

Chapter 1

ON BOUND ANAPHORA IN TYPE LOGICAL GRAMMAR

Glyn Morrill

In this paper we tackle the intrasentential anaphora quadrant of the intersentential and intrasentential anaphora and cataphora pie. Montague (1973) characterized personal pronouns as bound anaphors preceded by their antecedents and taking nominative forms in subject position and accusative forms in object position. In this paper we give a type logical formulation of such a view of anaphora (taking into account also reflexive pronouns). Jäger (2001) broaches much the same task by a proof-theoretic definition which, it is to be hoped, can be given a model-theoretic justification, whereas here we take ‘type logical’ to mean that the definition is model-theoretic in the first place.

Moortgat (1988) initiates discontinuity which is type logical in this sense, but encounters difficulties with indeterminacy of separation points. Versmissen (1991) proposes to mark a separation point by a pointer and Solias (1992) to encode separation points as the positions inbetween the operands of an absolutely free tuple operation. Morrill (2002) aspires to generalize various efforts by means of a separator constant operator which is close to Versmissen, except that there are multiple separator occurrences, and close to Solias, except that there is no nesting. A constant operator also appears in Moortgat (1996). In this paper, a development of Morrill (2000b), we invoke what would appear to be sufficient for a type logical treatment of intrasentential anaphora.

In section 1 we define the relevant formalism and in section 2 we present a type assignment calculus. In section 3 we present and exemplify the treatment of anaphora. In section 4 we discuss binding principles and argue that Principle B and the Delay of Principle B Ef-

*E-mail: morrill@lsi.upc.es, <http://www-lsi.upc.es/~morrill/>. Work partially supported by CICYT projects PB98-0937-C03-03 and TIC2002-04019-C03-01.

fect are not unexpected on our account. In the appendix we present a discontinuous sequent calculus which is free of structural rules.

1. FORMALISM

Let a *prosodic structure* be a free algebra $(L, +, 0, \$)$ of arity $(2, 0, 0)$ with generator set $V \cup \{\$\}$, such that $(L, +, 0)$ is a monoid. Let the mapping $| \cdot |_{\$}$ from L to naturals be defined by:

$$(1.1) \quad \begin{aligned} |s|_{\$} &= 0 & \text{if } s \in V \\ |\$|_{\$} &= 1 \\ |s_1 + s_2|_{\$} &= |s_1|_{\$} + |s_2|_{\$} \end{aligned}$$

We call $|s|_{\$}$ the *sort* of prosodic object s , and we define:

$$(1.2) \quad L^i = \{s \mid |s|_{\$} = i\}$$

Let the ternary relation W on L be the smallest such that:

$$(1.3) \quad W(s_1 + \$ + s_3, s_2, s_1 + s_2 + s_3) \text{ if } |s_3|_{\$} = 0$$

Note that when the sort of s_1 is greater than zero, there is a unique s_3 such that $W(s_1, s_2, s_3)$, which we denote $s_1 W s_2$: it is the result of replacing by s_2 the last $\$$ in s_1 .

Let there be a set A of *prosodic constants*, and a set B of *prosodic variables*, $\$ \notin A \cup B$. Then the sets $\mathcal{A}^i$ of *prosodic terms of sort i* for each natural i are the trees defined by:

$$(1.4) \quad \begin{aligned} \mathcal{A}^0 &::= A \mid B \\ \mathcal{A}^1 &::= \$ \\ \mathcal{A}^{i+j} &::= \mathcal{A}^i + \mathcal{A}^j \end{aligned}$$

A *prosodic form* is a prosodic term containing no prosodic variables.

A *prosodic interpretation* comprises a prosodic algebra $(L, +, 0, \$)$, a *prosodic assignment* w mapping from B into L^0 and a *prosodic valuation* v mapping from A into L^0 . Then the *prosodic value* $[\alpha] \in L^i$ of a prosodic term $\alpha \in \mathcal{A}^i$ relative to a prosodic interpretation with prosodic valuation v and prosodic assignment w is defined by:

$$(1.5) \quad \begin{aligned} [\mathbf{c}] &= v(\mathbf{c}) \\ [a] &= w(a) \\ [\$] &= \$ \\ [\alpha + \beta] &= [\alpha] + [\beta] \end{aligned}$$

Note that the prosodic value of a prosodic form is independent of any

prosodic assignment.

Prosodic terms α and β are *equivalent*, $\alpha \equiv \beta$, if and only if $[\alpha] = [\beta]$ in every prosodic interpretation. Clearly we have:

$$(1.6) \quad \alpha + (\beta + \gamma) \equiv (\alpha + \beta) + \gamma$$

So we can omit parentheses in prosodic terms.

Let the set $\mathcal{T}$ of *semantic types* be the trees defined by:

$$(1.7) \quad \mathcal{T} ::= e \mid o \mid \mathcal{T} \rightarrow \mathcal{T}$$

Then a *semantic structure* comprises a non-empty set E of *entities*, a non-empty set W of *worlds*, and a $\mathcal{T}$ -indexed family of sets $\{D_\tau\}_{\tau \in \mathcal{T}}$ such that:

$$(1.8) \quad \begin{aligned} D_e &= E \\ D_o &= \mathcal{P}(W) \text{ (propositions)} \\ D_{\tau \rightarrow \tau'} &= D_{\tau'}^{D_\tau} \end{aligned}$$

Let there be a set X_τ of *semantic variables* of each semantic type τ , and a set C_τ of *semantic constants* of each semantic type τ , including the *logical semantic constants*:

$$(1.9) \quad \begin{aligned} \neg &\in C_{o \rightarrow o} \\ \wedge, \vee, \rightarrow &\in C_{o \rightarrow (o \rightarrow o)} \\ \forall, \exists &\in C_{(e \rightarrow o) \rightarrow o} \end{aligned}$$

The sets Φ_τ of *semantic terms of type τ* for each semantic type τ are the trees defined by:

$$(1.10) \quad \begin{aligned} \Phi_\tau &::= C_\tau \mid X_\tau \mid (\Phi_{\tau' \rightarrow \tau} \Phi_{\tau'}) \\ \Phi_{\tau \rightarrow \tau'} &::= \lambda X_\tau \Phi_{\tau'} \end{aligned}$$

An occurrence of a semantic variable x in a semantic term is *free* if and only if it does not fall within any $\lambda x \phi$; otherwise it is *bound*. The result $\psi\{\phi/x\}$ of substituting semantic variable x (of semantic type τ) by semantic term ϕ (of semantic type τ) is the result of replacing by ϕ every free occurrence of x in ψ ; $\psi\{\phi/x\}$ is *free* if and only if no variable becomes bound in the process of replacement. A *semantic form* is a semantic term containing no free variables.

A *semantic interpretation* comprises a semantic structure $\{D_\tau\}_{\tau \in \mathcal{T}}$, a *semantic assignment* g mapping from each X_τ into D_τ , and a *semantic*

valuation f mapping from each C_τ into D_τ such that:

$$(1.11) \quad \begin{aligned} f(\neg)(m) &= \overline{m}^W \\ f(\wedge)(m) &= m' \mapsto m \cap m' \\ f(\vee)(m) &= m' \mapsto m \cup m' \\ f(\rightarrow)(m) &= m' \mapsto \overline{m}^W \cup m' \\ f(\forall)(m) &= \bigcap_{m' \in E} m(m') \\ f(\exists)(m) &= \bigcup_{m' \in E} m(m') \end{aligned}$$

The *semantic value* $[\phi]^g \in D_\tau$ of a semantic term $\phi \in \Phi_\tau$ with respect to a semantic interpretation with semantic valuation f and semantic assignment g is defined by:

$$(1.12) \quad \begin{aligned} [c]^g &= f(c) \\ [x]^g &= g(x) \\ [(\phi \ \psi)]^g &= [\phi]^g([\psi]^g) \\ [\lambda x_\tau \phi]^g &= D_\tau \ni m \mapsto [\phi]^{(g - \{\langle x, g(x) \rangle\}) \cup \{\langle x, m \rangle\}} \end{aligned}$$

Note that the semantic value of a semantic form is independent of any semantic assignment.

Semantic terms ϕ and ψ are *equivalent*, $\phi \equiv \psi$, if and only if $[\phi]^g = [\psi]^g$ in every semantic interpretation. We have:

$$(1.13) \quad \begin{aligned} \lambda x \phi &\equiv \lambda y(\phi\{y/x\}) && \alpha\text{-conversion} \\ &\text{provided } \phi\{y/x\} \text{ is free and } y \text{ is not free in } \phi \\ (\lambda x \phi \ \psi) &\equiv \phi\{\psi/x\} && \beta\text{-conversion} \\ &\text{provided } \phi\{\psi/x\} \text{ is free} \\ \lambda x(\phi \ x) &\equiv \phi && \eta\text{-conversion} \\ &\text{provided } x \text{ is not free in } \phi \end{aligned}$$

Let there be a set $\mathcal{A}$ of *atomic syntactic types*. Then the sets $\mathcal{F}^i$ of *syntactic types of sort i* for each natural i are defined by:

$$(1.14) \quad \begin{aligned} \mathcal{F}^0 &::= \mathcal{A} \\ \mathcal{F}^j &::= \mathcal{F}^i \setminus \mathcal{F}^{i+j} \\ \mathcal{F}^i &::= \mathcal{F}^{i+j} / \mathcal{F}^j \\ \mathcal{F}^j &::= \mathcal{F}^{i+1} \downarrow \mathcal{F}^{i+j} \\ \mathcal{F}^{i+1} &::= \mathcal{F}^{i+j} \uparrow \mathcal{F}^j \end{aligned}$$

Let the *sort map* S be the mapping from syntactic types to their sort:

$$(1.15) \quad \begin{aligned} S(P) &= 0 && \text{for atomic syntactic type } P \\ S(A \setminus C) &= S(C) - S(A) \\ S(C/B) &= S(C) - S(B) \\ S(A \downarrow C) &= S(C) + 1 - S(A) \\ S(C \uparrow B) &= S(C) + 1 - S(A) \end{aligned}$$

Let there be a *basic type map* t from atomic syntactic types to semantic types; then the *type map* T from syntactic types to semantic types is defined by:

$$(1.16) \quad \begin{aligned} T(P) &= t(P) && \text{for atomic syntactic type } P \\ T(A \setminus C) &= T(A) \rightarrow T(C) \\ T(C/B) &= T(B) \rightarrow T(C) \\ T(A \downarrow C) &= T(A) \rightarrow T(C) \\ T(C \uparrow B) &= T(B) \rightarrow T(C) \end{aligned}$$

A *syntactic interpretation* comprises a prosodic structure $(L, +, 0, \$)$, a semantic structure $\{D_\tau\}_{\tau \in \mathcal{T}}$, and a *syntactic valuation* F mapping each atomic syntactic type P into a subset of $L^0 \times D_{T(P)}$. The *syntactic value* $\llbracket A \rrbracket \subseteq L^{S(A)} \times D_{T(A)}$ of each syntactic type A with respect to a syntactic interpretation with syntactic valuation F is defined by:

$$(1.17) \quad \begin{aligned} \llbracket P \rrbracket &= F(P) && \text{for atomic syntactic type } P \\ \llbracket A \setminus C \rrbracket &= \{ \langle s_2, m_2 \rangle \mid \forall \langle s_1, m_1 \rangle \in \llbracket A \rrbracket, \langle s_1 + s_2, m_2(m_1) \rangle \in \llbracket C \rrbracket \} \\ \llbracket C/B \rrbracket &= \{ \langle s_1, m_1 \rangle \mid \forall \langle s_2, m_2 \rangle \in \llbracket B \rrbracket, \langle s_1 + s_2, m_1(m_2) \rangle \in \llbracket C \rrbracket \} \\ \llbracket A \downarrow C \rrbracket &= \{ \langle s_2, m_2 \rangle \mid \forall \langle s_1, m_1 \rangle \in \llbracket A \rrbracket, \langle s_1 W s_2, m_2(m_1) \rangle \in \llbracket C \rrbracket \} \\ \llbracket C \uparrow B \rrbracket &= \{ \langle s_1, m_1 \rangle \mid \forall \langle s_2, m_2 \rangle \in \llbracket B \rrbracket, \langle s_1 W s_2, m_1(m_2) \rangle \in \llbracket C \rrbracket \} \end{aligned}$$

A *type assignment statement* $\alpha - \phi: A$ comprises a syntactic type A , a prosodic term α of prosodic sort $S(A)$, and a semantic term ϕ of semantic type $T(A)$. A *semiotic interpretation* comprises a prosodic, semantic and syntactic interpretation with the same prosodic and semantic structures. A semiotic interpretation *satisfies a type assignment statement* $\alpha - \phi: A$ if and only if $\langle [\alpha], [\phi]^g \rangle \in \llbracket A \rrbracket$; it *satisfies a set* Σ *of type assignment statements* if and only if it satisfies every type assignment $\sigma \in \Sigma$. A set Σ of type assignment statements *entails* a type assignment statement σ , $\Sigma \models \sigma$ if and only if every semiotic interpretation which satisfies Σ also satisfies σ .

A *lexicon* is a set of type assignment statements with prosodic and semantic forms. The *language model* $\mathcal{L}(Lex)$ defined by a lexicon Lex is the set of type assignment statements entailed by Lex :

$$(1.18) \quad \mathcal{L}(Lex) = \{ \alpha - \phi: A \mid Lex \models \alpha - \phi: A \}$$

2. TYPE ASSIGNMENT CALCULUS

Derivations in the type assignment calculus are trees of type assignment statements with some leaves closed by coindexation with a dominating rule application. A derivation tree means that its (open) leaves

entail the root; thus if the open leaves are in the lexicon, the root is in the language model defined.

Any type assignment statement is a derivation of itself (as root) from itself (as leaf). Then the derivation trees are those that can be constructed by the following rules. For each operator there is a rule of elimination (E) and a rule of introduction (I); in rules of elimination the operator is eliminated reading from premisses to conclusions, and in rules of introduction the operator is introduced reading from premisses to conclusions.

(1.19) a.

$$\frac{\begin{array}{c} \vdots \\ \alpha - \phi : A \end{array} \quad \begin{array}{c} \vdots \\ \beta - \psi : A \setminus C \end{array}}{\alpha + \beta - (\psi \ \phi) : C} \setminus E$$

b.

$$\frac{\begin{array}{c} \overline{a_0 + \$ + \dots + \$ + a_{S(A)} - x : A}^i \\ \vdots \\ a_0 + \$ + \dots + \$ + a_{S(A)} + \beta - \chi : C \end{array}}{\beta - \lambda x \chi : A \setminus C} \setminus I^i$$

c.

$$\frac{\begin{array}{c} \vdots \\ \alpha - \phi : C / B \end{array} \quad \begin{array}{c} \vdots \\ \beta - \psi : B \end{array}}{\alpha + \beta - (\phi \ \psi) : C} / E$$

d.

$$\frac{\begin{array}{c} \overline{b_0 + \$ + \dots + \$ + b_{S(B)} - y : B}^i \\ \vdots \\ \alpha + b_0 + \$ + \dots + \$ + b_{S(B)} - \chi : C \end{array}}{\alpha - \lambda x \chi : C / B} / I^i$$

(1.20) a.

$$\frac{\begin{array}{c} \vdots \\ \alpha - \phi : A \end{array} \quad \begin{array}{c} \vdots \\ \beta - \psi : A \downarrow C \end{array}}{\alpha W \beta - (\psi \ \phi) : C} \downarrow E$$

b.

$$\frac{\begin{array}{c} \overline{a_0 + \$ + \dots + \$ + a_{S(A)} - x : A}^i \\ \vdots \\ (a_0 + \$ + \dots + \$ + a_{S(A)}) W \beta - \chi : C \end{array}}{\beta - \lambda x \chi : A \downarrow C} \downarrow I^i$$

c.

$$\frac{\begin{array}{c} \vdots \\ \alpha - \phi : C \uparrow B \end{array} \quad \begin{array}{c} \vdots \\ \beta - \psi : B \end{array}}{\alpha W \beta - (\phi \ \psi) : C} \uparrow E$$

d.

$$\frac{\begin{array}{c} \overline{b_0 + \$ + \dots + \$ + b_{S(B)} - y : B}^i \\ \vdots \\ \alpha W (b_0 + \$ + \dots + \$ + b_{S(B)}) - \chi : C \end{array}}{\alpha - \lambda x \chi : C \uparrow B} \uparrow I^i$$

3. GRAMMAR

For subject- and object-oriented reflexives respectively we assume the following lexical assignments:

$$\begin{aligned}
 (1.21) \quad \mathbf{himself} & - \lambda x \lambda y (x \ y \ y) \\
 & : ((N \setminus S) \uparrow N) \downarrow (N \setminus S) \\
 \mathbf{himself} & - \lambda x \lambda y (x \ y \ y) \\
 & : (((N \setminus S) \uparrow N) \uparrow N) \downarrow ((N \setminus S) \uparrow N)
 \end{aligned}$$

The example of object-oriented reflexivisation (1.22) is derived as shown in figure 1.1.

$$(1.22) \quad \text{Dorothy bet the Strawman half of himself that they would reach Emerald City.}$$

We exemplify personal pronouns by analysis of the sentence *A woman walks and she talks* chosen in Dowty, Wall and Peters (1981, 7. IV). Let there be the lexicon:

$$\begin{aligned}
 (1.23) \quad \mathbf{a} & - \lambda x \lambda y (\exists \lambda z (\wedge (x \ z) (y \ z))) \\
 & : ((S \uparrow N) \downarrow S) / \text{CN} \\
 \mathbf{and} & - \wedge \\
 & : (S \setminus S) / S \\
 \mathbf{her} & - \lambda x \lambda y (x \ \lambda z (z \ y) \ y) \\
 & : ((S \uparrow N) \uparrow (((N \setminus S) \uparrow N) \downarrow (N \setminus S))) \downarrow (S \uparrow N) \\
 \mathbf{she} & - \lambda x \lambda y (x \ \lambda z (z \ y) \ y) \\
 & : ((S \uparrow N) \uparrow (S / (N \setminus S))) \downarrow (S \uparrow N) \\
 \mathbf{talks} & - \text{talk} \\
 & : N \setminus S \\
 \mathbf{walks} & - \text{walk} \\
 & : N \setminus S \\
 \mathbf{woman} & - \text{man} \\
 & : \text{CN}
 \end{aligned}$$

Then the language model entailed includes:

$$(1.24) \quad \mathbf{a+woman+walks+and+she+talks-} \\
 (\exists \lambda z (\wedge (\text{woman } z) (\wedge (\text{talk } z) (\text{walk } z)))): S$$

There is the derivation given in figure 1.2.

On our account, an expression of syntactic type *B* containing an *A*-antecedent free anaphor will have syntactic type $((S \uparrow A) \uparrow B) \downarrow (S \uparrow A)$. For example, figure 1.3 derives (1.25).

$$(1.25) \quad \mathbf{she+talks-} \lambda x \lambda y (x \ (\text{walk } y) \ y): ((S \uparrow N) \uparrow S) \downarrow (S \uparrow N)$$

$$\begin{array}{c}
\frac{\frac{\frac{\text{bet} - \text{bet}; ((N \setminus S)/CP)/N}{a-x: N} \quad \frac{\frac{\text{half} + \text{of} - \text{half}; N/N}{b-y: N}}{i} \quad j}{\frac{\text{bet} + a - (bet \ x); ((N \setminus S)/CP)/N}{\text{half} + \text{of} + b - (half \ y): N}} / E \\
\frac{\text{bet} + a + \text{half} + \text{of} + b - (bet \ x \ (half \ y)); (N \setminus S)/CP}{\text{bet} + o + \text{half} + \text{of} + b + \text{ttwrEC} - (bet \ x \ (half \ y) \ ttwrEC); N \setminus S} / E \\
\frac{\text{bet} + o + \text{half} + \text{of} + b + \text{ttwrEC} - (bet \ x \ (half \ y) \ ttwrEC); N \setminus S}{\text{bet} + \$ + \text{half} + \text{of} + b + \text{ttwrEC} - \lambda x (bet \ x \ (half \ y) \ ttwrEC); (N \setminus S) \uparrow N} \uparrow i \\
\frac{\text{bet} + \$ + \text{half} + \text{of} + \$ + \text{ttwrEC} - \lambda y \lambda x (bet \ x \ (half \ y) \ ttwrEC); ((N \setminus S) \uparrow N) \uparrow N}{\text{bet} + \$ + \text{half} + \text{of} + \text{half} + \text{of} + \text{half} + \text{ttwrEC} - \lambda y (bet \ y \ (half \ y) \ ttwrEC); (N \setminus S) \uparrow N} \uparrow j \\
\frac{\text{bet} + \$ + \text{half} + \text{of} + \text{half} + \text{of} + \text{half} + \text{ttwrEC} - \lambda y (bet \ y \ (half \ y) \ ttwrEC); (N \setminus S) \uparrow N}{\text{bet} + \$ + \text{half} + \text{of} + \text{half} + \text{of} + \text{half} + \text{ttwrEC} - (bet \ s \ (half \ s) \ ttwrEC); N \setminus S} \downarrow E \\
\frac{\text{bet} + \$ + \text{half} + \text{of} + \text{half} + \text{of} + \text{half} + \text{ttwrEC} - (bet \ s \ (half \ s) \ ttwrEC); N \setminus S}{D + \text{bet} + \$ + \text{half} + \text{of} + \text{half} + \text{of} + \text{half} + \text{ttwrEC} - (bet \ s \ (half \ s) \ ttwrEC \ s); S} \uparrow E \\
\frac{D - d: N}{\text{D} + \text{bet} + \$ + \text{half} + \text{of} + \text{half} + \text{of} + \text{half} + \text{ttwrEC} - (bet \ s \ (half \ s) \ ttwrEC \ s); S} \uparrow E
\end{array}$$

Figure 1.1 Derivation of *Dorothy bet the Strawman half of himself that they would reach Emerald City.*

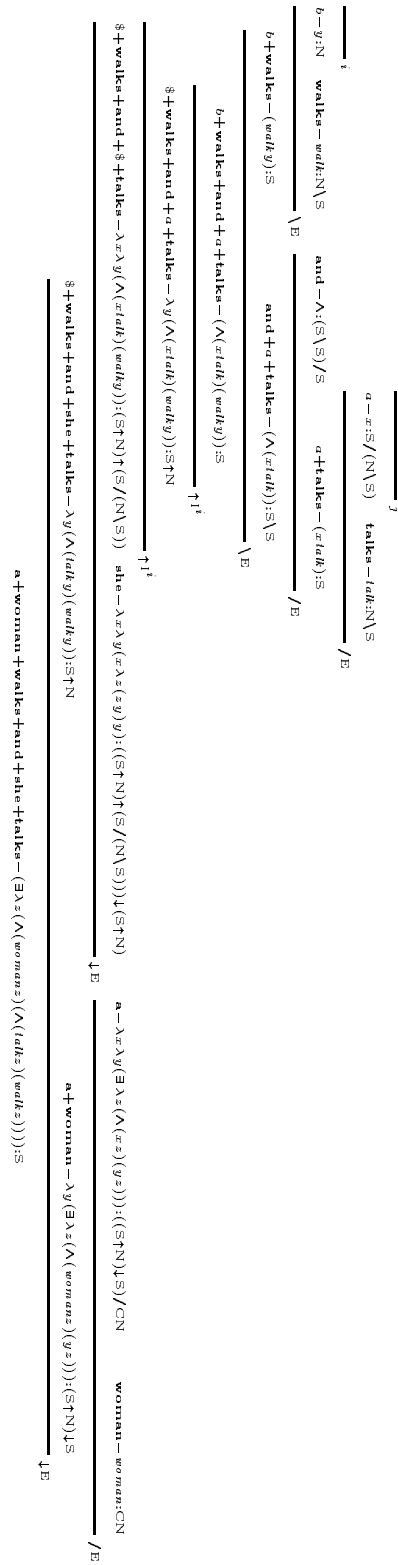


Figure 1.2 Derivation of (bound) *A woman walks and she talks*.

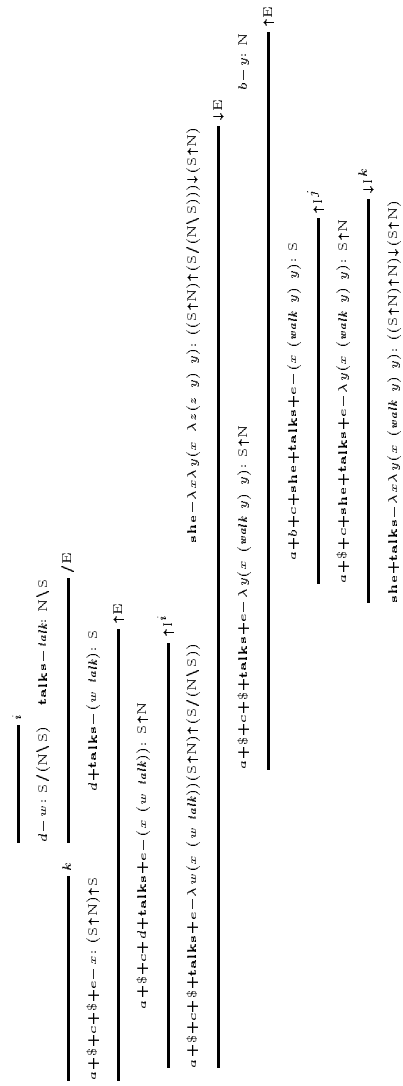


Figure 1.3 Derivation of (free) *She talks*.

In general, an expression of type B containing $A_1, \dots, A_n$ -antecedent free anaphors will have syntactic type:

$$(1.26) ((\dots(S \uparrow A_n) \uparrow \dots \uparrow A_1) \uparrow B) \downarrow (\dots(S \uparrow A_1) \uparrow \dots \uparrow A_n)$$

Let an *n-ary casting* be an n -ary function into propositions. Let an *n-ary context* be a function from propositions into n -ary castings. Let the *null n-ary context* be $\lambda x \lambda y_1 \dots \lambda y_n x$. Our semantics of a sentence containing $A_1, \dots, A_n$ -antecedent free anaphors will be a function from n -ary contexts into n -ary castings; the semantics ascribed by Jacobson (1999) and Jäger (2001) to a sentence containing $A_1, \dots, A_n$ -antecedent free anaphors can be obtained by applying our semantics to the null n -ary context. For example, applying the semantics in (1.25) to the null unary context evaluates:

$$(1.27) (\lambda x \lambda y (x (\text{walk } y) y) \lambda x \lambda y x) \equiv \lambda y (\text{walk } y)$$

4. BINDING PRINCIPLES

Chomsky (1981) proposes Principles A and B governing anaphoric relations. Roughly speaking, Principle A states that a reflexive or reciprocal pronoun must be bound by a local c-commanding antecedent, and Principle B states that a personal pronoun must not be bound by a local c-commanding antecedent. The ‘delay of Principle B effect’ is the term given to the phenomenon whereby in acquisition there seems to be non-compliance with principle B.

Morrill (1990) argues that the locality restriction on reflexives comes within grasp once we work within a modal calculus of intensional types. Regarding c-command for reflexives, our subject-oriented reflexive type observes c-command since a subject c-commands its verb phrase; however, our object-oriented reflexive type will allow both c-commanding and non-c-commanding antecedents. The capture of an apparent c-command condition on object-oriented reflexivisation remains for us illusive, but we note that without recourse to reanalysis a sentence like (1.28) is a counterexample to such a condition.

$$(1.28) \text{ Mary talked to John}_i \text{ about himself}_i.$$

According to Principle B a personal pronoun cannot be bound in its own clause. The grammar complies with Principle B in respect of nominative personal pronouns in virtue of their subject position and the right-branching structure of English. The grammar does not respect

principle B in respect of accusative personal pronouns. Thus according to our account, (1.29a) and (1.29b) are paraphrases.

- (1.29) a. *Every boy_i loves him_i.
b. Every boy_i loves himself_i.

However, according to our grammar, the syntactic type of the pronoun for (1.29a) is about twice as large as that required for (1.29b), and (1.29a) is ambiguous between a bound and a free reading whereas (1.29b) is not. Thus we claim that, in general, principle B violations are unacceptable because of their lack of communicative efficacy: if the local c-commanding reading was intended then the cheaper and less ambiguous reflexive form would have been used.

We suggest that argument for this position is supplied by the *Delay of Principle B Effect*. Chien and Wexler (1990) reports an experiment (experiment 1) in which children were presented with sentences of the form in (1.30).

- (1.30) a. Kitty says that Sarah should point to herself.
b. Kitty says that Sarah should point to her.

