Tree Adjoining Grammars in a Fragment of the Lambek Calculus

V. Michele Abrusci*
Universitá di Bari

Christophe Fouqueré†
Université Paris-Nord

Jacqueline Vauzeilles‡
Université Paris-Nord

This paper presents a logical formalization of Tree Adjoining Grammar (TAG). TAG deals with lexicalized trees and two operations are available: substitution and adjunction. Adjunction is generally presented as an insertion of one tree inside another, surrounding the subtree at the adjunction node. This seems to contradict standard logical ability. We prove that some logical formalisms, namely a fragment of the Lambek calculus, can handle adjunction.

We represent objects and operations of the TAG formalism in four steps: first trees (initial or derived) and the way they are constituted, then the operations (substitution and adjunction), and finally the elementary trees, i.e., the grammar. Trees (initial or derived) are obtained as the closure of the calculus under two rules that mimic the grammatical ones. We then prove the equivalence between the language generated by a TAG grammar and the closure under substitution and adjunction of its logical representation. Besides this nice property, we relate parse trees to logical proofs, and to their geometric representation: proofnets. *We briefly present them and give examples of parse trees as proofnets. This process can be interpreted as an assembling of blocks (proofnets corresponding to elementary trees of the grammar).*

1. Introduction

This paper presents a logical formalization of Tree Adjoining Grammar (TAG) (Joshi, Levy and Takahashi 1975). TAG deals with lexicalized trees and two operations are available: substitution and adjunction. A set of (elementary) trees is associated to each lexical item. TAG is a tree-rewriting system: the derivation process consists in applying operations to trees in order to obtain a (derived) tree whose sequence of leaves is a sentence. Adjunction increases the expressive power of the formalism in such a way that noncontext-free languages can be represented although the parse process is done in polynomial time. Adjunction is generally presented as an insertion of one tree inside another, surrounding the subtree at the adjunction node. This seems to contradict standard logic, but we show (in Section 4) that some logical formalisms, namely a fragment of the Lambek calculus (LC, first introduced by Lambek [1958]), can handle adjunction.

We represent objects and operations of the TAG formalism in four steps: first trees (initial or derived) and the way they are constituted, then the operations (substitution and adjunction), and finally the elementary trees, i.e., the grammar. Labels occurring

* CILA, 70121 Bari, Italy. E-mail: abrusci@caspur.it
† LIPN-CNRS URA 1507, 93430 Villetaneuse, France. E-mail: cf@lipn.univ-paris13.fr
‡ LIPN-CNRS URA 1507, 93430 Villetaneuse, France. E-mail: jv@lipn.univ-paris13.fr

in the grammar constitute the set of propositional variables we need. The sequent calculus is a restriction of the standard sequent calculus for LC: there are identity axioms ($A \vdash A$) and rules for introducing connectives ($\otimes$ at left-hand side, $\multimapinv$ at right-hand side). In LC, / is usually used for $\multimapinv$ and $\cdot$ for $\otimes$. We use this notation throughout the paper to relate our formalization to noncommutative linear logic. $\multimapinv$ is the left implication, $\otimes$ is a noncommutative "and" connective. We prove that this restricted calculus is closed under two rules that mimic the grammatical operations. Trees (initial or derived) are then obtained as the closure of the calculus under these two rules. In fact, trees are represented as (provable) sequents in an almost classical way. The right-hand side is the variable labeling the mother node of the tree. The left-hand side is a sequence of formulas of the following kinds: A for some leaf A of the tree, $A \multimapinv B_1 \otimes \ldots \otimes B_n$ where A is the label of some internal node and B_i are the labels of its daughters, $A \multimapinv A$ whenever A is a node where an adjunction can take place. This latter kind of formula can be grammatically interpreted as if such an A was split up into two nodes with the same label, linked by some "soft" relation. The set of elementary trees of a TAG grammar $\mathcal{G}'$ is then represented as a subset M of the sequents in the closure of the calculus under the two previous rules. We then prove the equivalence between the language generated in TAG by such a grammar $\mathcal{G}'$ and the closure under substitution and adjunction of the logical representation M. Note that our interpretation of adjunction is very close to the use of quasi trees described in Vijay-Shanker, (1992).

Besides this equivalence property, we relate parse trees to logical proofs, and to their geometric representation, **proofnets**. We briefly present proofnets, and the correspondence between proofs and proofnets, and give examples of parse trees viewed as proofnets. This enables a new point of view on the parse process. This process can be interpreted as an assembling of blocks (proofnets corresponding to elementary trees of the grammar), and also as a circulation of information through links relating nodes of the proofnets.

The remainder of the paper is organized in four parts. Section 2 describes the TAG formalism. We recall the terminology and show how substitution and adjunction operate on trees. Section 3 gives a survey of Lambek calculus viewed as a fragment of a noncommutative linear logic. We propose in Section 4 a logical formulation of TAG in a fragment of LC, and prove the correspondence between the two. Section 5 is devoted to the representation of proofs as proofnets; in this final section, we also study implications of this point of view. The proofs of propositions and theorems given in Section 4 are delayed to the appendix for the sake of clarity.

2. Tree Adjoining Grammars

The Tree Adjoining Grammar formalism is a tree-generating formalism introduced in Joshi, Levy, and Takahashi (1975), linguistically motivated (see, for example, Abeillé et al. [1990] and Kroch and Joshi [1985]), and with formal properties studied in Vijay-Shanker and Joshi (1985) and Vijay-Shanker and Weir (1994a, 1994b). A TAG is defined by two finite sets of trees composed by means of the substitution and adjunction operations.[1]

1 Originally, there was no need for a substitution operation, as initial trees were always rooted at S, thus labeling a sentence. In the Lexicalized-TAG formalism, this constraint disappears in favor of the substitution operation. Throughout the paper, we will use TAG to refer to the Lexicalized-TAG formalism.

Definition
A TAG G is a 5-tuple (V_N, V_T, S, I, A) where

- V_N is a finite set of nonterminal symbols,
- V_T is a finite set of terminal symbols,
- S is a distinguished nonterminal symbol, the **start** symbol,
- I is a set of **initial** trees,
- A is a set of **auxiliary** trees.

An elementary tree is either an initial tree or an auxiliary tree. Initial as well as auxiliary trees are trees with at least one leaf labeled by a terminal node (the grammar is a so-called lexicalized one). An auxiliary tree must furthermore have a leaf (the **foot** node, marked with a star $\star$) with the same label as the root node. Each nonterminal node is marked as adjoinable or nonadjoinable (in this case, the node is marked NA). Each internal node must obviously be labeled by a nonterminal node.[2] A derived tree is either an initial tree or a tree obtained from derived trees by means of the two available operations.

We add two constraints on TAG grammars: a node X cannot have a unique daughter labeled X, i.e., $\begin{array}{c}X\\ |\\ X\end{array}$ cannot be part of a tree. This condition is in no way an important constraint, as a grammar may always be transformed to conform to the constraint by substituting a unique node X for the partial tree. However, our logical representation makes use of a trick based on such trees: we replace nodes marked adjoinable by such partial trees (there is no mark at all in our logical representation). We also suppose that the type of each tree is unambiguous: an initial tree has no leaf with the same label as the root node, an auxiliary tree has only one leaf with the same label as the root node.

To conform with the literature, we will use α to refer to an initial tree, β to refer to an auxiliary tree, and γ to refer to some derived tree. Examples of initial and auxiliary trees are given in Figure 1. Two TAGs are defined: $G_1 = (\{S\}, \{a, b, c, d, \epsilon\}, S, \{\alpha_1\}, \{\beta_1\})$ (ϵ is the empty word) and $G_2 = (\{S, VP, NP, N\}, \{the, man, walks\}, S, \{\alpha_2, \alpha_3, \alpha_4\}, \emptyset)$.

The substitution operation is defined as usual. A nonterminal leaf of a tree may be expanded with a tree whose root node has the same label. We follow a conventional notation: leaves that accept substitution are marked with a down arrow $\downarrow$. This is not to be interpreted as a restriction on substitution, but only as a visual indication of what remains to be substituted to get a complete sentence. The adjunction operation is a little bit more complicated. It supposes a derived tree with a nonterminal node, say X, possibly internal and not marked NA, and an auxiliary tree with root node X. The operation consists in:

- excising the subtree with root labeled X in the derived tree,
- inserting the auxiliary tree at node labeled X in the derived tree,

2 In some versions, nonterminal nodes of elementary trees are labeled by a *set* of (auxiliary) trees that can be adjoined at this node. In case of the empty set, the node is obviously nonadjoinable. For the sake of clarity, we simplify the definition to only take into account the Boolean adjoinable property.

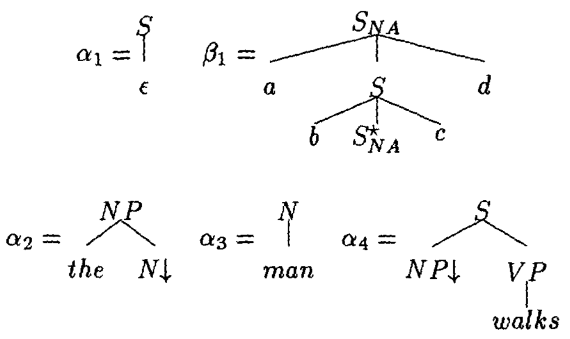


Figure 1
Elementary trees.

- finally, inserting the excised subtree at the foot node (hence labeled X and marked with a star $\star$) in the auxiliary tree.

Examples of these operations are given in Figure 2. To clearly show the adjunction operation, the links of the adjoined tree β_1 are represented by dashed lines in the derived trees γ_3 and γ_4. Obviously, there is only one kind of link. We write $\gamma_1 \Rightarrow_G \gamma_2$ when γ_2 is the result of an adjunction or a substitution of an elementary tree of a TAG G on the derived tree γ_1; $\Rightarrow^*_G$ is the reflexive, transitive closure of $\Rightarrow_G$. The set $\{\gamma / \exists \alpha \in G \text{ and } \alpha \Rightarrow^*_G \gamma\}$ is represented by $T(G)$. The language $L(G)$ generated by a TAG G is the set of strings, i.e., sequences of leaves of trees in $T(G)$ when the leaves of these trees are only labeled with terminal nodes, and whose root is the start symbol. Hence, $L(G_1) = \{a^n b^n c^n d^n / n \geq 0\}$ and $L(G_2) = \{\textit{the man walks}\}$.

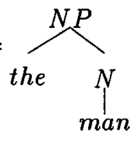
3. Lambek Calculus and Noncommutative Linear Logic

Lambek calculus is well known; we give only the language and sequent calculus in Figure 3.

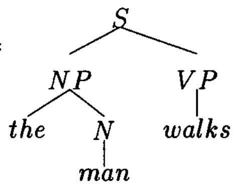
Lambek calculus will be sufficient to formalize TAG (see the next section). In Figures 4, 5, and 6, we give three examples of proofs to show how the sequent calculus can be used. The first one (Figure 4) is a straightforward use of a Lambek-style parsing, given the two implications and a set of proper axioms corresponding to the words. The two other proofs do not use proper axioms at all: rules labeled *lex* are provable sequents; as these sequents are obviously provable we omit their proof tree. The second proof (Figure 5) is in the same spirit as the first. However, for this second proof, descriptions of lexical items are included in the sequents. At the same time, it can easily be compared to the third proof: in the second proof, the structural information is located at the head of each structure as one formula; in the third proof, one formula represents a syntactic tree of level 1. The third proof (Figure 6) interprets the Lambek grammar in a derivation style, we only need one implication $\multimap$ and the connective times $\otimes$. The proofs use cuts: they can be withdrawn using the cut elimination theorem, but we think the cuts help in understanding the process. The following sections include other examples and emphasize the usefulness of noncommutative linear logic in the linguistic domain.

