Geometry of language and linguistic circuitry*

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

March 25, 2003

Abstract

We illustrate the potential for geometry of language and linguistic circuitry under the rendering of the syntactic structures of Lambek categorial grammar as proof nets. This empirical application 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 remaniscent of circuits, but whereas circuits are just generalisations of formulas, our syntactic structures are much more sublime objects: proofs.

*Thanks to the audiences at EACL, Bergen (Morrill 1999), the workshop on Linear Logic and Applications, Dagstuhl, August 1999, and the workshop on Grammar and Logic, Rennes, April 2000, and to Mario Fadda and Oriol Valentí for many related conversations. 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/.

The proof nets introduced by Girard (1987) for linear logic not only appear to be optimally parsimoneous, 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'. De Groote and Retoré (1996) show 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 Basic TLG

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

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

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

(2) $\quad s_1{+}(s_2{+}s_3) = (s_1{+}s_2){+}s_3$ $\quad$ + is associative

$\quad\quad 0{+}s = s = s{+}0$ $\quad$ 0 is an identity element for +

A *prosodic interpretation* comprises a prosodic structure $(L, +, 0)$ and a valuation v mapping from A into L. Then the prosodic object $[\alpha]_v$ denoted by a prosodic form α with respect to a prosodic interpretation with valuation v is defined by:

(3) $\quad [\mathbf{a}]_v = v(\mathbf{a})$ for prosodic constant $\mathbf{a}$

$\quad\quad [\alpha{+}\beta]_v = [\alpha]_v{+}[\beta]_v$

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

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

So we can drop parentheses in prosodic forms.

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

(5) $\mathcal{T} \quad ::= \quad \mathrm{e} \mid \mathrm{o} \mid \mathcal{T} \to \mathcal{T} \mid \mathcal{T}\&\mathcal{T}$

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) $\blacksquare, \square \in C_{\mathrm{o}}$
$\neg \in C_{\mathrm{o}\to\mathrm{o}}$
$\wedge, \vee, \to \in C_{\mathrm{o}\to(\mathrm{o}\to\mathrm{o})}$
$= \in C_{\mathrm{e}\to(\mathrm{e}\to\mathrm{o})}$
$\forall, \exists \in C_{(\mathrm{e}\to\mathrm{o})\to\mathrm{o}}$

Then the set $\Phi_{\tau,X}$ of *semantic terms of type* τ *with free variables* X for each semantic type τ and set X of semantic variables is defined by:

(7)
$$\begin{array}{lll}
\Phi_{\tau,\emptyset} & ::= & C_\tau \\
\Phi_{\tau,\{V\}} & ::= & V_\tau \\
\phi_{\tau,X\cup Y} & ::= & (\Phi_{\tau'\to\tau,X}\ \Phi_{\tau',Y}) \\
\Phi_{\tau,X} & ::= & \pi_1\Phi_{\tau\&\tau',X} \mid \pi_2\Phi_{\tau'\&\tau,X} \\
\Phi_{\tau\to\tau',X} & ::= & \lambda V_\tau \Phi_{\tau',X\cup\{V\}} \\
\Phi_{\tau\&\tau',X\cup Y} & ::= & (\Phi_{\tau,X}, \Phi_{\tau',Y})
\end{array}$$

An occurrence of a 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, i.e. a semantic term of $\Phi_{\tau,\emptyset}$.