Children's performance on the locality property of reflexives increased continuously from about 13% at age 2;6 to about 90% at 6;0, i.e. children older than 6;0 complied with the major properties of reflexives. By contrast, children's performance on the non-locality property of personal pronouns stayed roughly flat from 2;6 to 6;6 with only a slight improvement to 64%. That is, children between the ages of 6;0 and 6;6 demonstrated compliance with Principle A but appeared to allow violation of Principle B.

The claim that Principle A is a grammatical condition while Principle B is a pragmatic one allows for the possibility of the delay of Principle B if we can explain why Principle B, as a pragmatic condition, is resisted during acquisition. We claim that this is because there is pressure, coming from considerations of processing, leaning in favour of locality. It is argued in Morrill (2000a) that the number of axiom links crossing in between successive words in the syntactic structures of type logical grammar as represented by proof nets reflects the processing load on memory in the course of an incremental analysis. So far as processing of structural ambiguity is concerned, we expect a processing path with a lower profile to be favoured. When we compare the profiles of the local and non-local proof nets for (1.30b) in figures 1.4 and 1.5 we find

a marked differentiation in favour of the local analysis:

(1.31)	9							b	b
	8								
	7								
	6					b		b	
	5			b	b				a a
	4								
	3								
	2		ab			a		a	
	1	ab		a	a				
	0								ab
a.		Kitty	says	that	Sarah _i	should	point	to	her _i
b.		Kitty _i	says	that	Sarah	should	point	to	her _i

In summary, we claim that the locality of Principle A is a grammatically imposed Tensed-S constraint whereas Principle B in respect of accusative pronouns is a pragmatic condition. The acquisition of Principle B is impeded by the fact that non-locality requires dependencies to be held unresolved over long distances whereas a local analysis has a lower processing complexity. As processing becomes more resilient, Principle B as a pragmatic condition comes to be enforced; this is after the age of six, but the grammar was instilled before.

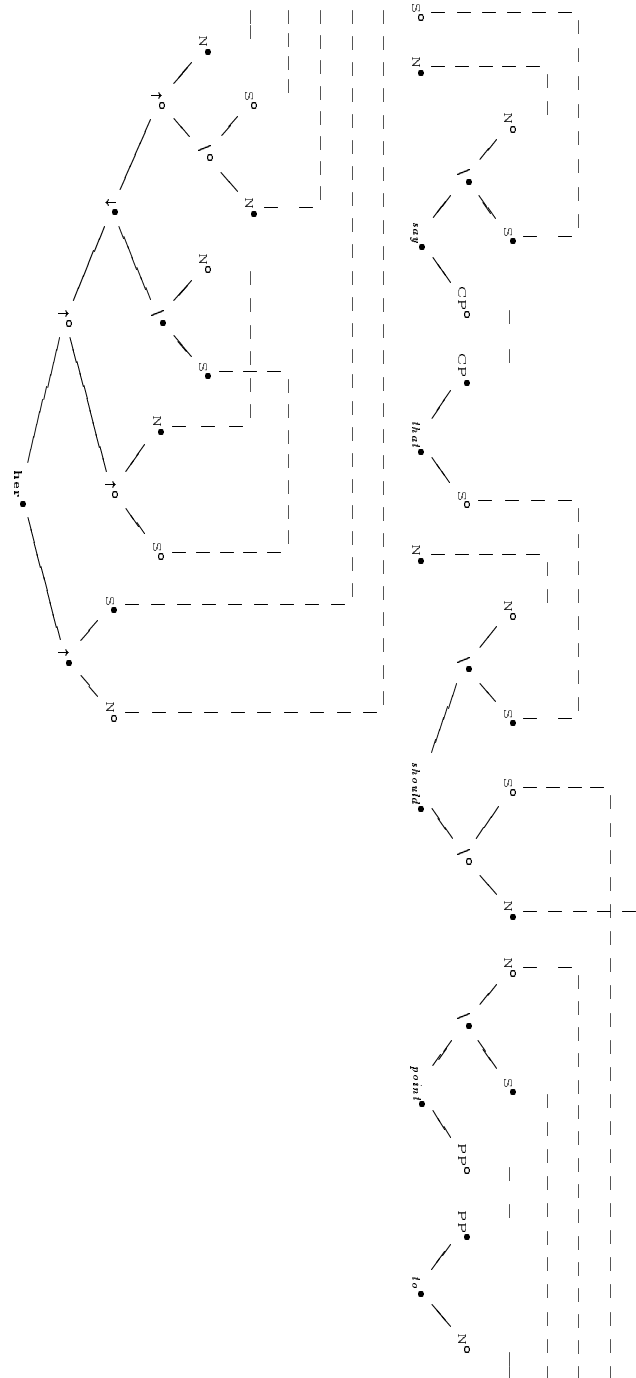


Figure 1.4 Proof net for *Kitty says that Sarah_i should point to her_i*.

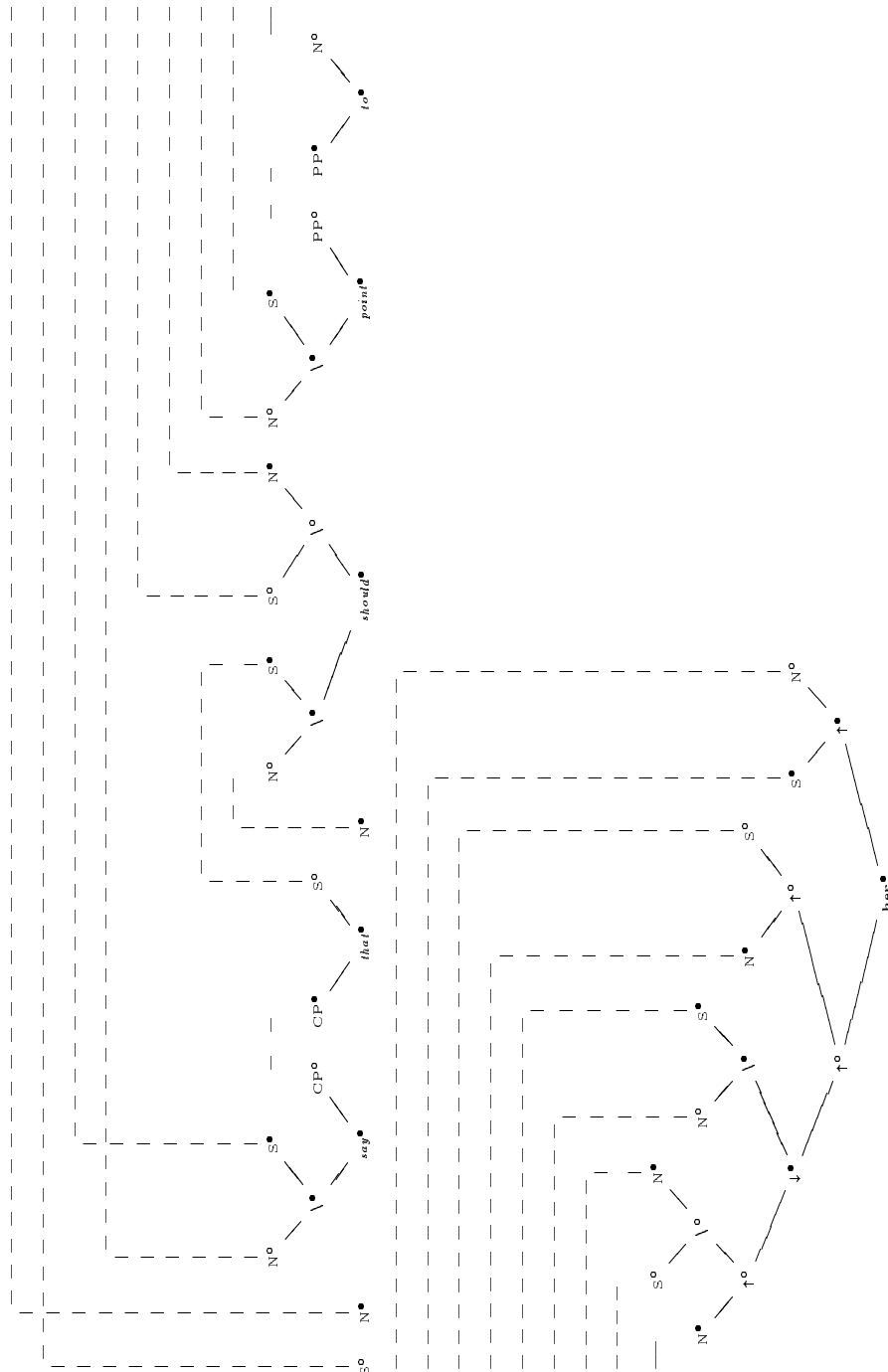


Figure 1.5 Proof net for *Kitty_i says that Sarah should point to her_i*.

Appendix: Hypersequent calculus

We present a sequent calculus for discontinuity which is ‘pure’ in that it has no structural rules. Its key feature is repetition of cotagged discontinuous types in sequents at their various loci of presence (Morrill 1997). A type of sort i has $i + 1$ loci. We refer to this as *hypersequent calculus*. Cotagged types are to be understood as a single type occurrence which manifests itself at various places. Types must be non-cotagged except where cotagging is enforced. The unique tag of a type of sort zero may be omitted.

The sets $\mathcal{O}_m^D$ of *configurations for succedent type D with tag m* are the strings defined by (Λ is the empty string):

$$\begin{aligned}
 (1.32) \quad \mathcal{O}_m^0 &::= \Lambda \\
 \mathcal{O}_m^1 &::= \overset{D}{m} \\
 \mathcal{O}_m^{S(A)+j} &::= \overset{A}{n}_0, \overset{D}{m}, \dots, \overset{D}{m}, \overset{A}{n}_{S(A)}, \mathcal{O}_m^j \\
 \mathcal{O}_m^{S(A)+j+k-1} &::= \overset{A}{n}_0, \overset{D}{m}, \dots, \overset{D}{m}, \overset{A}{n}_{S(A)-1}, \mathcal{O}_m^j, \overset{A}{n}_{S(A)}, \mathcal{O}_m^k \\
 \mathcal{O}_m &::= \mathcal{O}_m^{S(D)}
 \end{aligned}$$

The set $\mathcal{S}$ of *sequents* is defined by:

$$(1.33) \quad \mathcal{S} ::= \mathcal{O}_m^D \Rightarrow \overset{D}{m}$$

The sequent calculus is as follows, where $\Delta(\Gamma_1; \dots; \Gamma_n)$ signifies a configuration Δ with successive, but not necessarily contiguous, placed sub-configurations $\Gamma_1, \dots, \Gamma_n$, and repetitions of the same type variable in the same sequent are always cotagged unless otherwise indicated.

$$\begin{aligned}
 (1.34) \quad \text{a.} \quad & \frac{}{\overset{A}{n}, \overset{A}{m}, \dots, \overset{A}{m}, \overset{A}{n} \Rightarrow \overset{A}{m}} \text{id} \\
 \text{b.} \quad & \frac{\Gamma_0, \overset{A}{m}, \dots, \overset{A}{m}, \Gamma_{S(A)} \Rightarrow \overset{A}{m} \quad \Delta(\overset{A}{n}; \dots; \overset{A}{n}) \Rightarrow D}{\Delta(\Gamma_0; \dots; \Gamma_{S(A)}) \Rightarrow D} \text{Cut}
 \end{aligned}$$

$$\begin{aligned}
(1.35) \quad & \text{a.} \quad \frac{\Gamma_0, A, \dots, A, \Gamma_{S(A)} \Rightarrow A \quad \Delta(C; \dots; C; \dots; C) \Rightarrow D}{\Delta(\Gamma_0; \dots; \Gamma_{S(A)}, A \setminus C; \dots; A \setminus C) \Rightarrow D} \setminus L \\
& \text{b.} \quad \frac{A_0, C, \dots, C, A_{S(A)}, \Gamma_0, C, \dots, C, \Gamma_{S(C)-S(A)} \Rightarrow C}{\Gamma_0, A \setminus C, \dots, A \setminus C, \Gamma_{S(C)-S(A)} \Rightarrow A \setminus C} \setminus R \\
& \text{c.} \quad \frac{\Gamma_0, B, \dots, B, \Gamma_{S(B)} \Rightarrow B \quad \Delta(C; \dots; C; \dots; C) \Rightarrow D}{\Delta(C/B; \dots, C/B, \Gamma_0; \dots; \Gamma_{S(B)}) \Rightarrow D} /L \\
& \text{d.} \quad \frac{\Gamma_0, C, \dots, C, \Gamma_{S(C)-S(B)}, B_0, C, \dots, C, B_{S(B)} \Rightarrow C}{\Gamma_0, C/B, \dots, C/B, \Gamma_{S(C)-S(B)} \Rightarrow C/B} /R \\
(1.36) \quad & \text{a.} \quad \frac{\Gamma_0, A, \dots, A, \Gamma_{S(A)-1}, A, \Gamma_{S(A)} \Rightarrow A \quad \Delta(C; \dots; C; \dots; C) \Rightarrow D}{\Delta(\Gamma_0; \dots; \Gamma_{S(A)-1}, A \downarrow C; \dots; A \downarrow C, \Gamma_{S(A)}) \Rightarrow D} \downarrow L \\
& \text{b.} \quad \frac{A_0, C, \dots, C, A_{S(A)-1}, \Gamma_0, C, \dots, C, \Gamma_{S(C)+1-S(A)}, A_{S(A)} \Rightarrow C}{\Gamma_0, A \downarrow C, \dots, A \downarrow C, \Gamma_{S(C)+1-S(A)} \Rightarrow A \downarrow C} \downarrow R \\
& \text{c.} \quad \frac{\Gamma_0, B, \dots, B, \Gamma_{S(B)} \Rightarrow B \quad \Delta(C; \dots; C; \dots; C) \Rightarrow D}{\Delta(C \uparrow B; \dots; C \uparrow B, \Gamma_0; \dots; \Gamma_{S(B)}, C \uparrow B) \Rightarrow D} \uparrow L \\
& \text{d.} \quad \frac{\Gamma_0, C, \dots, C, \Gamma_{S(C)-S(B)}, B, C, \dots, C, B, \Gamma_{S(C)+1-S(B)} \Rightarrow C}{\Gamma_0, C \uparrow B, \dots, C \uparrow B, \Gamma_{S(C)+1-S(B)}, C \uparrow B, \Gamma_{S(C)+1-S(B)} \Rightarrow C \uparrow B} \uparrow R
\end{aligned}$$

Thus for example, the sequent underlying example (1.24) of section 3 is a theorem:

$$(1.37) \quad ((S \uparrow N) \downarrow S) / CN, \quad CN, \quad N \setminus S, \quad (S \setminus S) / S, \\
((S \uparrow N) \uparrow (S / (N \setminus S))) \downarrow (S \uparrow N), \quad N \setminus S \Rightarrow S$$

The derivation is given in figure 1.6.

$$\begin{array}{c}
\frac{N, N \setminus S, (S \setminus S)/S, S \Rightarrow S}{N \setminus S \Rightarrow N \setminus S} \uparrow_R \\
\frac{N \setminus S \Rightarrow N \setminus S \quad \frac{S \uparrow_N}{n}, N \setminus S, (S \setminus S)/S, S \Rightarrow S \uparrow_N}{S \uparrow_N, N \setminus S, (S \setminus S)/S, S/(N \setminus S), N \setminus S \Rightarrow S \uparrow_N} /L \\
\frac{(S \uparrow_N) \uparrow(S/(N \setminus S)), N \setminus S, (S \setminus S)/S, (S \uparrow_N) \uparrow(S/(N \setminus S)), N \setminus S \Rightarrow (S \uparrow_N) \uparrow(S/(N \setminus S)) \uparrow_R}{(S \uparrow_N) \uparrow(S/(N \setminus S)), N \setminus S, (S \setminus S)/S, ((S \uparrow_N) \uparrow(S/(N \setminus S))) \downarrow(S \uparrow_N), N \setminus S \Rightarrow S \uparrow_N} \uparrow_R \\
\frac{(S \uparrow_N) \uparrow(S/(N \setminus S)), N \setminus S, (S \setminus S)/S, ((S \uparrow_N) \uparrow(S/(N \setminus S))) \downarrow(S \uparrow_N), N \setminus S \Rightarrow S \uparrow_N}{(S \uparrow_N) \uparrow(S/(N \setminus S)), N \setminus S, (S \setminus S)/S, ((S \uparrow_N) \uparrow(S/(N \setminus S))) \downarrow(S \uparrow_N), N \setminus S \Rightarrow S} \downarrow_L \\
\frac{CN \Rightarrow CN \quad ((S \uparrow_N) \downarrow S)/CN, CN, N \setminus S, (S \setminus S)/S, ((S \uparrow_N) \uparrow(S/(N \setminus S))) \downarrow(S \uparrow_N), N \setminus S \Rightarrow S}{((S \uparrow_N) \downarrow S)/CN, CN, N \setminus S, (S \setminus S)/S, ((S \uparrow_N) \uparrow(S/(N \setminus S))) \downarrow(S \uparrow_N), N \setminus S \Rightarrow S} /L
\end{array}$$

Figure 1.6 Derivation of $((S \uparrow N) \downarrow S)/CN, CN, N \setminus S, (S \setminus S)/S, ((S \uparrow N) \uparrow(S/(N \setminus S))) \downarrow(S \uparrow N), N \setminus S \Rightarrow S$.

References

- Chien, Yu-Chin and Kenneth Wexler: 1990 ‘Children’s Knowledge of Locality Conditions in Binding as Evidence for the Modularity of Syntax and Pragmatics’, *Language Aquisition*, **1**, 3, 225–295.
- Chomsky, Noam: 1981, *Lectures on Government and Binding*, Foris, Dordrecht.
- Dowty, David R., Robert E. Wall and Stanley Peters: 1981, *Introduction to Montague Semantics*, Synthese Language Library Volume 11, D. Reidel, Dordrecht.
- Jacobson, Pauline: 1999, ‘Towards a Variable-Free Semantics’, *Linguistics and Philosophy*, **22**, 2, 117–184.
- Jäger, Gerhard: 2001, *Anaphora in Type Logical Grammar*, UIL-OTS Working Paper, 01004-CL/TL, OTS, Rijksuniversiteit Utrecht.
- Montague, Richard: 1973, ‘The proper treatment of quantification in ordinary English’, in J. Hintikka, J.M.E. Moravcsik and P. Suppes (eds.) *Approaches to Natural Language*, D. Reidel, Dordrecht. Reprinted in R.H. Thomason (ed.): 1974, *Formal Philosophy: Selected Papers of Richard Montague*, Yale University Press, New Haven, 247–270.
- Moortgat, Michael: 1988, *Categorial Investigations: Logical and Linguistic Aspects of the Lambek Calculus*, Foris, Dordrecht.
- Moortgat, Michael: 1996, ‘In situ binding: A modal analysis’, in Dekker, P. and M. Stokhof (eds.) *Proceedings of the 10th Amsterdam Colloquium*, ILLC, Universiteit van Amsterdam, 235–240.
- Morrill, Glyn: 1990, ‘Intensionality and Boundedness’, *Linguistics and Philosophy* **13** 6, 699–726.
- Morrill, Glyn: 1997, ‘Proof syntax of discontinuity’, in Dekker, P., M. Stockhof and Y. Venema (eds.) *Proceedings of the 11th Amsterdam Colloquium*, ILLC, Universiteit van Amsterdam, 235–240.
- Morrill, Glyn: 2000a, ‘Incremental Processing and Acceptability’, *Computational Linguistics*, **26**, 3, 319–338.
- Morrill, Glyn: 2000b, ‘Type-Logical Anaphora’, Report de Recerca LSI-00–59-R, LSI, Universitat Politècnica de Catalunya.
- Morrill, Glyn: 2002, ‘Towards Generalised Discontinuity’, in Jäger, G., P. Monachesi, G. Penn and S. Wintner (eds.) *Proceedings of the 7th Conference on Formal Grammar*, ESSLLI, Trento, 103–111.
- Solias, Teresa: 1992, *Gramáticas Catoriales, Coordinación Generalizada y Elisión*, Ph.D. dissertation, Universidad Autónoma de Madrid.
- Versmissen, Koen: 1991, ‘Discontinuous Type Constructors in Catorial Grammar’, ms. Rijksuniversiteit Utrecht.

On Aphasic Comprehension and Working Memory Load

Glyn Morrill¹

*Departament de Llenguatges i Sistemes Informàtics,
Universitat Politècnica de Catalunya,
Mòdul C 5 - Campus Nord,
Jordi Girona Salgado 1-3,
08034 Barcelona*

Anna Gavarró²

*Departament de Filologia Catalana,
Universitat Autònoma de Barcelona,
08193 Bellaterra,
Barcelona*

Abstract

There is debate in the literature on aphasia as to the extent to which aphasia arises from loss of knowledge of language vs. loss of processing resources. In this paper we compare a simple measure of the complexity of processing Type Logical Grammar with the aphasic performance observed in a large study carried out by Caplan and his colleagues. To the extent that a correlation is found, and the type logical model is taken to be psychologically plausible, our analysis suggests that aphasics suffer a deficit of working memory capacity in the incremental comprehension of language.

Key words: aphasia, type logical grammar, proof nets, memory

Email addresses: morrill@lsi.upc.es (Glyn Morrill), anna.gavarro@uab.es (Anna Gavarró).

URLs: <http://www-lsi.upc.es/~morrill> (Glyn Morrill), <http://seneca.uab.es/ggt/Gavarro/gavarro.html> (Anna Gavarró).

¹ Work supported by CICYT project TIC2002-04019-C03-01.

² Work supported by CICYT project BFF2003-08364-C02-01.

1 Introduction

According to the dependency locality theory (DLT) of [7], 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. [11] and [15] instantiate this view by reference to proof nets as the syntactic structures of Type Logical Grammar ([14]). In type logical proof nets, dependencies are identity links and the number of unresolved syntactic dependencies at a word boundary is simply the number of identity links spanning that point. [11] argues that the maximal number of unresolved dependencies so-measured underlies the unacceptability of centre-embedding. [15] argues that the measure of the incremental course of complexity (complexity profile) of analyses underlies also processing phenomena such as garden pathing, right association (right attachment), left-to-right quantifier scoping in passives and actives, preference for passivisation of nested sentential subjects, and heavy noun phrase shift, in what we might call type logical DLT.

Complexity-based accounts are not always computationally rigorous, thus [9, p.56] writes: “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.” On the other hand, computational rigor often requires ancillary assumptions, as noted by [17, p.1145]: “...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.” Type logical DLT, we suggest, squares this circle. At the same time that it is computationally rigorous, assuming uniform cost for dependencies no ancillary assumptions are required to draw the complexity profile prediction of processing complexity. In this paper we analyse whether *aphasic* processing complexities correlate with type logical DLT complexity profiles. We find that they do, suggesting that aphasics suffer a deficit of working memory capacity in the incremental comprehension of language.

[10] describes a computational model of aphasic sentence comprehension based on the premise that all aphasics share a common deficit in the activation resources of working memory. They simulate the data from [2] (see [3]) of aphasic patients from all major syndrome types. In their system “As each word comes in, the model attempts to incorporate it as much as possible into the evolving syntactic and semantic representation. First, the word is perceptually encoded. Then, lexical access makes available its meaning and syntactic class and, in the case of verbs, also its argument structure. Based on its word class and a grammar, the word is integrated into a parse tree representation. The thematic role mapping component computes thematic-role bindings.” ([10, p.88]). “The hypothesized resource reduction in aphasia was then induced by

decreasing the model’s working memory capacity considerably, reducing it by half to a level of 15 activation units, to optimize the fit with the Caplan et al. data.” ([10, p.96]).

Our result echoes that of [10] with a different grammar architecture, one that does not make reference to distinct syntactic structure, argument structure and thematic-role bindings.

We focus on experiment 2 reported in [3] Chapter 4. In this experiment the subjects were 37 largely unselected native English speaker aphasics. The experimenter read a sentence with normal nonemphatic intonation and the subjects were required to perform an object-manipulation task using toys to demonstrate semantic features of the sentence. The types of sentences were as follows:

(1)	Active	A	The rat hit the dog.
	Dative	D	The rat gave the dog to the cow.
	Conjoined	C	The rat hit the dog and kissed the cow.
	Passive	P	The rat was hit by the dog.
	Dative Passive	DP	The rat was given to the dog by the cow.
	Object-Subject relative	OS	The rat hit the dog that kissed the cow.
	Subject-Object relative	SO	The rat that the dog hit kissed the cow.
	Cleft-Subject	CS	It was the rat that hit the dog.
	Cleft-Object	CO	It was the rat that the dog hit.

The sentences are all unambiguous except the Object-Subject relative (OS) type, which has a right extraposed reading in which the relative clause modifies the subject. This is presumably an unintended defect in experimental design since apparent aphasic error could be the consequence of interference from the grammatical extraposition reading. (Indeed, in only this case attributed error is much higher than predicted by our model.)

2 The formalism of basic Type Logical Grammar

We present the formalism of basic Type Logical Grammar, sufficient for our current purposes, this being an embellishment of the Lambek calculus ([12]) with semantics along the lines of [13].

2.1 Prosodics

Let a *prosodic structure* be a semigroup, i.e. an algebra $(L, +)$ of arity (2) such that $+$ is associative:

$$(2) \quad s_1 + (s_2 + s_3) = (s_1 + s_2) + s_3$$

Let there be a set A of *prosodic constants*. Then the set $\mathbf{A}$ of *prosodic forms* is defined by:

$$(3) \quad \mathbf{A} ::= A \mid \mathbf{A} + \mathbf{A}$$

A *prosodic interpretation* comprises a prosodic structure $(L, +)$ and a *prosodic valuation* mapping from A into L . Then the *prosodic value* $[\alpha]$ of a prosodic form α relative to a prosodic interpretation with prosodic valuation v is defined by:

$$(4) \quad \begin{aligned} [\mathbf{c}] &= v(\mathbf{c}) \text{ for prosodic constant } \mathbf{c} \\ [\alpha + \beta] &= [\alpha] + [\beta] \end{aligned}$$

Prosodic forms α and β are *equivalent*, $\alpha \equiv \beta$, if and only if $[\alpha] = [\beta]$ in every prosodic interpretation. Clearly we have:

$$(5) \quad \alpha + (\beta + \gamma) \equiv (\alpha + \beta) + \gamma$$

So we can omit parentheses in prosodic forms.

2.2 Semantics

Let the set $\mathbf{T}$ of *semantic types* be defined by:

$$(6) \quad \mathbf{T} ::= e \mid t \mid \mathbf{T} \rightarrow \mathbf{T} \mid \mathbf{T} \& \mathbf{T}$$

Then the *semantic structure* induced by a non-empty set E of *entities* is the $\mathbf{T}$ -indexed family of sets $\{D_\tau\}_{\tau \in \mathbf{T}}$ such that:

$$\begin{aligned}
(7) \quad & D_e = E \\
& D_o = \mathbf{P}(\{\emptyset\}) = \{\emptyset, \{\emptyset\}\} \\
& D_{\tau \rightarrow \tau'} = D_{\tau'}^{D_\tau} \\
& D_{\tau \& \tau'} = D_\tau \times D_{\tau'}
\end{aligned}$$

Let there be a set X_τ of *semantic variables* of each semantic type τ , and a set C_τ of *semantic constants* of each semantic type τ , including the *logical semantic constants*:

$$\begin{aligned}
(8) \quad & \neg \in C_{t \rightarrow t} \\
& \wedge, \vee, \rightarrow \in C_{t \rightarrow (t \rightarrow t)} \\
& = \in C_{e \rightarrow (e \rightarrow t)} \\
& \forall, \exists \in C_{(e \rightarrow t) \rightarrow t} \\
& \iota \in C_{(e \rightarrow t) \rightarrow e}
\end{aligned}$$

The sets Φ_τ of *semantic terms of type τ* for each semantic type τ are defined by:

$$\begin{aligned}
(9) \quad & \Phi_\tau ::= C_\tau \mid X_\tau \mid (\Phi_{\tau' \rightarrow \tau} \Phi_{\tau'}) \mid \pi_1 \Phi_{\tau \& \tau'} \mid \pi_2 \Phi_{\tau' \& \tau} \\
& \Phi_{\tau \rightarrow \tau'} ::= \lambda X_\tau \Phi_{\tau'} \\
& \Phi_{\tau \& \tau'} ::= (\Phi_\tau, \Phi_{\tau'})
\end{aligned}$$

We assume a convention of left-association such that $((\phi \psi) \chi)$ may be abbreviated as $(\phi \psi \chi)$. An occurrence of a semantic variable x in a semantic term is *free* if and only if it does not fall within any $\lambda x \phi$; otherwise it is *bound*. The result $\psi\{\phi/x\}$ of substituting semantic variable x (of semantic type τ) by semantic term ϕ (of semantic type τ) is the result of replacing by ϕ every free occurrence of x in ψ ; the substitution $\psi\{\phi/x\}$ is *free* if and only if no variable becomes bound in the process of replacement. A *semantic form* is a semantic term containing no free variables.