A natural way to extend Lambek calculus consists in embedding it in a classical system, in the sense that the connectives "and" and "or" are dual. Indeed, LC is an "intuitionistic" system as there can be only one conclusion in the sequents, this is not

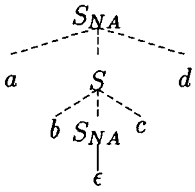
Substitution of α_3 on α_2 gives γ_1 =



Substitution of γ_1 on α_4 gives γ_2 =



Adjunction of β_1 on α_1 gives γ_3 =



Adjunction of β_1 on γ_3 gives γ_4 =

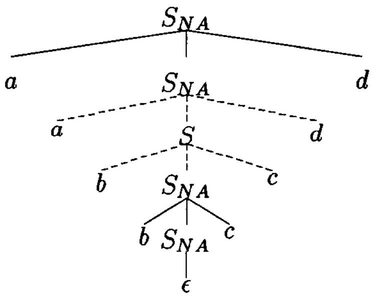


Figure 2
Substitution and adjunction results.

the case with noncommutative linear logic. Allowing multiple conclusions may give valuable benefits from a linguistic point of view, but we will only consider in this paper the geometrical representation available for such a system, i.e., proofnets. In the appendix, we give a brief description of linear logic, and the relations between classical, linear, and noncommutative linear logics. We hope this will help readers to understand the overall framework.

4. The Calculus $\mathcal{A}$ (A Fragment of LC)

The formalization of TAG in LC relies mainly on a logical presentation of the two operations substitution and adjunction, together with a correspondence between proofs and trees. As already shown in the previous section, the substitution operation is nothing but the application of the cut rule restricted to atomic formulas, which we call the **atomic cut rule**. Interpreting the adjunction operation is really the main difficulty. The adjunction results from two atomic cut rules between the sequent corresponding to the adjunction tree and two suitable sequents corresponding to two subparts of the

$$\frac{}{A \vdash A}\ (axiom) \qquad \frac{\Gamma \vdash A \quad \Gamma_1, A, \Gamma_2 \vdash B}{\Gamma_1, \Gamma, \Gamma_2 \vdash B}\ (cut)$$

$$\frac{\Gamma \vdash A \quad \Delta \vdash B}{\Gamma, \Delta \vdash A \otimes B}\ (r - \otimes) \qquad \frac{\Gamma_1, A, B, \Gamma_2 \vdash C}{\Gamma_1, A \otimes B, \Gamma_2 \vdash C}\ (l - \otimes)$$

$$\frac{\Gamma_1, A, \Gamma_2 \vdash C \quad \Delta \vdash B}{\Gamma_1, A \circ\!- B, \Delta, \Gamma_2 \vdash C}\ (l\!-\circ\!-) \qquad \frac{\Gamma, B \vdash A}{\Gamma \vdash A \circ\!- B}\ (r\!-\circ\!-)$$

$$\frac{\Gamma_1, A, \Gamma_2 \vdash C \quad \Delta \vdash B}{\Gamma_1, \Delta, B \multimap A, \Gamma_2 \vdash C}\ (l\!-\multimap) \qquad \frac{B, \Gamma \vdash A}{\Gamma \vdash B \multimap A}\ (r\!-\multimap)$$

Figure 3
Language and sequent calculus for the Lambek calculus.

John	$\vdash$	NP
gives	$\vdash$	((NP$\multimap$ S)$\circ\!-$NP)$\circ\!-$NP
Mary	$\vdash$	NP
a	$\vdash$	NP$\circ\!-$N
book	$\vdash$	N

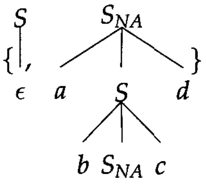
$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{NP \vdash NP \quad S \vdash S}{NP, NP \multimap S \vdash S} \quad NP \vdash NP}{NP, (NP \multimap S) \circ\!- NP, NP \vdash S} \quad NP \vdash NP}{NP, ((NP \multimap S) \circ\!- NP) \circ\!- NP, NP, NP \vdash S} \quad N \vdash N}{NP, ((NP \multimap S) \circ\!- NP) \circ\!- NP, NP, (NP \circ\!- N), N \vdash S}}{\dots\ \dots\ \dots\ \dots .\ \dots .}\ \text{(cuts wrt the}}{\text{John, gives, Mary, a, book} \vdash S}\ \text{proper axioms)}$$

Figure 4
Lexicon and proof of *John gives Mary a book*: (Lambek-style) with proper axioms.

$$\dfrac{\dfrac{}{NP \circ\!- John, John \vdash NP}\text{(lex)} \quad \dfrac{\dfrac{\dfrac{}{NP, (((NP \multimap S) \circ\!- NP) \circ\!- NP) \circ\!- gives, gives, NP, NP \vdash S}\text{(lex)} \quad \dfrac{}{NP \circ\!- Mary, Mary \vdash NP}\text{(lex)}}{NP, (((NP \multimap S) \circ\!- NP) \circ\!- NP) \circ\!- gives, gives, NP \circ\!- Mary, Mary, NP \vdash S}\text{(cut)} \quad \dfrac{\dfrac{}{(NP \circ\!- N) \circ\!- a, a, N \vdash NP}\text{(lex)} \quad \dfrac{}{N \circ\!- book, book \vdash N}\text{(lex)}}{(NP \circ\!- N) \circ\!- a, a, N \circ\!- book, book \vdash NP}}{NP, (((NP \multimap S) \circ\!- NP) \circ\!- NP) \circ\!- gives, gives, NP \circ\!- Mary, Mary, (NP \circ\!- N) \circ\!- a, a, N \circ\!- book, book \vdash S}\text{(cut)}}{NP \circ\!- John, John, (((NP \multimap S) \circ\!- NP) \circ\!- NP) \circ\!- gives, gives, NP \circ\!- Mary, Mary, (NP \circ\!- N) \circ\!- a, a, N \circ\!- book, book \vdash S}\text{(cut)}$$

Figure 5
Proof of *John gives Mary a book*: (Lambek-style) two implications.

tree where adjunction is done. Consider, for example, the following TAG grammar:

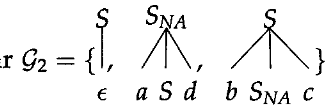
Grammar $\mathcal{G}_2' = \{$  $\}$

$$\dfrac{\dfrac{}{NP \circ\!- John, John \vdash NP}\text{(lex)} \quad \dfrac{\dfrac{\dfrac{}{S \circ\!- NP \otimes VP, NP, VP \circ\!- V \otimes NP \otimes NP, V \circ\!- gives, gives, NP, NP \vdash S}\text{(lex)} \quad \dfrac{}{NP \circ\!- Mary, Mary \vdash NP}\text{(lex)}}{S \circ\!- NP \otimes VP, NP, VP \circ\!- V \otimes NP \otimes NP, V \circ\!- gives, gives, NP \circ\!- Mary, Mary, NP \vdash S}\text{(cut)} \quad \dfrac{\dfrac{}{NP \circ\!- Det \otimes N, Det \circ\!- a, a, N \vdash NP}\text{(lex)} \quad \dfrac{}{N \circ\!- book, book \vdash N}\text{(lex)}}{NP \circ\!- Det \otimes N, Det \circ\!- a, a, N \circ\!- book, book \vdash NP}\text{(cut)}}{S \circ\!- NP \otimes VP, NP, VP \circ\!- V \otimes NP \otimes NP, V \circ\!- gives, gives, NP \circ\!- Mary, Mary, NP \circ\!- Det \otimes N, Det \circ\!- a, a, N \circ\!- book, book \vdash S}\text{(cut)}}{S \circ\!- NP \otimes VP, NP \circ\!- John, John, VP \circ\!- V \otimes NP \otimes NP, V \circ\!- gives, gives, NP \circ\!- Mary, Mary, NP \circ\!- Det \otimes N, Det \circ\!- a, a, N \circ\!- book, book \vdash S}\text{(cut)}$$

Figure 6
Proof of *John gives Mary a book*: one implication and times.

This set of trees may be viewed as a subset of the closure $\mathcal{T}(\mathcal{G}_2)$ under substitution (possibly with the declaration of adjunction nodes) of the following set of trees of level 1:

Grammar $\mathcal{G}_2 =$



Note that the result of the adjunction of the second tree of $\mathcal{G}'_2$ on itself is exactly the result of substitutions on trees of $\mathcal{G}_2$. However it is obvious that trees resulting from substitution operations on $\mathcal{G}_2$ do not always correspond to results of adjunction operations on $\mathcal{G}'_2$.

We logically represent the set of trees $\mathcal{T}(\mathcal{G}_2)$ as (the set of provable theorems of) a calculus $\mathcal{A}(\mathcal{G}_2)$: the formulas are built with the alphabet $\{\epsilon, a, b, c, d, S\}$ and the set of connectives $\{\otimes, \circ\!-\}$, the sequent calculus consists of the axioms $s \vdash s$ and the rules (in both axioms and rules, s is a propositional letter):

$$\frac{\Gamma \vdash \epsilon \quad \Gamma_1, S, \Gamma_2 \vdash B}{\Gamma_1, S \circ\!- \epsilon, \Gamma, \Gamma_2 \vdash B} \qquad \frac{\Gamma \vdash a \otimes S \otimes d \quad \Gamma_1, S, \Gamma_2 \vdash B}{\Gamma_1, S \circ\!- a \otimes S \otimes d, \Gamma, \Gamma_2 \vdash B} \qquad \frac{\Gamma \vdash b \otimes S \otimes c \quad \Gamma_1, S, \Gamma_2 \vdash B}{\Gamma_1, S \circ\!- b \otimes S \otimes c, \Gamma, \Gamma_2 \vdash B}$$

$$\frac{\Gamma \vdash A \quad \Delta \vdash B}{\Gamma, \Delta \vdash A \otimes B}\,(\otimes) \qquad \frac{s \vdash s \quad \Gamma_1, s, \Gamma_2 \vdash B}{\Gamma_1, s \circ\!- s, s, \Gamma_2 \vdash B}$$

The introduction of a left implication ($\circ\!-$) corresponds to the building of a partial tree. Such introductions are then restricted either to the formalization of the trees of the grammar (the first three rules correspond exactly to the trees of $\mathcal{G}_2$), or to the formalization of adjunction nodes (the formula $s \circ\!- s$ "marks" s as being an adjunction node, i.e., the adjunction rule may be applied only on this kind of node as it will be clear below).

The grammar $\mathcal{G}'_2$ can then be logically represented as a subset $M(\mathcal{G}'_2)$ of the set of provable sequents of the calculus $\mathcal{A}(\mathcal{G}_2)$:

$$M(\mathcal{G}'_2) = \{S \circ\!- a \otimes S \otimes d, a, S \circ\!- S, S \circ\!- b \otimes S \otimes c, b, S, c, d \vdash S, S \circ\!- S, S \circ\!- \epsilon, \epsilon \vdash S\}$$

In AB-grammars (Bar-Hillel 1953), only one implication is used without any "and" connective. The grammar would be represented in AB-grammars as two provable sequents (note that "daughters of a node" are explicitly ordered):

$$((S \circ\!- d) \circ\!- S) \circ\!- a, a, S \circ\!- S, ((S \circ\!- c) \circ\!- S) \circ\!- b, b, S, c, d \vdash S, \quad S \circ\!- S, S \circ\!- \epsilon, \epsilon \vdash S$$

We will prove later that, besides the cut rule, there exists another derived rule for the calculus $\mathcal{A}(\mathcal{G}_2)$ (and in fact for each calculus of this kind) mimicking the adjunction operation. Reducing the calculus, then, to a closure of the substitution and adjunction

$$\begin{tikzcd}
 & \mathcal{A}(\mathcal{G}) \arrow[loop above, "\begin{matrix}\text{cut}\\ \text{adj}\end{matrix}"] \arrow[dd, shift right, "\text{Tree}"'] \arrow[r, phantom, "\supset"] & \mathrm{M}(\mathcal{G}') \arrow[r, "\begin{matrix}\text{cut}\\ \text{adj}\end{matrix}"] & \mathrm{CL}(\mathrm{M}(\mathcal{G}')) \\
\mathcal{G} \arrow[ur, "\begin{matrix}\circ\!\!-\\ \otimes\end{matrix}"] \arrow[dr, "\begin{matrix}\text{NA}\\ \text{subst}\end{matrix}"'] & & & \\
 & \mathcal{T}(\mathcal{G}) \arrow[uu, shift right, "\text{Seq}"'] \arrow[r, phantom, "\supset"] & \mathcal{G}' \arrow[uu] \arrow[r, "\begin{matrix}\text{subst}\\ \text{adj}\end{matrix}"'] & \mathrm{CL}(\mathcal{G}') \arrow[uu]
\end{tikzcd}$$

Figure 7
Summary of the logical interpretation of the TAG formalism.

rules on $M(\mathcal{G}'_2)$, we get exactly the logical representations of the set of trees under the TAG grammar $\mathcal{G}'_2$.