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; there is *accidental capture* in the application of a substitution to a semantic term if and only if some 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:

(8) D_{e} is a non-empty set E of *entities*
D_{o} is the powerset of a non-empty set W of *worlds*
$D_{\tau\&\tau'} = D_\tau \times D_{\tau'}$
$D_{\tau\to\tau'} = D_{\tau'}^{D_{\tau}}$

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

(9)
$$
\begin{aligned}
f(\blacksquare) &= W\\
f(\square) &= \emptyset\\
f(\neg) &= m \mapsto \overline{m}^W\\
f(\wedge) &= m \mapsto (m' \mapsto m \cap m')\\
f(\vee) &= m \mapsto (m' \mapsto m \cup m')\\
f(\to) &= m \mapsto (m' \mapsto \overline{m}^W \cup m')\\
f(=) &= m \mapsto (m' \mapsto (W \text{ if } m = m' \text{ else } \emptyset))\\
f(\forall) &= m \mapsto \textstyle\bigcap_{m'\in E} m(m')\\
f(\exists) &= m \mapsto \textstyle\bigcup_{m'\in E} m(m')
\end{aligned}
$$

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

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

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

(11)
$$\lambda x\phi \quad \equiv \quad \lambda y(\phi\{y/x\}) \qquad \alpha\text{-conversion}$$
provided y is not free in ϕ and there is no accidental capture in $\phi\{y/x\}$

$$(\lambda x\phi\ \psi) \quad \equiv \quad \phi\{\psi/x\} \qquad \beta\text{-conversion}$$
provided there is no accidental capture in $\phi\{\psi/x\}$

$$\pi_1(\phi,\psi) \quad \equiv \quad \phi \qquad\qquad \pi_2(\phi,\psi) \quad \equiv \quad \psi$$

$$\lambda x(\phi\ x) \quad \equiv \quad \phi \qquad \eta\text{-conversion}$$
provided x is not free in ϕ

$$(\pi_1\phi,\pi_2\phi) \quad \equiv \quad \phi$$

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

(12)
$$\begin{array}{rcl}
((=\phi)\ \phi) & \equiv & \blacksquare \\
((\wedge\ \phi)\ \blacksquare) & \equiv & \phi \\
(\exists\ \lambda x((\wedge\ \phi)\ ((=x)\ \psi))) & \equiv & \phi\{\psi/x\} \quad \text{provided there is no accidental capture in } \phi\{\psi/x\}
\end{array}$$

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

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

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

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

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 $[\![A]\!]$ of a syntactic type with respect to a syntactic interpretation is defined by:

(15)
$$\begin{array}{lcl} [\![P]\!] & = & F(P) \text{ for atomic syntactic type } P \\ [\![A{\cdot}B]\!] & = & \{\langle s_3, \langle m_1, m_2\rangle\rangle \mid \exists\langle s_1, m_1\rangle \in [\![A]\!], \langle s_2, m_2\rangle \in [\![B]\!], s_3 = s_1{+}s_2\} \\ [\![A\backslash C]\!] & = & \{\langle s_2, m_2\rangle \mid \forall\langle s_1, m_1\rangle \in [\![A]\!], \langle s_1{+}s_2, m_2(m_1) \in [\![C]\!]\rangle\rangle \\ [\![C/B]\!] & = & \{\langle s_1, m_1\rangle \mid \forall\langle s_2, m_2\rangle \in [\![B]\!], \langle s_1{+}s_2, m_1(m_2) \in [\![C]\!]\rangle\rangle \end{array}$$

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 $\langle[\alpha], [\phi]\rangle \in [\![A]\!]$; 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) $\mathcal{L}(Lex) \quad = \quad \{\alpha{-}\phi\text{: } A \mid Lex \models \alpha{-}\phi\text{: } A\}$

For example, let there be the following lexicon:

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

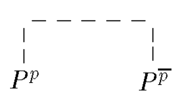
2 Nets for BTLG

2.1 Prosodic nets

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

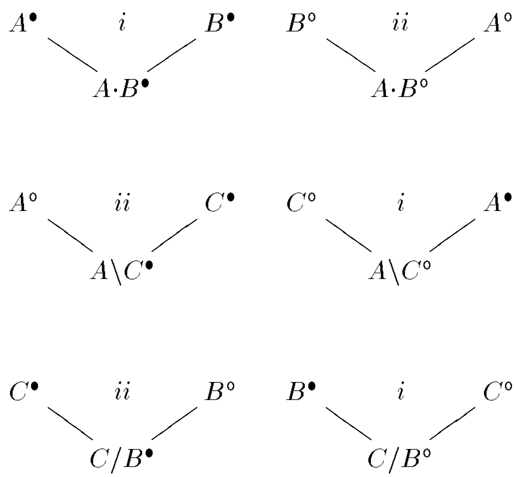
An *identity link* is of the form:

(19)



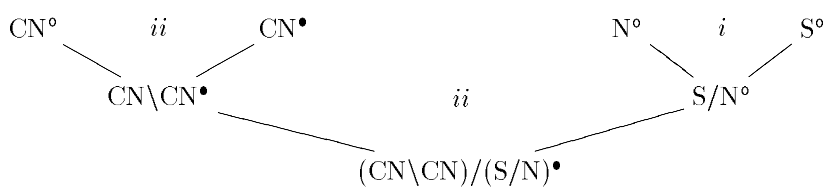
A *logical link* is one of the forms:

(20)



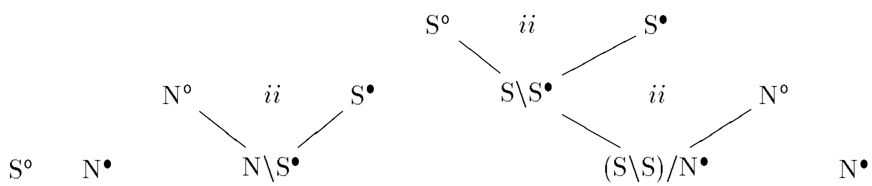
A *prosodic tree* is a tree the leaves of which are literals and each local tree of which is a logical link. Each label is the root of a unique prosodic tree, which is the result of unfolding the label upwards according to the logical links. For example, the prosodic tree for $(\mathrm{CN}\backslash\mathrm{CN})/(\mathrm{S}/\mathrm{N})^\bullet$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

(21)



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

(22)

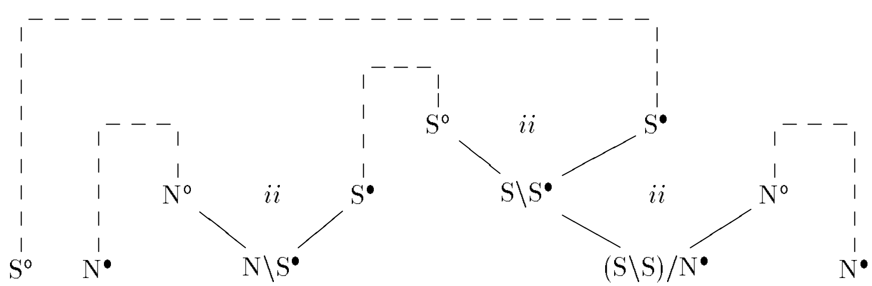


A *prosodic net* is the result of connecting by an identity link every leaf in a prosodic frame with a complementary leaf such that:

(23) *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:

(24)



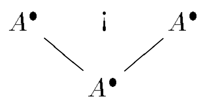
Then we have the following:

(25) **Claim** (*Correctness of prosodic nets for BTLG*).
There is a prosodic net with roots $A_0{}^\circ, A_1{}^\bullet, \ldots, A_n{}^\bullet$ if and only if $\{\alpha_1 \colon A_1, \ldots, \alpha_n \colon A_n\} \models \alpha_1 + \cdots + \alpha_n \colon A_0$ for all $\alpha_1, \ldots, \alpha_n$.