A *semantic interpretation* comprises a semantic structure $\{D_\tau\}_{\tau \in \mathbf{T}}$, a *semantic assignment* g mapping from each X_τ into D_τ , and a *semantic valuation* f

mapping from each C_τ into D_τ such that:

$$\begin{aligned}
(10) \quad & f(\neg)(m) = \overline{m}^{\{\emptyset\}} \\
& f(\wedge)(m) = m' \mapsto m \cap m' \\
& f(\vee)(m) = m' \mapsto m \cup m' \\
& f(\rightarrow)(m) = m' \mapsto \overline{m}^{\{\emptyset\}} \cup m' \\
& f(=)(m) = m' \mapsto \{\emptyset\} \text{ if } m = m' \text{ else } \emptyset \\
& f(\forall)(m) = \bigcap_{m' \in E} m(m') \\
& f(\exists)(m) = \bigcup_{m' \in E} m(m') \\
& f(\iota)(\{m\}) = m
\end{aligned}$$

The *semantic value* $[\phi]^g \in D_\tau$ of a semantic term $\phi \in \Phi_\tau$ with respect to a semantic interpretation with semantic valuation f and semantic assignment g is defined by:

$$\begin{aligned}
(11) \quad & [c]^g = f(c) \text{ for semantic constant } c \\
& [x]^g = g(x) \text{ for semantic variable } x \\
& [(\phi \ \psi)]^g = [\phi]^g([\psi]^g) \quad \text{functional application} \\
& [\pi_1 \phi]^g = \mathbf{fst}([\phi]^g) \quad \text{first projection} \\
& [\pi_2 \phi]^g = \mathbf{snd}([\phi]^g) \quad \text{second projection} \\
& [\lambda x \phi]^g = D_\tau \ni m \mapsto [\phi]^{(g - \{(x, g(x))\}) \cup \{(x, m)\}} \text{ for } x \in V_\tau \quad \text{functional abstraction} \\
& [(\phi, \psi)]^g = \langle [\phi]^g, [\psi]^g \rangle \quad \text{pair formation}
\end{aligned}$$

Semantic terms ϕ and ψ are *equivalent*, $\phi \equiv \psi$, if and only if $[\alpha]^g = [\beta]^g$ in every semantic interpretation. We have the laws of $\alpha\beta\eta$ -conversion:

- (12) a. $\lambda x \phi \equiv \lambda y (\phi\{y/x\})$ α -conversion
provided y is not free in ϕ and $\phi\{y/x\}$ is free
- b. $(\lambda x \phi \psi) \equiv \phi\{\psi/x\}$ β -conversion
provided $\phi\{\psi/x\}$ is free
 $\pi_1(\phi, \psi) \equiv \phi$ $\pi_2(\phi, \psi) \equiv \psi$
- c. $\lambda x (\phi x) \equiv \phi$ η -conversion
provided x is not free in ϕ
 $(\pi_1 \phi, \pi_2 \phi) \equiv \phi$

We also have semantic equivalences arising in virtue of the logical semantic constants, for example:

- (13) $(\wedge (= \phi \phi) \psi) \equiv \psi$
 $(\exists \lambda x (\wedge \phi (= x \psi))) \equiv \phi\{\psi/x\}$ provided $\phi\{\psi/x\}$ is free

2.3 Syntax

Let there be a set $\mathbf{A}$ of *atomic syntactic types*. Then the set $\mathbf{F}$ of *syntactic types* is defined by:

- (14) $\mathbf{F} ::= \mathbf{A} \mid \mathbf{F} \cdot \mathbf{F} \mid \mathbf{F} \backslash \mathbf{F} \mid \mathbf{F} / \mathbf{F}$

The operators $\cdot$, $\backslash$ and $/$ are referred to product, left division and right division respectively. Let there be a *basic type map* t mapping from $\mathbf{A}$ into $\mathbf{T}$. This

induces the *type map* T from $\mathbf{F}$ into $\mathbf{T}$ such that:

$$\begin{aligned}
(15) \quad T(P) &= t(P) \text{ for atomic syntactic type } P \\
T(A \cdot B) &= T(A) \& T(B) \\
T(A \setminus C) &= T(A) \rightarrow T(C) \\
T(C/B) &= T(B) \rightarrow T(C)
\end{aligned}$$

A *syntactic interpretation* comprises a prosodic structure $(L, +)$, a semantic structure $\{D\}_{\tau \in \mathbf{T}}$, and a syntactic valuation F sending each $P \in \mathbf{A}$ into a subset of $L \times D_{t(P)}$. Then the value $\llbracket A \rrbracket \subseteq L \times D_{T(A)}$ of a syntactic type with respect to a syntactic interpretation with valuation F is defined by:

$$\begin{aligned}
(16) \quad \llbracket P \rrbracket &= F(P) \text{ for atomic syntactic type } P \\
\llbracket A \cdot B \rrbracket &= \{(s_1 + s_2, \langle m_1, m_2 \rangle) \mid \exists (s_1, m_1) \in \llbracket A \rrbracket, (s_2, m_2) \in \llbracket B \rrbracket\} \\
\llbracket A \setminus C \rrbracket &= \{(s_2, m_2) \mid \forall (s_1, m_1) \in \llbracket A \rrbracket, (s_1 + s_2, m_2(m_1)) \in \llbracket C \rrbracket\} \\
\llbracket C/B \rrbracket &= \{(s_1, m_1) \mid \forall (s_2, m_2) \in \llbracket B \rrbracket, (s_1 + s_2, m_1(m_2)) \in \llbracket C \rrbracket\}
\end{aligned}$$

A *type assignment statement* $\alpha - \phi$: A comprises a syntactic type A , a prosodic form α , and a semantic form ϕ of semantic type $T(A)$. A prosodic, semantic and syntactic interpretation is a *model of a type assignment statement* $\alpha - \phi$: A if and only if $\langle [\alpha], [\phi]^g \rangle \in \llbracket A \rrbracket$; it is a *model of a set Σ of type assignment statements* if and only if it is a model of every type assignment statement $\sigma \in \Sigma$.

A set Σ of type assignment statements *entails* a type assignment statement σ , $\Sigma \models \sigma$, if and only if every model of Σ is also a model of σ . A *lexicon* is a set of type assignment statements. The *language* $\mathbf{L}(Lex)$ defined by a lexicon Lex is the set of type assignment statements that it entails:

$$(17) \quad \mathbf{L}(Lex) = \{\alpha - \phi: A \mid Lex \models \alpha - \phi: A\}$$

and— $\lambda x \lambda y \lambda z (\wedge (y \ z) (x \ z))$: $((N \backslash S) \backslash (N \backslash S)) / (N \backslash S)$
by— $\lambda x \lambda y \lambda z (\wedge (y \ z) (= x \ z))$: $((N \backslash S-) \backslash (N \backslash S-)) / N$
cow—*cow*: CN
dog—*dog*: CN
gave—*give*: $((N \backslash S) / PP) / N$
given— $(\lambda x \lambda y \lambda z (\wedge (y \ z) (\exists (x \ z))), \textit{give})$: $((CN \backslash CN) / (N \backslash (N \backslash S-))) \cdot (N \backslash ((N \backslash S-) / PP))$
hit—*hit*: $(N \backslash S) / N$
hit— $(\lambda x \lambda y \lambda z (\wedge (y \ z) (\exists (x \ z))), \textit{hit})$: $((CN \backslash CN) / (N \backslash (N \backslash S-))) \cdot (N \backslash (N \backslash S-))$
it— $\lambda x x$: NPit
kissed—*kiss*: $(N \backslash S) / N$
rat—*rat*: CN
that— $\lambda x \lambda y \lambda z (\wedge (y \ z) (x \ z))$: $(CN \backslash CN) / (N \backslash S)$
that— $\lambda x \lambda y \lambda z (\wedge (y \ z) (x \ z))$: $(CN \backslash CN) / (S / N)$
the— ι : N / CN
to— $\lambda x x$: PP / N
was— $\lambda x \lambda y (x (= y) y)$: $(N \backslash S) / (CN \backslash CN)$
was— $\lambda x \lambda y \lambda z (z (y (= x) x))$: $((NPit \backslash S) / (CN \backslash CN)) / N$

Fig. 1. Type logical lexicon

3 Grammar

Let there be the following basic types and the lexicon in figure 1.

(18) Atomic syntactic type	P	$t(P)$
count noun	CN	$e \rightarrow t$
proper name	N	e
expletive pronoun	NPit	$t \rightarrow t$
prepositional phrase	PP	e
declarative sentence	S	t
abstract passive sentence	S-	t

Then the language model defined includes the following:

- (19) A **the+rat+hit+the+dog**–
 $(hit (\iota dog) (\iota rat))$: S
- D **the+rat+gave+the+dog+to+the+cow**–
 $(give (\iota dog) (\iota cow) (\iota rat))$: S
- C **the+rat+hit+the+dog+and+kissed+the+cow**–
 $(\wedge (hit (\iota dog) (\iota rat)) (kiss (\iota cow) (\iota rat)))$: S
- P **the+rat+was+hit+by+the+dog**–
 $(hit (\iota rat) (\iota dog))$: S
- DP **the+rat+was+given+to+the+dog+by+the+cow**–
 $(give (\iota rat) (\iota dog) (\iota cow))$: S
- OS **the+rat+hit+the+dog+that+kissed+the+cow**–
 $(hit (\iota \lambda z (\wedge (dog z) (kiss (\iota cow) z))) (\iota rat))$: S
- SO **the+rat+that+the+dog+hit+kissed+the+cow**–
 $(kiss (\iota cow) (\iota \lambda z (\wedge (rat z) (hit z (\iota dog))))$: S
- CS **it+was+the+rat+that+hit+the+dog**–
 $(hit (\iota dog) (\iota rat))$: S
- CO **it+was+the+rat+that+the+dog+hit**–
 $(hit (\iota rat) (\iota dog))$: S

The derivations, or syntactic structures, of TLG are (proof) nets ([8], [16]), which we define in the next section.

4 Nets for basic Type Logical Grammar

A *label* is a syntactic type together with a polarity input ($\bullet$) or output ($\circ$). Where p is a polarity, $\bar{p}$ is the opposite polarity. Labels A^p and $A^{\bar{p}}$ are *complementary*. A *literal* is a label the type of which is atomic.

A *link* has a list of *premise* labels (above) and a list of *conclusion* labels

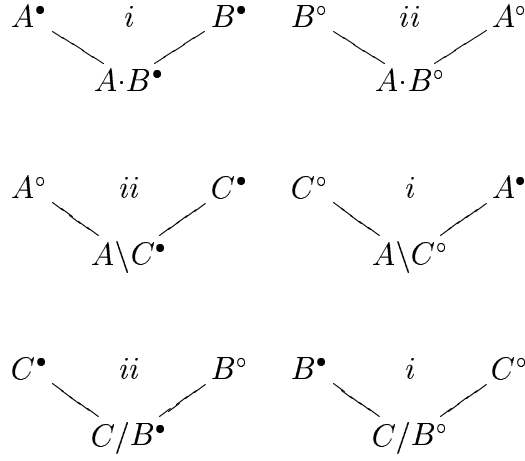


Fig. 2. Logical links

(below) with some edges between them. An *identity link* is of the form:

$$(20) \quad \begin{array}{ccc} & \text{---} & \\ | & & | \\ P^p & & P^{\bar{p}} \end{array}$$

A *logical link* is one of the local trees given in Figure 2. A *polar type tree* is a tree the leaves of which are literals and each local tree of which is a logical link. An *input polar type tree* is one the root of which has input polarity; an *output polar type tree* is one the root of which has output polarity. Each label is the root of a unique polar type tree, which is the result of unfolding the label upwards according to the logical links. For example, the (input) polar type tree for $(CN \setminus CN)/(S/N)^{\bullet}$ is:

$$(21) \quad \begin{array}{ccccc} CN^{\circ} & & ii & & CN^{\bullet} & & N^{\bullet} & & i & & S^{\circ} \\ & \swarrow & & \searrow & & & & \swarrow & & \searrow & \\ & CN \setminus CN^{\bullet} & & & & & & S/N^{\circ} & & & \\ & & & & ii & & & & & & \\ & & & & (CN \setminus CN)/(S/N)^{\bullet} & & & & & & \end{array}$$

A *frame* is a list comprising an output polar type tree followed by some input polar type trees. For example, the following is a frame:

$$(22) \quad \begin{array}{ccccccc} & & & S^{\circ} & & ii & & S^{\bullet} \\ & & & \swarrow & & \searrow & & \\ & & & S \setminus S^{\bullet} & & & & ii & & N^{\circ} \\ & N^{\circ} & & ii & & S^{\bullet} & & & & \\ S^{\circ} & N^{\bullet} & & N \setminus S^{\bullet} & & & & (S \setminus S)/N^{\bullet} & & N^{\bullet} \end{array}$$

A *net* is the result of connecting by an identity link every leaf in a frame with

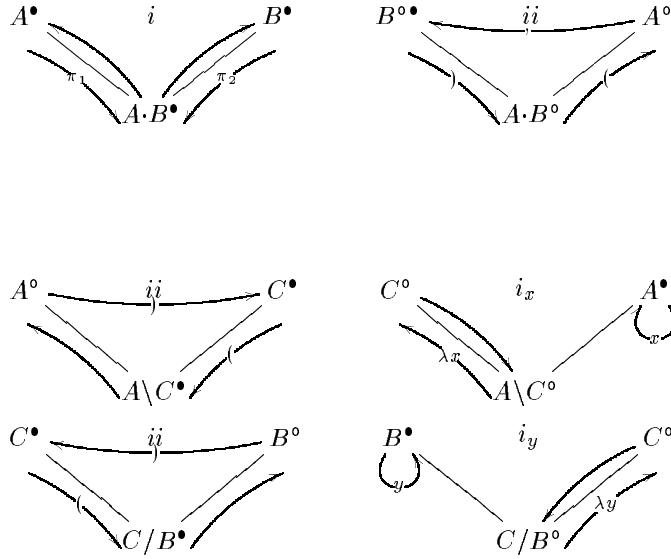
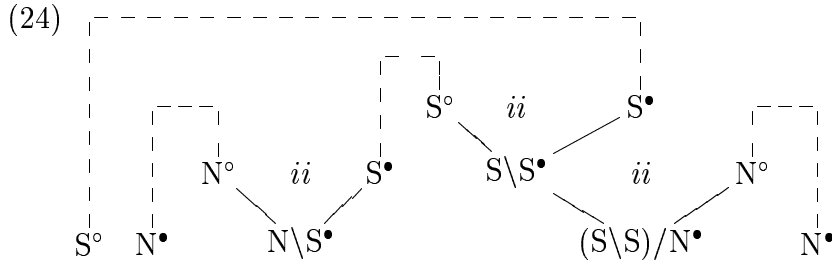


Fig. 3. Semantic trip travel instructions

a unique complementary leaf such that:

- (23) *planarity* The identity links are planar in the list ordering.
acyclicity Every cycle crosses both edges of some *i*-link.
no subtending No identity link connects the leftmost and rightmost
descendent leaves of an output division node.

(For planarity see [16] and [1]; for acyclicity see [4]; for no subtending see [6].)
For example, the following is a net:



Let there be semantic forms of the appropriate type associated with each input polar tree in a net. The *semantic trip* of the net with each output division logical link assigned a distinct semantic variable is the trip which originates upwards at the unique output root and generates a textual form proceeding as shown in Figure 3 and bouncing with the associated semantic form at input roots ([5]). The semantic trip ends when it returns downwards at the origin. The *reading* ϕ_{Π} of a net Π is the textual form generated by its semantic trip.

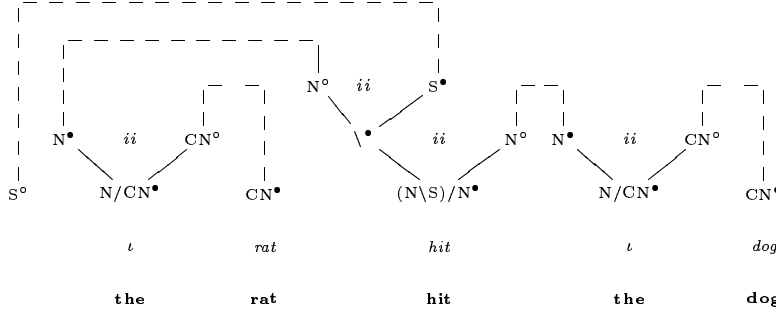


Fig. 4. Net for an Active (A) sentence

Then we have the following:

Proposition 1 (Soundness). *If $\alpha_1 - \phi_1: A_1, \dots, \alpha_n - \phi_n: A_n \in Lex$ then for every net Π over the frame $A_0^\circ, \overset{A_1^\bullet}{\phi_1}, \dots, \overset{A_n^\bullet}{\phi_n}, \alpha_1 + \dots + \alpha_n - \phi_\Pi: A_0 \in \mathbf{L}(Lex)$.*

Proposition 2 (Completeness). *If $\alpha - \phi: A_0 \in \mathbf{L}(Lex)$ then there exist $\alpha_1 - \phi_1: A_1, \dots, \alpha_n - \phi_n: A_n \in Lex$ and a net Π over the frame $A_0^\circ, \overset{A_1^\bullet}{\phi_1}, \dots, \overset{A_n^\bullet}{\phi_n}$ such that $\alpha \equiv \alpha_1 + \dots + \alpha_n$ and $\phi \equiv \phi_\Pi$.*

5 Sentences

In the following subsections we give the nets and semantics for each of the sentence types. We also give the complexity profiles, these being a plot of the number of unbounded dependencies (overarching identity links) at each word boundary. Following Johnson (1988) we refer to this number as the ‘cut’ and we note maximal cuts and average cuts.

5.1 Active (A)

The net for a sentence of this type is given in figure 4. The result of the semantic trip is the following:

$$(25) (hit (\iota \text{ dog}) (\iota \text{ rat}))$$

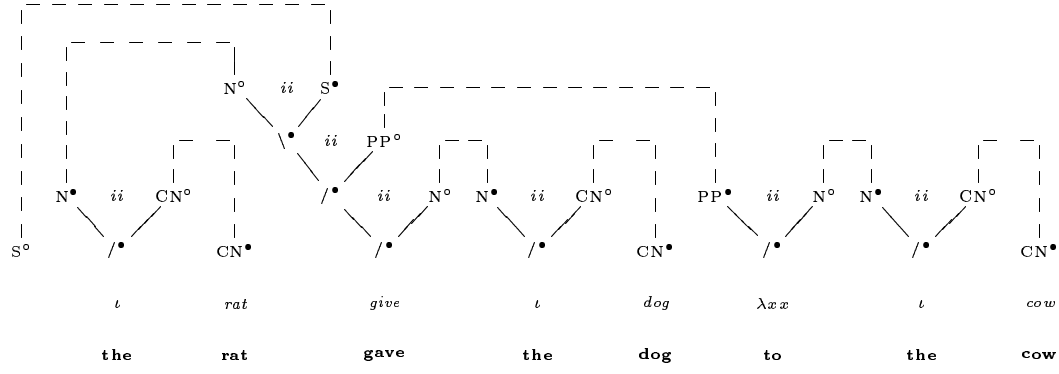


Fig. 5. Net for a Dative (D) sentence

The complexity profile is thus:

(26)	3		a						
	2			a					
	1	a			a		a		
	0							a	
		The rat hit the dog.							

The maximal cut is 3; the average cut is 1.33.

5.2 Dative (D)

The net for a sentence of this type is given in figure 5. The result of the semantic trip is (27a) which normalizes to (27b).

- (27) a. $(give (\iota dog) (\lambda xx (\iota cow) (\iota rat)))$
 b. $(give (\iota dog) (\iota cow) (\iota rat))$

The complexity profile is as follows:

(28)	3		a						
	2			a		a		a	
	1	a					a	a	a
	0								a
		The rat gave the dog to the cow.							

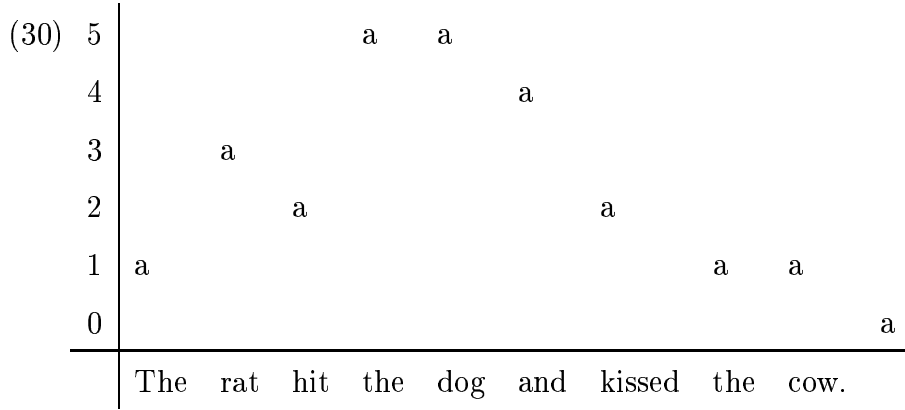
The maximal cut is 3; the average cut is 1.44.

5.3 Conjoined (C)

The net for a sentence of this type is given in figure 6. The normalized semantics is as follows.

$$(29) (\wedge (hit (\iota dog) (\iota rat)) (kiss (\iota cow) (\iota rat)))$$

The complexity profile is thus:



The maximal cut is 5; the average cut is 2.40.

5.4 Passive (P)

The net for a sentence of this type is given in figure 7; the analysis of passive is that of [15]. The normalized semantics is (31a) which is logically equivalent to (31b).

$$(31) \text{ a. } (\wedge (= (\iota rat) (\iota rat)) (\exists \lambda z (\wedge (hit (\iota rat) z) (= (\iota dog) z))))$$

$$\text{ b. } (hit (\iota rat) (\iota dog))$$

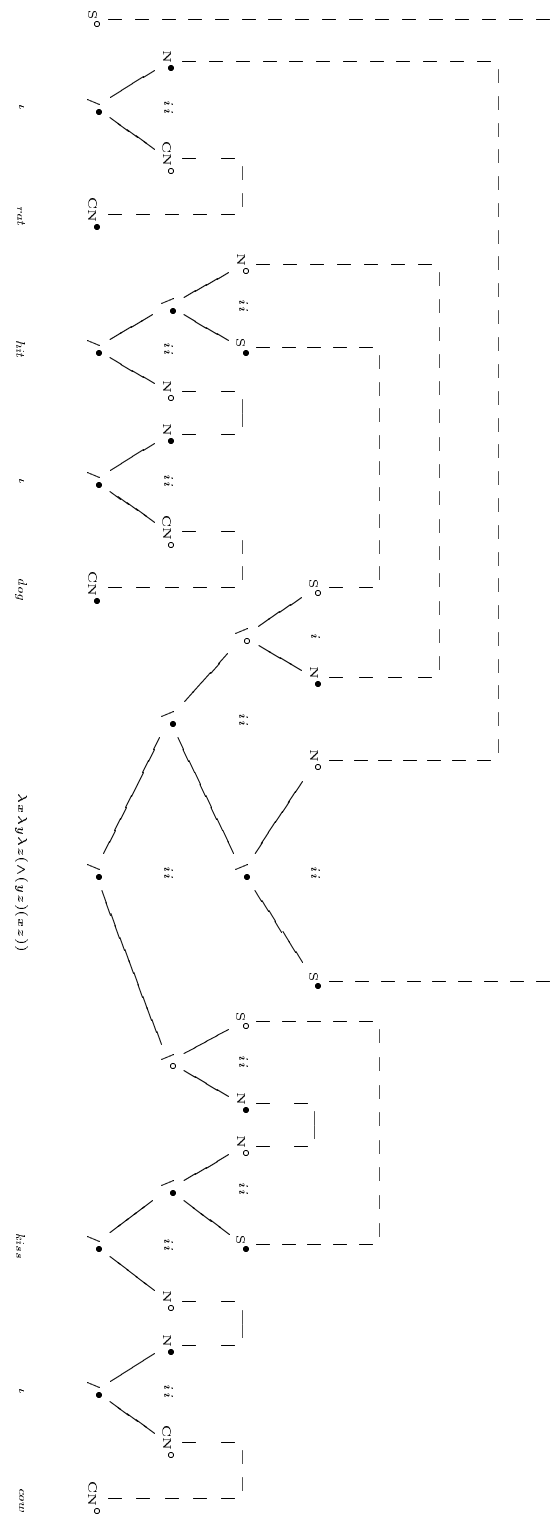
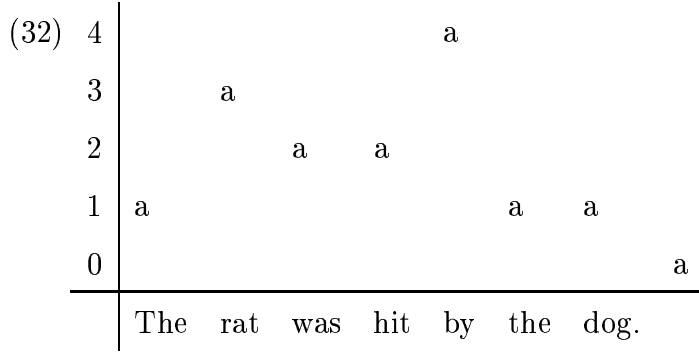


Fig. 6. Net for a Conjoined (C) sentence



The complexity profile is as follows:



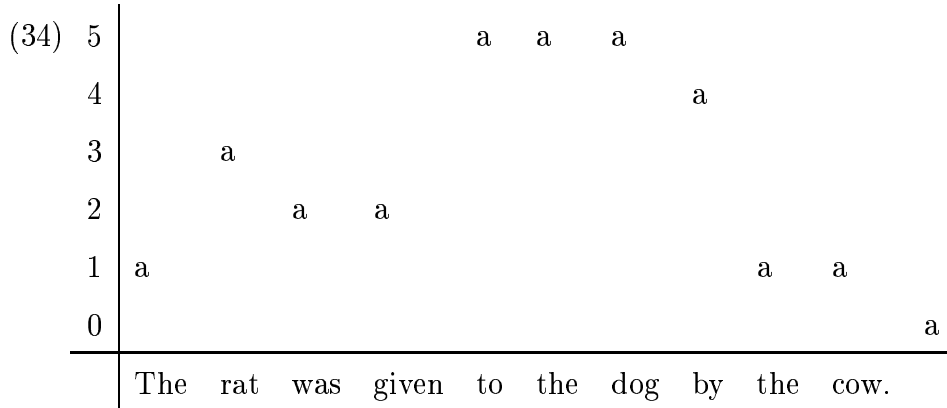
The maximal cut is 4; the average cut is 1.75.

5.5 Dative Passive (DP)

The net for a sentence of this type is given in figures 8 and 9; the analysis of dative passive is a generalization of that of passive of [15]. The normalized semantics is (33a) which is logically equivalent to (33b).

- (33) a. $(\wedge (= (\iota \text{ rat}) (\iota \text{ rat})) (\exists \lambda z (\wedge (\text{give } (\iota \text{ rat}) (\iota \text{ dog}) z) (= (\iota \text{ cow}) z))))$
b. $(\text{give } (\iota \text{ rat}) (\iota \text{ dog}) (\iota \text{ cow}))$

The complexity profile is as follows:



The maximal cut is 5; the average cut is 2.64.