The adjunction rule must be logically justified: there must be only one way to combine the pieces (i.e., provable sequents corresponding to trees of level 1), given the substitution node, such that the order of the elements is as requested.

To prove this, we show that for a suitable fragment of LC there is a *unique* way to decompose a sequent $\Gamma, a \mathrel{\circ\!\!-} A, \Delta \vdash B$ in two sequents $\Gamma, a, \Delta_2 \vdash B$ and $\Delta_1 \vdash A$, where $\Delta = \Delta_1, \Delta_2$. In this section, we clarify the calculus $\mathcal{A}$ used to interpret TAG: it includes a cut rule and an adjunction rule that mimic the grammatical operations. As pointed out previously, these two rules are correct with respect to logic. We give the basic properties satisfied by this calculus $\mathcal{A}$. In order to represent TAG in LC, we first construct the set $\mathcal{G}$ of subtrees of depth 1 of trees appearing in a TAG grammar $\mathcal{G}'$. The TAG grammar $\mathcal{G}'$ is then a subset of the closure $\mathcal{T}(\mathcal{G})$ of the set $\mathcal{G}$ under substitution (indicated by *subst*) and the declaration of nodes where adjunction is not available (indicated by *NA*). The interpretation of elements of $\mathcal{G}$ as provable sequents of $\mathcal{A}$ is straightforward. This leads to a calculus $\mathcal{A}(\mathcal{G})$ where the operations are restricted with respect to $\mathcal{G}$. The TAG grammar $\mathcal{G}'$ is then in correspondence with a subset $\mathrm{M}(\mathcal{G}')$ of $\mathcal{A}(\mathcal{G})$ and we prove the equivalence between the language $\mathrm{CL}(\mathcal{G}')$ generated by $\mathcal{G}'$ and the set of sequents $\mathrm{CL}(\mathrm{M}(\mathcal{G}'))$ obtained by closure on $\mathrm{M}(\mathcal{G}')$ by the cut and adjunction rules (we use M instead of $\mathrm{M}(\mathcal{G}')$ whenever there is no ambiguity). Proofs of propositions are delayed to the appendix. The various components of our approach are summarized in Figure 7.

Consider the following fragment $\mathcal{A}$ of LC:

Definition The Calculus $\mathcal{A}$

- Alphabet of $\mathcal{A}$: propositional letters $a, b, \ldots$, connectives $\otimes, \mathrel{\circ\!\!-}$.
- Formulas: usual definition. A is a simple $\otimes$-formula iff A is a propositional letter or A is a formula $b_1 \otimes \ldots \otimes b_n$ where $b_1, \ldots, b_n$ are propositional letters. B is a $\mathrel{\circ\!\!-}$-formula iff $B = a \mathrel{\circ\!\!-} A$ where a is a propositional letter and A is a simple $\otimes$-formula.

- Sequents: $\Gamma \vdash A$, where Γ is a finite sequence of formulas and A is a formula.
- Sequent calculus:

  — Axiom: $a \vdash a$

  — Rules: $\dfrac{\Gamma \vdash A \quad \Delta \vdash B}{\Gamma, \Delta \vdash A \otimes B}\ (\otimes)$ $\qquad \dfrac{\Gamma \vdash A \quad \Gamma_1, C, \Gamma_2 \vdash B}{\Gamma_1, C \multimapinv A, \Gamma, \Gamma_2 \vdash B}\ (\multimapinv)$

In the following, we only consider sequents such that formulas in the left side are either propositional letters, or $\multimapinv$-formulas. So, in the rule introducing $\multimapinv$, C stands for a propositional letter. As we have only one propositional letter before $\multimapinv$, we model trees: C is the (unique) mother and the $\otimes$-formula A is the sequence of its daughters.

Proposition Main properties of calculus $\mathcal{A}$
(proofs in the appendix)

1. If $\Gamma \vdash A \otimes B$ is provable in $\mathcal{A}$, then
   - A and B are simple $\otimes$-formulas;
   - there is a unique pair (Γ_1, Γ_2) s.t. $\Gamma = \Gamma_1, \Gamma_2$ and both the sequents $\Gamma_1 \vdash A$ and $\Gamma_2 \vdash B$ are provable in $\mathcal{A}$.
2. If $\Gamma, a \multimapinv A, \Delta \vdash B$ is provable in $\mathcal{A}$, then
   - A and B are simple $\otimes$-formulas;
   - there is a unique pair (Δ_1, Δ_2) s.t. $\Delta = \Delta_1, \Delta_2$ and both the sequents $\Delta_1 \vdash A$ and $\Gamma, a, \Delta_2 \vdash B$ are provable in $\mathcal{A}$.

   Such a pair (Δ_1, Δ_2) will be called "the splitting pair in $\Gamma, a \multimapinv A, \Delta \vdash B$ for Δ." Note that this pair can be computed easily: the first element Δ_1 of the splitting pair must satisfy a counting condition on each propositional letter occurring in it (see the appendix).
3. The calculus $\mathcal{A}$ is closed under the atomic cut rule (which we also call the substitution rule)
   $$\frac{\Gamma \vdash a \quad \Delta_1, a, \Delta_2 \vdash A}{\Delta_1, \Gamma, \Delta_2 \vdash A}\ (cut)$$
   i.e., if the sequents $\Gamma \vdash a$ and $\Delta_1, a, \Delta_2 \vdash A$ are provable in $\mathcal{A}$, then the sequent $\Delta_1, \Gamma, \Delta_2 \vdash A$ is also provable in $\mathcal{A}$.
4. The calculus $\mathcal{A}$ is closed under the adjoining rule
   $$\frac{\Gamma_1, a, \Gamma_2 \vdash a \quad \Delta, a \multimapinv a, \Lambda \vdash b}{\Delta, \Gamma_1, \Lambda_1, \Gamma_2, \Lambda_2 \vdash b}\ (adj)$$
   where (Λ_1, Λ_2) is the splitting pair of Λ in $\Delta, a \multimapinv a, \Lambda \vdash b$.

Note that Λ_1 and Λ_2 are *uniquely defined* from the premises, so the previous deduction is really a logical rule.

Definition The Calculus $\mathcal{A}(\mathcal{G})$

Let $\mathcal{G}$ be a family of labeled trees, of depth 1, not of the form $\begin{array}{c} X \\ | \\ X \end{array}$. Let $\mathcal{T}(\mathcal{G})$ be the closure of $\mathcal{G}$ under the rules:

- substitution with or without the declaration of a new possibly internal point on which the adjoining operation may be performed,
- adjoining operation.

$\mathcal{A}(\mathcal{G})$ is the calculus obtained from $\mathcal{A}$ as follows:

- propositional letters are exactly all the labels of the trees in $\mathcal{G}$,
- the rule $(\multimapinv)$ is restricted as follows:

  $$\frac{\Gamma \vdash A \quad \Gamma_1, a, \Gamma_2 \vdash B}{\Gamma_1, a \multimapinv A, \Gamma, \Gamma_2 \vdash B}\ (\multimapinv, \mathcal{G})$$

  where A, B are simple $\otimes$-formulas of the language of $\mathcal{A}(\mathcal{G})$, a is a propositional letter of the language of $\mathcal{A}(\mathcal{G})$ and one of the following cases occurs:

  — A is a

  — A is a propositional letter b different from a, and the tree $\begin{array}{c} a \\ | \\ b \end{array} \in \mathcal{G}$

  — A is $b_1 \otimes \ldots \otimes b_n$, and the tree $\begin{array}{c} a \\ \diagup \ | \ \diagdown \\ b_1 \ \ldots \ b_n \end{array} \in \mathcal{G}$

The following propositions state the correspondence between sequents and trees. The first two provide a precise translation between the two notions. Basically, a sequent $\Gamma \vdash a$ (in the previous language) is the logical equivalent of a tree with root a, and there is exactly one formula in Γ for each leaf, for each subtree (of depth 1), for each adjunction node, and nothing else. *Seq()* (respectively, *Tree()*) associates a sequent (respectively, a tree) to each tree (respectively, each sequent), and we prove the two are converse. The last three propositions are properties concerning the logical counterpart of a TAG grammar. The last one is in fact the most important: the closure under (logical) adjunction and substitution of the set of sequents corresponding to a set of elementary trees is exactly the set of sequents corresponding to the closure under (grammatical) adjunction and substitution of this set of elementary trees. In other words, the logical calculus (the restricted logical calculus we defined above) and the grammatical calculus (the TAG calculus) coincide.

Proposition Main properties of calculus $\mathcal{A}(\mathcal{G})$
(proofs in the appendix)

Properties 1–4 of $\mathcal{A}$ are also properties of $\mathcal{A}(\mathcal{G})$. Moreover the following properties hold for $\mathcal{A}(\mathcal{G})$:

- To $T \in \mathcal{T}(\mathcal{G})$, we associate a sequent *Seq(T)* of $\mathcal{A}(\mathcal{G})$ s.t.

  — if a is the root of T, and the terminal points of T (ordered from left to right) are $a_1, \ldots, a_m$, then *Seq(T)* is

  $$\Gamma \vdash a$$

where the sequence of all the propositional variables occurring in Γ is $a_1, \ldots, a_m$ and there is a formula $c \multimap c$ in Γ iff c is a possibly internal point of T on which the adjoining operation may be performed;

— *Seq(T)* is provable in $\mathcal{A}(\mathcal{G})$.

- To every provable sequent $\Gamma \vdash A$ in $\mathcal{A}(\mathcal{G})$, we associate *Tree*($\Gamma \vdash A$) s.t.

— if A is a propositional letter, then *Tree*($\Gamma \vdash A$) $\in \mathcal{T}(\mathcal{G})$ where the root is A, the terminal points (from left to right) are exactly all the propositional letters occurring in Γ and in the same order in which they occur in Γ, and the possibly internal points on which the adjoining operation may be performed are exactly all the propositional letters c s.t. $c \multimap c$ occurs in Γ;

— if A is $b_1 \otimes \ldots \otimes b_n$, and so $\Gamma = \Gamma_1, \ldots, \Gamma_n$ with the sequents $\Gamma_i \vdash b_i$ provable in $\mathcal{A}(\mathcal{G})$ for every $1 \leq i \leq n$, then *Tree*($\Gamma \vdash A$) is a sequence $T_1, \ldots, T_n$ of trees $\in \mathcal{T}(\mathcal{G})$, s.t. $T_i =$ *Tree*($\Gamma_i \vdash b_i$).

- If $\Gamma \vdash a$ is provable in $\mathcal{A}(\mathcal{G})$, then *Seq(Tree*($\Gamma \vdash a$)) $= \Gamma \vdash a$. If T is a tree of $\mathcal{G}$, then *Tree(Seq(T))* $= T$.

- Let M be a set of provable sequents in $\mathcal{A}(\mathcal{G})$. Define CL(M) as follows:

— M$\subseteq$CL(M)

— (closure under atomic cut rule) if $\Gamma \vdash a \in$ CL(M) and $\Delta_1, a, \Delta_2 \vdash B \in$CL(M), then $\Delta_1, \Gamma, \Delta_2 \vdash B \in$CL(M)

— (closure under adjoining operation) if $\Gamma_1, a, \Gamma_2 \vdash a \in$CL(M) and $\Delta, a \multimap a, \Lambda \vdash b \in$CL(M), then $\Delta, \Gamma_1, \Lambda_1, \Gamma_2, \Lambda_2 \vdash b \in$CL(M), where (Λ_1, Λ_2) is the splitting pair of Λ in $\Delta, a, \Lambda \vdash b$

— nothing else belongs to CL(M).

- If $\Gamma \vdash A \in$CL(M), then $\Gamma \vdash A$ is provable in $\mathcal{A}(\mathcal{G})$.

- If $\mathcal{G}' \subseteq \mathcal{T}(\mathcal{G}$, let CL($\mathcal{G}'$) be the closure of $\mathcal{G}'$ under:

— substitution,

— adjoining operation.

Clearly, CL($\mathcal{G}'$) $\subseteq \mathcal{T}(\mathcal{G})$. Let $M = \{Seq(T)/T \in \mathcal{G}'\}$, then $CL(M) = \{Seq(T)/T \in CL(\mathcal{G}')\}$.

