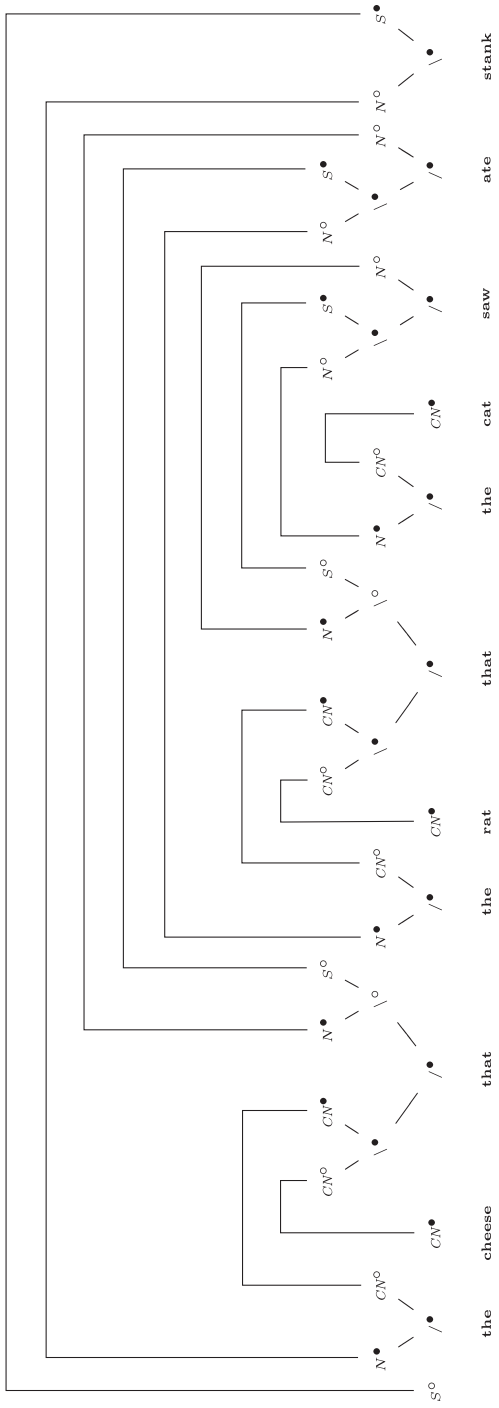


FIGURE 4.10. *the cheese that the rat that the cat saw ate stank*

While both sentences in (30) have both quantifier scopings, the preferred readings give the first quantifier wide scope.

A rudimentary account of sentence-peripheral quantifier phrase scoping is obtained in Lambek categorial grammar by means of lexical assignments such as the following (for a more refined treatment, without lexical ambiguity, see Chapter 6):

(31) **everyone** : $St/(N\backslash St)$: $\lambda x\forall y(x\ y)$
everyone : $(St/N)\backslash St$: $\lambda x\forall y(x\ y)$
someone : $St/(N\backslash St)$: $\lambda x\exists y(x\ y)$
someone : $(St/N)\backslash St$: $\lambda x\exists y(x\ y)$

Then one analysis of (28a) is that given in Fig. 4.11. This is the subject wide scope analysis: its extracted and simplified semantics is as in (32).

(32) a. $(\lambda x\exists y(x\ y)\ \lambda u(\lambda x\forall y(x\ y)\ \lambda v(love\ v\ u)))$
b. $\exists x\forall y(love\ y\ x)$

A second analysis is that given in Fig. 4.12. This is the object wide scope analysis: its extracted and simplified semantics is as in (33).