2.2 Semantic nets

A *contraction link* is of the form:

(26)



A *Cut link* is of the form:

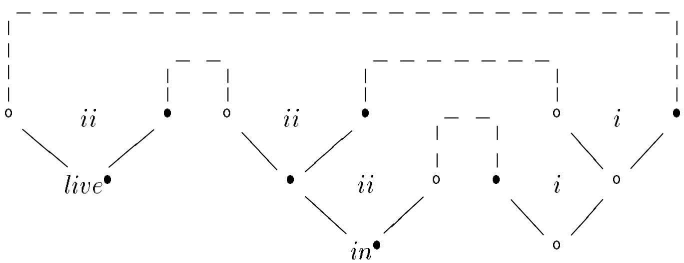
(27)

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 exactly one of which has an output root. Note that prosodic frames are semantic frames when we forget about the cyclic order. A *semantic module* is the result of i) possibly connecting in a semantic frame some leaves with a complementary leaf by an identity link and some roots with a complementary root by a Cut link, and ii) associating semantic constants to every open input root, such that:

(28) *acyclicity* Every cycle crosses both edges of some i-link.

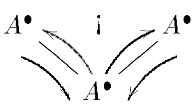
A *semantic net* is a semantic module with no open leaves. For example, the following is a semantic net:

(29)

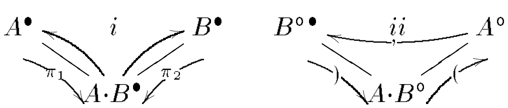


The *semantic trip* of a semantic net is the trip which starts upwards at the unique open output root and generates a semantic form proceeding as follows and bouncing with the associated semantic form at input roots:

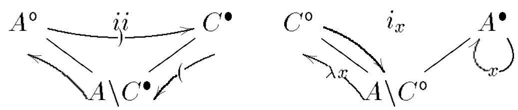
(30)



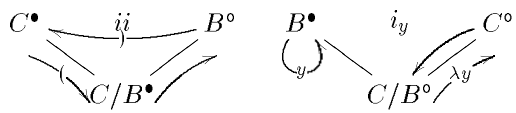
(31)



(32)



(33)



The semantic trip ends when it returns to the unique open output root. The *reading* ϕ_Π of a semantic net Π is the semantic term generated by its semantic trip. For example, the semantic reading of (29) is

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

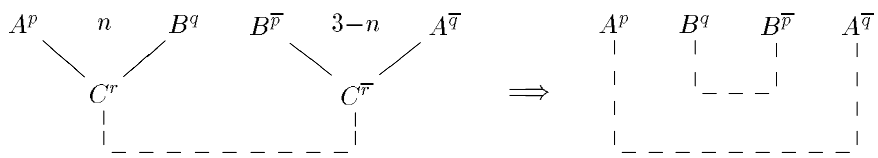
Then we have the following:

(35) **Claim.**

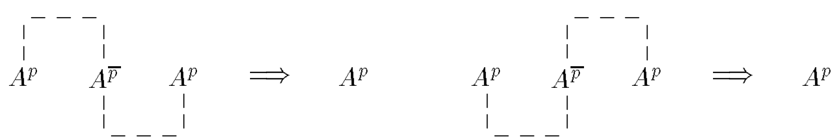
The readings of the semantic nets are the semantics forms.

The following conversions on semantic nets preserve equivalence of readings:

(36)



(37)



2.3 Syntactic nets

A *syntactic module* is a bag of semantic modules with no open output root and with a strict ordering on their open leaves. A *syntactic frame* is a cyclic list of syntactic 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 complementary leaf such that:

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

Let an *initial module* for a lexical assignment $\alpha{-}\phi$: A be a syntactic module which results from connecting by a Cut link the prosodic tree of A with a semantic net the reading of which is ϕ.

Then we have the following:

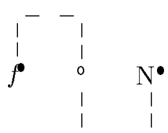
(39) **Claim** (*Correctness of syntactic nets for BTLG*).