Starting from this last proposition, it is possible to prove that the language accepted by a TAG grammar $\mathcal{G}'$ is exactly the language accepted by M($\mathcal{G}'$). We can define the language accepted by such a calculus as follows: Let us take only those sequents in CL(M($\mathcal{G}'$)) whose right part is the propositional variable S (the start symbol of the grammar), and such that propositional variables of the left part of the sequent correspond to terminal symbols of the grammar, i.e., words of the language. The language accepted by M($\mathcal{G}'$) is then the set of sequences of words in the same order as they appear in the previous sequents.

5. TAG Analysis Using Noncommutative Proofnets

A proof in sequent calculus contains many useless properties in its contexts. Girard (1987) has defined, in a purely geometric way, a class of graphs of formulas, called proofnets: for each proof of a sequent $\vdash \Gamma$ in the one-sided sequent calculus for multiplicative linear logic, there is a corresponding proofnet whose conclusions are exactly the formulas in Γ, and for each proofnet, there is at least one corresponding proof of the sequent $\vdash \Gamma$ in the one-sided sequent calculus for multiplicative linear logic (where Γ is a sequence of all the conclusions of the proofnet). Similarly, Abrusci (1991) defined in a purely geometric way a class of graphs, called **noncommutative proofnets,** relative to multiplicative noncommutative linear logic. Roorda (1992) also described proofnets for Lambek calculus. Other criteria exist by now for characterizing proofnets for commutative or noncommutative, intuitionistic or nonintuitionistic linear logic. We present here Abrusci's criteria.

5.1 Noncommutative Proofnets

Proofnets are defined on one-sided sequent calculi. Presentations of the one-sided sequent calculus, and of proofnets are given in the appendix. Let us recall that $\parr$ is the "or" connective associated to $\otimes$ (the "and" connective), such that $A \multimap B = A^{\perp} \parr B$. To every proof π of a sequent $\vdash \Gamma$ in the one-sided sequent calculus for multiplicative noncommutative linear logic, we can associate (by induction on the construction of the proof π) a **noncommutative proofnet with conclusions** Γ, i.e., an oriented planar graph π' of occurrencies of formulas such that:

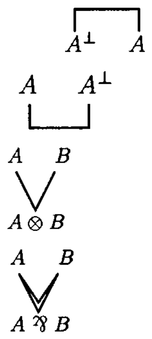
- The conclusions of π' are exactly the formulas in Γ.
- π' is a noncommutative proof structure, i.e., it is constructed by means of the following links[3]:

  — Axiom-link (two conclusions, no premise)

  — Cut-link (two premises, no conclusion)

  — $\otimes$-link (two premises, one conclusion)

  — $\parr$-link (two premises, one conclusion)



  and every occurrence of formula is a premise of at most one link and is a conclusion of exactly one link.

- The translation π' of π is a proofnet, i.e., it admits no **shorttrip**. A shorttrip is a trip that does not contain each node twice. A **trip** is a sequence of nodes, going from one node to another according to the graph and to a switch for each $\otimes$-link and each $\parr$-link, in a bideterministic way: the traversal of nodes is done according to Figure 8.
- Every assignment for π' is total: two integer variables are associated to each label (one for each "side" of the variable). Constraints are imposed on variables with respect to how trips are done throughout the net. The assignment is total if the set of constraints has a solution.

3 The $\parr$-link is graphically distinguished from the $\otimes$-link. However this is a moot point because the graph has only one kind of edge.

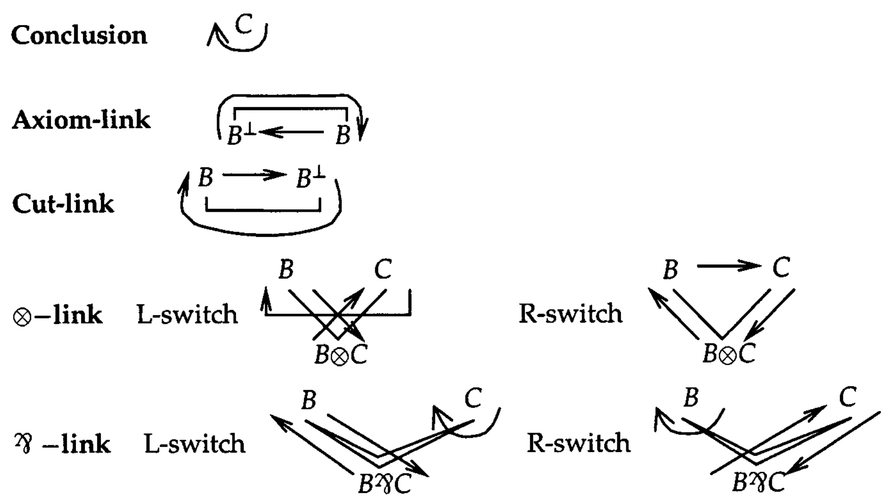


Figure 8
Travels through proof structures.

- π' induces the linear order Γ of the conclusions, i.e., iff the precedence relation is a chain and each conclusion occurs exactly once in the chain.

Precise definitions, examples, explanations and the proof of the following theorem may be found in Abrusci (1995).

Theorem
π' is a noncommutative proofnet with conclusions Γ iff there exists a proof π of the sequent $\vdash \Gamma$ in the sequent calculus for multiplicative noncommutative linear logic such that π' is associated to π.

Note that every noncommutative proofnet is a planar graph.

5.2 Parse Examples

In this section, we give two simple examples of parses. The aim of this section is to show the strong connection between the structure of proofs of sequents and a standard TAG derived structure. Moreover, it emphasizes the interest of a proofnet approach as the syntax (and parsing process) is concretely designed as a logical manipulation of logical structures. In the next section, we develop this approach and show how lexical rules can be integrated into it. Finally, we briefly mention that this can also give a logical formalization of D-trees (Vijay-Shanker 1992).

The first example requires only substitution, i.e., the cut rule in the logical point of view. We first give the sequents (provable in $\mathcal{A}$) associated to the lexical items. Their meanings are straightforward, e.g., "*John* and *Mary* are noun phrases (NP)" or "*saw* requires a complement NP to obtain a verb phrase (VP) and a subject NP to obtain a sentence (S)." Note that VP is an adjunction node so the sequent associated to the item *saw* includes the formula $VP \multimap VP$. The next example uses this specification.

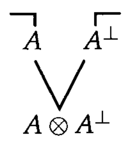
John	$NP \circ\!\!- John, John \vdash NP$
Mary	$NP \circ\!\!- Mary, Mary \vdash NP$
saw	$S \circ\!\!- NP \otimes VP, NP, VP \circ\!\!- VP, VP \circ\!\!- V \otimes NP, V \circ\!\!- saw, saw, NP \vdash S$

$$\dfrac{NP \multimapinv John, John \vdash NP \qquad \dfrac{S \multimapinv NP \otimes VP, NP, VP \multimapinv VP, VP \multimapinv V \otimes NP, V \multimapinv saw, saw, NP \vdash S \quad NP \multimapinv Mary, Mary \vdash NP}{S \multimapinv NP \otimes VP, NP, VP \multimapinv VP, VP \multimapinv V \otimes NP, V \multimapinv saw, saw, NP \multimapinv Mary, Mary \vdash S}\,(cut)}{S \multimapinv NP \otimes VP, NP \multimapinv John, John, VP \multimapinv VP, VP \multimapinv V \otimes NP, V \multimapinv saw, saw, NP \multimapinv Mary, Mary \vdash S}\,(cut)$$

$$\dfrac{NP \multimapinv Mary, Mary \vdash NP \qquad \dfrac{S \multimapinv NP \otimes VP, NP, VP \multimapinv VP, VP \multimapinv V \otimes NP, V \multimapinv saw, saw, NP \vdash S \quad NP \multimapinv John, John \vdash NP}{S \multimapinv NP \otimes VP, NP \multimapinv John, John, VP \multimapinv VP, VP \multimapinv V \otimes NP, V \multimapinv saw, saw, NP \vdash S}\,(cut)}{S \multimapinv NP \otimes VP, NP \multimapinv John, John, VP \multimapinv VP, VP \multimapinv V \otimes NP, V \multimapinv saw, saw, NP \multimapinv Mary, Mary \vdash S}\,(cut)$$

Figure 9
$\mathcal{A}(\mathcal{G})$ proofs of *John saw Mary*.

The proof associated to the analysis of *John saw Mary* requires two cuts. The two sequent proofs given in Figure 9 are the only two possibilities for this sentence in the fragment $\mathcal{A}(\mathcal{G})$. This pinpoints the fact that the order in which the cuts are done is not significant with respect to the derived structure. Proofnets allow the expression of this equivalence. Hence the two proofs have the same associated proofnet, given in Figure 10. For the sake of clarity, the cut rules are bold lines, and subnets associated to lexical items are circled. Obviously, if we delete the two cut lines, we are left with three proofnets referring to (provable) sequents. The proofnet in Figure 10 still contains some superfluous information, namely, nodes that cannot be targeted by the only available operations in $\mathcal{A}(\mathcal{G})$—the cut rule and the adjunction rule on a propositional variable. In fact, we only need to keep nodes (i) that refer to conclusions of the proofnet that are propositional variables or negation of propositional variables (a cut can be done on such a literal), and (ii) that belong to subgraphs of the following form (corresponding to the existence of a formula $A \multimapinv A$ in the left part of a sequent, i.e., its negation $A \otimes A^{\perp}$ in the one-sided associated sequent):



We can then simplify the graph and replace the internal logical machinery by black boxes (shown in the figures as solid black circles). The conclusions of each basic proofnet are labeled: outputs (i.e., conclusions that are propositional variables) are drawn as closed half circles, inputs (i.e., conclusions that are the negation of propositional variables) are drawn as open half circles. Plain lines link black boxes to black boxes or conclusions, and subgraphs corresponding to adjunction points are drawn as dashed lines. The previous proofnet is then redrawn as in Figure 11. We obviously find the derived tree (neglecting some minor differences). The logical proofnet can then be seen as an "explanation" of the structure of the tree, that is to say the operations available on the tree are the result of some focus of what can be done on the proofnet. On the one hand, the use of black boxes is necessary to clarify the structure of the analysis; on the other hand, this hides proof details that can be useful for some linguistic operations (as is the case for adjunction with respect to the classical structure of a derived tree). We show in the next subsection another application of such a (logical) refinement.

The last example discussed in this section is the analysis of the sentence *John saw Mary today*. The sequent associated to the adverb *today* is the following one:

today $\quad VP \multimapinv VP \otimes today, VP, today \vdash VP$

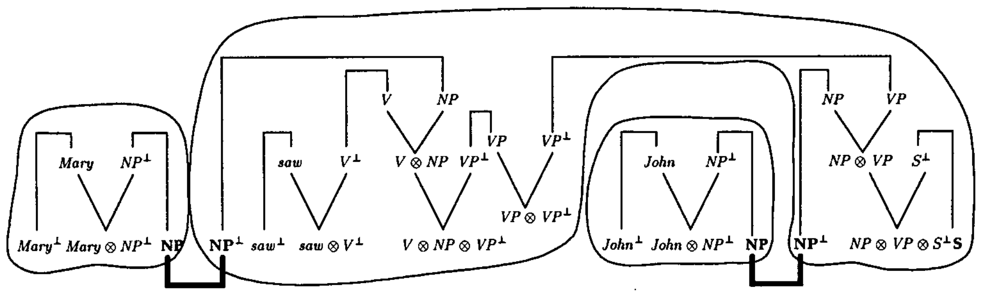


Figure 10
John saw Mary.

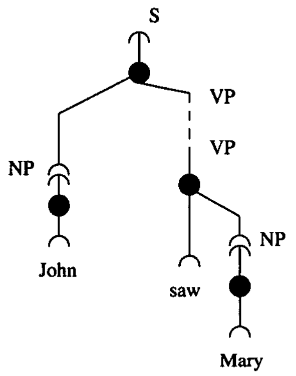


Figure 11
A simplified proof for *John saw Mary.*

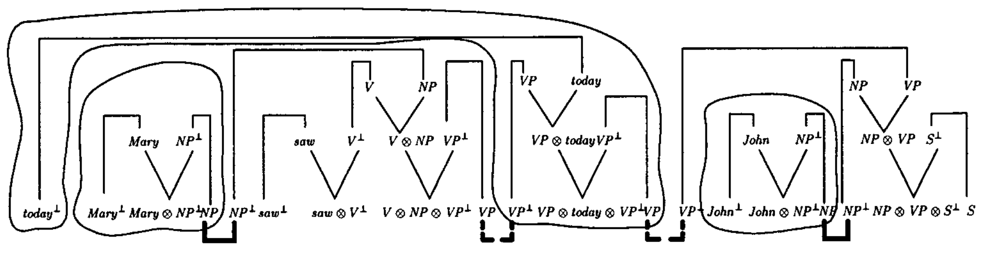


Figure 12
John saw Mary today.