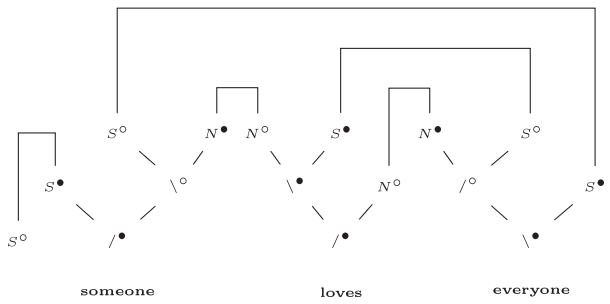


FIGURE 4.11. *someone loves everyone* (∃∀)

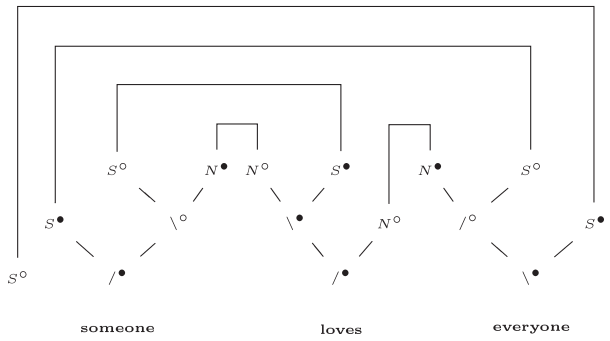


FIGURE 4.12. *someone loves everyone* (∀∃)

(33) a. $(\lambda x \forall y(x\ y)\ \lambda v(\lambda x \exists y(x\ y)\ \lambda u(love\ v\ u)))$
b. $\forall y \exists x(love\ y\ x)$

Let us compare the complexity profiles of the two readings:

(34)

4		b			
3			ab		
2		a			
1	ab				
0				ab	
a.	someone	loves	everyone		∃∀ (subject wide scope, Fig. 4.11)
b.					∀∃ (object wide scope, Fig. 4.12)

At the only point of difference the subject wide scope reading, the preferred reading, has the lower complexity.

For the passive (28b) let there be assignments as in (35). The preposition 'by' projects an agentive adverbial phrase; 'is' is a functor over (post-)nominal modifiers (*the man outside*, *John is outside*, etc.) and passive 'loved' is treated exactly like passive 'raced' in (23).

(35) **by** : $((N\backslash S{-})\backslash(N\backslash S{-}))/N\ :\ \lambda x\lambda y\lambda z[[z = x] \wedge (y\ z)]$
is : $(N\backslash S{+})/(CN\backslash CN)\ :\ \lambda x\lambda y(x\ \lambda z[z = y]\ y)$
loved : $((CN\backslash CN)/(N\backslash(N\backslash S{-})))\bullet(N\backslash(N\backslash S{-}))$
: $(\lambda x\lambda y\lambda z[(y\ z) \wedge \exists w(x\ z\ w)], love)$

A ∀∃ analysis of (28b) is given in Fig. 4.13. This has semantics, after some simplification, as in (36), which is equivalent to (33).

(36) $\forall x \exists y \exists z[[y = z] \wedge (love\ x\ y)]$

An ∃∀ analysis of (28b) is given in Fig. 4.14. This has semantics, after some simplification, as in (37), which is equivalent to (32).

(37) $\exists y \forall x \exists z[[z = y] \wedge (love\ x\ z)]$

Again, the preferred reading has the lower complexity profile:

(38)

6				b			
5							
4		b	b	a			
3					ab		
2		a	a				
1	ab						
0						ab	
a.	everyone	is	loved	by	someone		∀∃ (Fig. 13)
b.							∃∀ (Fig. 14)