There is a syntactic net with reading ϕ on the syntactic frame comprising the prosodic tree of $A_0{}^\circ$ and initial modules for $\alpha_1 - \phi_1 \colon A_1, \ldots, \alpha_n - \phi_n \colon A_n$ if and only if $\{\alpha_1 - \phi_1 \colon A_1, \ldots, \alpha_n - \phi_n \colon A_n\} \models \alpha_1 + \cdots + \alpha_n - \phi' \colon A_0$ for all $\phi' \equiv \phi, \alpha_1, \ldots, \alpha_n, \phi_1, \ldots, \phi_n$.

A *lexical module* for a lexical assignment $\alpha - \phi \colon A$ is a result of normalizing according to (36) an initial module for $\alpha - \phi \colon A$.

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

(40)



Whixh simplifies to the lexical module:

(41)

$$\underset{f}{\mathrm{N}^\bullet}$$

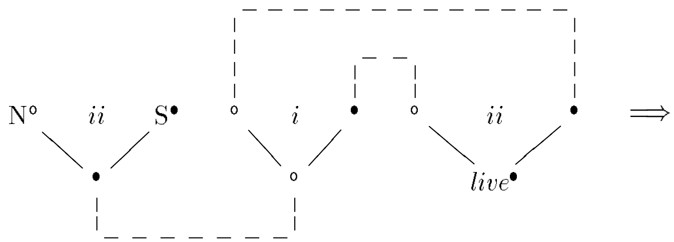
Similarly, for **bag+end**$-b$: N we obtain the lexical module:

(42)

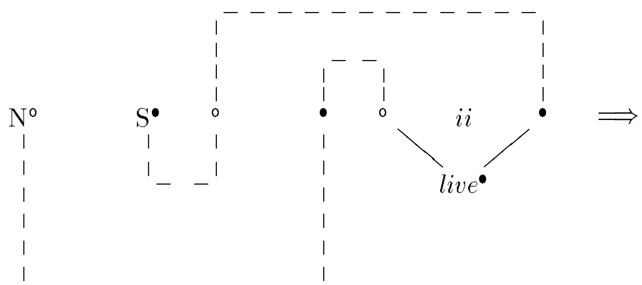
$$\underset{b}{\mathrm{N}^\bullet}$$

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

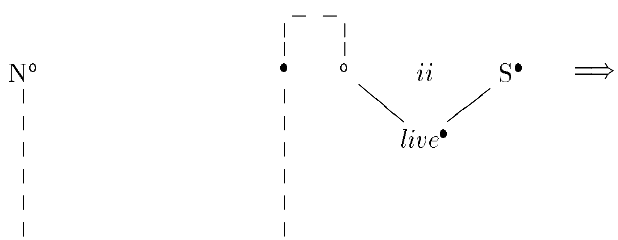
(43)



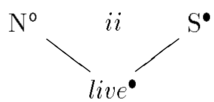
(44)



(45)

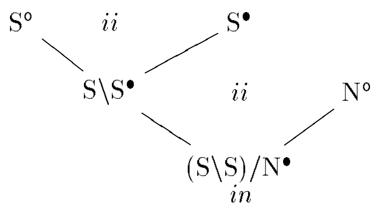


(46)



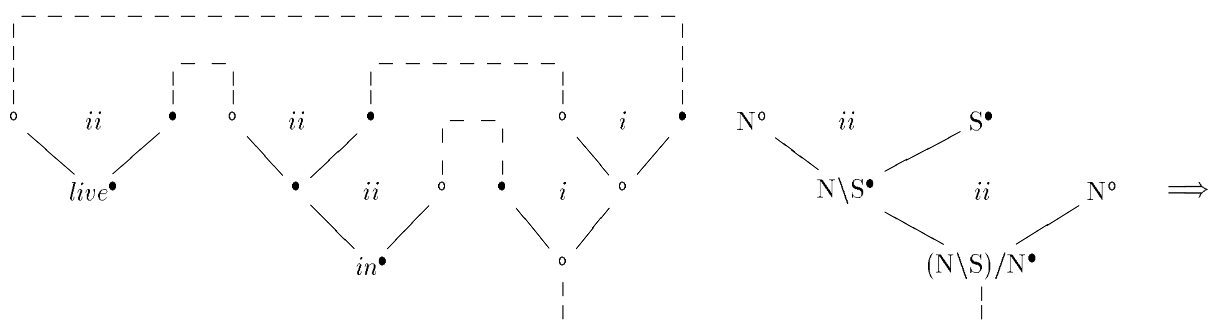
Similarly, for **in**–*in*: (S\S)/N we obtain:

(47)

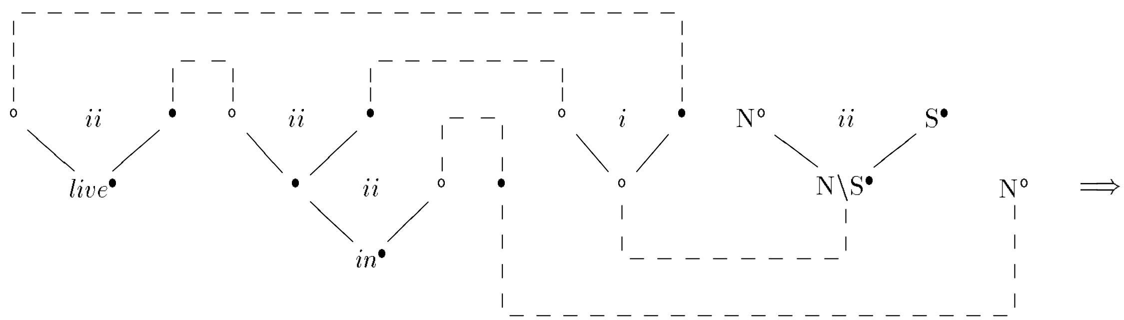


For **inhabits**–$\lambda x \lambda y((in\ x)\ (live\ y))$: (N\S)/N we have the initial module, preevaluation and lexical module:

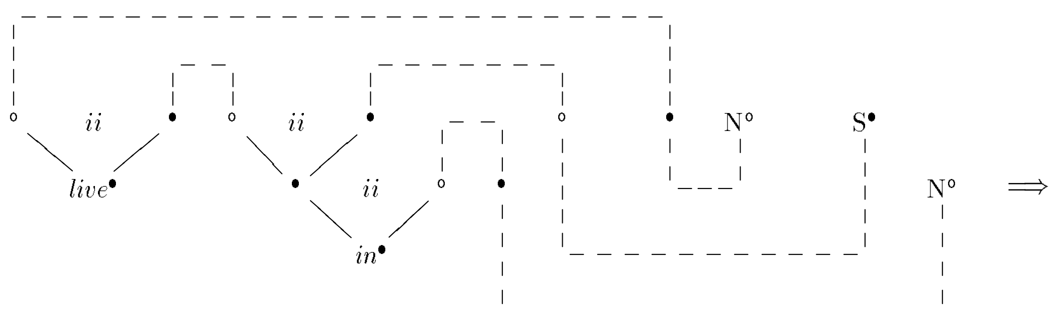
(48)



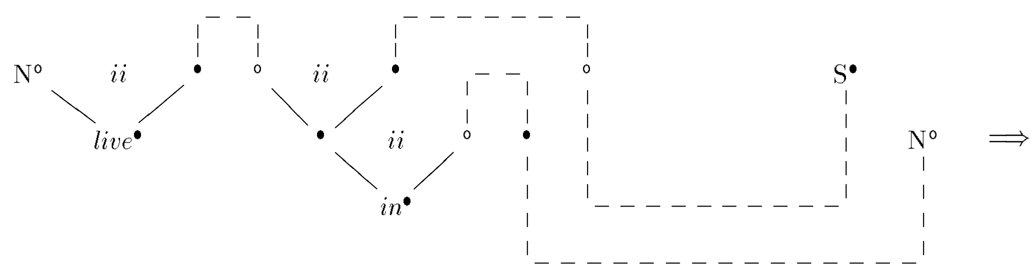
(49)