The logical analysis includes the two operations substitution and adjunction, i.e., two cut rules and an adjunction rule. In Figure 12 the adjunction rule is shown as a double-thick dashed line: this (logically) mimics the adjunction as it is shown in the derived tree given in Figure 13. Note that the adverb has to be placed after the complement (rightmost in the proofnet) in order to keep the graph planar. The proofnet in Figure 14 is the proofnet corresponding to a cut-free proof.

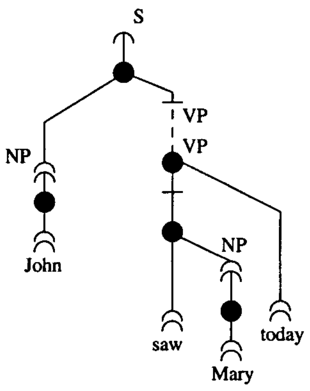


Figure 13
A simplified proof for *John saw Mary today.*

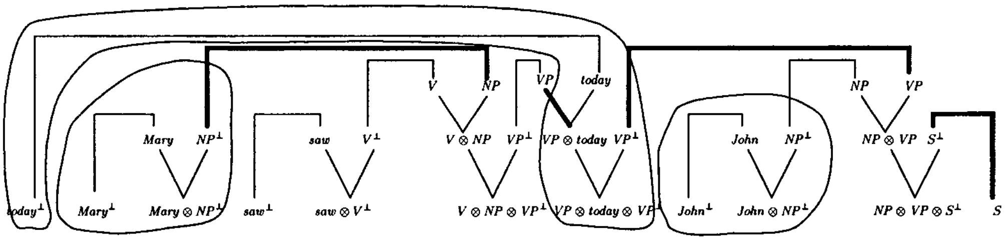


Figure 14
Cut-free proofnet for *John saw Mary today.*

5.3 On Some Extensions

As usual in lexicalized formalisms, TAG states rules to generate the lexicon from a basic set of descriptions. Among these, we find rules for passivization, interrogative forms or *wh*-sentences. We focus here on one example (namely *who*) to show to what extent the previous paradigm can be used also to logically interpret these lexical rules. We expect this will help in understanding the underlying mechanisms. The formulation we propose is the simplest one. This is also closely related to the approach used in categorial grammars (the raising rule is simply the introduction of an implication; see also Joshi and Kulick [1995] for such a relation and the way *who* can be defined). Figures 15, 16, and 17 present proofnets and simplified proofnets for the two noun adjuncts *who John meets* and *who meets John.* The analysis of complete sentences including these adjuncts is then similar to the process developed in the previous section. The corresponding (provable) sequents are given below. The basic lexical descriptions are the following (we have deleted the adjunction declarations for sake of clarity; the (logical) adjunction rule has to be slightly extended in order to take care of these new structures):

John	$NP \multimap John, John \vdash NP$
meets	$S \multimap NP \otimes VP, NP, VP \multimap V \otimes NP, V \multimap meets, meets, NP \vdash S$
who	$N \multimap N \otimes who \otimes (S \multimap NP), N, who, S \multimap NP \vdash N$

Let M($\mathcal{G}_1'$) denote the set of the three previous sequents. From these basic descriptions, the following entries are computed, i.e., the part of the lexicon relevant to these words

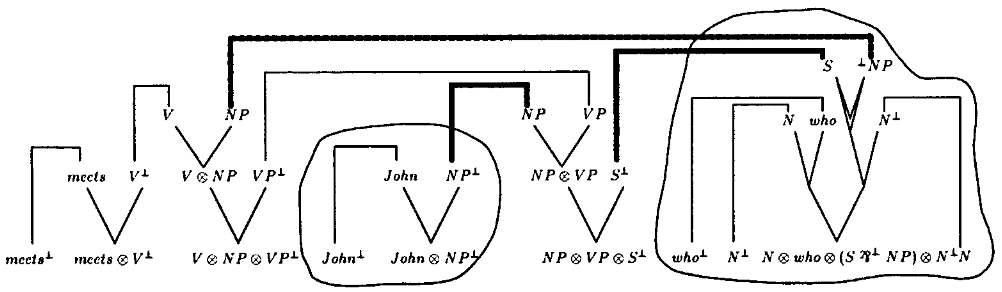


Figure 15
Cut-free proofnet for *who John meets.*

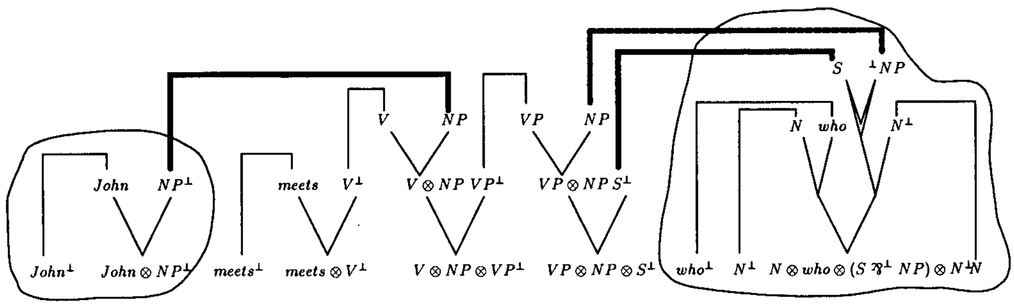


Figure 16
Cut-free proofnet for *who meets John.*

(M($\mathcal{G}_2'$) denotes this new set):

John	$NP \multimap John, John \vdash NP$
meets	$S \multimap NP \otimes VP, NP, VP \multimap V \otimes NP, V \multimap meets, meets, NP \vdash S$
who meets _	$N \multimap N \otimes who \otimes (S \multimap NP), N, who, (S \multimap NP) \multimap VP, VP \multimap V \otimes NP, V \multimap meets, meets, NP \vdash N$
who _ meets	$N \multimap N \otimes who \otimes (S \multimap NP), N, who, S \multimap NP \otimes VP, NP, VP \multimap V \otimes NP, V \multimap meets, meets \vdash N$

It should be noted that the two sequents given below are provable in the calculus M($\mathcal{G}_2'$) (cut and adjunction rules only).

who meets John	$N \multimap N \otimes who \otimes (S \multimap NP), N, who, (S \multimap NP) \multimap VP,$ $VP \multimap V \otimes NP, V \multimap meets, meets, NP \multimap John, John \vdash N$
who John meets	$N \multimap N \otimes who \otimes (S \multimap NP), N, who, S \multimap NP \otimes VP, NP \multimap John, John,$ $VP \multimap VP, VP \multimap V \otimes NP, V \multimap meets, meets \vdash N$

But they are not provable with the cut and adjunction rules from M($\mathcal{G}_1'$). In other words, we should consider the construction of the language in two steps. The first step is the construction of the lexicon (a TAG grammar) from a basic set of descriptions using complex rules. The second step is the closure of the TAG grammar with the cut and adjunction rules. This point of view needs to be further developed but could be a first approach to a complete integration of lexicon and grammar.

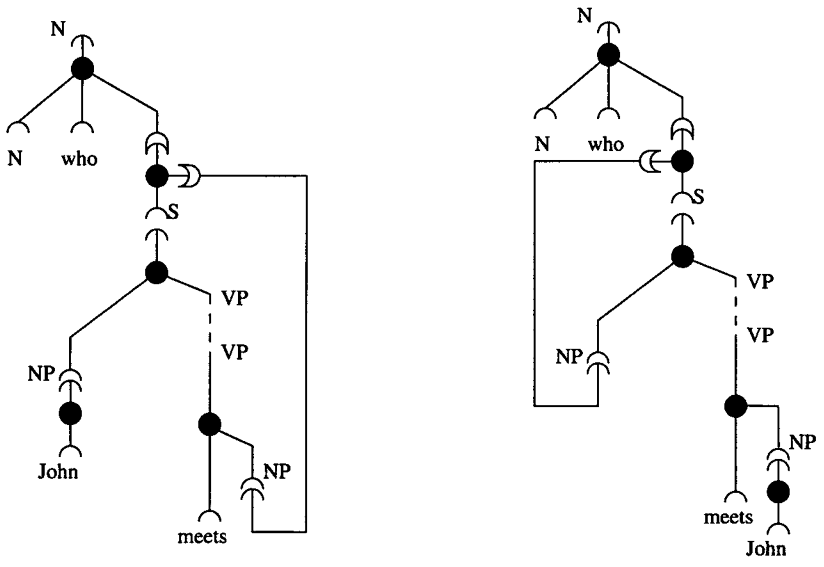


Figure 17
Simplified proofs for *who John meets* and *who meets John*.

6. Conclusion

The use of logic as a framework to describe natural language is not a new idea. Works on Lambek calculus and logic programming are famous examples. However, linguistic formalisms have fundamentally evolved in the past two decades. Though theoretical research has been done on unification and attribute-value structures, operations on syntactic trees have been investigated mainly by comparing different solutions (Vijay-Shanker and Weir 1994a, 1994b). In this paper, we consider another way to look at these operations. We focus on the adjunction operation available in Tree Adjoining Grammars, as it seems to be the simplest way to augment the expressive power of a formalism. We prove that noncommutative intuitionistic linear logic is a good framework and we define a fragment equivalent to TAG. We show, furthermore, to what extent geometric representations of proofs (proofnets) may be useful in understanding how black boxes (i.e., relations between nodes in a syntactic tree) help simplify a parse but also hide interesting mechanisms. There is still a lot to do in this direction. For one thing, generalized categorial grammars also have to be logically investigated, the objective being to relate GCG operations to logical operations (completed if necessary). The preceding discussions also show the relationship between our point of view and the idea of quasi trees developed by Vijay-Shanker (1992). He proposes to consider partial descriptions of trees, i.e., adjunction nodes represented by means of loose relations whose meaning is a domination relation. In this case, the adjunction operation is identified by a pair of substitution operations. The strong relation with what precedes is clear. However, in order to take into account exactly this presentation, the axiom of identity $A \vdash A$, where A is a propositional variable, must be added to the calculus $\mathcal{A}(\mathcal{G})$ given in Section 4. In this way, adjunction nodes can be deleted from sequents. In this new calculus, the following rule is satisfied:

$$\frac{A \vdash A \quad \Gamma, A \multimapinv A, \Delta \vdash B}{\Gamma, \Delta \vdash B} \ (adjunction)$$

Hence, we obtain the following equivalence:

Proposition
A parse tree is *correct*

iff the two nodes in a domination relation have the same label

iff there is a proof whose conclusions that are propositional variables are the words of the sentence in the same order, and without any formula of the form $A \multimapinv A$.

Appendix

A.1 A Brief Description of Noncommutative Linear Logic

Linear logic was introduced by Girard (1987) as a "resource conscious logic." In other words, though classical logic deals with static descriptions, linear logic considers propositions as finite resources. Hence, while "A" and "A *and* A" are equivalent in classical logic, this is (generally) not the case in linear logic. The easiest technical way to investigate this difference is to consider the Gentzen sequent calculus for these logics. A sequent is of the form $\Gamma \vdash \Delta$ where Γ and Δ stand for sequences of formulas well-formed with respect to the language of the logic. It expresses the fact that the (multiplicative) disjunction of formulas in Δ is a consequence of the (multiplicative) conjunction of formulas in Γ. Remember that a sequent calculus is a set of rules specifying the provable sequents, given a set of axioms. A proof of a sequent is then the successive application of sequent rules beginning with axioms, i.e., a tree with the proved sequent as the root of the tree (at the bottom) and whose leaves are axioms (on top). Besides axioms and rules introducing connectives at the right or left part of a sequent, we find structural rules that govern the structure of a sequent. In classical logic, the set of structural rules consists in weakening, contraction, and exchange (see Figure 18 where A, B are formulas, $\Gamma, \Gamma', \Delta, \Delta'$ are sequences of formulas). Weakening and contraction allow the arbitrary copying of formulas: having a formula A as a hypothesis or conclusion is equivalent to having it twice (or more). This point of view contradicts the notion of resource, hence these two structural rules are omitted in linear logic. However special connectives, namely the **exponentials** *of-course* "!" and *why-not* "?" have these properties. The exchange rule is responsible for commutativity of the comma (in the right side and in the left side): the order of hypotheses or conclusions does not matter. This rule is no longer valid in the noncommutative version of linear logic.