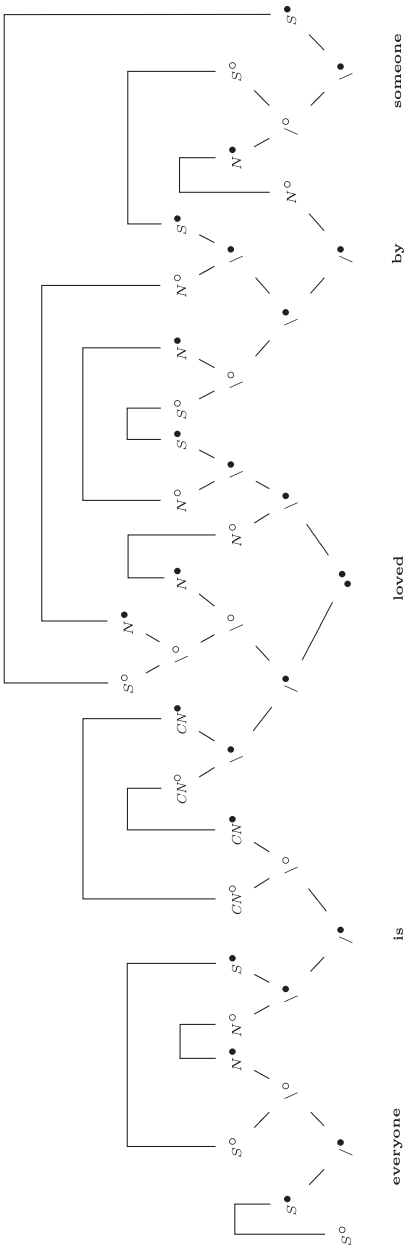


FIGURE 4.13. *everyone is loved by someone* (∀∃)

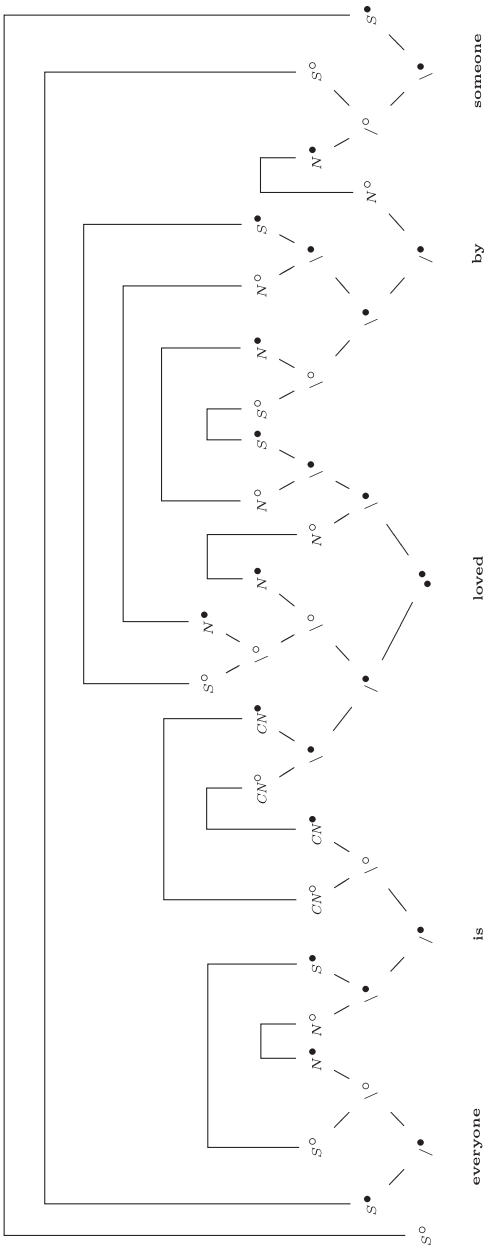


FIGURE 4.14. *everyone is loved by someone* (∃∀)

978–0–19–958985–2 (Hbk) 978–0–19–958986–9 (Pbk) 04-Morrill-c04 Morrill (Typeset by SPi, Chennai) 73 of 80 July 1, 2010 21:24
OUP UNCORRECTED PROOF – REVISES, 1/7/2010, SPi

4.5.3 Preference for lower attachment

Kimball (1973: 27) observes that in a sentence such as (39), three ways ambiguous according to the attachment of the adverb, the lower the attachment is, the higher the preference (what he terms 'Right Association').

(39) Joe said that Martha believed that Ingrid fell today.