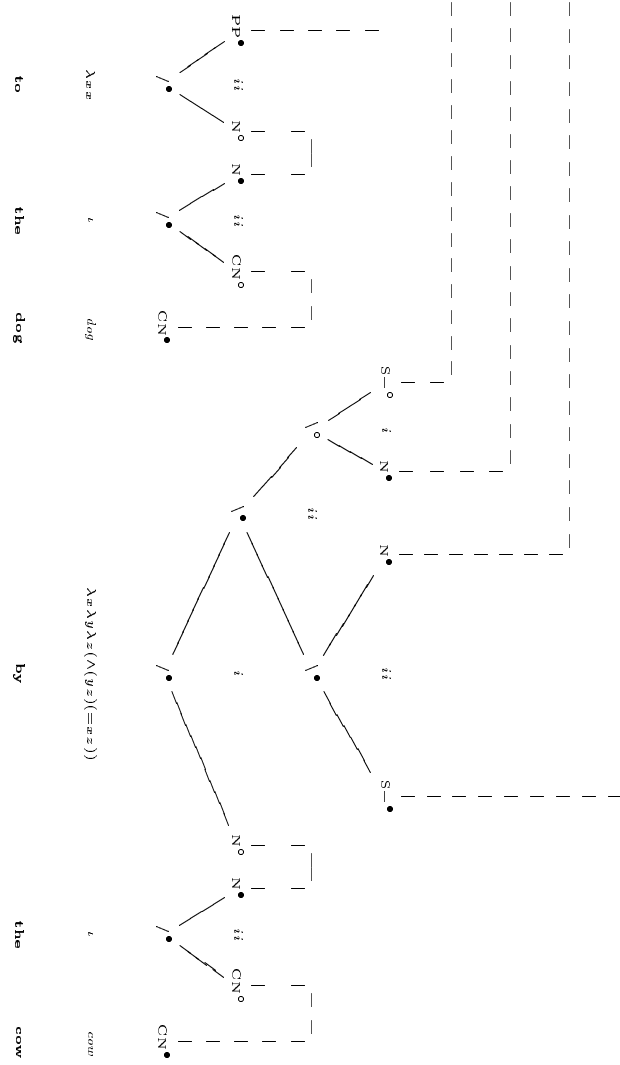
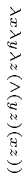


Fig. 9. Net for a Dative Passive (DP) sentence, part II

5.6 Object-Subject relative (OS)

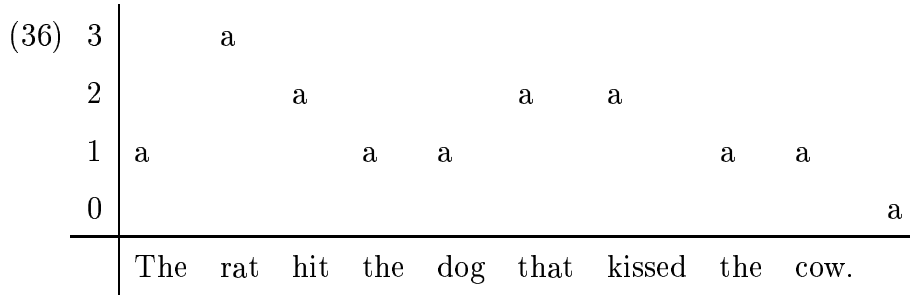
The net for a sentence of this type is given in figure 10. The normalized semantics is as follows:

$$(35) (hit (\iota \lambda z (\wedge (dog z) (kiss (\iota cow) z))) (\iota rat))$$



21

The complexity profile is thus:



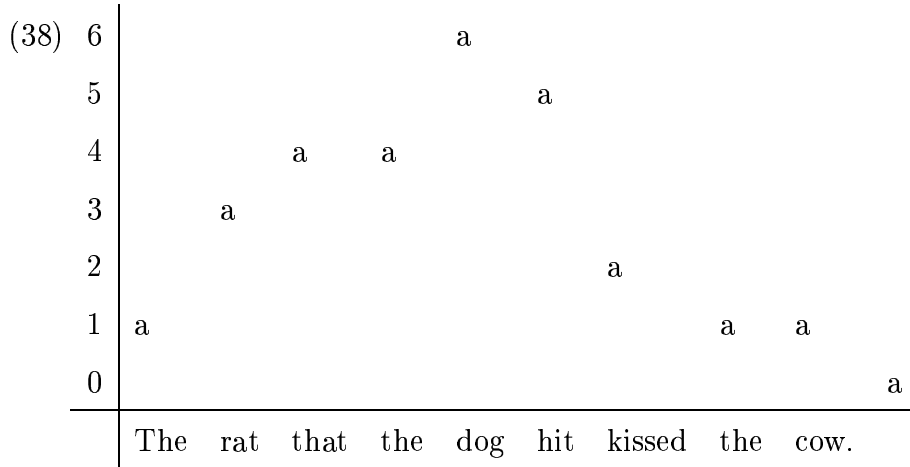
The maximal cut is 3; the average cut is 1.40.

5.7 Subject-Object relative (SO)

The net for a sentence of this type is given in figure 11. The normalized semantics is as follows:

$$(37) (kiss (\iota \text{ cow}) (\iota \lambda z (\wedge (\text{rat } z) (\text{hit } z (\iota \text{ dog}))))))$$

The complexity profile is thus:



The maximal cut is 6; the average cut is 2.70.

5.8 Cleft-Subject (CS)

The net for a sentence of this type is given in figure 12; the analysis of the expletive subject of cleft constructions would appear to be new to this paper.

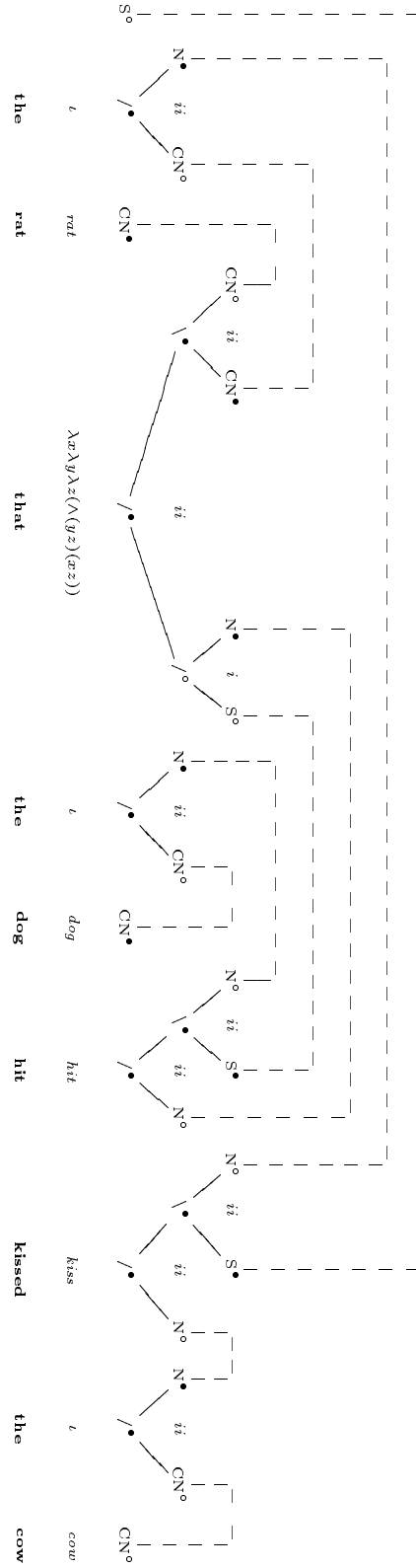


Fig. 11. Net for a Subject-Object relative (SO) sentence

The normalized semantics is (39a) which is logically equivalent to (39b).

- (39) a. $(\wedge (= (\iota \textit{ rat}) (\iota \textit{ rat})) (\textit{hit} (\iota \textit{ dog}) (\iota \textit{ rat})))$
 b. $(\textit{hit} (\iota \textit{ dog}) (\iota \textit{ rat}))$

The complexity profile is as follows:

(40)	3			a		a						
	2		a				a		a			
	1	a								a		a
	0											a
		It	was	the	rat	that	hit	the	dog.			

The maximal cut is 3; the average cut is 1.67.

5.9 Cleft-Object (CO)

The net for a sentence of this type is given in figure 13. The normalised semantics is (41a) which is logically equivalent to (41b).

- (41) a. $(\wedge (= (\iota \textit{ rat}) (\iota \textit{ rat})) (\textit{hit} (\iota \textit{ dog}) (\iota \textit{ rat})))$
 b. $(\textit{hit} (\iota \textit{ dog}) (\iota \textit{ rat}))$

The complexity profile is as follows:

(42)



Word	Position	Count
It	1	1
was	2	2
the	3	3
rat	4	3
that	5	2
the	6	2
dog	7	4
hit.	8	0

The maximal cut is 4; the average cut is 2.22.

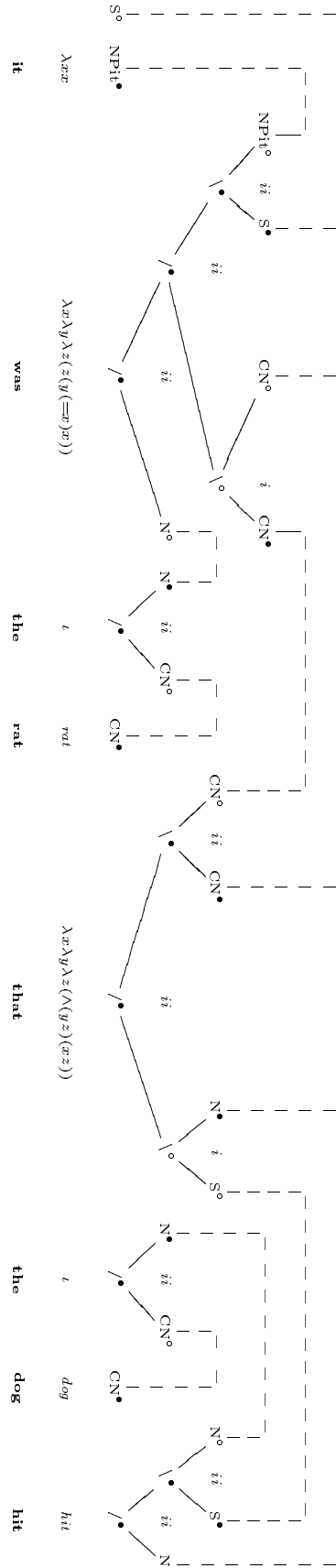


Fig. 13. Net for a Cleft-Object (CO) sentence

6 Results

The mean comprehension results on a scale of 0 (worst) to 5 (best) were as follows ([3, p.105]), compared to the maximal cut and average cut of the proof net analyses:

(43)	Sentence type	mean comprehension	maximal cut	average cut
	A	4.4	3	1.33
	CS	4.2	3	1.67
	D	3.2	3	1.44
	P	2.9	4	1.75
	C	2.7	5	2.40
	CO	2.6	4	2.22
	OS	2.3	3	1.40
	DP	2.0	5	2.64
	SO	1.3	6	2.70

The maximal cut and average cut correlate quite well with the mean comprehension with the exception of the OS sentence type. As we noted in the introduction, this sentence type is ambiguous, allowing a right extraposed reading, so that the existence of this reading would be expected to detract from the prescribed interpretation. Otherwise, in particular, the A sentence type has the best comprehension and the lowest cut complexity and the SO sentence type has the worst comprehension and the highest cut complexity.

7 Conclusion

We argue that the aphasic comprehension of the sentence types of the Caplan et al. experiment correlates with the proof net complexity of these sentence types. The measure of complexity with which we correlate aphasic comprehension is an immediate reflection of grammar, no ancilliary assumption is made beyond that that dependencies are of uniform cost.

References

- [1] V. Michele Abrusci. Noncommutative Proof Nets. In Yves Lafont, Jean-Yves Girard, and Laurent Regnier, editors, *Advances in Linear Logic*, London Mathematical Society Lecture Note Series 222, pages 271–296. Cambridge University Press, Cambridge, 1995.
- [2] D. Caplan, C. Baker, and F. Dehaut. Syntactic determinants of sentence comprehension in aphasia. *Cognition*, 21: 117–175, 1985.
- [3] David Caplan and Nancy Hildecrandt. *Disorders of Syntactic Comprehension*. The MIT Press, Cambridge, Massachusetts, 1988.
- [4] V. Danos and L. Regnier. The structure of multiplicatives. *Archive for Mathematical Logic*, **28**: 181–203, 1989.
- [5] Philippe de Groote and Christian Retoré. On the Semantic Readings of Proof-Nets. In Geert-Jan Kruijff, Glyn Morrill, and Richard T. Oehrle, editors, *Proceedings of Formal Grammar 1996*, pages 57–70, Prague, 1996.
- [6] Philippe de Groote and Christian Retoré. *Proof-theoretic methods in computational linguistics*. Lecture notes 15th European Summer School in Logic, Language and Information, Vienna, 2003.
- [7] Edward Gibson. Linguistic complexity: locality of syntactic dependencies. *Cognition*, **68**:1–76, 1988.
- [8] Jean-Yves Girard. Linear Logic. *Theoretical Computer Science*, **50**: 1–102, 1987.
- [9] Yosef Grodzinsky. The neurology of syntax: Language use without Broca’s area. *Behavioural and Brain Sciences*, **23**:1–71, 2000.
- [10] Henk J. Haarmann, Marcel Adam Just, and Patricia A. Carpenter. Aphasic Sentence Comprehension as a Resource Deficit: A Computational Approach. *Brain and Language*, 59: 76–100, 1997.
- [11] M. Johnson. Proof nets and the complexity of processing center-embedded constructions. *Journal of Logic, Language and Information*, 7(4):433–447, 1988.
- [12] J. Lambek. The mathematics of sentence structure. *American Mathematical Monthly*, **65**: 154–170, 1958.
- [13] J. Lambek. Categorical and Categorical Grammars. In Richard T. Oehrle, Emmon Bach and Deidre Wheeler, editors, *Categorical Grammars and Natural Language Structures*, Studies in Linguistics and Philosophy Volume 32, pages 297–317. D. Reidel, Dordrecht, 1988.
- [14] Glyn Morrill. *Type Logical Grammar: Categorical Logic of Signs*. Kluwer Academic Publishers, Dordrecht, 1994.
- [15] Glyn Morrill. Incremental Processing and Acceptability. *Computational Linguistics*, 26(3), 319–338. 2000.

- [16] Dirk Roorda. *Resource Logics: Proof-Theoretical Investigations*. PhD thesis, Universiteit van Amsterdam, 1991.
- [17] Michael K. Tanenhaus. Sentence Processing. In Lynn Nadal, editor, *Encyclopedia of Cognitive Science, Volume 3*, pages 1142–1148. Nature Publishing Group, Macmillan, London, 2003.

Incremental Processing and Acceptability

Glyn Morrill*

Universitat Politècnica de Catalunya

We present a left to right incremental algorithm for the processing of Lambek categorial grammar by proof net construction. A simple metric of complexity, the profile in time of the number of unresolved valencies, correctly predicts a wide variety of performance phenomena: garden pathing, left to right quantifier scope preference, centre embedding unacceptability, preference for lower attachment, and heavy noun phrase shift.

Introduction

Contemporary linguistics rests on abstractions and idealizations which, however fruitful, must eventually be integrated in a mature cognitive science with human computational performance in language use. In this paper we consider the case of language processing on the basis of Lambek categorial grammar (Lambek 1958). We argue that an incremental algorithm of proof net construction affords an account of various processing phenomena, including:

- (1)
 - garden pathing
 - left to right quantifier scope preference
 - centre embedding unacceptability

Garden pathing (Bever 1970) is illustrated by the following contrasts:

- (2)
 - a. The horse raced past the barn.
 - b. ?The horse raced past the barn fell.
- (3)
 - a. The boat floated down the river.
 - b. ?The boat floated down the river sank.
- (4)
 - 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 due to a strong tendency to interpret their initial segments as in the 'a' sentences.

Left to right quantifier scope preference is illustrated by:

- (5)
 - a. Someone loves everyone.
 - b. Everyone is loved by someone.

Both sentences exhibit both quantifier scopings:

* Departament de Llenguatges i Sistemes Informàtics, Mòdul C 5 - Campus Nord, Jordi Girona Salgado 1-3, E-08034 Barcelona. E-mail: morrill@lsi.upc.es; <http://www-lsi.upc.es/~glyn/>.

- (6) a. $\exists x \forall y (\text{love } y \ x)$
 b. $\forall y \exists x (\text{love } y \ x)$

However, while the dominant reading of (5a) is (6a), that of (5b) is (6b), i.e. the preference is for the first quantifier to have wider scope. Note that the same effect is observed when the nature of the quantifications is swapped:

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

While both sentences in (7) have both quantifier scopings, the preferred readings give the first quantifier wide scope.

Centre embedding unacceptability is illustrated by the fact that while the nested subject relativizations of (8) exhibit little variation in acceptability, the nested object relativizations (9) exhibit a severe deterioration in acceptability (Chomsky 1965, ch. 1).

- (8) 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.
- (9) 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.

We argue that a single simple metric of categorial processing complexity explains these and other performance phenomena.

1 Lambek calculus

We shall assume some familiarity with Lambek categorial grammar as presented in e.g. Moortgat (1988), Morrill (1994), Moortgat (1997) or Carpenter (1998), and limit ourselves here to reviewing some technical and computational aspects.

The types, or (category) formulas, of Lambek calculus are freely generated from a set of primitives by binary infix connectives $/$ (“over”), $\backslash$ (“under”) (directional divisions) and $\cdot$ (“product”). With respect to a semigroup algebra $(L, +)$ (i.e. a set L closed under an associative binary operation $+$ of adjunction) each formula A is interpreted as a subset $\llbracket A \rrbracket$ of L by residuation as follows.

- (10) $\llbracket A \cdot B \rrbracket = \{s_1 + s_2 \mid s_1 \in \llbracket A \rrbracket \wedge s_2 \in \llbracket B \rrbracket\}$
 $\llbracket A \backslash B \rrbracket = \{s \mid \forall s' \in \llbracket A \rrbracket, s' + s \in \llbracket B \rrbracket\}$
 $\llbracket B / A \rrbracket = \{s \mid \forall s' \in \llbracket A \rrbracket, s + s' \in \llbracket B \rrbracket\}$

A sequent, $\Gamma \Rightarrow A$, comprises a succedent formula A and an antecedent configuration Γ which is a finite sequence of formulas.¹ A sequent is valid if and only if in all interpretations the ordered adjunction of elements inhabiting the antecedent formulas always yields an element inhabiting the succedent formula. The following Gentzen-style sequent presentation is sound and complete for this interpretation (Buszkowski 1986, Došen 1992), and indeed for free semigroups (Pentus 1994): hence the Lambek calculus

¹ Officially the antecedent is non-empty, a detail we gloss over.

can make an impressive claim to be *the* logic of concatenation; the parenthetical notation $\Gamma(\Delta)$ represents a configuration containing a distinguished subconfiguration Δ .

$$\begin{aligned}
 (11) \quad & \text{a. } A \Rightarrow A \quad \text{id} \quad \frac{\Gamma \Rightarrow A \quad \Delta(A) \Rightarrow B}{\Delta(\Gamma) \Rightarrow B} \text{Cut} \\
 & \text{b. } \frac{\Gamma \Rightarrow A \quad \Delta(B) \Rightarrow C}{\Delta(\Gamma, A \setminus B) \Rightarrow C} \setminus L \quad \frac{A, \Gamma \Rightarrow B}{\Gamma \Rightarrow A \setminus B} \setminus R \\
 & \text{c. } \frac{\Gamma \Rightarrow A \quad \Delta(B) \Rightarrow C}{\Delta(B/A, \Gamma) \Rightarrow C} /L \quad \frac{\Gamma, A \Rightarrow B}{\Gamma \Rightarrow B/A} /R \\
 & \text{d. } \frac{\Gamma(A, B) \Rightarrow C}{\Gamma(A \cdot B) \Rightarrow C} \cdot L \quad \frac{\Gamma \Rightarrow A \quad \Delta \Rightarrow B}{\Gamma, \Delta \Rightarrow A \cdot B} \cdot R
 \end{aligned}$$

By way of example, “lifting” $A \Rightarrow B/(A \setminus B)$ is generated as follows:

$$(12) \quad \frac{\frac{A \Rightarrow A \quad B \Rightarrow B}{A, A \setminus B \Rightarrow B} \setminus L}{A \Rightarrow B/(A \setminus B)} /R$$

And “composition” $A \setminus B, B \setminus C \Rightarrow A \setminus C$ is thus:

$$(13) \quad \frac{\frac{A \Rightarrow A \quad \frac{B \Rightarrow B \quad C \Rightarrow C}{B, B \setminus C \Rightarrow C} \setminus L}{A, A \setminus B, B \setminus C \Rightarrow C} \setminus L}{A \setminus B, B \setminus C \Rightarrow A \setminus C} \setminus R$$

Every rule with the exception of Cut, where the Cut formula A does not appear in the conclusion, has exactly one connective occurrence less in its premisses than in its conclusion; Lambek (1958) proved Cut elimination—that every proof has a Cut-free counterpart—hence a decision procedure for theoremhood is given by backward-chaining proof search in the Cut-free calculus. The non-atomic instances of the id axiom are derivable from atomic instances by the rules for the connectives. But even in the Cut-free atomic-id calculus there is spurious ambiguity: equivalent derivations differing only in irrelevant rule ordering.

One approach to this problem consists in defining normal form derivations in which the succession of rule application is regulated (König 1989, Hepple 1990; see e.g. Hendriks 1993 ch. 3). Each sequent has a distinguished category formula (underlined) on which rule applications are keyed:

$$\begin{aligned}
 (14) \quad & \text{a. } \underline{P} \Rightarrow P \quad \text{id} \\
 & \text{b. } \frac{\Gamma \Rightarrow \underline{A} \quad \Delta(\underline{B}) \Rightarrow C}{\Delta(\Gamma, \underline{A \setminus B}) \Rightarrow C} \setminus L \quad \frac{A, \Gamma \Rightarrow \underline{B}}{\Gamma \Rightarrow \underline{A \setminus B}} \setminus R
 \end{aligned}$$

- c.
$$\frac{\Gamma \Rightarrow \underline{A} \quad \Delta(\underline{B}) \Rightarrow C}{\Delta(\underline{B/A}, \Gamma) \Rightarrow C} /L \quad \frac{\Gamma, A \Rightarrow \underline{B}}{\Gamma \Rightarrow \underline{B/A}} /R$$
- d.
$$\frac{\Gamma \Rightarrow \underline{A} \quad \Delta \Rightarrow \underline{B}}{\Gamma, \Delta \Rightarrow \underline{A \cdot B}} R$$
- e.
$$\frac{\Gamma(\underline{A}) \Rightarrow B}{\Gamma(A) \Rightarrow \underline{B}}$$

In the regulated calculus there is no spurious ambiguity, and provided there is no explicit or implicit antecedent product, i.e. provided $\cdot L$ is not needed, $\Gamma \Rightarrow A$ is a theorem of the Lambek calculus iff $\Gamma \Rightarrow \underline{A}$ is a theorem of the regulated calculus. However, apart from the issue regarding $\cdot L$, there is a general cause for dissatisfaction with this approach: it assumes the initial presence of the entire sequent to be proved, i.e. it is in principle non-incremental; on the other hand, allowing incrementality on the basis of Cut would reinstate with avengance the problem of spurious ambiguity: for then what are to be the Cut formulas? Consequently, the sequent approach is ill-equipped to address the basic asymmetry of language, the asymmetry of its processing in time, and has never been forwarded in a model of the kind of processing phenomena cited in the introduction.

An alternative, combinatory, presentation would comprise combinatory schemata such as the following (together with a Cut rule, feeding one rule application into another):

(15)

- a. $A, A \backslash B \Rightarrow B \quad B/A, A \Rightarrow B$
- b. $A \Rightarrow (B/A) \backslash B \quad A \Rightarrow B/(A \backslash B)$
- c. $A \backslash B, B \backslash C \Rightarrow A \backslash C \quad C/B, B/A \Rightarrow C/A$

By a result of Zielonka (1981) the Lambek calculus is not axiomatizable by any finite set of combinatory schemata, so no such combinatory presentation can constitute *the* logic of concatenation in the sense of Lambek calculus. Combinatory categorial grammar (Steedman 1997) does not concern itself with the capture of all (or only) the concatenatively valid combinatory schemata, but from its inception (Ades and Steedman 1981) it has emphasized the capacity of schemata such as those of (15) to produce left-branching, and therefore incrementally processable, analyses, e.g. on a shift-reduce design. An approach, also based on regulation of the succession of rule application, to the associated problem of spurious ambiguity is given in Hepple and Morrill (1989) but again, to our knowledge, there is no predictive relation between incremental combinatory processing and the kind of processing phenomena cited in the introduction.

2 Proof nets

Lambek categorial derivations are usually presented in the style of natural deduction or sequent calculus. Here we concern ourselves with categorial proof nets (Roorda 1991) as the fundamental structures of proof in categorial logic, in the same sense that linear

proof nets were originally introduced by Girard (1987) as the fundamental structures of proof in linear logic. (Cut-free) proof nets exhibit no spurious ambiguity and play the role in categorical grammar that parse trees play in phrase structure grammar.

A *polar category formula* is a Lambek categorical type labelled with *input* ($\bullet$) or *output* ($\circ$) polarity. A *polar category formula tree* is a binary ordered tree in which the leaves are labelled with polar atoms (*literals*) and each local tree is one of the following (*logical links*):

(16)

$$\begin{array}{ll}
 \text{a.} & \frac{A^\circ \quad B^\bullet}{A \backslash B^\bullet} \text{ii} \quad \frac{B^\circ \quad A^\bullet}{A \backslash B^\circ} \text{i} \\
 \text{b.} & \frac{B^\bullet \quad A^\circ}{B / A^\bullet} \text{ii} \quad \frac{A^\bullet \quad B^\circ}{B / A^\circ} \text{i} \\
 \text{c.} & \frac{A^\bullet \quad B^\bullet}{A \cdot B^\bullet} \text{i} \quad \frac{B^\circ \quad A^\circ}{A \cdot B^\circ} \text{ii}
 \end{array}$$

Without polarities, a formula tree is a kind of construction tree of the formula at its root: daughters are labelled with the immediate subformulas of their mothers. The polarities indicate sequent sidedness, input for antecedent and output for succedent; the polarity propagation follows the sidedness of subformulas in the sequent rules: in the antecedent (input) rule for $A \backslash B$ the subformula A goes in a succedent (output) and the subformula B goes in an antecedent (input), in the succedent (output) rule for $A \backslash B$ the subformula A goes in an antecedent (input) and the subformula B goes in a succedent (output), etc. The labels i and ii indicate whether the corresponding sequent rule is unary or binary. Note that in the output links the order of the subformulas is switched; this corresponds to a cyclic reading of sequents: the succedent type is adjacent to the first antecedent type.

A *proof frame* is a finite sequence of polar category formula trees exactly one of which has a root of output polarity (corresponding to the unique succedent of sequents).

An *axiom linking* on a set of literal labelled leaves is a partitioning of the set into pairs of complementary leaves which is planar in their ordering, i.e. there are no two pairs $\{L_1, L_3\}, \{L_2, L_4\}$ such that $L_1 < L_2 < L_3 < L_4$. Geometrically, planarity means that where the leaves are ordered on a line, paired leaves can be connected in the half plane without crossing. Axiom links correspond to id instances in a sequent proof.

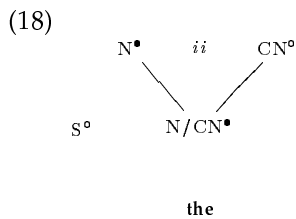
A *proof structure* is a proof frame together with an axiom linking on its leaves. A *proof net* is a proof structure in which every elementary (i.e. visiting vertices at most once) cycle crosses the edges of some i-link.² Geometrically, an elementary cycle is the perimeter of a face or cluster of faces in a planar proof structure. There is a proof net with roots $A^\circ, A_1^\bullet, \dots, A_n^\bullet$ iff $A_1, \dots, A_n \Rightarrow A$ is a valid sequent.

² This criterion, adapted from that of Lecomte and Retoré (1995), derives from Girard's (1987) 'long trip condition', which is a highly involved mathematical result. Danos and Regnier (1989) express it in terms of acyclicity and connectivity of certain subgraphs. Intuitively, acyclicity assures that the subformulas of ii-links (binary rules) occur in *different* subproofs, whereas connectivity assures that the subformulas of i-links (unary rules) occur in the *same* subproofs (attributed to Jean Gallier by Philippe de Groote, p.c.). However the single-succedent (intuitionistic) nature of Cut-free categorical proofs in fact renders the connectivity requirement redundant, hence we have just an acyclicity test.