However, and this is already true in linear logic, the logical interpretation of "and" and "or" is not as simple as it is in classical logic. We need to distinguish two "and" ($\otimes$ meaning 'times' and & meaning 'with') and two "or" ($\parr$ meaning 'par' and $\oplus$ meaning 'plus'), hence inducing four constants: $1, \top, \bot, 0$ (respective neutral elements for the previous connectives). In fact, connectives are related in such a way that they form two groups: the multiplicative group ($\otimes, \parr, 1, \bot$) and the additive group ($\&, \oplus, \top, 0$). Hereafter, we use only the multiplicative group. There are obviously fundamental reasons for this proliferation but these explanations are outside the scope of this paper. Negation and implication are however of special interest. In (commutative) linear logic, there is only one negation $\cdot^{\perp}$ and one (linear) implication $\multimap$. In the noncommutative case, negation and implication have to be split: there is pre- ${}^{\perp}\cdot$ and post- negation $\cdot^{\perp}$ and pre- $\multimapinv$ and post- implication $\multimap$. These two implications have to be related with two operations in Lambek calculus: $\multimap$ with $\backslash$ and $\multimapinv$ with $/$. The implications may

$$\frac{\Gamma \vdash \Delta}{\Gamma, A \vdash \Delta} \; (l - weakening) \qquad \frac{\Gamma \vdash \Delta}{\Gamma \vdash \Delta, A} \; (r - weakening)$$

$$\frac{\Gamma, A, A \vdash \Delta}{\Gamma, A \vdash \Delta} \; (l - contraction) \qquad \frac{\Gamma \vdash \Delta, A, A}{\Gamma \vdash \Delta, A} \; (r - contraction)$$

$$\frac{\Gamma, B, A, \Gamma' \vdash \Delta}{\Gamma, A, B, \Gamma' \vdash \Delta} \; (l - exchange) \qquad \frac{\Gamma \vdash \Delta, B, A, \Delta'}{\Gamma \vdash \Delta, A, B, \Delta'} \; (r - exchange)$$

Figure 18
Structural rules.

be defined in the following way: $B \circ\!- A \equiv B ⅋ {}^{\perp}A$ and $A -\!\circ B \equiv A^{\perp}⅋B$. In Figure 19, we give the one-sided sequent calculus for the multiplicative fragment of noncommutative linear logic (N-LL), and in Figure 3 in Section 3, the two-sided sequent calculus for the multiplicative fragment of intuitionistic noncommutative linear logic (N-ILL): sequent calculus for N-LL and sequent calculus for N-ILL satisfy the cut elimination theorem, i.e., for each proof there exists a cut-free proof with the same conclusion; however, we make use of cut rules in Section 4. Note that if $\Gamma \vdash A$ is provable in the multiplicative intuitionistic noncommutative linear logic, then $\vdash (\Gamma^*)^{\perp}, A^*$ is provable in the multiplicative noncommutative linear logic, where:

- for each formula A of intuitionistic noncommutative linear logic, A^* is a formula of noncommutative linear logic defined as follows
  - $p^* = p$, for every propositional letter p
  - $(B \otimes C)^* = B^* \otimes C^*$, $(B -\!\circ C)^* = (B^*)^{\perp}⅋C^*$, $(B \circ\!- C)^* = B^*⅋\,{}^{\perp}(C^*)$
- for each finite sequence $A_1, \ldots, A_n$ of formulas of intuitionistic noncommutative linear logic, $(A_1, \ldots, A_n)^* = (A_1)^*, \ldots, (A_n)^*$
- for each finite sequence $A_1, \ldots, A_n$ of formulas of noncommutative linear logic, $(A_1, \ldots, A_n)^{\perp} = (A_n)^{\perp}, \ldots, (A_1)^{\perp}$

A.2 The Calculus $\mathcal{A}$ (a Fragment of N-ILL) (proofs)

In this section, we give the proofs for the various propositions presented in the paper. We repeat the definitions and propositions for clarity.

Definition The Calculus $\mathcal{A}$

- Alphabet of $\mathcal{A}$: propositional letters $a, b, \ldots,$ connectives $\otimes, \circ\!-$.
- Formulas: usual definition. A is a simple $\otimes$-formula iff A is a propositional letter or A is a formula $b_1 \otimes \ldots \otimes b_n$ where $b_1, \ldots, b_n$ are propositional letters. B is a $\circ\!-$-formula iff $C = a \circ\!- A$ where a is a propositional letter and A is a simple $\otimes$-formula.
- Sequents: $\Gamma \vdash A$, where Γ is a finite sequence of formulas and A is a formula.

Alphabet:

- propositional letters: $a, b, c, \ldots$
- for each propositional letter p and each integer $n > 0$

$$p^{\overbrace{\perp \cdots \perp}^{n \text{ times}}} \text{ and } {}^{\overbrace{\perp \cdots \perp}^{n \text{ times}}}p$$

- connectives: $\otimes, \parr$

Formulas: usual definition

Sequents: $\vdash \Gamma$ where Γ is a finite sequence of formulas

Metalinguistic definition of $A^{\perp}$ and ${}^{\perp}A$ s.t. ${}^{\perp}(A^{\perp}) = ({}^{\perp}A)^{\perp} = A$, for every formula A:

$$(p^{\overbrace{\perp \ldots \perp}^{n \text{ times}}})^{\perp} = p^{\overbrace{\perp \ldots \perp}^{n+1 \text{ times}}} \qquad ({}^{\overbrace{\perp \ldots \perp}^{n \text{ times}}}p)^{\perp} = {}^{\overbrace{\perp \ldots \perp}^{n-1 \text{ times}}}p$$

$${}^{\perp}(p^{\overbrace{\perp \ldots \perp}^{n \text{ times}}}) = p^{\overbrace{\perp \ldots \perp}^{n-1 \text{ times}}} \qquad {}^{\perp}({}^{\overbrace{\perp \ldots \perp}^{n \text{ times}}}p) = {}^{\overbrace{\perp \ldots \perp}^{n+1 \text{ times}}}p$$

$$(B \otimes C)^{\perp} = C^{\perp} \parr B^{\perp} \qquad (B \parr C)^{\perp} = C^{\perp} \otimes B^{\perp}$$

$${}^{\perp}(B \otimes C) = {}^{\perp}C \parr {}^{\perp}B \qquad {}^{\perp}(B \parr C) = {}^{\perp}C \otimes {}^{\perp}B$$

Rules of sequent calculus:

$$\frac{}{\vdash A^{\perp}, A}\ (axiom) \qquad \frac{\vdash \Gamma_1, A, \Gamma_2 \quad \vdash A^{\perp}, \Delta}{\vdash \Gamma_1, \Delta, \Gamma_2}\ (cut-1) \qquad \frac{\vdash \Gamma, A \quad \vdash \Delta_1, A^{\perp}, \Delta_2}{\vdash \Delta_1, \Gamma, \Delta_2}\ (cut-2)$$

$$\frac{\vdash \Delta_1, A, B, \Delta_2}{\vdash \Delta_1, A \parr B, \Delta_2}\ (r-\parr) \qquad \frac{\vdash \Gamma_1, A, \Gamma_2 \quad \vdash B, \Delta}{\vdash \Gamma_1, A \otimes B, \Delta, \Gamma_2}\ (r1-\otimes) \qquad \frac{\vdash \Gamma, A \quad \vdash \Delta_1, B, \Delta_2}{\vdash \Delta_1, \Gamma, A \otimes B, \Delta_2}\ (r2-\otimes)$$

Figure 19
Language and sequent calculus for multiplicative noncommutative linear logic.

- Sequent calculus:
  - Axiom: $a \vdash a$
  - Rules: $\dfrac{\Gamma \vdash A \quad \Delta \vdash B}{\Gamma, \Delta \vdash A \otimes B}\ (\otimes) \qquad \dfrac{\Gamma \vdash A \quad \Gamma_1, C, \Gamma_2 \vdash B}{\Gamma_1, C \multimapinv A, \Gamma, \Gamma_2 \vdash B}\ (\multimapinv)$

In the following, we only consider sequents such that formulas in the left side are either propositional letters, or $\multimapinv$-formulas. So, in the rule introducing $\multimapinv$, C stands for a propositional letter. This consists in considering trees: C is the (unique) mother and the $\otimes$-formula A is the sequence of its daughters.

Proposition Calculus $\mathcal{A}$

1. If $\Gamma \vdash A \otimes B$ is provable in $\mathcal{A}$, then

   - A and B are simple $\otimes$-formulas;
   - there is a unique pair (Γ_1, Γ_2) s.t. $\Gamma = \Gamma_1, \Gamma_2$ and both the sequents $\Gamma_1 \vdash A$ and $\Gamma_2 \vdash B$ are provable in $\mathcal{A}$.

Proof
By induction on the proof π of $\Gamma \vdash A \otimes B$ in $\mathcal{A}$. Note that π cannot be an axiom.

- If the last rule in π is $(\otimes)$, then π is $\dfrac{\overset{\vdots}{\Psi \vdash C} \quad \overset{\vdots}{\Phi \vdash D}}{\Gamma \vdash C \otimes D}\,(\otimes)$ with $\Gamma = \Psi, \Phi$ and $A \otimes B = C \otimes D$ (disregarding the brackets).
  If $A = C$ and $B = D$, the result is obvious. If $A \otimes C' = C$ and $C' \otimes D = B$, then by induction hypothesis there exist unique Ψ_1 and Ψ_2 such that $\Psi_1 \vdash A$ and $\Psi_2 \vdash C'$ are provable for $\Psi = \Psi_1, \Psi_2$; and then, by $(\otimes)$ $\Psi_2, \Phi \vdash B$ is provable, so that $\Gamma_1 = \Psi_1$ and $\Gamma_2 = \Psi_2, \Phi$ are unique and satisfy the property. If $A = C \otimes D'$ and $D' \otimes B = D$, then by induction hypothesis, there exist unique Φ_1 and Φ_2 such that $\Phi_1 \vdash D'$ and $\Phi_2 \vdash B$ are provable for $\Phi = \Phi_1, \Phi_2$; and then, by $(\otimes)$ $\Psi, \Phi_1 \vdash A$ is provable, so that $\Gamma_1 = \Psi, \Phi_1$ and $\Gamma_2 = \Phi_2$ are unique and satisfy the property.

- If the last rule in π is $(\multimapinv)$, then π is $\dfrac{\overset{\vdots}{\Phi \vdash C} \quad \overset{\vdots}{\Psi_1, a, \Psi_2 \vdash A \otimes B}}{\Gamma \vdash A \otimes B}\,(\multimapinv)$ with $\Gamma = \Psi_1, a \multimapinv C, \Phi, \Psi_2$. We apply the induction hypothesis on $\overset{\vdots}{\Psi_1, a, \Psi_2 \vdash A \otimes B}$ (a proof shorter than π). If $\Psi_1, a, \Delta \vdash A$ and $\Delta' \vdash B$ are provable with $\Psi_2 = \Delta, \Delta'$, then $\dfrac{\overset{\vdots}{\Phi \vdash C} \quad \overset{\vdots}{\Psi_1, a, \Delta \vdash A}}{\Psi_1, a \multimapinv C, \Phi, \Delta \vdash A}\,(\multimapinv)$ so that $\Gamma_1 = \Psi_1, a \multimapinv C, \Phi, \Delta$ and $\Gamma_2 = \Delta'$ are unique and satisfy the property. If $\Delta \vdash A$ and $\Delta', a, \Psi_2 \vdash B$ are provable with $\Psi_1 = \Delta, \Delta'$, then
  $\dfrac{\overset{\vdots}{\Phi \vdash C} \quad \overset{\vdots}{\Delta', a, \Psi_2 \vdash B}}{\Delta', a \multimapinv C, \Phi, \Psi_2 \vdash B}\,(\multimapinv)$ so that $\Gamma_1 = \Delta$, and $\Gamma_2 = \Delta', a \multimapinv C, \Phi, \Psi_2$ are unique and satisfy the property.