In Fig. 4.15 we give the analyses for the highest, the middle, and the lowest attachments. (We now abbreviate proof nets by flattening formula trees into their linear representations; since this conceals the order switching of output links in this notation the axiom links may cross, belying the underlying planarity.) The complexity profiles are:

(40)

6					c			bc		
5			c	c		bc	bc			
4									abc	
3										
2		abc			ab			a		
1	abc		ab	ab		a	a			
0										abc
a.										lowest attachment
b.	Joe	said	that	Martha	believed	that	Ingrid	fell	today	middle attachment
c.										highest attachment

The same effect occurs strongly in (41), where the preferred reading is the one given by the lowest attachment, even though that one is the nonsensical reading.

(41) the book that shocked Mary's title

The analyses are given in Fig. 4.16. The complexities are thus:

(42)

4			a	a					
3		a				a			
2			b	b			ab		
1	ab	b				b		ab	
0									ab
a.									sensical
b.	the	book	that	shocked	Mary	's	title		nonsensical

4.5.4 Heavy noun phrase shift

Our account appears to explain the preference for heavy noun phrases to appear at the end of the verb phrase (heavy noun phrase shift). Of the following the second is more acceptable:

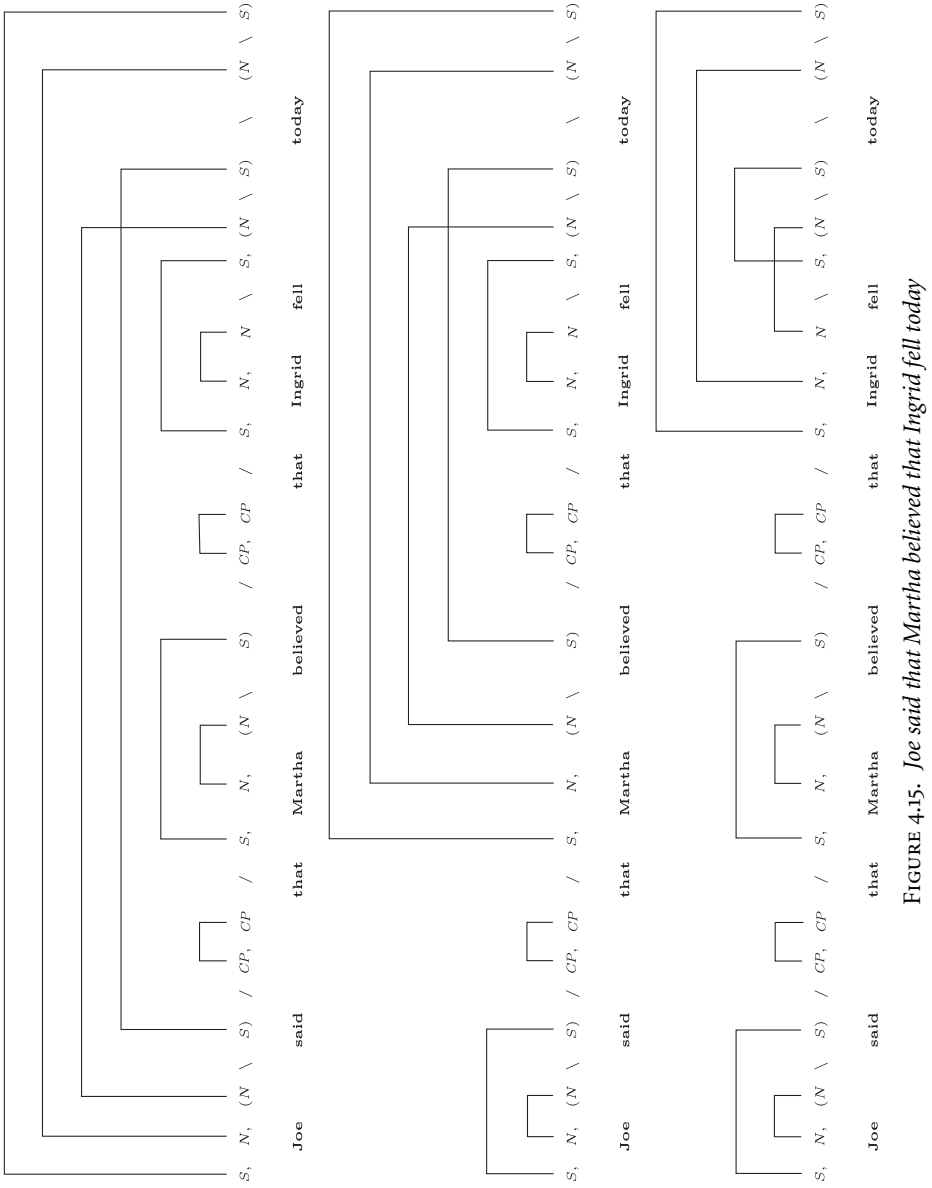


FIGURE 4.15. *Joe said that Martha believed that Ingrid fell today*

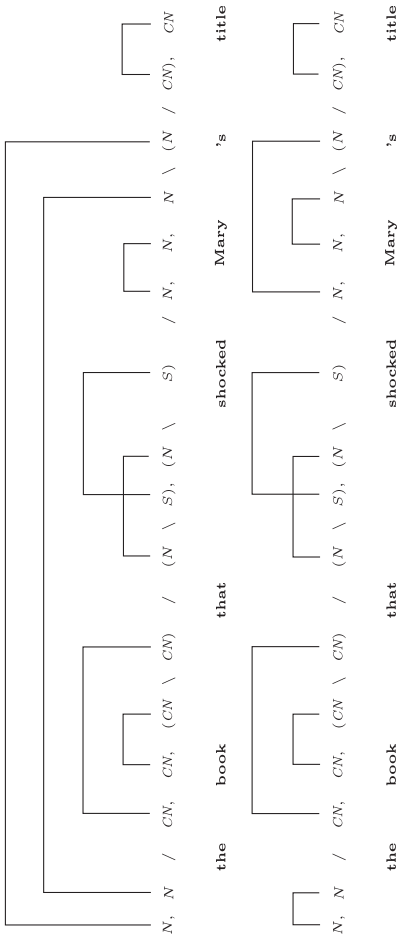


FIGURE 4.16. *the book that shocked Mary's title* (sensical and nonsensical)

FIGURE 4.17. *John gave the painting that Mary loved to Bill* vs. *John gave Bill the painting that Mary loved*

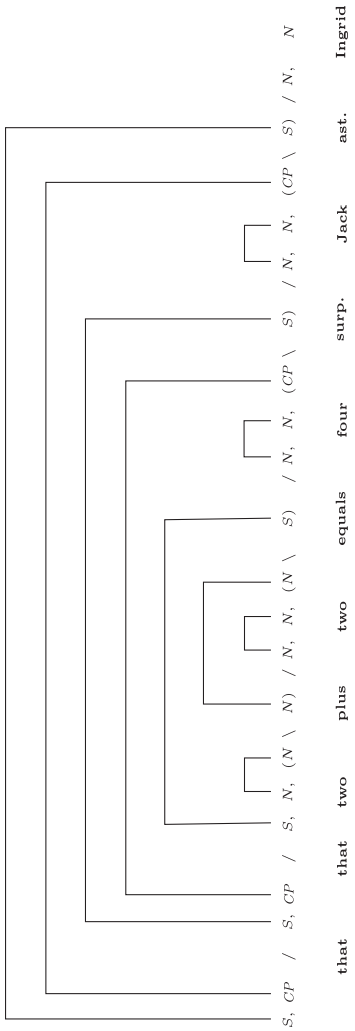


FIGURE 4.18. *that that two plus two equals four surprised Jack astonished Ingrid*