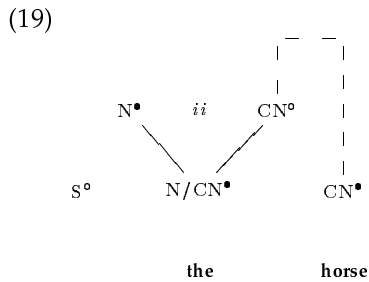
Let us assume the following lexical assignments:

- (17)
- | | | |
|--------------|----|--|
| barn | – | <i>barn</i> |
| | := | CN |
| horse | – | <i>horse</i> |
| | := | CN |
| past | – | $\lambda x \lambda y \lambda z (past\ x\ (y\ z))$ |
| | := | $((N \backslash St) \backslash (N \backslash St)) / N$ |
| raced | – | <i>race</i> |
| | := | $N \backslash S +$ |
| the | – | <i>the</i> |
| | := | N / CN |

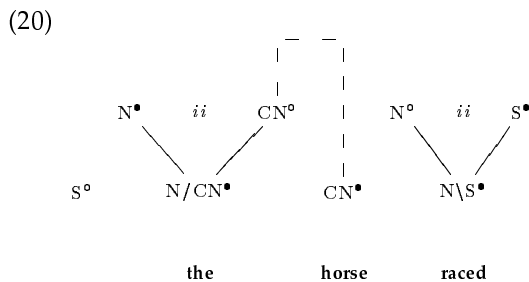
The feature + on S marks the projection of a tensed verb form; a verb phrase modified by “past” need not be tensed. Let us consider the incremental processing of (2a) as proof net construction. We assume initially that an S is expected; after perception of the word “the” there is the following partial proof net (for simplicity we omit features, included in lexical entries, from proof nets):



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



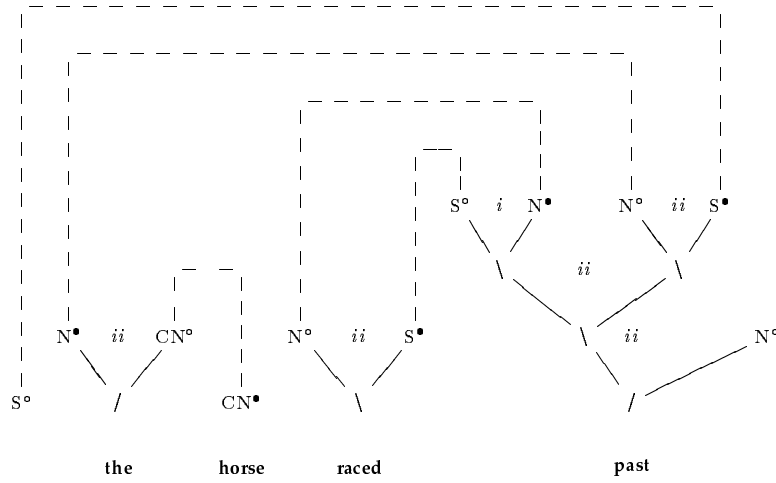
Now there are only two unmatched valencies. After “raced” we have, on the correct analysis, the following:



Note that linking the Ns is possible, but we are interested in the history of the *correct*

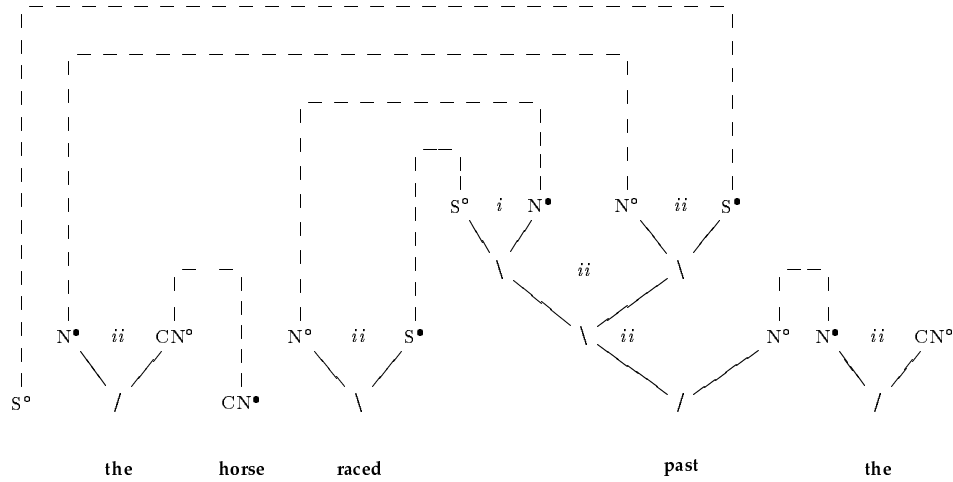
analysis, and in that the verb valencies are matched by the adverb that follows (henceforth we indicate only the principal connective of a mother node):

(21)



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

(22)



The final proof net analysis is given in figure 1.

The semantics associated with a categorial proof net, i.e. 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

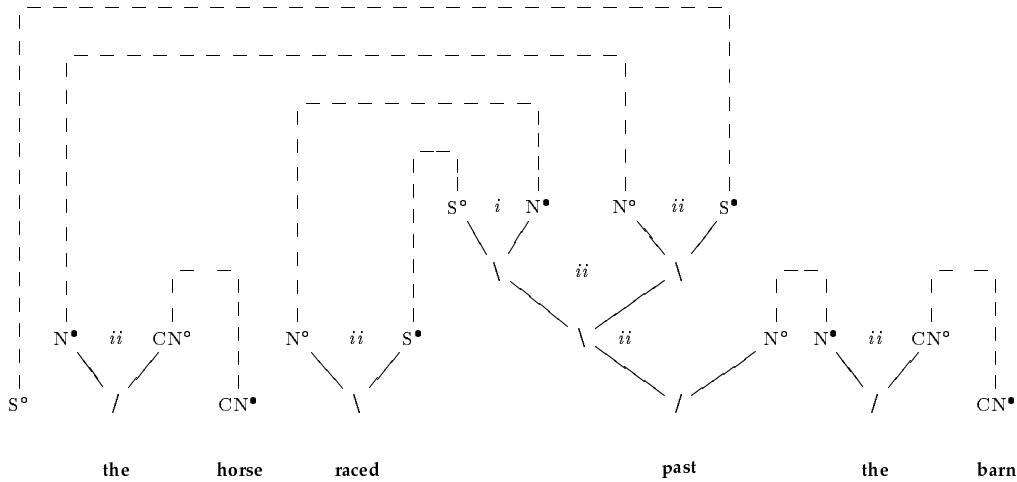


Figure 1
"the horse raced past the barn"

follows, starting by going up at the unique output root (de Groote and Retoré 1996):

- (23)
- 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;
 - travelling down at a root, return the associated lexical semantics.

Thus for our example we obtain (24a), which is logically equivalent to (24b).

- (24) a. $(\lambda x \lambda y \lambda z (\text{past } x (y z)) (\text{the barn}) \lambda 1 (\text{race } 1) (\text{the horse}))$
 b. $(\text{past } (\text{the barn}) (\text{race } (\text{the horse})))$

The analysis of (2b) is less straightforward. Whereas in (2a) “raced” expresses a one-place predication (“go quickly”), in (2b) 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.³

- (25) **fell** – *fall*
 $:= N \backslash S +$
raced – $(\lambda x \lambda y \lambda z [(y z) \wedge \exists w (x z w)], \text{race2})$
 $:= ((CN \backslash CN) / (N \backslash (N \backslash S -))) \cdot (N \backslash (N \backslash S -))$

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, (2b) has the proof net analysis given in figure 2. The semantics extracted is (26a), equivalent to (26b)

- (26) a. $(\text{fall } (\text{the } (\pi_1 (\lambda x \lambda y \lambda z [(y z) \wedge \exists w (x z w)], \text{race2}) \lambda 29 \lambda 30$
 $(\lambda u \lambda v \lambda w (\text{past } u (v w)) (\text{the barn}) \lambda 41 (($
 $(\pi_2 (\lambda p \lambda s \lambda t [(s t) \wedge \exists q (p t q)], \text{race2}) 29 41) 30) \text{horse})))$
 b. $(\text{fall } (\text{the } \lambda 8 [(\text{horse } 8) \wedge \exists 7 (\text{past } (\text{the barn}) (\text{race2 } 8 7))]))$

Let us assign to each proof net analysis a complexity profile which indicates, before and after each word, the number of unmatched literals, i.e. unresolved valencies or dependencies, following 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 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 rightwards (forwards in time) at that point. Thus for (2a) and (2b) analysed in figures 1 and 2 the complexity profiles are as follows:

- (27)
- | | | | | | | | | | |
|---|----|-----|-------|-------|------|-----|------|------|---|
| 6 | | | | | | | | | b |
| 5 | | | | | | | | | |
| 4 | | | | | | | | | |
| 3 | | | | | | | | | |
| 2 | | ab | | | | | | | |
| 1 | ab | | a | | | | | | |
| 0 | | | | | | | | | |
| | 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 (2b) is con-

³ In general grammar requires the expressivity of more powerful categorial logics than just Lambek calculus; 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 sophisticated categorial logic refinements, because the latter concern principally generalisations of word order, whereas the semantic dependencies on which our complexity metric depends remain the same.

sistently higher than that of its ‘garden path’ (2a), just as would be expected on the assumption that in (2b) the less costly but incorrect analysis is assailable.

Let us consider now quantifier scope preference. A rudimentary account of sentence-peripheral quantifier phrase scoping is obtained in Lambek calculus by means of lexical assignments such as the following:

- (28)
- | | | |
|-----------------|----|-------------------------------|
| everyone | – | $\lambda x \forall y (x \ y)$ |
| | := | $St / (N \backslash St)$ |
| everyone | – | $\lambda x \forall y (x \ y)$ |
| | := | $(St / N) \backslash St$ |
| someone | – | $\lambda x \exists y (x \ y)$ |
| | := | $St / (N \backslash St)$ |
| someone | – | $\lambda x \exists y (x \ y)$ |
| | := | $(St / N) \backslash St$ |

Then one analysis of (5a) is that given in figure 3. This is the subject wide scope analysis: its extracted and simplified semantics is as in (29).

- (29)
- | | |
|----|--|
| a. | $(\lambda x \exists y (x \ y) \ \lambda 1 (\lambda x \forall y (x \ y) \ \lambda 2 (love \ 2 \ 1)))$ |
| b. | $\exists x \forall y (love \ y \ x)$ |

A second analysis is that given in figure 4. This is the object wide scope analysis: its extracted and simplified semantics is as in (30).

- (30)
- | | |
|----|--|
| a. | $(\lambda x \forall y (x \ y) \ \lambda 2 (\lambda x \exists y (x \ y) \ \lambda 1 (love \ 2 \ 1)))$ |
| b. | $\forall y \exists x (love \ y \ x)$ |

Let us compare the complexity profiles for the two readings:

- (31)
- | | | | |
|----|---------|-------|--|
| 4 | | b | |
| 3 | | | ab |
| 2 | | a | |
| 1 | ab | | |
| 0 | | | ab |
| a. | someone | loves | everyone |
| b. | | | $\exists \forall$ (subject wide scope, figure 3) |
| | | | $\forall \exists$ (object wide scope, figure 4) |

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

For the passive (5b) let there be assignments as in (32). 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 (25).

- (32)
- | | | |
|--------------|----|--|
| by | – | $\lambda x \lambda y \lambda z [(z = x) \wedge (y \ z)]$ |
| | := | $((N \backslash S -) \backslash (N \backslash S -)) / N$ |
| is | – | $\lambda x \lambda y (x \ \lambda z [z = y] \ y)$ |
| | := | $(N \backslash S +) / (CN \backslash CN)$ |
| loved | – | $(\lambda x \lambda y \lambda z [(y \ z) \wedge \exists w (x \ z \ w)], love)$ |
| | := | $((CN \backslash CN) / (N \backslash (N \backslash S -))) \cdot (N \backslash (N \backslash S -))$ |

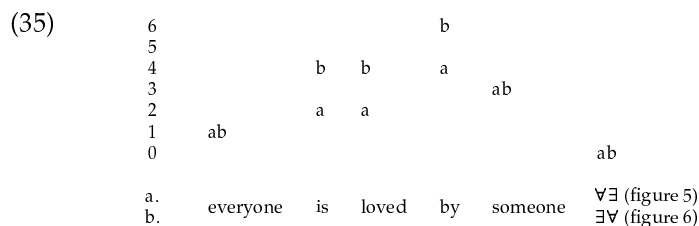
A $\forall \exists$ analysis of (5b) is given in figure 5. This has semantics, after some simplification, (33), which is equivalent to (30).

- (33)
- $$\forall 16 \exists 9 \exists 7 [[9 = 7] \wedge (love \ 16 \ 9)]$$

An $\forall\exists$ analysis of (5b) is given in figure 6. This has semantics, after some simplification, (34), which is equivalent to (29).

$$(34) \quad \exists 16 \forall 14 \exists 7 [[7 = 16] \wedge (love\ 14\ 7)]$$

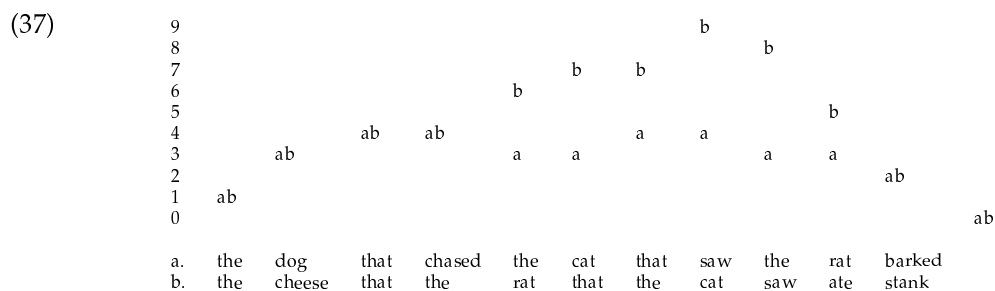
Again, the preferred reading has the lower complexity profile:



Turning now to the final performance phenomenon listed in the introduction, for subject and object relativisation we assume the relative pronoun lexical assignments (36).

$$\begin{array}{ll}
\text{that} & - \lambda x \lambda y \lambda z [(y \ z) \wedge (x \ z)] \\
(36) & := (\text{CN} \setminus \text{CN}) / (\text{N} \setminus \text{S}+) \\
\text{that} & - \lambda x \lambda y \lambda z [(y \ z) \wedge (x \ z)] \\
& := (\text{CN} \setminus \text{CN}) / (\text{S}+ / \text{N})
\end{array}$$

Sentence (8b) is analysed in figure 7. Sentence (9b) is analysed in figure 8. Let us compare the complexities:



Again, the profile of (9b) is higher; indeed it rises above 7–8, thus reaching what are usually taken to be the limits of short term memory.

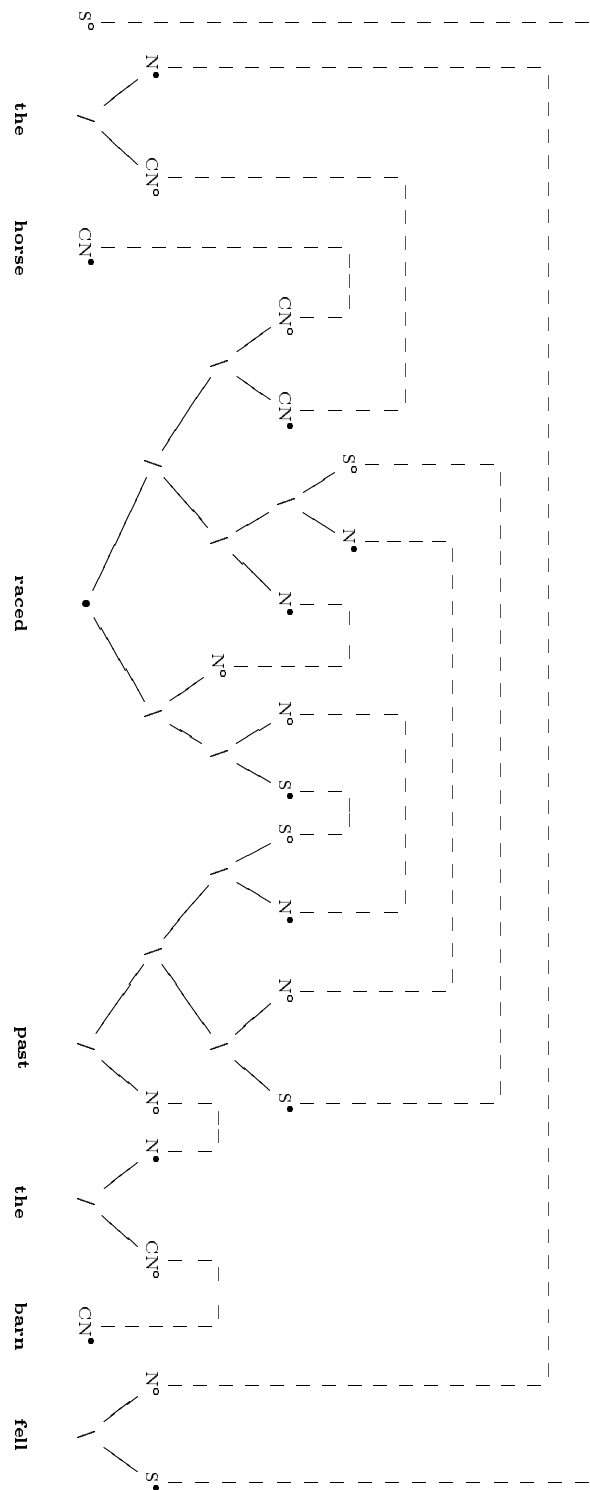


Figure 2
"the horse raced past the barn fell"

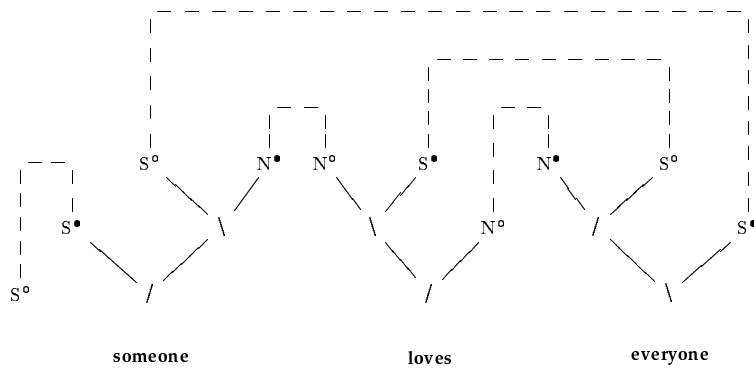


Figure 3
"someone loves everyone" ($\exists\forall$)

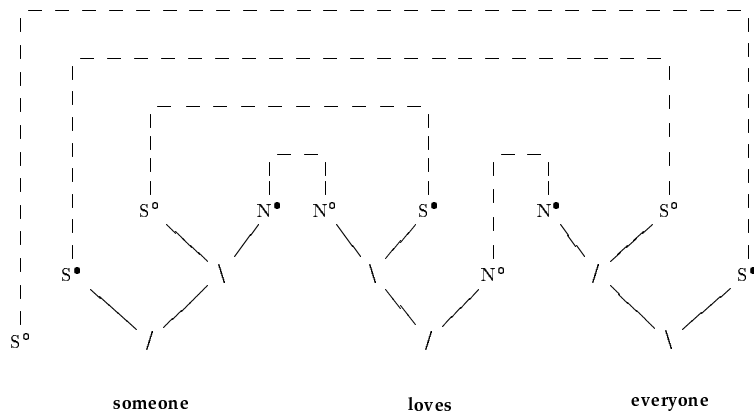


Figure 4
"someone loves everyone" ($\forall\exists$)

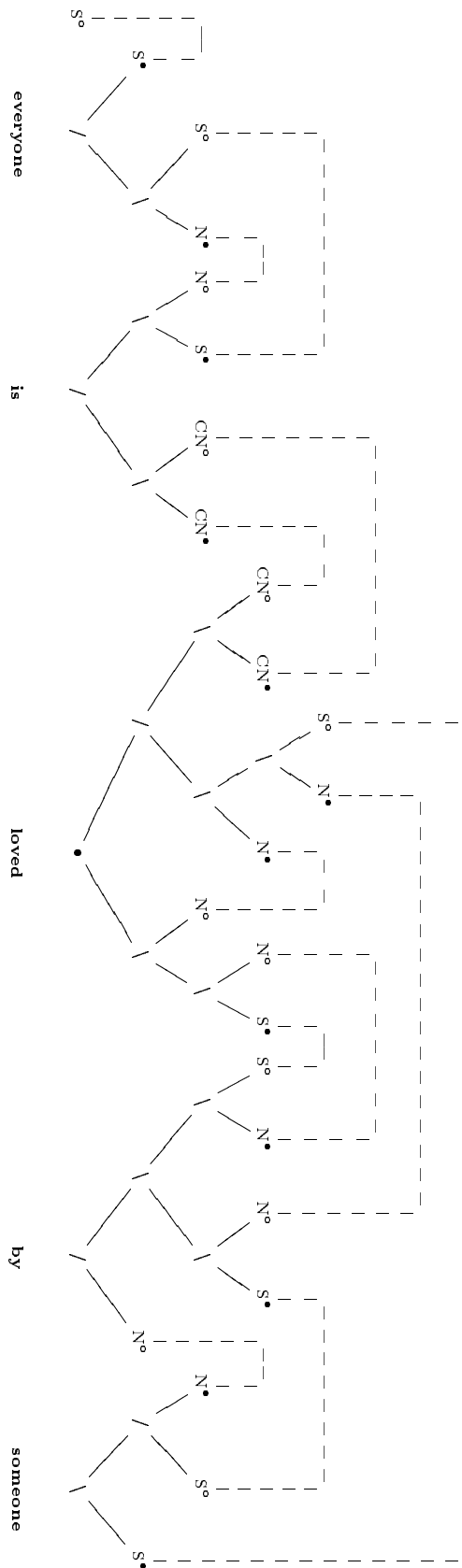


Figure 5
 "everyone is loved by someone" ($\forall\exists$)

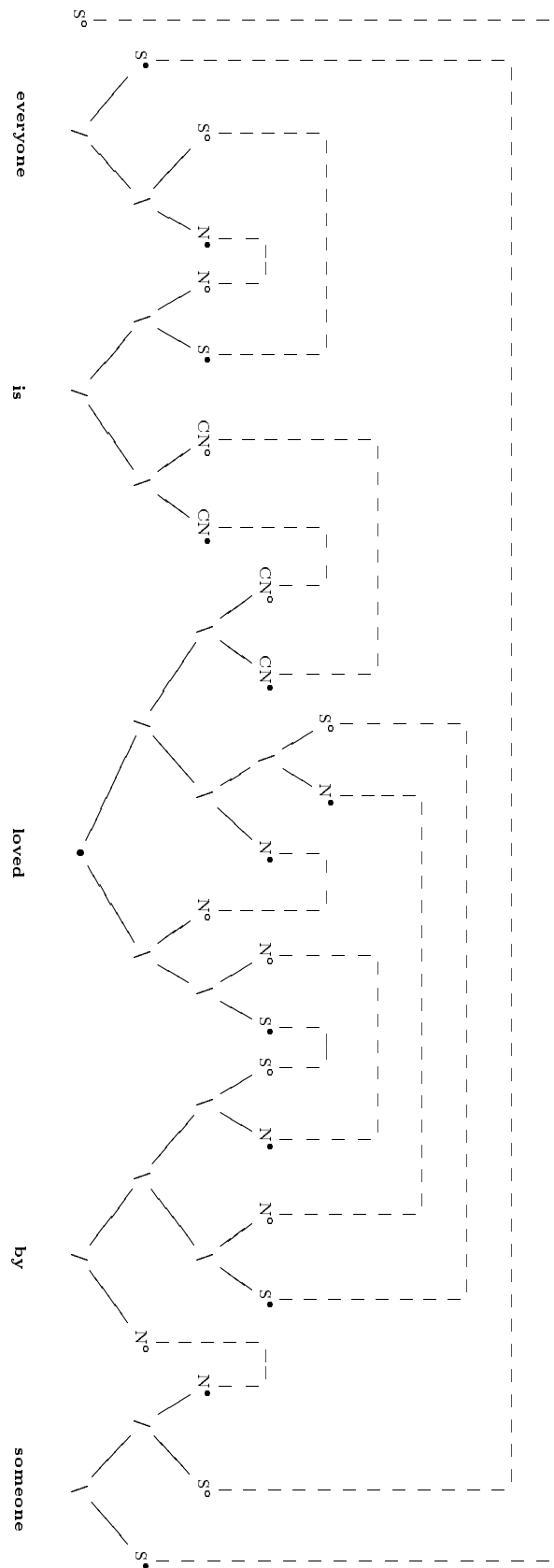


Figure 6
 "everyone is loved by someone" (AE)

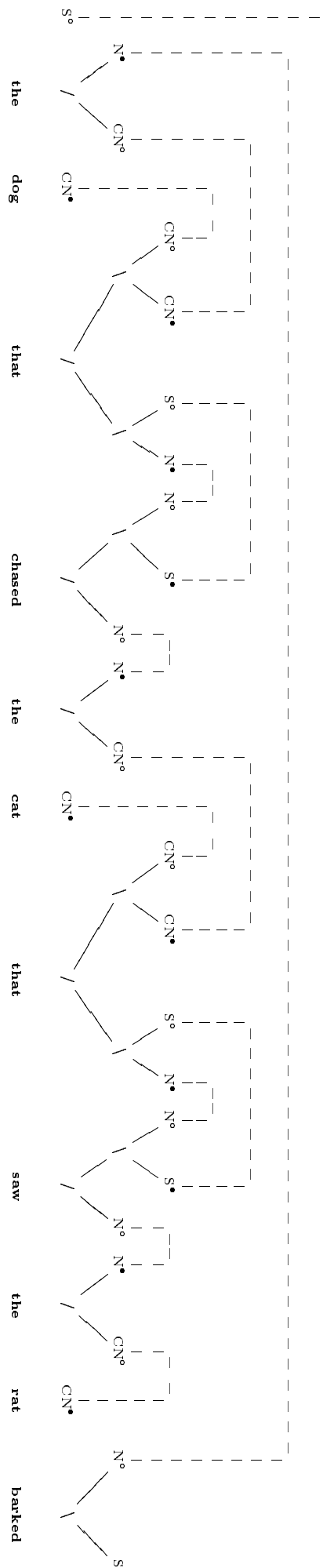


Figure 7

"the dog that chased the cat that saw the rat barked"

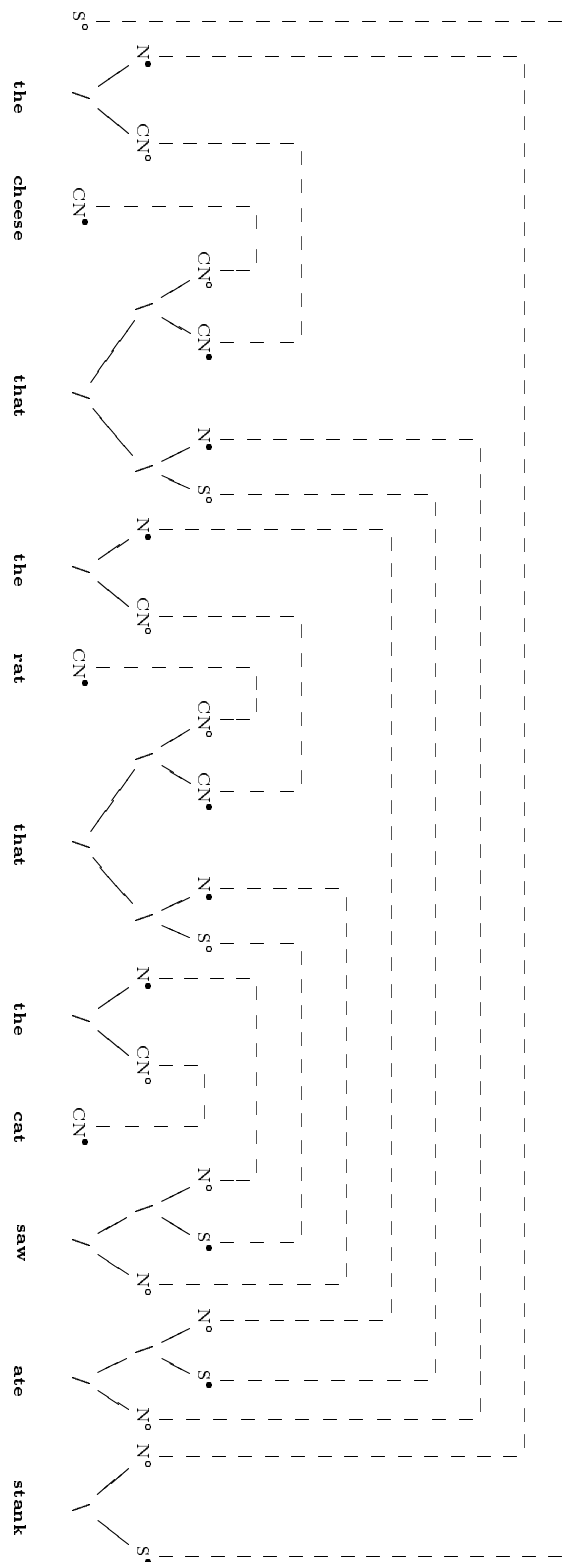


Figure 8
 "the cheese that the rat that the cat saw ate stank"

3 Further cases

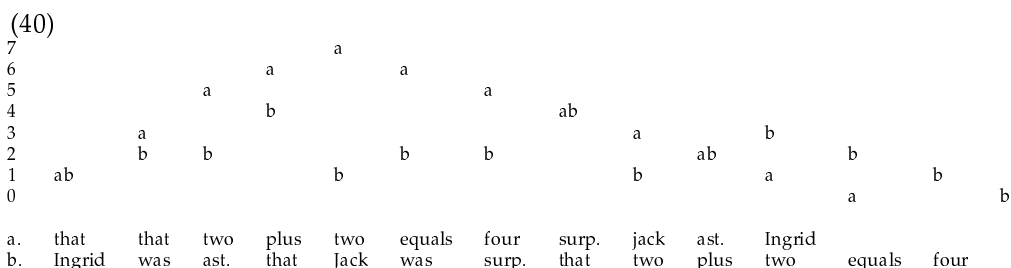
Another dramatic example of unacceptability is provided by the following:

- (38)
- 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 parafrases, however, seem more or less equally acceptable:

- (39)
- 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.