2. If $\Gamma, a \multimapinv A, \Delta \vdash B$ is provable in $\mathcal{A}$, then

   - A and B are simple $\otimes$-formulas;
   - there is a unique pair (Δ_1, Δ_2) s.t. $\Delta = \Delta_1, \Delta_2$ and both the sequents $\Delta_1 \vdash A$ and $\Gamma, a, \Delta_2 \vdash B$ are provable in $\mathcal{A}$.

   Such a pair (Δ_1, Δ_2) will be called "the splitting pair for Δ in $\Gamma, a \multimapinv A, \Delta \vdash B$."

Proof
By induction on the proof π of $\Gamma, a \multimapinv A, \Delta \vdash B$ in $\mathcal{A}$.

This pair can be computed easily: the first element Δ_1 of the splitting pair must satisfy a counting condition on each propositional variable occurring in it as defined below. This property will enable us to consider an adjunction rule based on such splitting pairs.

Definition
Let A be a simple $\otimes$-formula or a $\circ\!-$-formula (calculus $\mathcal{A}$) and a a propositional variable, the number of positive occurrences $p(a, A)$ (and negative occurrences $n(a, A)$) of a in A is defined by:

- if $A \equiv a$ then $p(a, A) = 1$, $n(a, A) = 0$
- if $A \equiv b$ and b is a propositional variable distinct from a, then $p(a, A) = 0$, $n(a, A) = 0$
- if $A \equiv B \otimes C$, then $p(a, A) = p(a, B) + p(a, C)$, $n(a, A) = n(a, B) + n(a, C)$
- if $A \equiv B \circ\!- A_1 \otimes \ldots \otimes A_n$, then $p(a, A) = p(a, B)$ and $n(a, A) = p(a, A_1 \otimes \ldots \otimes A_n)$ as $A_1, \ldots, A_n$ are $\otimes$-simple formulas, cf. the calculus $\mathcal{A}$.

Let $\mathcal{S}$ be the sequent $C_1, \ldots, C_n \vdash A$ defined as for calculus $\mathcal{A}$ ($\otimes$ and $\circ\!-$):

- $p(a, \mathcal{S}) = p(a, A) + n(a, C_1) + \cdots + n(a, C_n)$
- $n(a, \mathcal{S}) = p(a, C_1) + \cdots + p(a, C_n)$

It is easy to prove (for $\mathcal{S}$ provable in the calculus $\mathcal{A}$) by induction on a proof of $\mathcal{S}$ that (i) for each propositional variable a occurring in $\mathcal{S}$, $p(a, \mathcal{S}) = n(a, \mathcal{S})$, and also that (ii) if $\mathcal{S}$ is the sequent $C_1, \ldots, C_n \vdash A$ then C_n is a propositional variable (we denote this variable by $e(\mathcal{S})$). Moreover, for $k \leq n$, if we denote the sequent $C_1, \ldots, C_k \vdash A$ by $\mathcal{S}_k$, then (iii) $p(a, \mathcal{S}_k) \geq n(a, \mathcal{S}_k)$. We can then deduce that (iv), for $k < n$, there exists at least one propositional variable s.t. $p(a, \mathcal{S}_k) > n(a, \mathcal{S}_k)$. Note that $p(e(\mathcal{S}), \mathcal{S}_{n-1}) > n(e(\mathcal{S}), \mathcal{S}_{n-1})$.

Proposition
Let $\mathcal{S}$: $\Gamma, B \circ\!- C, D_1, \ldots, D_n \vdash A$ be a provable sequent in $\mathcal{A}$, then the splitting pair for $D_1, \ldots, D_n$ in $\mathcal{S}$ is uniquely determined by the sequent $\mathcal{S}'$: $D_1, \ldots, D_j \vdash C$, $j \leq n$, such that for each propositional variable a occurring in $\mathcal{S}'$, the following condition is satisfied:

$$p(a, \mathcal{S}') = n(a, \mathcal{S}')$$

Proof
Note that $\mathcal{S}'$ is provable. Hence the property (i) is true for such a sequent. The uniqueness results from property (iv) stated previously.

3. The calculus $\mathcal{A}$ is closed under the atomic cut rule

$$\frac{\Gamma \vdash a \quad \Delta_1, a, \Delta_2 \vdash A}{\Delta_1, \Gamma, \Delta_2 \vdash A} \; (cut)$$

i.e., if the sequents $\Gamma \vdash a$ and $\Delta_1, a, \Delta_2 \vdash A$ are provable in $\mathcal{A}$, then the sequent $\Delta_1, \Gamma, \Delta_2 \vdash A$ is also provable in $\mathcal{A}$.

Proof
By induction on the proof π of $\Gamma \vdash a$, by using the properties 1 and 2. If π is an axiom, the result is trivial. If π is not an axiom, the last rule in π is $(\circ\!\!-)$, and so π has the form

$$\dfrac{\overset{\vdots}{\Psi \vdash B} \quad \overset{\vdots}{\Phi_1, b, \Phi_2 \vdash a}}{\Gamma \vdash a}\;(\circ\!\!-)$$

By induction hypothesis, since $\Phi_1, b, \Phi_2 \vdash a$ is provable (with a shorter proof than π) and $\Delta_1, a, \Delta_2 \vdash A$ is provable, then $\Delta_1, \Phi_1, b, \Phi_2, \Delta_2 \vdash A$ is provable, and then we get

$$\dfrac{\overset{\vdots}{\Psi \vdash B} \quad \overset{\vdots}{\Delta_1, \Phi_1, b, \Phi_2, \Delta_2 \vdash a}}{\Gamma \vdash a}\;(\circ\!\!-)$$

4. The calculus $\mathcal{A}$ is closed under the adjoining rule

$$\dfrac{\Gamma_1, a, \Gamma_2 \vdash a \quad \Delta, a \circ\!\!- a, \Lambda \vdash b}{\Delta, \Gamma_1, \Lambda_1, \Gamma_2, \Lambda_2 \vdash b}\;(adj)$$

where (Λ_1, Λ_2) is the splitting pair of Λ in $\Delta, a \circ\!\!- a, \Lambda \vdash b$.

Proof
Indeed, suppose the sequents $\Gamma_1, a, \Gamma_2 \vdash a$ and $\Delta, a \circ\!\!- a, \Lambda \vdash b$ are provable in $\mathcal{A}$. Since $\Delta, a \circ\!\!- a, \Lambda \vdash b$ is provable, by the property 2, there is a unique pair (Λ_1, Λ_2) s.t. $\Lambda = \Lambda_1, \Lambda_2$ and both the sequents $\Lambda_1 \vdash a$ and $\Delta, a, \Lambda_2 \vdash b$ are provable in $\mathcal{A}$. Now since $\Gamma_1, a, \Gamma_2 \vdash a$ and $\Delta, a, \Lambda_2 \vdash b$ are provable in $\mathcal{A}$, by the property 3 the sequent $\Delta, \Gamma_1, a, \Gamma_2, \Lambda_2 \vdash b$ is also provable in $\mathcal{A}$; and now, since $\Lambda_1 \vdash a$ and $\Delta, \Gamma_1, a, \Gamma_2, \Lambda_2 \vdash b$ are provable in $\mathcal{A}$, the sequent $\Delta, \Gamma_1, \Lambda_1, \Gamma_2, \Lambda_2 \vdash b$ is also provable in $\mathcal{A}$.

Definition The calculus $\mathcal{A}(\mathcal{G})$

Let $\mathcal{G}$ be a family of labeled trees, of depth 1, not of the form $\begin{array}{c}X\\ |\\ X\end{array}$. Let $\mathcal{T}(\mathcal{G})$ be the closure of $\mathcal{G}$ under the rules:

- substitution with or without the declaration of a new possibly internal point on which the adjoining operation may be performed,
- adjoining operation.

$\mathcal{A}(\mathcal{G})$ is the calculus obtained from $\mathcal{A}$ as follows:

- propositional letters are exactly all the labels of the trees in $\mathcal{G}$,
- the rule $(\circ\!\!-)$ is restricted as follows:

$$\dfrac{\Gamma \vdash A \quad \Gamma_1, a, \Gamma_2 \vdash B}{\Gamma_1, a \circ\!\!- A, \Gamma, \Gamma_2 \vdash B}\;(\circ\!\!-, \mathcal{G})$$

where A, B are simple $\otimes$-formulas of $\mathcal{A}(\mathcal{G})$, a is a propositional letter of $\mathcal{A}(\mathcal{G})$, and one of the following cases occurs:

— A is a

— A is a propositional letter b different from a, and the tree $\begin{array}{c} a \\ | \\ b \end{array} \in \mathcal{G}$

— A is $b_1 \otimes \ldots \otimes b_n$, and the tree $\begin{array}{c} a \\ \diagup \mid \diagdown \\ b_1 \ \ldots \ b_n \end{array} \in \mathcal{G}$

Proposition Calculus $\mathcal{A}(\mathcal{G})$
Properties 1–4 of $\mathcal{A}$ are also properties of $\mathcal{A}(\mathcal{G})$. Moreover the following properties hold for $\mathcal{A}(\mathcal{G})$:

- To $T \in \mathcal{T}(\mathcal{G})$, we associate a sequent *Seq(T)* of $\mathcal{A}(\mathcal{G})$ s.t.:

  — if a is the root of T, and the terminal points of T (ordered from left to right) are $a_1, \ldots, a_m$, then *Seq(T)* is

$$\Gamma \vdash a$$

  where in Γ the sequence of all the occurring propositional variables is $a_1, \ldots, a_m$ and in Γ there is a formula $c \multimapinv c$ iff c is a possibly internal point of T on which the adjoining operation may be performed;

  — *Seq(T)* is provable in $\mathcal{A}(\mathcal{G})$.

Proof
By induction on the class of all the trees of $\mathcal{T}(\mathcal{G})$.

Let $T \in \mathcal{G}$, i.e. T is $\begin{array}{c} a \\ \diagup \mid \diagdown \\ b_1 \ \ldots \ b_n \end{array} \in \mathcal{G}$

Define $Seq(T) \equiv a \multimapinv b_1 \otimes \ldots \otimes b_n, b_1, \ldots, b_n \vdash a$. Trivially, $Seq(T)$ satisfies (i) and (ii).

Let T be a tree obtained from a tree $T_1 \in \mathcal{T}(\mathcal{G})$ with root a and a tree $T_2 \in \mathcal{T}(\mathcal{G})$ with a terminal point a, by substitution *with* the declaration that a is a point in T on which the adjoining operation may be performed. Suppose b is the root of T_2, and so b is the root of T. By induction hypothesis, to T_1 is associated a sequent $Seq(T_1) \equiv \Gamma \vdash a$ satisfying (i) and (ii), and to T_2 is associated a sequent $Seq(T_2)$ satisfying (i) and (ii) so that $Seq(T_2) \equiv \Delta_1, a, \Delta_2 \vdash b$ where all the terminal points of T_2 before a occur in Δ_1 in the same order as in T_2 and all the terminal points of T_2 after a occur in Δ_2 in the same order as in T_2. Define $Seq(T) \equiv \Delta_1, a \multimapinv a, \Gamma, \Delta_2 \vdash b$. It is easy to prove that $Seq(T)$ satisfies (i). $Seq(T)$ is obtained from $Seq(T_1)$ and $Seq(T_2)$ by using $(\multimapinv, \mathcal{G})$, so that it is provable in $\mathcal{A}(\mathcal{G})$ since $Seq(T_1)$ and $Seq(T_2)$ are provable in $\mathcal{A}(\mathcal{G})$ by induction hypothesis.

Let T be a tree obtained from a tree $T_1 \in \mathcal{T}(\mathcal{G})$ with root a and a tree $T_2 \in \mathcal{T}(\mathcal{G})$ with a terminal point a, by substitution *without* the declaration that a is in T a point on which the adjoining operation may be performed. Suppose b is the root of T_2, and so b is the root of T. By induction hypothesis, to T_1 is associated a sequent $Seq(T_1) \equiv \Gamma \vdash a$ satisfying (i) and (ii), and to T_2 is associated a sequent $Seq(T_2)$ satisfying (i) and (ii) so that $Seq(T_2) \equiv \Delta_1, a, \Delta_2 \vdash b$ where all the terminal points of T_2 before a occur in Δ_1 in the same order as in T_2 and all the terminal points of T_2 after a occur in Δ_2 in the same order as in T_2. Define $Seq(T) \equiv \Delta_1, \Gamma, \Delta_2 \vdash b$. It is easy to prove that $Seq(T)$ satisfies (i). $Seq(T)$ is obtained from $Seq(T_1)$ and $Seq(T_2)$ by using the atomic cut rule, so that by property 3 it is provable in $\mathcal{A}(\mathcal{G})$ since $Seq(T_1)$ and $Seq(T_2)$ are provable in $\mathcal{A}(\mathcal{G})$ by induction hypothesis.