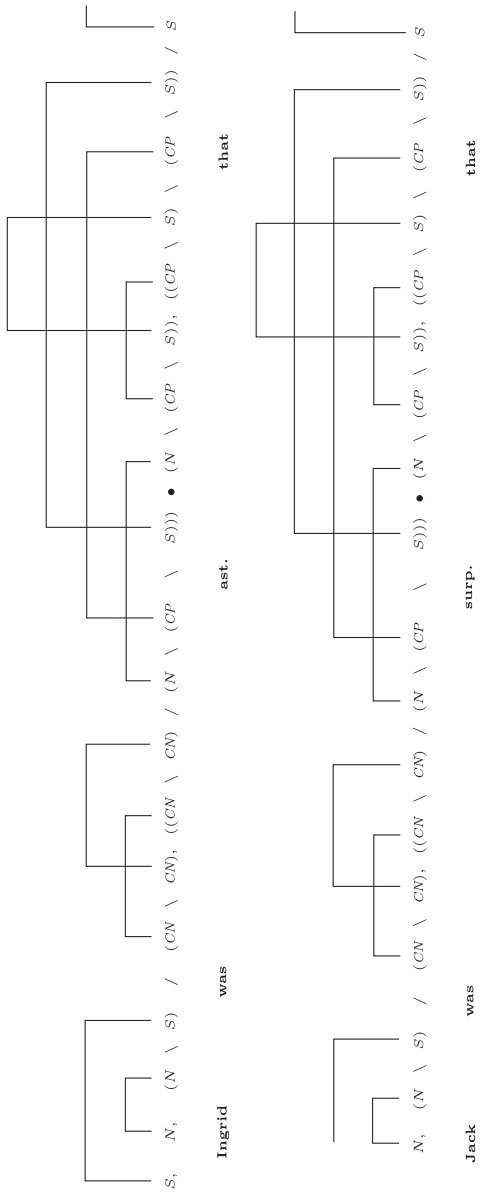


FIGURE 4.19. *Ingrid was astonished that Jack was surprised that two plus two equals four*

FIGURE 4.20. Complexity profiles for active and passive nested sentential subjects

978–0–19–958985–2 (Hbk) 978–0–19–958986–9 (Pbk) 04-Morrill-c04 Morrill (Typeset by SPi, Chennai) 80 of 80 July 1, 2010 21:24

(43) a. ?John gave the painting that Mary loved to Bill.
b. John gave Bill the painting that Mary loved.

The analyses are given in Fig. 4.17. The complexities are thus:

(44)

4							a			
3					a	a		b		
2		ab	ab	a		b	b			
1	ab			b	b			a	a	
0									b	a
a.	John	gave	the	painting	that	Mary	loved	to	Bill	
b.	John	gave	Bill	the	painting	that	Mary	loved		

4.5.5 Preference for passivization of nested sentential subjects

A final dramatic example of unacceptability is provided by the following:

(45) a. That two plus two equals four surprised Jack.
b. ?That that two plus two equals four surprised Jack astonished Ingrid.
c. ??That that that two plus two equals four surprised Jack astonished Ingrid bothered Frank.

The passive paraphrases, however, seem more or less equally acceptable:

(46) a. Jack was surprised that two plus two equals four.
b. Ingrid was astonished that Jack was surprised that two plus two equals four.
c. Frank was bothered that Ingrid was astonished that Jack was surprised that two plus two equals four.

This is puzzling, since just about any theory of processing would expect passives to be no more acceptable than actives. For example, Clark and Clark (1977: 144) cite such examples as reasons for the abandonment of the theory of derivational complexity in transformational grammar.

Let us consider the predictions of our theory. In Fig. 4.18 we give the analysis of (45b) and in Fig. 4.19 that of (46b). It is very interesting to observe that in accordance with the actual acceptabilities the complexity profile of the latter is in general lower even though the analysis has more than twice the total number of links; the complexity profiles are given in Fig. 4.20.