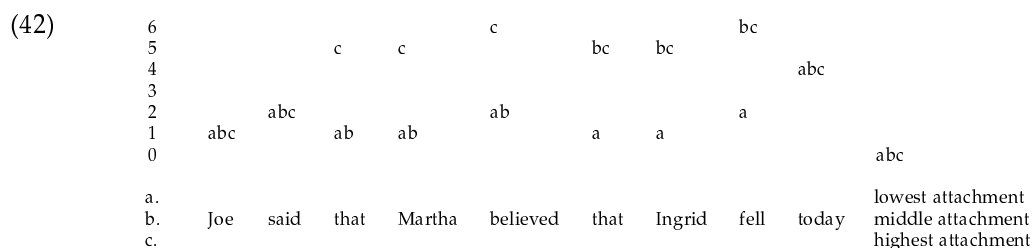
In figure 9 we give the analysis of (38b) and in figure 10 that of (39b). We now abbreviate proof nets by flattening formula trees into their linear representations; since this conceals the order switching of output links the notation belies the underlying planarity. It is very interesting to observe that the complexity profile of the latter is in general lower even though the analysis has more than twice the total number of links.



By way of another example, Kimball (1973, p.27) observes that in a sentence such as (41), three ways ambiguous according to the attachment of the adverb, the lower the attachment is, the higher the preference (what he terms 'Right Association').

- (41) Joe said that Martha believed that Ingrid fell today.

In figure 11 we give the analyses for the highest, the middle, and the lowest attachments. Accordingly, the complexity profiles are:



The same effect occurs strongly in (43), where the preferred reading is the one given by the lowest attachment, even though that one is the nonsensical reading.

- (43) the book that shocked Mary's title

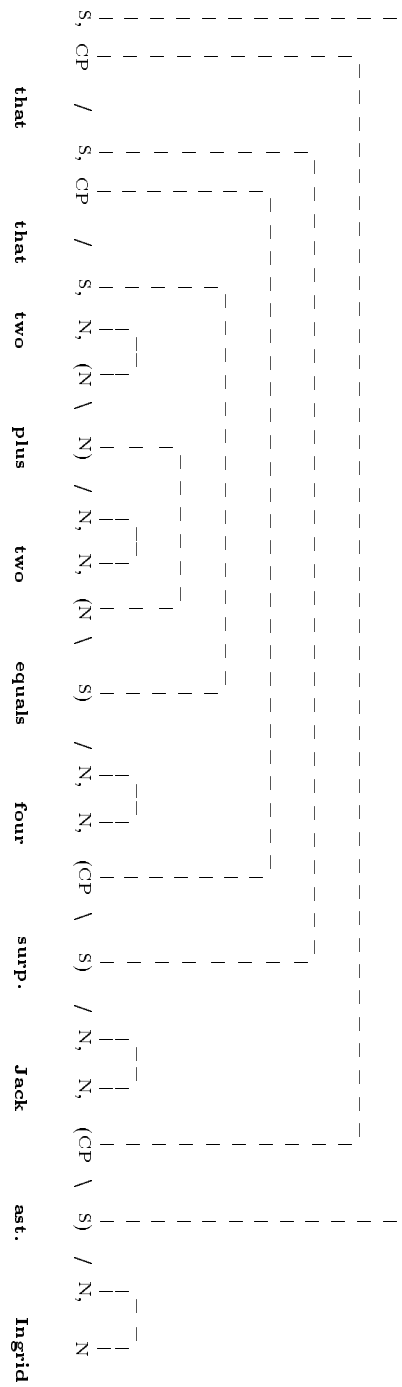


Figure 9
 "that that two plus two equals four surprised Jack astonished Ingrid"

The analyses are given in figure 12. The complexities are thus:

(44)	4			a	a					
	3		a			a				
	2			b	b		ab			
	1	ab	b			b		ab		
	0								ab	
	a.	the	book	that	shocked	Mary	's	title	sensical	
	b.								nonsensical	

Finally, 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:

- (45) a. ?John gave the painting that Mary hated to Bill.
 b. John gave Bill the painting that Mary hated.

The analyses are given in figure 13. The complexities are thus:

(46)	4						a			
	3					a		b		
	2		ab	ab	a		b			
	1	ab		b	b			a	a	
	0								b	a
	a.	John	gave	the	painting	that	Mary	hated	to	Bill
	b.	John	gave	Bill	the	painting	that	Mary	hated	

References

- Ades, Anthony E. and Mark J. Steedman: 1982, 'On the Order of Words', *Linguistics and Philosophy* **4**, 517–558.
- Bever, T.: 1970, 'The cognitive basis for linguistic structures', in J. R. Hayes (ed.) *Cognition and the Growth of Language*, Wiley, New York.
- Buszkowski, Wojciech: 1986, 'Completeness results for Lambek syntactic calculus', *Zeitschrift für mathematische Logik und Grundlage der Mathematik* **32**, 13–28.
- Carpenter, Bob: 1998, *Type-Logical Semantics*, MIT Press, Cambridge, Massachusetts.
- Chomsky, Noam: 1965, *Aspects of the Theory of Syntax*, MIT Press, Cambridge, Massachusetts.
- Danos, V. and L. Regnier: 1989, 'The structure of multiplicatives', *Archive for Mathematical Logic* **28**, 181–203.
- Došen, K.: 1992, 'A brief survey of frames for the Lambek calculus', *Zeitschrift für mathematische Logik und Grundlage der Mathematik* **38**, 179–187.
- de Groote, Philippe and Christian Retoré: 1996, 'On the Semantic Readings of Proof-Nets', in G.-J. Kruijff, G. Morrill and R. T. Oehrle (eds.) *Proceedings of formal Grammar 1996*, Prague, 57–70.
- Girard, Jean-Yves: 1987, 'Linear Logic', *Theoretical Computer Science* **50**, 1–102.
- Hendriks, Herman: 1993, *Studied Flexibility: Categories and Types in Syntax and Semantics*, Ph.D. dissertation, Universiteit van Amsterdam.
- Hepple, Mark: 1990, 'Normal form theorem proving for the Lambek calculus', in H. Karlgren (ed.), *Proceedings of COLING 1990*, Stockholm.

- Hepple, Mark and Glyn Morrill: 1989, 'Parsing and Derivational Equivalence', in *Proceedings of the Fourth Conference of the European Chapter of the Association for Computational Linguistics*, Manchester.
- Kimball, John: 1973, 'Seven principles of surface structure parsing in natural language', *Cognition* **2**, 15–47.
- König, E.: 1989, 'Parsing as natural deduction', in *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, Vancouver.
- Lambek, J.: 1958, 'The mathematics of sentence structure', *American Mathematical Monthly* **65**, 154–170, also in Buszkowski, W., W. Marciszewski, and J. van Benthem (eds.): 1988, *Categorical Grammar*, Linguistic & Literary Studies in Eastern Europe Volume 25, John Benjamins, Amsterdam, 153–172.
- Lecomte, Alain and Christian Retoré: 1995, 'An alternative categorial grammar', in G. Morrill and R. T. Oehrlé (eds.) *Proceedings of formal Grammar 1995*, Barcelona.
- Moortgat, Michael: 1988, *Categorical Investigations: Logical and Linguistic Aspects of the Lambek Calculus*, Foris, Dordrecht.
- Moortgat, Michael: 1997, 'Categorial Type Logics', in J. van Benthem and A. ter Meulen (eds.) *Handbook of Logic and Language*, Elsevier, Amsterdam, 93–177.
- Morrill, Glyn: 1994, *Type Logical Grammar: Categorial Logic of Signs*, Kluwer Academic Publishers, Dordrecht.
- Pentus, M.: 1994, 'Language completeness of the Lambek calculus', *Proceedings of the Eighth Annual IEEE Symposium on Logic in Computer Science*, Paris.
- Roorda, Dirk: 1991, *Resource Logics: proof-theoretical investigations*, Ph.D. dissertation, Universiteit van Amsterdam.
- Steedman, Mark J.: 1997, *Surface Structure and Interpretation*, MIT Press, Cambridge, Massachusetts.
- Zielonka, W.: 1981, 'Axiomatizability of Ajdukiewicz-Lambek calculus by means of cancellation schemes', *Zeitschrift für mathematische Logik und Grundlage der Mathematik* **27**, 215–224.

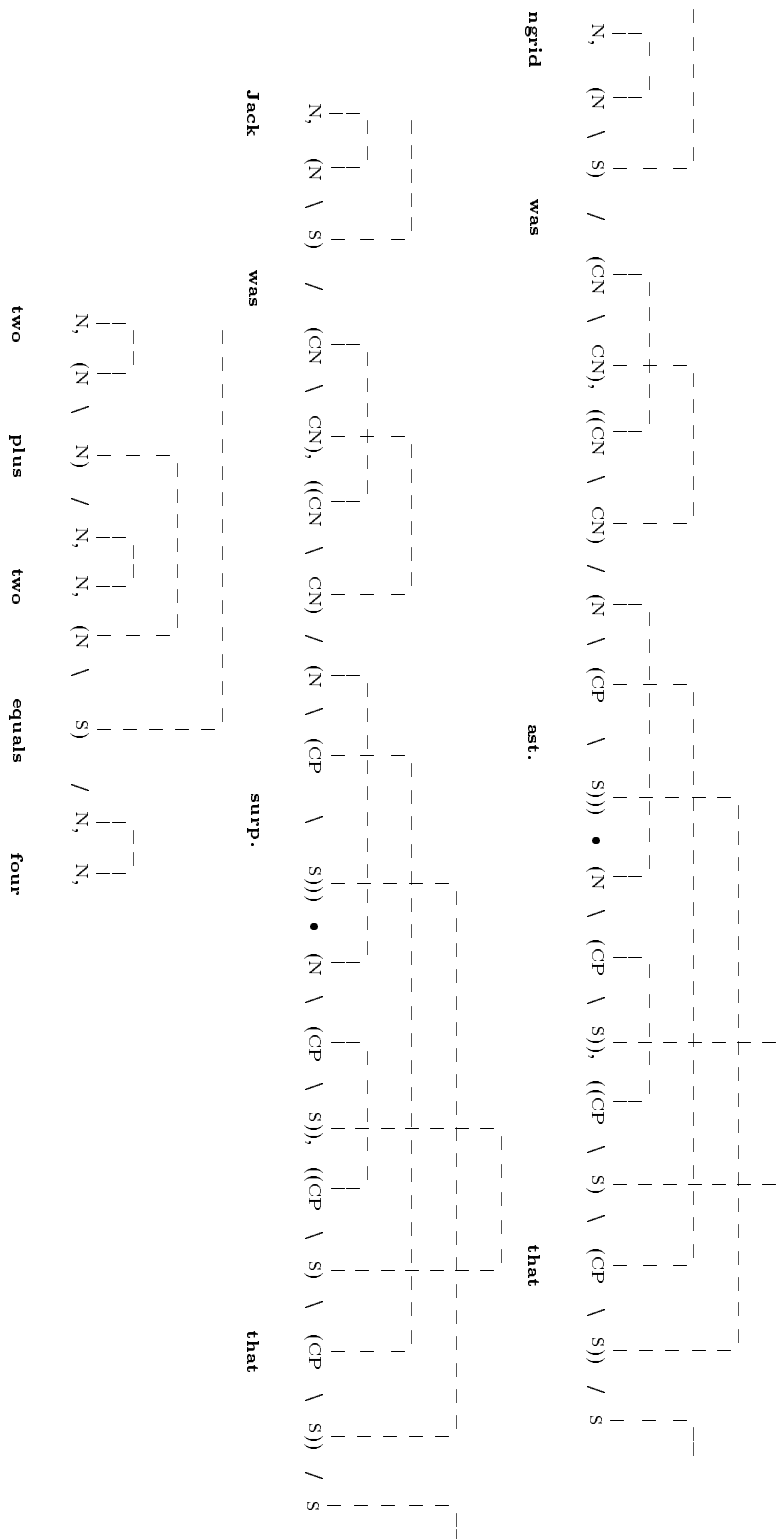


Figure 10

"Ingrid was astonished that Jack was surprised that two plus two equals four"

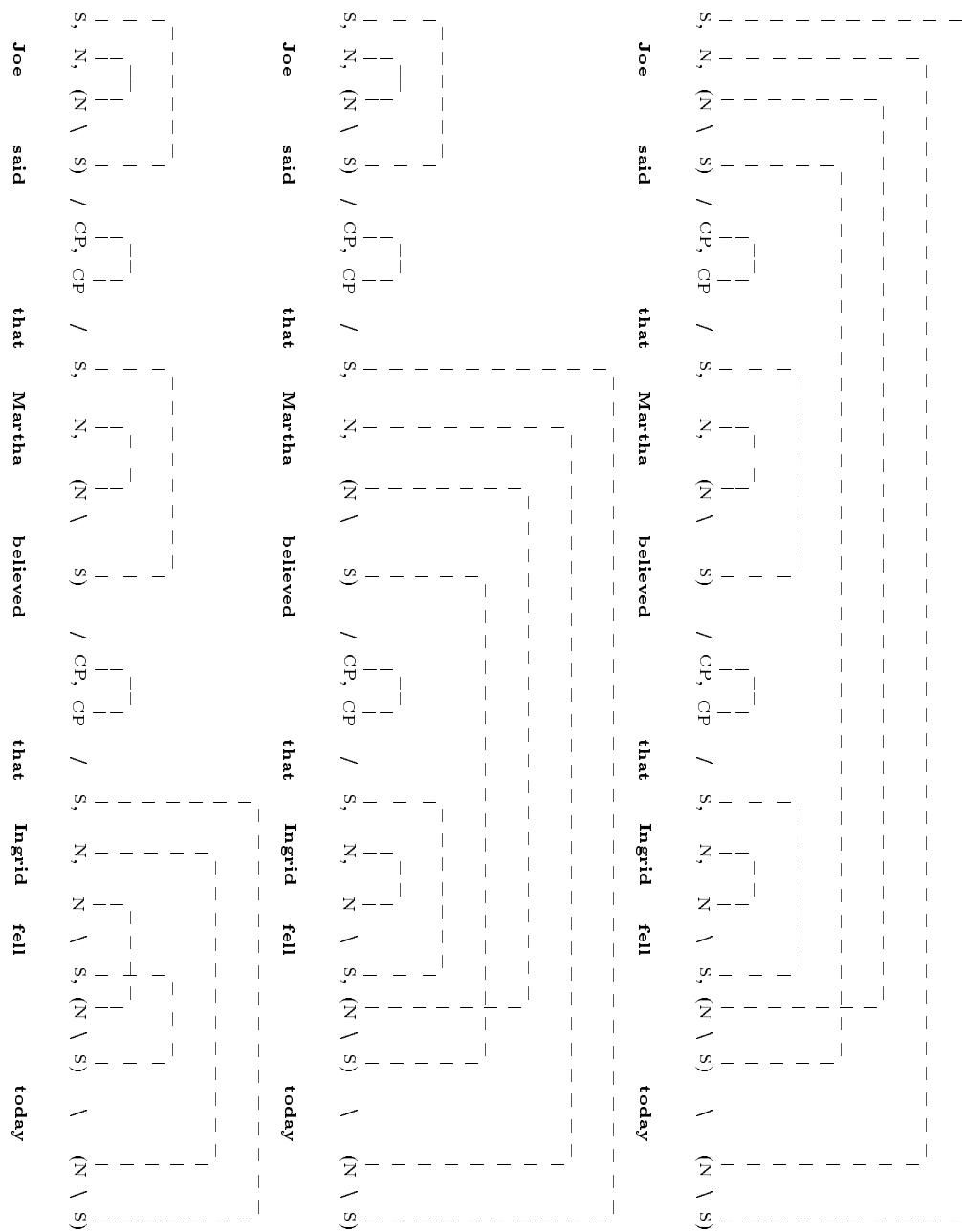


Figure 11
 "Joe said that Martha believed that Ingrid fell today"

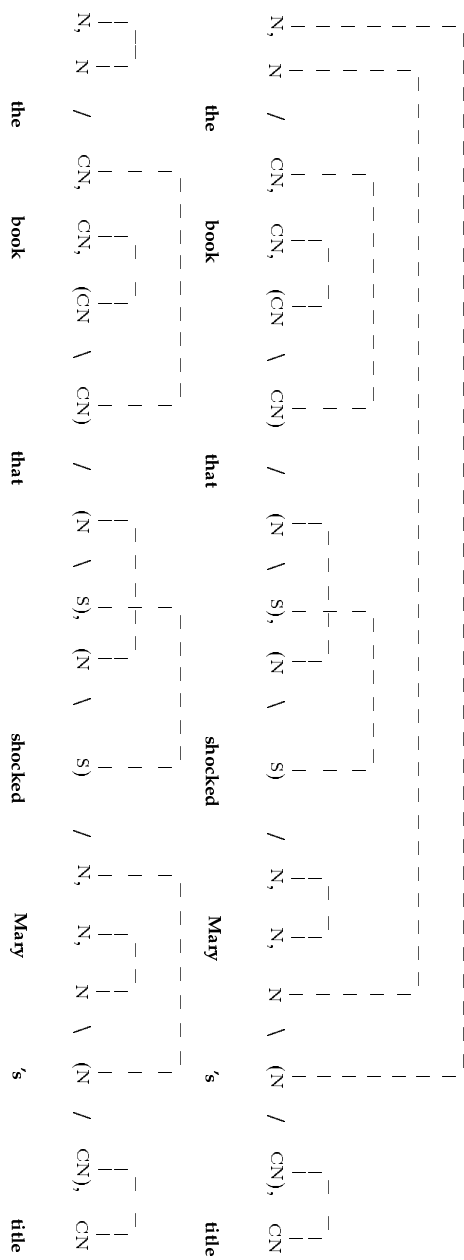


Figure 12
 "the book that shocked Mary's title" (sensical and nonsensical)

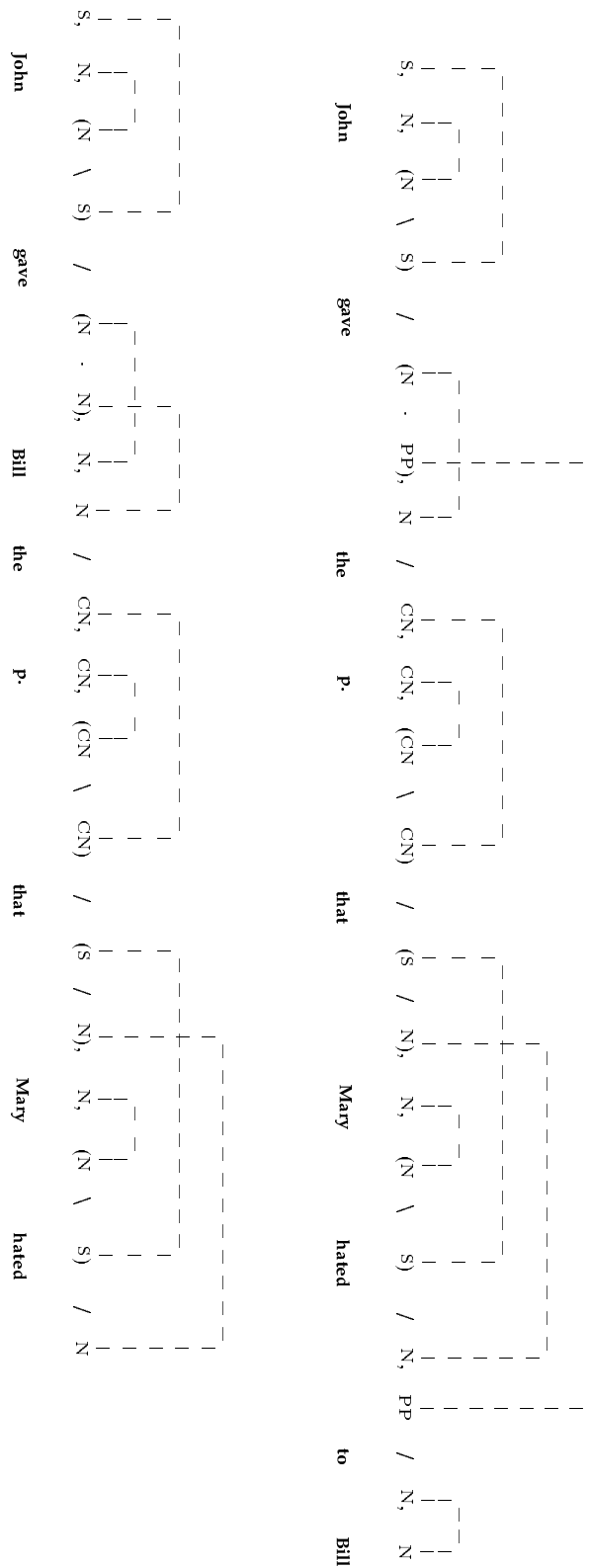


Figure 13
 "John gave the painting that Mary hated to Bill" vs. "John gave Bill the painting that Mary hated"

1

Geometry of language and linguistic circuitry

GLYN MORRILL

Dear Jim,

Suppose we sought common ground between what Frege, Ajdukiewicz, Chomsky and Montague have said about syntax and semantics, and a corresponding mathematics of language, and picture of linguistic flow.

Then it might be something like this, and at its heart is the Lambek calculus.

Best always,

“for every dimension presupposes a medium within which it can act, and if, in the spiral unwinding of things, space warps into something akin to time, and time, in its turn, warps into something akin to thought, then, surely, another dimension follows – a special Space maybe, not the old one, we trust, unless spirals become vicious circles again.”

(Nabokov, 1969, p.231)

We illustrate the potential for geometry of language and linguistic circuitry under the rendering of the syntactic structures of Lambek categorical grammar as proof nets.¹ This empirical application of proof nets

¹Thanks to the audiences at EACL99, Bergen, the workshop on Linear Logic and Applications, Dagstuhl, August 1999, and the workshop on Grammar and Logic, Rennes, April 2000, to Mario Fadda and Oriol Valentí for many related conver-

sees sentences as proof nets and words as partial proof nets, and well-formedness/meaningfulness as a global harmony of categorial syntactic connection. The global cohesion coincides with a dynamic connectivity reminiscent of circuits, but whereas circuits are just generalisations of formulas, our syntactic structures are much more sublime objects: proofs.

The proof nets introduced by Girard (1987) for linear logic not only appear to be optimally parsimonious, but reduce Cut-elimination to *local* graph transformations. This paper continues the line of Roorda (1991) and others propounding the use of proof nets for Lambek categorial grammar, in what might be put under the slogan ‘syntactic structures as proof nets’.

On this view, words are *partial* proof nets, which are called *modules*, and Lecomte and Retoré (1995) advocate ‘words as modules’. It is shown in de Groote and Retoré (1996) how semantics as well as syntax can be represented in the same formalism of proof nets. Morrill (1999) observes that within this general arrangement, much Cut-elimination between syntax and semantics can be preevaluated in a partial execution of the lexicon.

The present paper attempts to spell out and illustrate these possibilities. Section 1 defines Basic Type Logical Grammar and section 2 defines nets for Basic TLG and illustrates the lexical preevaluation. Section 3 illustrates lexical words as modules and section 4, the syntactic structures of sentences as proof nets.

1.1 Basic TLG

This section sets out categorial grammar based on the Lambek calculus (Lambek (1958), Lambek (1988)): Basic Type Logical Grammar (Morrill (1994), Carpenter (1997), Moortgat (1997)).

Let there be a set C of *prosodic constants*. Then the set A of *prosodic forms* is defined by:

$$(1) \quad A ::= C \mid A+A$$

A *prosodic structure* is a monoid $(L, +, 0)$, i.e. an algebra of arity $(2, 0)$ such that:

$$(2) \quad \begin{array}{lcl} s_1+(s_2+s_3) & = & (s_1+s_2)+s_3 \quad + \text{ is associative} \\ 0+s & = & s = s+0 \quad 0 \text{ is neutral for } + \end{array}$$

A *prosodic interpretation* comprises a prosodic structure $(L, +, 0)$ and

sations, and to an anonymous reviewer. All errors are my own. Work supported by CICYT projects PB98-0937-C03-03 and TIC2002-04019-C03-01. E-mail: morrill@lsi.upc.es, <http://www.lsi.upc.es/~morrill/>.

GEOMETRY OF LANGUAGE AND LINGUISTIC CIRCUITRY / 3

a valuation v mapping from C into L . Then the prosodic object $[\alpha]$ denoted by a prosodic form α with respect to a prosodic interpretation with valuation v is defined by:

$$(3) \quad \begin{aligned} [\mathbf{a}] &= v(\mathbf{a}) \text{ for prosodic constant } \mathbf{a} \\ [\alpha + \beta] &= [\alpha] + [\beta] \end{aligned}$$

Prosodic forms α and β are *equivalent*, $\alpha \equiv \beta$, if and only if $[\alpha] = [\beta]$ in every prosodic interpretation. Clearly:

$$(4) \quad \alpha + (\beta + \gamma) \equiv (\alpha + \beta) + \gamma$$

So we can drop parentheses in prosodic forms.

The set $\mathcal{T}$ of semantic types is defined by:

$$(5) \quad \mathcal{T} ::= e \mid t \mid \mathcal{T} \rightarrow \mathcal{T} \mid \mathcal{T} \& \mathcal{T}$$

Let there be a set V_τ of *semantic variables* and a set C_τ of *semantic constants* for each semantic type τ including the *logical semantic constants*:

$$(6) \quad \begin{aligned} \blacksquare, \square &\in C_t \\ \neg &\in C_{t \rightarrow t} \\ \wedge, \vee, \rightarrow &\in C_{t \rightarrow (t \rightarrow t)} \\ = &\in C_{e \rightarrow (e \rightarrow t)} \\ \forall, \exists &\in C_{(e \rightarrow t) \rightarrow t} \end{aligned}$$

Then the set Φ_τ of *semantic terms of type τ* for each semantic type τ is defined by:

$$(7) \quad \begin{aligned} \Phi_\tau &::= C_\tau \\ \Phi_\tau &::= V_\tau \\ \Phi_\tau &::= (\Phi_{\tau' \rightarrow \tau} \Phi_{\tau'}) \\ \Phi_\tau &::= \pi_1 \Phi_{\tau \& \tau'} \mid \pi_2 \Phi_{\tau' \& \tau} \\ \Phi_{\tau \rightarrow \tau'} &::= \lambda V_\tau \Phi_{\tau'} \\ \Phi_{\tau \& \tau'} &::= (\Phi_\tau, \Phi_{\tau'}) \end{aligned}$$

We may abbreviate $((\phi \psi) \chi)$ as $(\phi \psi \chi)$. An occurrence of a semantic variable x in a semantic term is *free* if and only if it does not fall within any subterm of the form $\lambda x \phi$; otherwise it is *bound*. A *semantic form* is a semantic term containing no free variables.

The *application* to a semantic term ϕ of the *substitution* of the semantic variable x (of semantic type τ) by the semantic term ψ (of semantic type τ), $\phi\{\psi/x\}$, is the result of replacing by ψ every free occurrence of x in ϕ ; the application of a substitution to a semantic term is *free* if and only no semantic variable becomes bound in the process of replacement.