Let T be a tree obtained by adjoining operation from a tree $T_1 \in \mathcal{T}(\mathcal{G})$ with root a and a terminal point a, and a tree $T_2 \in \mathcal{T}(\mathcal{G})$ with a possibly internal point a on which the adjoining operation may be performed. Suppose b is the root of T_2, and so b is the root of T. By induction hypothesis, to T_1 is associated a sequent $Seq(T_1)$ satisfying (i) and (ii), so that $Seq(T_1) \equiv \Gamma_1, a, \Gamma_2 \vdash a$ where all the terminal points of T_1 before a occur in Γ_1 in the same order as in T_1 and all the terminal points of T_1 after a occur in Γ_2 in the same order as in T_1; and to T_2 is associated a sequent $Seq(T_2)$ satisfying (i) and (ii) so that $Seq(T_2) \equiv \Delta, a \multimapinv a, \Lambda \vdash b$. Since $Seq(T_2)$ is provable in $\mathcal{A}(\mathcal{G})$ by induction hypothesis, then by property 2, there is a unique pair (Λ_1, Λ_2) s.t. $\Lambda = \Lambda_1, \Lambda_2$ and the sequents $\Lambda_1 \vdash a$ and $\Delta, a, \Lambda_2 \vdash b$ are both provable in $\mathcal{A}(\mathcal{G})$. Define $Seq(T) \equiv \Delta, \Gamma_1, \Lambda_1, \Gamma_2, \Lambda_2 \vdash b$. It is easy to prove that $Seq(T)$ satisfies (i). $Seq(T)$ is obtained from $Seq(T_1)$ and $Seq(T_2)$ by using adjoining rule, so that by property 3 it is provable in $\mathcal{A}(\mathcal{G})$ since $Seq(T_1)$ and $Seq(T_2)$ are provable in $\mathcal{A}(\mathcal{G})$ by induction hypothesis.

- To every provable sequent $\Gamma \vdash A$ in $\mathcal{A}(\mathcal{G})$, we associate $Tree(\Gamma \vdash A)$ s.t.
  - — if A is a propositional letter, then $Tree(\Gamma \vdash A) \in \mathcal{T}(\mathcal{G})$ where the root is A, the terminal points (from left to right) are exactly all the propositional letters occurring in Γ and in the same order in which they occur in Γ, and the possibly internal points on which the adjoining operation may be performed are exactly all the propositional letters c s.t. $c \multimapinv c$ occur in Γ;
  - — if A is $b_1 \otimes \ldots \otimes b_n$, and so $\Gamma = \Gamma_1 \ldots \Gamma_n$ with the sequents $\Gamma_i \vdash b_i$ provable in $\mathcal{A}(\mathcal{G})$ for every $1 \leq i \leq n$, then $Tree(\Gamma \vdash A)$ is a sequence $T_1, \ldots, T_n$ of trees $\in \mathcal{T}(\mathcal{G})$, s.t. $T_i = Tree(\Gamma_i \vdash b_i)$.

Proof
By induction on the proof π of $\Gamma \vdash A$.

$Tree(a \vdash a) = a$

If $A = B \otimes C$ and the last rule of π is $(\otimes)$ with principal formula $B \otimes C$ and premises $\Gamma_1 \vdash B$ and $\Gamma_2 \vdash C$, then $Tree(\Gamma \vdash B \otimes C) = Tree(\Gamma_1 \vdash B), Tree(\Gamma_2 \vdash C)$.

If the last rule of π is $(\multimapinv, \mathcal{G})$ with principal formula $a \multimapinv a$ and premises $\Gamma \vdash a$ and $\Delta_1, a, \Delta_2 \vdash b$, then $Tree(\Delta_1, a \multimapinv a, \Delta_2 \vdash b)$ is the tree obtained by substitution from $Tree(\Gamma \vdash a)$ and $Tree(\Delta_1, a, \Delta_2 \vdash b)$ with the declaration that the possibly internal point a is a point on which the adjoining operation may be performed.

If the last rule of π is $(\multimapinv, \mathcal{G})$ with principal formula $a \multimapinv a$ and premises $\Gamma \vdash a$ and $\Delta_1, a, \Delta_2 \vdash b_1 \otimes \ldots \otimes b_n$, and a occurs in Γ_i s.t. $\Gamma_i \vdash b_i$, then $Tree(\Delta_1, a \multimapinv a, \Delta_2 \vdash b_1 \otimes \ldots \otimes b_n)$ is obtained from $Tree(\Delta_1, a, \Delta_2 \vdash b_1 \otimes \ldots \otimes b_n) = Tree(\Gamma_1 \vdash b_1), \ldots, Tree(\Gamma_n \vdash b_n)$ by replacing $Tree(\Gamma_i \vdash b_i)$ by the tree obtained by substitution from $Tree(\Gamma \vdash a)$ and $Tree(\Gamma_i \vdash b_i)$ with the declaration that the possibly internal point a is a point on which the adjoining operation may be performed.

If the last rule of π is $(\multimapinv, \mathcal{G})$ with principal formula $a \multimapinv A$ and premises $\Gamma \vdash A$ and $\Delta_1, a, \Delta_2 \vdash b$, then $Tree(\Delta_1, a \multimapinv A, \Delta_2 \vdash b)$ is the tree obtained as follows: first add a root a common to all the trees $Tree(\Gamma \vdash A)$ by using (if $A = b_1 \otimes \ldots \otimes b_n$) the link

$$\begin{tikzcd} & a \arrow[dl, no head] \arrow[d, no head] \arrow[dr, no head] & \\ b_1 & \ldots & b_n \end{tikzcd} \in \mathcal{G},$$

or (if $A = b$) the link $\begin{array}{c} a \\ | \\ b \end{array} \in \mathcal{G}$, and then compose this tree with the tree $Tree(\Delta_1, a, \Delta_2) \vdash b)$.

If the last rule of π is $(\circ\!-, \mathcal{G})$ with principal formula $a \circ\!- A$ and premises $\Gamma \vdash A$ and $\Delta_1, a, \Delta_2 \vdash b_1 \otimes \ldots \otimes b_n$, and a occurs in Γ_i s.t. $\Gamma_i \vdash b_i$, then $Tree(\Delta_1, a \circ\!- A, \Delta_2 \vdash b_1 \otimes \ldots \otimes b_n)$ is obtained from $Tree(\Delta_1, a, \Delta_2 \vdash b_1 \otimes \ldots \otimes b_n) = Tree(\Gamma_1 \vdash b_1), \ldots, Tree(\Gamma_n \vdash b_n)$ by replacing $Tree(\Gamma_i \vdash b_i)$ by the tree obtained as above from $Tree(\Gamma \vdash A)$ and $Tree(\Gamma_i \vdash b_i)$.

- If $\Gamma \vdash a$ is provable in $\mathcal{A}(\mathcal{G})$, then $Seq(Tree(\Gamma \vdash a)) = \Gamma \vdash a$. If T is a tree of $\mathcal{G}$, then $Tree(Seq(T)) = T$.
- Let M be a set of provable sequents in $\mathcal{A}(\mathcal{G})$. Define CL(M) as follows:

  — M$\subseteq$CL(M)
  — (closure under atomic cut rule) if $\Gamma \vdash a \in$ CL(M) and $\Delta_1, a, \Delta_2 \vdash B \in$CL(M), then $\Delta_1, \Gamma, \Delta_2 \vdash B \in$CL(M)
  — (closure under adjoining operation) if $\Gamma_1, a, \Gamma_2 \vdash a \in$CL(M) and $\Delta, a \circ\!- a, \Lambda \vdash b \in$CL(M), then $\Delta, \Gamma_1, \Lambda_1, \Gamma_2, \Lambda_2 \vdash b \in$CL(M), where (Λ_1, Λ_2) is the splitting pair of Λ in $\Delta, a, \Lambda \vdash b$;
  — nothing else belongs to CL(M).

- If $\Gamma \vdash A \in$CL(M), then $\Gamma \vdash A$ is provable in $\mathcal{A}(\mathcal{G})$.

Proof
By induction on CL(M).

If $\Gamma \vdash A \in$M, then by hypothesis $\Gamma \vdash A$ is provable.

If $\Gamma \vdash A$ is obtained from two other sequents, by atomic cut rule, then $\Gamma \vdash A$ is provable by property 3 since (by induction hypothesis) the two sequents are provable.

If $\Gamma \vdash A$ is obtained from two other sequents, by adjoining operation, then $\Gamma \vdash A$ is provable by property 4 since (by induction hypothesis) the two sequents are provable.

- If $\mathcal{G}' \subseteq \mathcal{G}$, let $\mathcal{T}(\mathcal{G}')$ be the closure of $\mathcal{G}'$ under:

  — substitution;
  — adjoining operation.

  Clearly, CL$(\mathcal{G}') \subseteq \mathcal{T}(\mathcal{G})$. Let $M = \{Seq(T)/T \in \mathcal{G}'\}$, then $CL(M) = \{Seq(T)/T \in \mathcal{T}(\mathcal{G}')\}$.

Proof
The proof follows previous results.

References

Abeillé, Anne, K. Bishop, S. Cote, and Yves Schabes. 1990. A lexicalized tree-adjoining grammar for English. Technical Report MS-CIS-90-24, LINC LAB 170, Computer Science Department, University of Pennsylvania, Philadelphia, PA.

Abrusci, Michele. 1991. Phase semantics and sequent calculus for pure noncommutative classical linear propositional logic. *The Journal of Symbolic Logic*, 56(4):1,403–1,451.

Abrusci, Michele. 1995. Noncommutative proof nets. In Jean-Yves Girard, Yves Lafont, and Laurent Regnier, editors, *Advances in Linear Logic*, volume 222. Cambridge University Press, pages 271–296. Proceedings of the Workshop on Linear Logic, Ithaca, NY, June 1993.

Bar-Hillel, Yoshua. 1953. A quasi-arithmetical notation for syntactic description. *Language*, 29:47–58.

Girard, Jean-Yves. 1987. Linear logic. *Theoretical Computer Science*, 50:1–102.

Joshi, Aravind K. and Seth Kulick. 1995. Partial proof trees as building blocks for a categorial grammar. In Glyn Morrill and Richard T. Oehrle, editors, *Formal Grammar, Proceedings of the Conference of the European Summer School of Logic, Language and Information*, Barcelona, August. Also as Technical Report, Institute for Research in Cognitive Science, University of Pennsylvania, Philadelphia, PA, March 1996.

Joshi, Aravind K., Leon S. Levy, and M. Takahashi. 1975. Tree adjunct grammars. *Journal of Computer and System Sciences*, 10(1):136–163.

Kroch, Anthony S. and Aravind K. Joshi. 1985. Linguistic relevance of tree adjoining grammars. Technical Report MS-CIS-85-18, LINC LAB 170, Computer Science Department, University of Pennsylvania, Philadelphia, PA.

Lambek, Joachim. 1958. The mathematics of sentence structure. *American Math. Monthly*, 65:154–169.

Roorda, Dirk. 1992. Proof nets for Lambek calculus. *Journal of Logic and Computation*, 2(2):211–231.

Vijay-Shanker, K. 1992. Using descriptions of trees in a tree adjoining grammar. *Computational Linguistics*, 18(4):481–517.

Vijay-Shanker, K. and Aravind K. Joshi. 1985. Some computational properties of tree adjoining grammars. In *Proceedings of the 23rd Annual Meeting*, pages 82–93. Association for Computational Linguistics.

Vijay-Shanker, K. and David J. Weir. 1994a. The equivalence of four extensions of context-free grammars. *Mathematical Systems Theory*, 27:511–545.

Vijay-Shanker, K. and David J. Weir. 1994b. Parsing some constrained grammar formalisms. *Computational Linguistics*, 19(4):591–636.