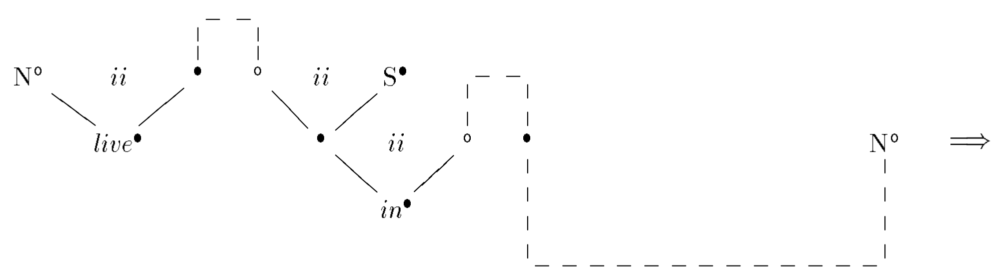
(50)



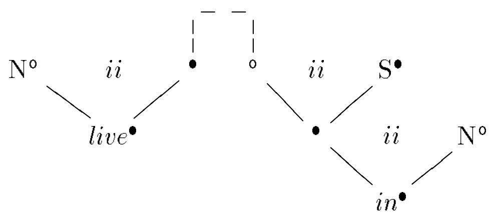
(51)



(52)

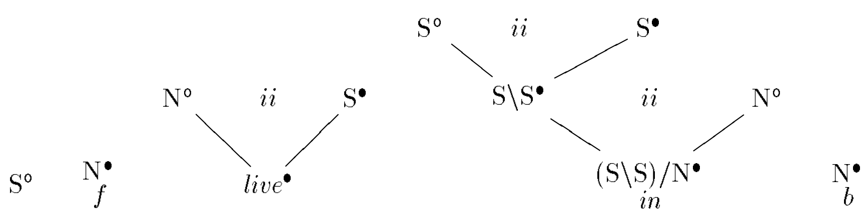


(53)



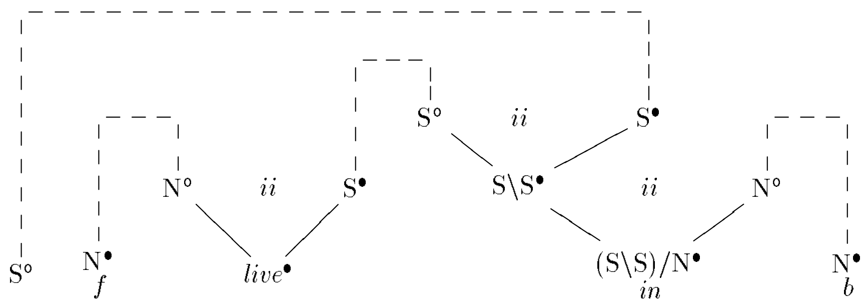
For *Frodo lives in Bag End* we have the syntactic frame:

(54)



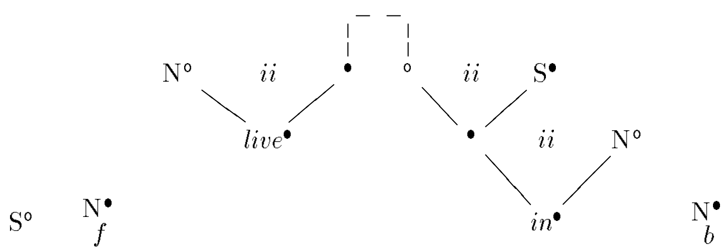
And the syntactic net:

(55)



For *Frodo inhabits Bag End* we have the syntactic frame (56) and hence the same syntactic net (55).

(56)



The semantic reading of (55) is ((*in* *b*) (*live* *f*)).