A *semantic structure* is a $\mathcal{T}$ -indexed family of sets $\{D_\tau\}_{\tau \in \mathcal{T}}$ where:

4 / GLYN MORRILL

- (8) D_e is a non-empty set E of *entities*
 D_t is $\mathcal{P}(\{\emptyset\}) = \{\{\}, \{\{\}\}\}$
 $D_{\tau \& \tau'}$ = $D_\tau \times D_{\tau'}$
 $D_{\tau \rightarrow \tau'}$ = $D_{\tau'}^{D_\tau}$

A *semantic interpretation* comprises a semantic structure, an assignment g mapping from each V_τ into D_τ , and a valuation f mapping from each C_τ into D_τ such that:

- (9) $f(\blacksquare)$ = $\{\{\}\}$
 $f(\square)$ = $\{\}$
 $f(\neg)$ = $m \mapsto \overline{m}^{\{\emptyset\}}$
 $f(\wedge)$ = $m \mapsto (m' \mapsto m \cap m')$
 $f(\vee)$ = $m \mapsto (m' \mapsto m \cup m')$
 $f(\rightarrow)$ = $m \mapsto (m' \mapsto \overline{m}^{\{\emptyset\}} \cup m')$
 $f(=)$ = $m \mapsto (m' \mapsto (\{\{\}\}) \text{ if } m = m' \text{ else } \{\})$
 $f(\forall)$ = $m \mapsto \bigcap_{m' \in E} m(m')$
 $f(\exists)$ = $m \mapsto \bigcup_{m' \in E} m(m')$

The semantic object $[\phi]^g$ denoted by a semantic term ϕ with respect to a semantic interpretation with valuation f and assignment g is defined by:

- (10) $[c]^g$ = $f(c)$ for semantic constant c
 $[x]^g$ = $g(x)$ for semantic variable x
 $[(\phi \ \psi)]^g$ = $[\phi]^g([\psi]^g)$
 $[\pi_1 \phi]^g$ = $\mathbf{fst}([\phi]^g)$
 $[\pi_2 \phi]^g$ = $\mathbf{snd}([\phi]^g)$
 $[\lambda x \phi]^g$ = $D_\tau \ni m \mapsto [\phi]^{(g - \{(x, g(x))\}) \cup \{(x, m)\}}$ for $x \in V_\tau$
 $[(\phi, \psi)]^g$ = $\langle [\phi]^g, [\psi]^g \rangle$

Semantic terms ϕ and ψ are *equivalent*, $\phi \equiv \psi$, if and only if $[\alpha]^g = [\beta]^g$ in every semantic interpretation. We have:

- (11) a. $\lambda x \phi \equiv \lambda y (\phi \{y/x\})$ α -conversion
provided y is not free in ϕ and $\phi \{y/x\}$ is free

b. $(\lambda x \phi \ \psi) \equiv \phi \{\psi/x\}$ β -conversion
provided $\phi \{\psi/x\}$ is free
 $\pi_1(\phi, \psi) \equiv \phi$ $\pi_2(\phi, \psi) \equiv \psi$

c. $\lambda x (\phi \ x) \equiv \phi$ η -conversion
provided x is not free in ϕ
 $(\pi_1 \phi, \pi_2 \phi) \equiv \phi$

We also have semantic equivalences arising in virtue of the logical se-

GEOMETRY OF LANGUAGE AND LINGUISTIC CIRCUITRY / 5

mantic constants, for example:

$$\begin{aligned}
 (12) \quad & ((= \phi) \phi) \equiv \blacksquare \\
 & ((\wedge \phi) \blacksquare) \equiv \phi \\
 & (\exists \lambda x ((\wedge \phi) ((= x) \psi))) \equiv \phi\{\psi/x\} \text{ provided } \phi\{\psi/x\} \text{ is free}
 \end{aligned}$$

Let there be a set $\mathcal{A}$ of *atomic syntactic types*. Then the set $\mathcal{F}$ of *syntactic types* is defined by:

$$(13) \quad \mathcal{F} ::= \mathcal{A} \mid \mathcal{F} \cdot \mathcal{F} \mid \mathcal{F} \backslash \mathcal{F} \mid \mathcal{F} / \mathcal{F}$$

Let there be a *basic type map* t mapping from $\mathcal{A}$ into $\mathcal{T}$. This induces the *type map* T from $\mathcal{F}$ into $\mathcal{T}$ such that:

$$\begin{aligned}
 (14) \quad & T(P) = t(P) \text{ for atomic syntactic type } P \\
 & T(A \cdot B) = T(A) \& T(B) \\
 & T(A \backslash C) = T(A) \rightarrow T(C) \\
 & T(C / B) = T(B) \rightarrow T(C)
 \end{aligned}$$

A *syntactic interpretation* comprises a prosodic structure $(L, +, 0)$, a semantic structure $\{D\}_{\tau \in \mathcal{T}}$, and a valuation F sending each $P \in \mathcal{A}$ into a subset of $L \times D_{t(P)}$. Then the value $\llbracket A \rrbracket \subseteq L \times D_{T(A)}$ of a syntactic type with respect to a syntactic interpretation with valuation F is defined by:

$$\begin{aligned}
 (15) \quad & \llbracket P \rrbracket = F(P) \text{ for atomic syntactic type } P \\
 & \llbracket A \cdot B \rrbracket = \{ \langle s_1 + s_2, \langle m_1, m_2 \rangle \rangle \mid \exists \langle s_1, m_1 \rangle \in \llbracket A \rrbracket, \langle s_2, m_2 \rangle \in \llbracket B \rrbracket \} \\
 & \llbracket A \backslash C \rrbracket = \{ \langle s_2, m_2 \rangle \mid \forall \langle s_1, m_1 \rangle \in \llbracket A \rrbracket, \langle s_1 + s_2, m_2(m_1) \rangle \in \llbracket C \rrbracket \} \\
 & \llbracket C / B \rrbracket = \{ \langle s_1, m_1 \rangle \mid \forall \langle s_2, m_2 \rangle \in \llbracket B \rrbracket, \langle s_1 + s_2, m_1(m_2) \rangle \in \llbracket C \rrbracket \}
 \end{aligned}$$

A *type assignment statement* $\alpha - \phi : A$ comprises a syntactic type A , a prosodic form α , and a semantic form ϕ of semantic type $T(A)$. A prosodic, semantic and syntactic interpretation is a *model of a type assignment statement* $\alpha - \phi : A$ if and only if $\langle [\alpha], [\phi]^g \rangle \in \llbracket A \rrbracket$; it is a *model of a set Σ of type assignment statements* if and only if it is a model of every type assignment statement $\sigma \in \Sigma$.

A set Σ of type assignment statements *entails* a type assignment statement σ , $\Sigma \models \sigma$, if and only if every model of Σ is also a model of σ . A *lexicon* is a set of type assignment statements. The *language* $\mathcal{L}(Lex)$ defined by a lexicon Lex is the set of type assignment statements that it entails:

$$(16) \quad \mathcal{L}(Lex) = \{ \alpha - \phi : A \mid Lex \models \alpha - \phi : A \}$$

For example, let there be the following lexicon:

6 / GLYN MORRILL

(17)	bag+end	—	b
		:=	N
	frodo	—	f
		:=	N
	in	—	in
		:=	$(S \backslash S) / N$
	inhabits	—	$\lambda x \lambda y ((in\ x)\ (live\ y))$
		:=	$(N \backslash S) / N$
	lives	—	$live$
		:=	$N \backslash S$

Then the language defined includes the following type assignment statements:

- (18) a. **frodo+lives+in+bag+end**— $((in\ b)\ (live\ f))$: S
 b. **frodo+inhabits+bag+end**— $((in\ b)\ (live\ f))$: S

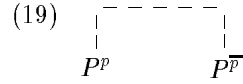
1.2 Nets for BTLG

The following subsections define respectively prosodic, semantic and syntactic proof nets for Basic Type Logical Grammar.

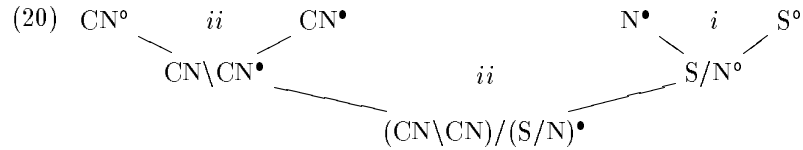
1.2.1 Prosodic nets

A *label* is a syntactic type together with a polarity input ($\bullet$) or output ($^\circ$). Where p is a polarity, $\bar{p}$ is the opposite polarity. Labels A^p and $A^{\bar{p}}$ are *complementary*. A *literal* is a label the type of which is atomic.

A *link* has a list of *premise* labels (above) and a list of *conclusion* labels (below) with some edges between them. An *identity link* is of the form:



A *logical link* is one of the local trees given in Figure 1. A *prosodic tree* is a tree the leaves of which are literals and each local tree of which is a logical link. Each label is the root of a unique prosodic tree, which is the result of unfolding the label upwards according to the logical links. For example, the prosodic tree for $(CN \backslash CN) / (S/N)^\bullet$ is:



A *prosodic frame* is a cyclic list of prosodic trees exactly one of which has an output root. For example, the following (read as a cyclic list) is

GEOMETRY OF LANGUAGE AND LINGUISTIC CIRCUITRY / 7

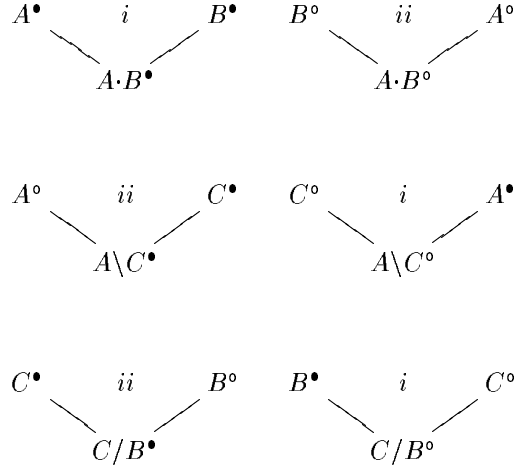
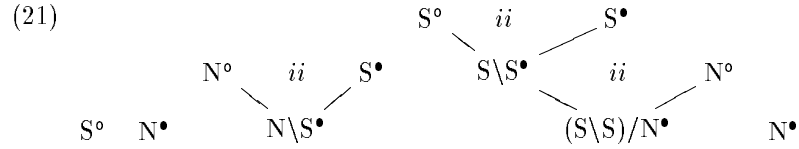


FIGURE 1 Logical links

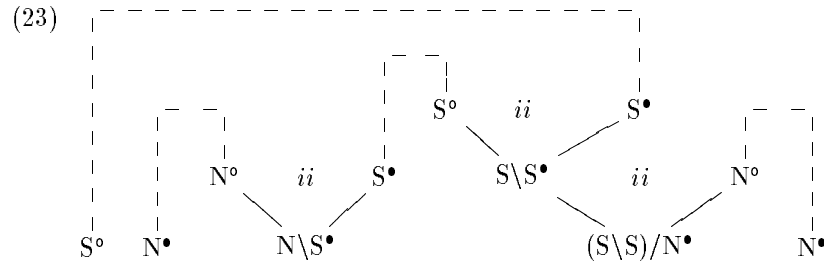
a prosodic frame:



A *prosodic net* is the result of connecting by an identity link every leaf in a prosodic frame with a unique complementary leaf such that:

- (22) *acyclicity* Every cycle crosses both edges of some i -link.
planarity The identity links are planar in the cyclic ordering.

For example, the following is a prosodic net:



Then we have the following:

Proposition 1 (Correctness of prosodic nets for BTLG). *There is a prosodic net with roots $A_0^\circ, A_1^\bullet, \dots, A_n^\bullet$ if and only if for all*

8 / GLYN MORRILL

$$\alpha_1, \dots, \alpha_n, \{\alpha_1 : A_1, \dots, \alpha_n : A_n\} \models \alpha_1 + \dots + \alpha_n : A_0.$$

See Buszkowski (1986) for completeness of Lambek calculus and Danos and Regnier (1989) and Roorda (1991) for correctness of proof nets.²

1.2.2 Semantic nets

A *contraction link* is of the form:

$$(24) \quad \begin{array}{ccc} A^\bullet & i & A^\bullet \\ & \searrow \quad \swarrow & \\ & A^\bullet & \end{array}$$

A *Cut link* is of the form:

$$(25) \quad \begin{array}{ccc} A^p & & A^{\overline{p}} \\ | & & | \\ | & & | \end{array}$$

A *semantic tree* is a tree the leaves of which are literals and each local tree of which is a logical link or a contraction link. Note that prosodic trees are semantic trees — without contraction links. A *semantic frame* is a bag of semantic trees. Note that prosodic frames are semantic frames — forgetting about the cyclic order. A *semantic module* is the result of i) possibly connecting, in a semantic frame, some leaves with unique complementary leaves by an identity link, and some roots with unique complementary roots by a Cut link, and ii) associating semantic constants to every open input root, such that:

(26) *acyclicity* Every cycle crosses both edges of some i or j -link.

A *semantic net* is a semantic module with no open leaves and exactly one output root. For example, the following is a semantic net:

(27)

The diagram (27) is a complex graph structure enclosed in a dashed rectangular frame. It features several nodes and edges:

- Nodes:**
 - live:** A solid black dot in the lower-left quadrant.
 - ii:** Two labels for solid black dots. One is in the upper-left quadrant, and the other is in the center.
 - in:** A solid black dot in the lower-right quadrant.
 - i:** A label for a solid black dot in the far-right quadrant.
 - o:** Several open circles are scattered throughout the diagram, some connected to the main structure and others isolated.
- Edges:**
 - Solid lines:** Connect 'live' to its two 'ii' nodes. Connect the central 'ii' node to the 'in' node. Connect the 'in' node to the 'i' node.
 - Dashed lines:** Form a large rectangular frame around the entire diagram. Additionally, dashed lines connect some of the 'o' nodes to the frame and to each other.

The *semantic trip* of a semantic net with each $\setminus^\circ$ and $/^\circ$ logical link assigned a distinct semantic variable is the trip which starts upwards at the unique open output root and generates a textual form proceeding as shown in Figure 2 and bouncing with the associated semantic constant

²An anonymous reviewer suggests that the correctness condition as formulated here might have been originally proposed by Arnaud Fleury.

GEOMETRY OF LANGUAGE AND LINGUISTIC CIRCUITRY / 9

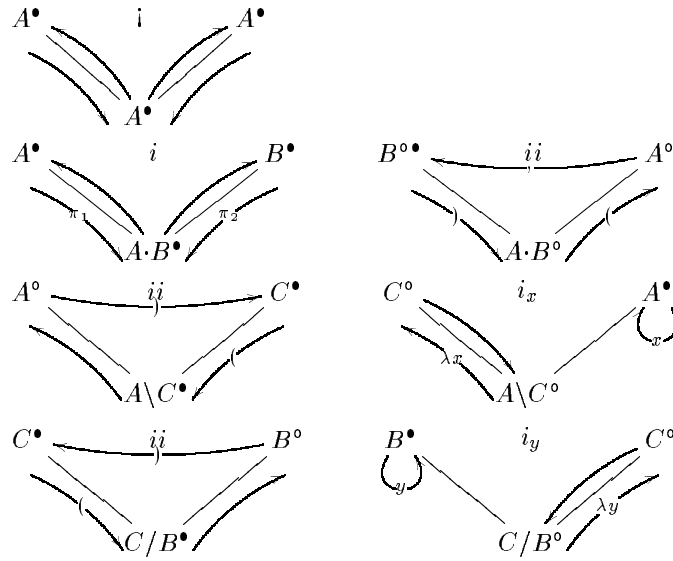


FIGURE 2 Semantic trip travel instructions

at input roots. The semantic trip ends when it returns to the unique open output root. The *reading* of a semantic net is the textual form generated by its semantic trip. For example, the semantic reading of (27) is

$$(28) \lambda x \lambda y ((in\ x) (live\ y))$$

Then we have the following:

Proposition 2 *The reading of a semantic net is a semantic form.*

See de Groote and Retoré (1996). The following conversions on semantic nets preserve equivalence of readings:

$$(29) \begin{array}{c} A^p \quad n \quad B^q \quad B^{\bar{p}} \quad 3-n \quad A^{\bar{q}} \\ \diagdown \quad \diagup \quad \diagdown \quad \diagup \\ C^r \quad C^{\bar{r}} \end{array} \Rightarrow \begin{array}{c} A^p \quad B^q \quad B^{\bar{p}} \quad A^{\bar{q}} \\ | \quad | \quad | \quad | \\ \text{---} \end{array}$$

$$(30) \begin{array}{c} \text{---} \quad \text{---} \\ | \quad | \\ A^p \quad A^{\bar{p}} \quad A^p \\ | \quad | \\ \text{---} \end{array} \Rightarrow A^p \quad \begin{array}{c} \text{---} \quad \text{---} \\ | \quad | \\ A^p \quad A^{\bar{p}} \quad A^p \\ | \quad | \\ \text{---} \end{array} \Rightarrow A^p$$

In view of the equivalence (30) there are infinitely many semantic nets for any given semantic form.

1.2.3 Syntactic nets

A *lexical module* is a semantic module with only input roots and with a strict ordering on the open leaves. A *syntactic frame* is a cyclic list of lexical modules and exactly one output prosodic tree. A *syntactic net* is the result of connecting by an identity link every leaf in a syntactic frame with a unique complementary leaf such that:

- (31) *acyclicity* Every cycle crosses both edges of some i or j -link.
planarity The identity links added are planar in the cyclic ordering.

Let an *initial lexical module* for a lexical assignment $\alpha - \phi: A$ be a lexical module which results from connecting by a Cut link the prosodic tree of $A^\bullet$ with a semantic net the reading of which is ϕ . Then we have the following:

Proposition 3 (Correctness of syntactic nets for BTLG). *There is a syntactic net with reading ϕ on the syntactic frame comprising the prosodic tree of A_0° and initial modules for $\alpha_1 - \phi_1: A_1, \dots, \alpha_n - \phi_n: A_n$ if and only if for all $\alpha_1, \dots, \alpha_n, \phi_1, \dots, \phi_n$, and $\phi' \equiv \phi$, $\{\alpha_1 - \phi_1: A_1, \dots, \alpha_n - \phi_n: A_n\} \models \alpha_1 + \dots + \alpha_n - \phi': A_0$.*

A *final lexical module* for a lexical assignment $\alpha - \phi: A$ is a result of normalizing according to (29) and (30) an initial lexical module for $\alpha - \phi: A$, with the contraction (30) required to preserve the order on open leaves.

For example, for **frodo**— $f: N$ we have the initial lexical module:

$$(32) \quad \begin{array}{c} \begin{array}{cc} \text{---} & \text{---} \\ | & | \\ f^\bullet & \circ \\ | & | \\ \text{---} & \text{---} \end{array} \quad N^\bullet \end{array}$$

Which simplifies to the final lexical module:

$$(33) \quad \begin{array}{c} N^\bullet \\ f \end{array}$$

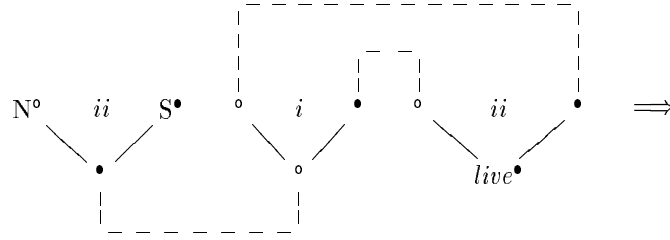
Similarly, for **bag+end**— $b: N$ we obtain the final lexical module:

$$(34) \quad \begin{array}{c} N^\bullet \\ b \end{array}$$

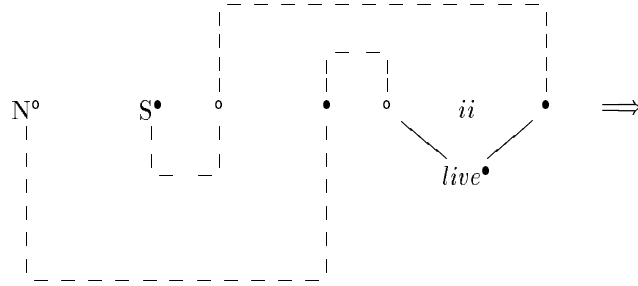
For **lives**— $live: N \setminus S$ we have the initial lexical module, preevaluation and final lexical module:

GEOMETRY OF LANGUAGE AND LINGUISTIC CIRCUITRY / 11

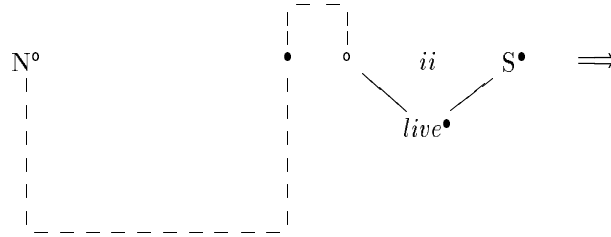
(35) a.



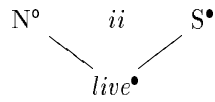
b.



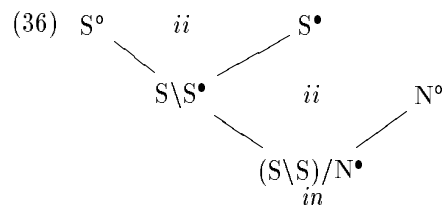
c.



d.

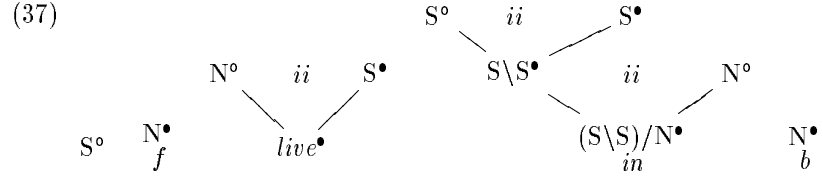


Similarly, for **in**–*in*: $(S \backslash S)/N$ we obtain the final lexical module:

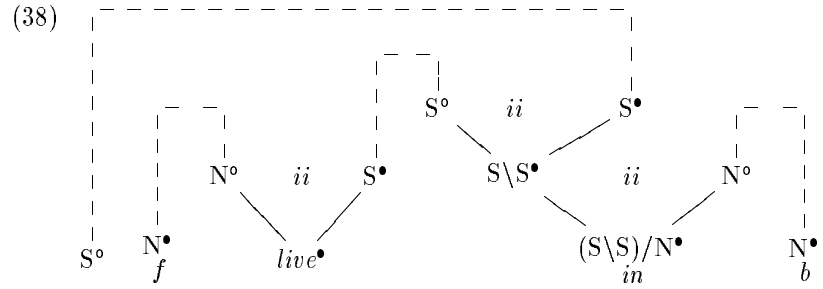


For **inhabits**– $\lambda x \lambda y ((in\ x) (live\ y))$: $(N \backslash S)/N$ we have the initial lexical module, preevaluation and final lexical module given in Figures 3 to 5.

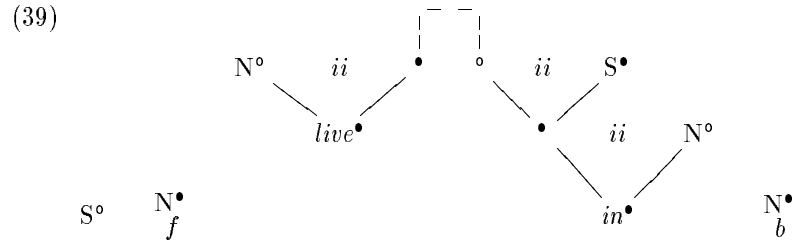
So, for example, from the final lexical modules, we have for the sentence *Frodo lives in Bag End* the syntactic frame:



And the syntactic net:



And, for *Frodo inhabits Bag End*, from the final lexical modules we have the syntactic frame (39) and hence the same syntactic net (38).



The semantic reading of (38) is $((in\ b)\ (live\ f))$.

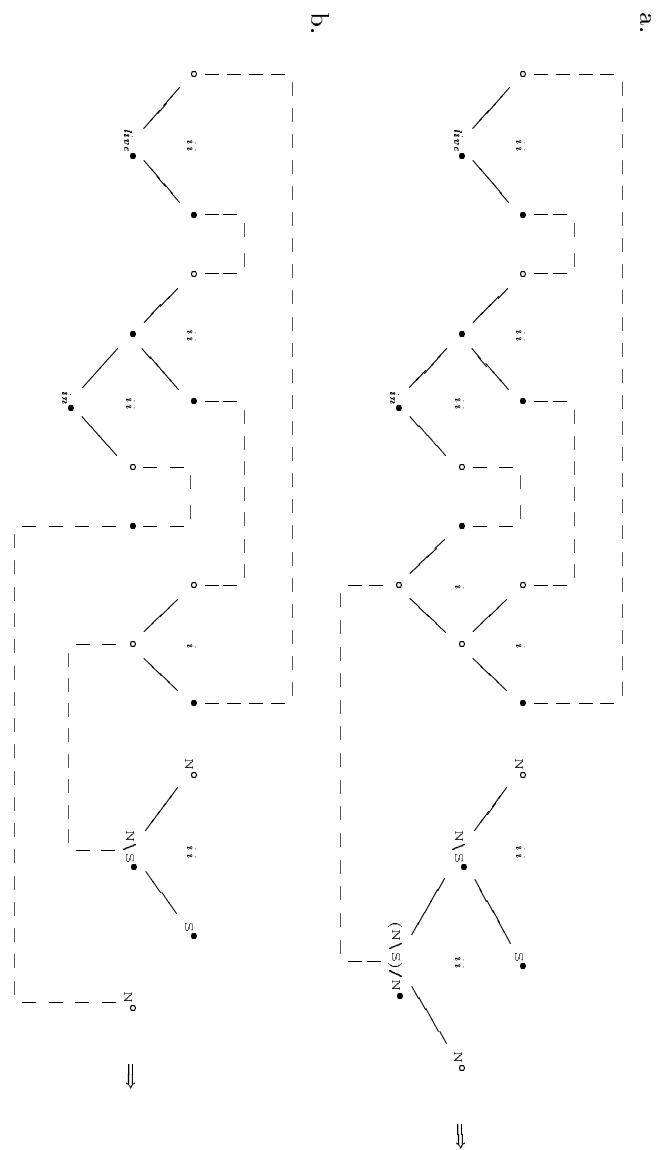


FIGURE 3 Lexical execution, part I

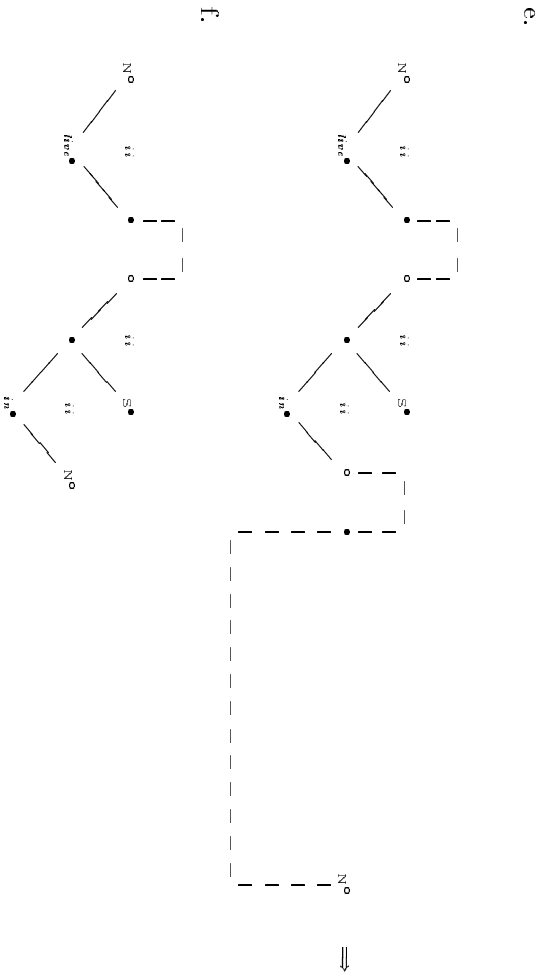


FIGURE 5 Lexical execution, part III

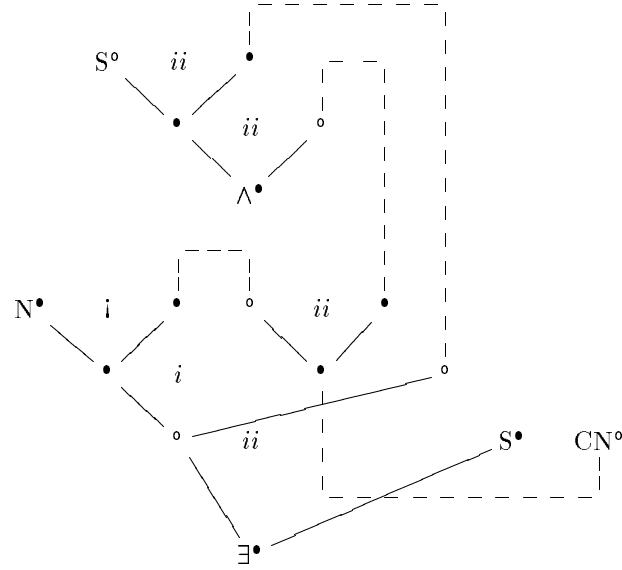
1.3 Words

In this section we illustrate final lexical modules for sample lexical assignments. Note that, as is the case with partial execution in general, the relation between the source and the result is not necessarily transparent, but the compilations follow according to the principles of the previous section.

1.3.1 **a**— $\lambda x \lambda y (\exists \lambda z (\wedge (x \ z) (y \ z)))$: $((S/N) \backslash S)/CN$

Indefinite article of verb phrase-final noun phrase.

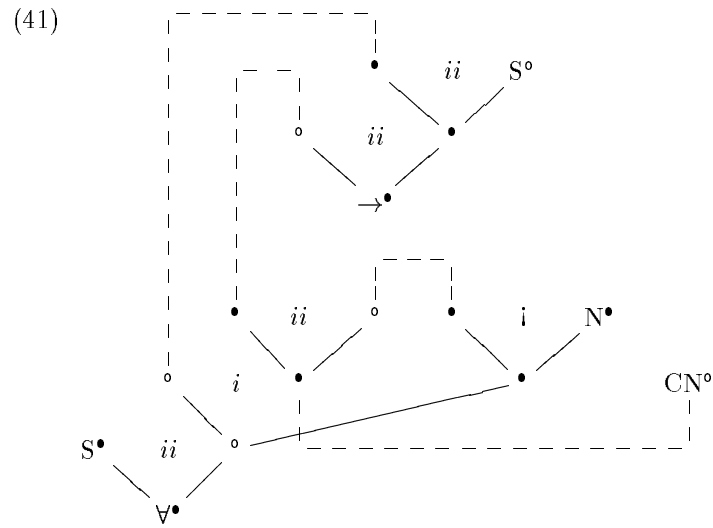
(40)



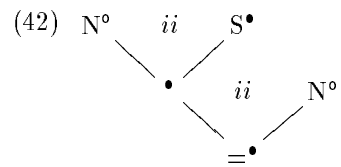
1.3.2 **every**— $\lambda x \lambda y (\forall \lambda z (\rightarrow (x \ z) (y \ z)))$: $(S/(N \backslash S))/CN$

Universal quantifier of subject noun phrase.

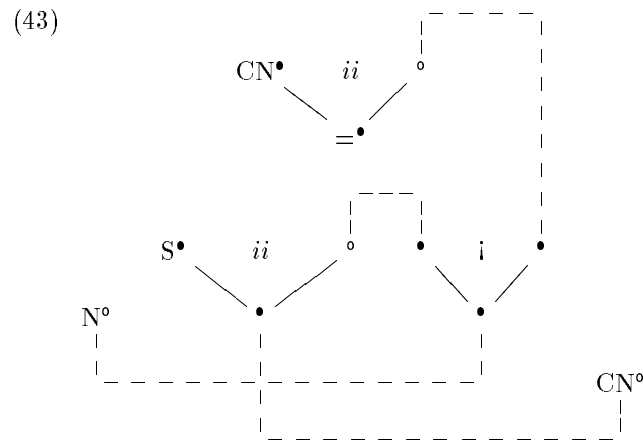
GEOMETRY OF LANGUAGE AND LINGUISTIC CIRCUITRY / 17

**1.3.3** $is \equiv: (N \setminus S) / N$

Equative copula.

**1.3.4** $is - \lambda x \lambda y (x (= y) y): (N \setminus S) / (CN / CN)$

Predicative copula.



18 / GLYN MORRILL

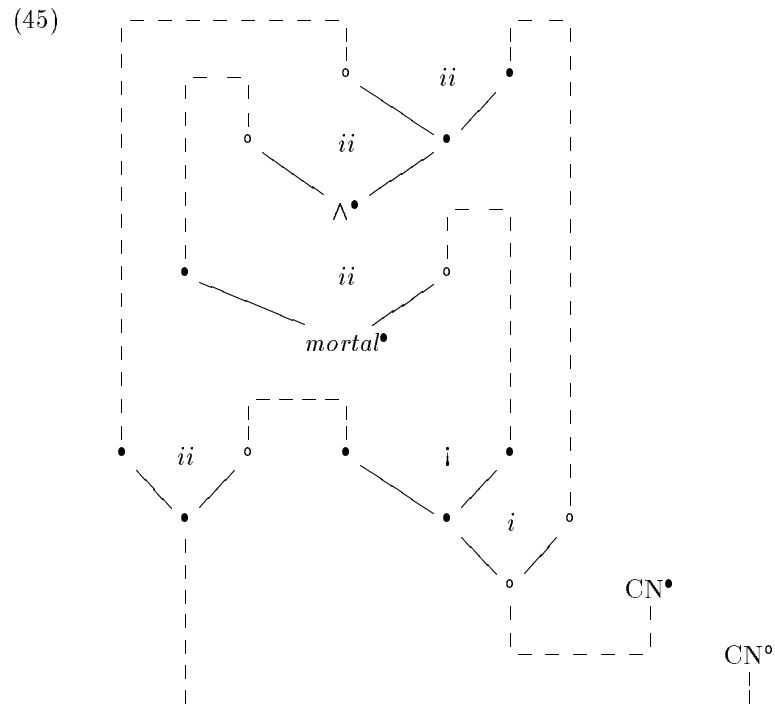
1.3.5 **man**—*man*: CN

Count noun.

(44) $\text{CN}^\bullet$
man

1.3.6 **mortal**— $\lambda x \lambda y (\wedge (mortal\ y) (x\ y))$: CN/CN

Intersective adjective.



1.3.7 **socrates**—*s*: N

Proper name.

(46) $\text{N}^\bullet$
s

1.4 Sentences

We analyse the classical syllogism: *Socrates is a man*, *Every man is mortal* / *Socrates is mortal*.

1.4.1 **Socrates is a man**

The syntactic net is given in Figure 6. The semantic reading is:

GEOMETRY OF LANGUAGE AND LINGUISTIC CIRCUITRY / 19

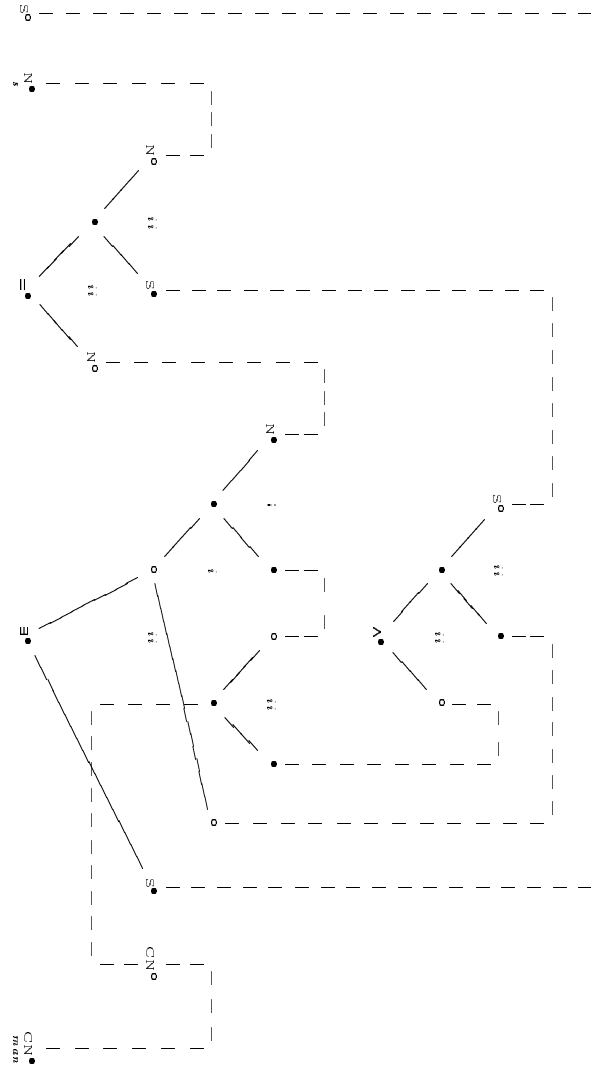


FIGURE 6 Syntactic net for *Socrates is a man*

20 / GLYN MORRILL

$$\begin{aligned}
(47) \quad & (\exists \lambda x ((\wedge (\text{man } x)) ((= x) s))) \\
& \equiv \\
& (\text{man } s)
\end{aligned}$$

1.4.2 Every man is mortal

The syntactic net is given in Figures 7 to 8. The semantic reading is:

$$\begin{aligned}
(48) \quad & (\forall \lambda x ((\rightarrow (\text{man } x)) (\lambda y ((\wedge (\text{mortal } y)) ((= x) y)) x))) \\
& \equiv \\
& (\forall \lambda x (\rightarrow (\text{man } x) (\text{mortal } x)))
\end{aligned}$$

Note that the syntactic analysis creates a β -redex which is not anticipated by the isolated partial execution of lexical modules.

1.4.3 Socrates is mortal

The syntactic net is given in Figure 9. The semantic reading is:

$$\begin{aligned}
(49) \quad & (\lambda x ((\wedge (\text{mortal } x)) ((= s) x)) s) \\
& \equiv \\
& (\text{mortal } s)
\end{aligned}$$

Indeed, in every semantic interpretation in which the premisses of the syllogism evaluate to truthhood, the conclusion of the syllogism evaluates to truthhood.

Appendix

We give a simple Prolog program computing the analyses of section 4. It implements the procedure of Morrill (2000) as a ‘double’ shift-reduce parser in which the stack is split into a global stack and a local stack, to deal one by one with the leaves of lexical modules; the only subtlety is to check that reduction (placement of axiom links) complies with *acyclicity*. The nets are incrementally built encoding their edges dynamically in the Prolog database.

Program listing

`t(?N)` is the top-level call to process as a sentence a string indexed `N`.

```

t(N) :- s(N, Str),
        nl, nl, write(N), write(' '), write(Str), clean,
        p([out(s, '$VAR'(0))], [], Str).

```

`p(+Global_stack, +Local_stack, +Buffer)` is the main engine to process the local stack `Local_stack` and the remaining string `Buffer` in the context of `Global_stack`.

```

p([], [], []) :- nl, nl, semup('$VAR'(0)), fail.

```

GEOMETRY OF LANGUAGE AND LINGUISTIC CIRCUITRY / 21

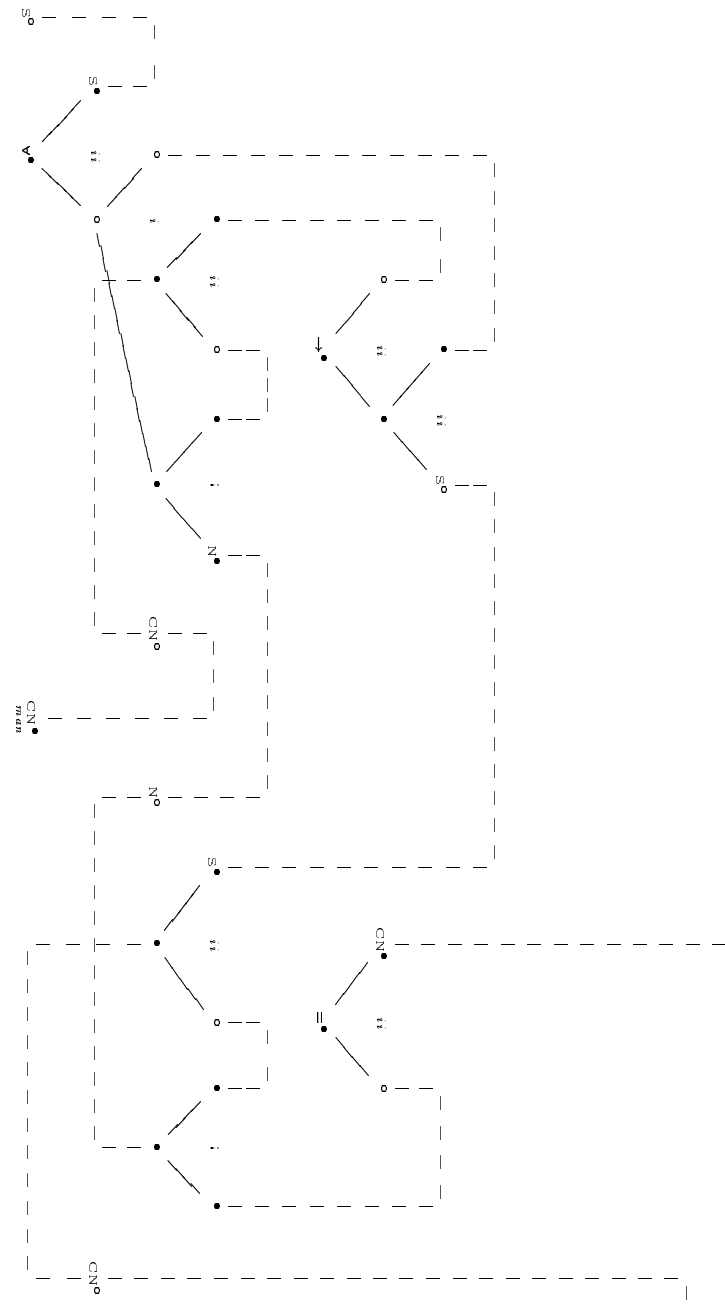


FIGURE 7 Syntactic net for *Every man is mortal*, part I

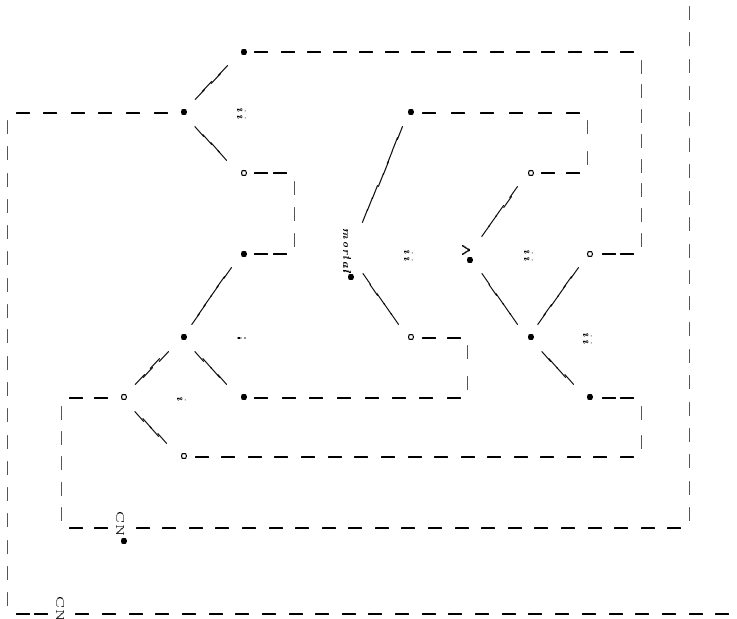


FIGURE 8 Syntactic net for *Every man is mortal*, part II

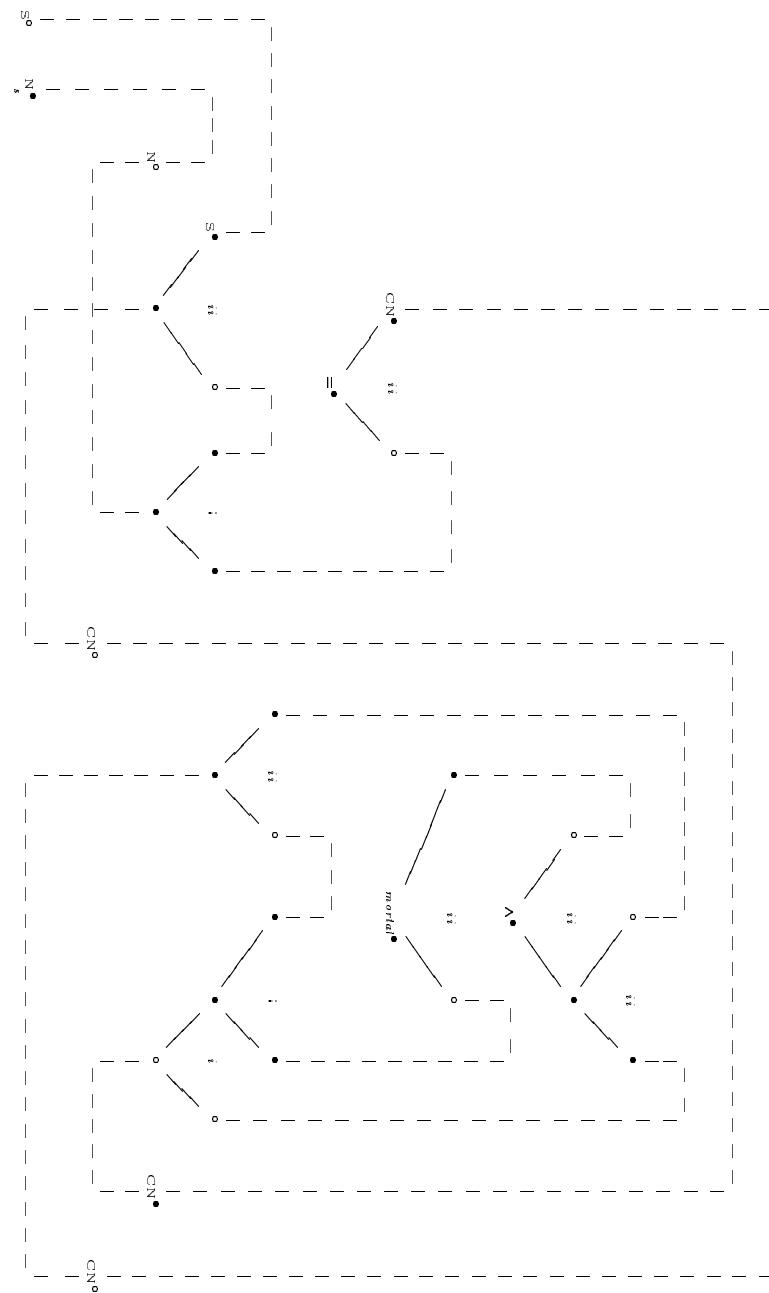


FIGURE 9 Syntactic net for *Socrates is mortal*

24 / GLYN MORRILL

```
p(GS, [], [W|Str]) :- !, lex2(W, Fringe),
    p(GS, Fringe, Str).
```

```
p([L1|GS], [L2|LS], Str) :- compl2(L1, L2),
    p(GS, LS, Str).
```

```
p(GS, [L|LS], Str) :- p([L|GS], LS, Str).
```

`compl2(+L1, +L2)` means that the leaves `L1` and `L2` are complementary and are not connected by a *i* and *j*-free path, that is a path that does not cross both edges of any *i* or *j*-link. So placing an identity link between `L1` and `L2` will not create a ‘vicious circle’, a violation of *acyclicity*.

```
compl2(L1, L2) :- compl(L1, L2, N1, N2),
    \+dn(N1, N2),
    groundlist([id(N1, N2)]).
```

```
groundlist(Links) :- assertlist(Links).
```

```
groundlist(Links) :- retractlist(Links).
```

```
retractlist([H|T]) :- retract(H),
    retractlist(T).
```

```
assertlist([]).
```

```
assertlist([H|T]) :- assert(H),
    assertlist(T).
```

`dn(+N1, +N2)` means that there is a *i* and *j*-free path starting downwards at `N1` and ending upwards at `N2` and `up(+N1, +N2)` means that there is a *i* and *j*-free path starting upwards at `N1` and ending upwards at `N2`; they are mutually recursive.

```
dn(N1, N2) :- ii(N1, C, P2, _), !, (up(P2, N2); dn(C, N2)).
```

```
dn(N1, N2) :- ii(P1, C, N1, _), !, (up(P1, N2); dn(C, N2)).
```

```
dn(N1, N2) :- i(N1, C, _, _), !, dn(C, N2).
```

```
dn(N1, N2) :- i(_, C, N1, _), !, dn(C, N2).
```

```
dn(N, M) :- cut(N, N1), up(N1, M).
```

GEOMETRY OF LANGUAGE AND LINGUISTIC CIRCUITRY / 25

`dn(N, M) :- cut(N1, N), up(N1, M).`

`up(N, N).`

`up(N, M) :- i(P1, N, P2, _), !, (up(P1, M); up(P2, M)).`

`up(N, M) :- ii(P1, N, P2, _), !, (up(P1, M); up(P2, M)).`

`up(N, M) :- id(N, N1), dn(N1, M).`

`up(N, M) :- id(N1, N), dn(N1, M).`

`compl(+L1, +L2, -N, -M)` means that leaves `L1` and `L2` are complementary and are numbered `N` and `M` respectively.

`compl(inp(P, N), out(P, M), N, M).`

`compl(out(P, N), inp(P, M), N, M).`

`lex2(+W, -Fringe)` means that word `W` has a lexical module with ordered leaves `Fringe`.

```
lex2(W, Fringe) :- lex(W, Fringe, Links),
    retract(count(N)),
    numbervars(Links, N, N2),
    assert(count(N2)),
    groundlist(Links).
```

`lex(+W, -Fringe, -Module)` means that word `W` has lexical module `Module` with ordered leaves `Fringe`.

```
lex(a, [inp(n, A), out(s, B), inp(s, C), out(cn, D)],
[id(Y, Z), id(P, V), id(Q, N),
i(A, X, Y, c), ii(B, T, P, u), ii(W, R, C, u),
cut(U, D), ii(T, S, Q, o), ii(Z, U, N, u), i(X, W, V, o),
rs(R, 'E'), rs(S, '&')]).
lex(every, [inp(s, A), out(s, B), inp(n, C), out(cn, D)],
[ii(X, Z, B, o), ii(Y, X1, Z, u), ii(X2, X6, X3, o),
ii(A, X9, X8, o), i(X4, X7, C, c), i(X5, X8, X7, u), cut(X6, D),
id(X5, X), id(X2, Y), id(X3, X4), rs(X1, '->'), rs(X9, 'A')]).
lex(is, [out(n, A), inp(s, B), out(n, C)],
[ii(A, X, B, u), ii(X, Y, C, o), rs(Y, '=')]).
lex(is, [out(n, A), inp(s, B), inp(cn, C), out(cn, D)],
[ii(X, X1, B, u), ii(C, X2, Y, o), i(Z, X4, X3, c),
id(Z, X), id(Y, X3), cut(A, X4), cut(X1, D), rs(X2, '=')]).
lex(man, [inp(cn, A)], [rs(A, man)]).
lex(mortal, [inp(cn, A), out(cn, B)],
```

26 / GLYN MORRILL

```
[ii(X, Z, Y, u), ii(Z, X2, X1, o), ii(X3, X5, X4, u),
i(X6, X10, X7, c), ii(X8, X13, X9, o), i(X10, X12, X11, o),
id(X8, X), id(Y, X11), id(X9, X6), id(X1, X4), id(X7, X3),
cut(X13, B), cut(X12, A), rs(X2, '&'), rs(X5, mortal)]]).
lex(socrates, [inp(n, A)], [rs(A, s)]).
```

clean resets the Prolog database regarding net predicates.

```
clean :- retractall(count(_)),
         retractall(cut(_, _)),
         retractall(rs(_, _)),
         retractall(id(_, _)),
         retractall(i(_, _, _, _)),
         retractall(ii(_, _, _, _)),
         assert(count(1)).
```

semup(+N) performs the semantic trip starting upwards at node N and
 semdn(+N) performs the semantic trip starting downwards at node N;
 they are mutually recursive.

```
semup(N) :- id(N, M),
           semdn(M).
```

```
semup(N) :- id(M, N),
           semdn(M).
```

```
semup(N) :- i(P1, N, P2, p),
           write('('), semup(P2), write(', '), semup(P1), write(')').
```

```
semup('$VAR'(I)) :- i(P1, '$VAR'(I), _, u),
                  write('L'), write(I), semup(P1).
```

```
semup('$VAR'(I)) :- i(_, '$VAR'(I), P2, o),
                  write('L'), write(I), semup(P2).
```

```
semdn(N) :- rs(N, Phi),
           write(Phi).
```

```
semdn(N) :- cut(N, M),
           semup(M).
```

```
semdn(N) :- cut(M, N),
           semup(M).
```

GEOMETRY OF LANGUAGE AND LINGUISTIC CIRCUITRY / 27

```

semdn(N) :- i(N, C, _, c),
             semdn(C).

semdn(N) :- i(_, C, N, c),
             semdn(C).

semdn(N) :- ii(N, C, _, p),
             write('p1'), semdn(C).

semdn(N) :- ii(_, C, N, p),
             write('p2'), semdn(C).

semdn(N) :- ii(P1, C, N, u),
             write('('), semdn(C), write(' '), semup(P1), write(')').

semdn(N) :- ii(N, C, P2, o),
             write('('), semdn(C), write(' '), semup(P2), write(')').

semdn(N) :- i(_, '$VAR'(I), N, u),
             write(I).

semdn(N) :- i(N, '$VAR'(I), _, o),
             write(I).

s(?N, -String) means that string String has index N.

s(1, [socrates, is, a, man]).
s(2, [every, man, is, mortal]).
s(3, [socrates, is, mortal]).

```

Log

```

Script started on Thu Sep 25 12:18:55 2003
[david] ~/Catlog/Net2 >sicstus
SICStus 2.1 #9: Thu Jun 19 13:03:29 MET DST 1997
| ?- [trip030925].
{consulting /amd/entalpia/home3/usuaris/morrill/Catlog/Net2/trip030925.pl...}
{/amd/entalpia/home3/usuaris/morrill/Catlog/Net2/trip030925.pl consulted, 567 msec 19

yes
| ?- t(_).

```

```

1. [socrates,is,a,man]

```

28 / GLYN MORRILL

(E L35((& (man 35)) ((= 35) s)))

2. [every,man,is,mortal]

(A L11((-> (man 11)) (L120((& (mortal 120)) ((= 11) 120)) 11)))

3. [socrates,is,mortal]

(L104((& (mortal 104)) ((= s) 104)) s)

no

| ?- ^D

[david] ~/Catlog/Net2 >^D

script done on Thu Sep 25 12:19:37 2003

References

- Buszkowski, W. 1986. Completeness results for Lambek syntactic calculus. *Z. Math. Logik Grundlag. Math.* 32:13–28.
- Carpenter, Bob. 1997. *Type-Logical Semantics*. Cambridge, MA: MIT Press.
- Danos, V. and L. Regnier. 1989. The structure of multiplicatives. *Archive for Mathematical Logic* 28:181–203.
- de Groote, Philippe and Christian Retoré. 1996. On the Semantic Readings of Proof-Nets. In G.-J. Kruijff, G. Morrill, and R. T. Oehrle, eds., *Proceedings of Formal Grammar 1996*, pages 57–70. Prague.
- Girard, Jean-Yves. 1987. Linear Logic. *Theoretical Computer Science* 50:1–102.
- Lambek, J. 1958. The mathematics of sentence structure. *American Mathematical Monthly* 65:154–170.
- Lambek, J. 1988. Categorical and Categorical Grammars. In R. T. Oehrle, E. Bach, and D. Wheeler, eds., *Categorical Grammars and Natural Language Structures*, Studies in Linguistics and Philosophy Volume 32, pages 297–317. Dordrecht: D. Reidel.
- Lecomte, Alain and Christian Retoré. 1995. An alternative categorial grammar. In G. Morrill and R. T. Oehrle, eds., *Proceedings of Formal Grammar 1995*, pages 181–196. Barcelona.
- Moortgat, Michael. 1997. Categorical type logics. In J. van Benthem and A. ter Meulen, eds., *Handbook of Logic and Language*, pages 93–177. Amsterdam: Elsevier.
- Morrill, Glyn. 1994. *Type Logical Grammar: Categorical Logic of Signs*. Dordrecht: Kluwer Academic Publishers.
- Morrill, Glyn. 1999. Geometry of Lexico-Syntactic Interaction. In *Proceedings of the Ninth Conference of the European Chapter of the Association for Computational Linguistics*, pages 61–70. Bergen.

REFERENCES / 29

- Morrill, Glyn. 2000. Incremental Processing and Acceptability. *Computational Linguistics* 26(3):319–338.
- Nabokov, Vladimir. 1969. *Speak, Memory*. London: Penguin Books.
- Roorda, Dirk. 1991. *Resource Logics: Proof-Theoretical Investigations*. Ph.D. thesis, Universiteit van Amsterdam.

October 7, 2003