October 4, 2004

Language and Grammar

C. Casadio, P.J. Scott, R.A.G. Seely (eds.)

Studies in Mathematical Linguistics and Natural Language

**CENTER FOR THE STUDY
OF LANGUAGE
AND INFORMATION**

Contents

Contributors

Daniele Bargelli
McGill University, Montreal
Dept. of Mathematics and Statistics
805 Sherbrooke Street West
Montreal, QC, Canada H3A 2K6
bargelld@math.mcgill.ca

Johan van Benthem
ILLC University of Amsterdam
Plantage Muidergracht 24
NL-1018 TV Amsterdam
johan@wins.uva.nl

Michael Barr
Department of Mathematics
and Statistics, McGill University
805 Sherbrooke St. West
Montreal, QC H3A 2K6
barr@barrs.org

Wojciech Buszkowski
Faculty of Mathematics
and Computer Science
Adam Mickiewicz University
Poznań Poland
buszko@amu.edu.pl

Claudia Casadio
Dept. of Philosophy
University "G. D'Annunzio"
66100 Chieti Scalo CH - Italy
casadio@unich.it

Philippe de Groote
LORIA UMR 7503 – INRIA
Campus Scientifique, B.P. 239
54506 Vandoeuvre ls Nancy
Cedex - France
Philippe.de.Groote@loria.fr

Mario Fadda
Universitat Politécnica de Catalunya
Jordi Girona Salgado 1-3
E - 08034 Barcelona
fadda@lsi.upc.es

Brendan S. Gillon
McGill University
Department of Linguistics
1085 Docteur-Penfield Avenue
Montreal, Quebec H3A 1A7
brendan.gillon@mcgill.ca

Maciej Kandulski
Faculty of Mathematics and
Computer Science
Adam Mickiewicz University
61-614 Poznań, Poland
mkandu@amu.edu.pl

Edward L. Keenan
Professor and Chair
UCLA Dept of Linguistics
keenan@humnet.ucla.edu

Joachim Lambek
McGill University
Dept. of Mathematics and Statistics
805 Sherbrooke Street West
Montreal, QC, Canada H3A 2K6
lambek@math.mcgill.ca

Alain Lecomte
IMAG, Grenoble
and SIGNES-INRIA
Bordeaux
Alain.Lecomte@upmf-grenoble.fr

Michael Moortgat
Utrecht Institute of Linguistics OTS
Utrecht University
Trans 10, 3512 JK Utrecht
The Netherlands
michael.moortgat@phil.uu.nl

Glyn Morrill
Universitat Politécnica de Catalunya
Jordi Girona Salgado 1-3
E - 08034 Barcelona
morrill@lsi.upc.es

Richard T. Oehrle
502 Lobos Avenue
Pacific Grove, CA 93950 - USA
oehrle@linc.cis.upenn.edu

Mati Pentus
Department of Mathematical Logic
and Theory of Algorithms
Faculty of Mechanics and Mathematics
Moscow State University
Moscow, Russia, 119992
http://lpcs.math.msu.su/ ~pentus/

Philip J. Scott
Dept. of Mathematics and Statistics
University of Ottawa
585 King Edward
Ottawa, Ontario
Canada K1N 6N5
phil@site.uottawa.ca

Robert A. G. Seely
Department of Mathematics and Statistics
McGill University
805 Sherbrooke St W
Montreal, QC, Canada H3A 2K6
rags@math.mcgill.ca

This volume is dedicated to Joachim Lambek by the editors and the other authors, to honour the contributions he has made to the field of mathematical linguistics. Without his work the volume would have been impossible.

Introduction: the Lambek Program

C. Casadio, P.J. Scott, R.A.G. Seely

> A categorical grammar of a language may be viewed as consisting of the syntactic calculus freely generated from a finite set $\{S, N, \dots\}$ of basic types together with a dictionary which assigns to each word of the language a finite set of types composed from the basic types and 1 by the three binary operations. ... The question now arises, given types A and B, when is $A \to B$ a theorem, that is, when is there a proof $f\colon A \to B$? (Lambek 1988, 304)

Applying logic to grammar is a fundamental issue in philosophy, propounded by such eminent philosophers as Leibniz, Bolzano, Frege and Husserl. Categorial grammars and type logical grammars occupy a central place in this line of investigation, although they have received less attention than well-known linguistic theories such as transformational grammar or lexical-functional grammar. But during the latter part of the twentieth century they have attracted the interest of a select, indeed growing, group of scholars so that now, at the beginning of the twenty-first century, type logical grammars are promising models of reasoning and computation.

Categorial grammars are interesting for the analysis of natural language, both for their elegance and simplicity and for the straightforward relation they establish between a lexically based syntax (essentially consisting of the morpho-phonological properties of the lexical items along with their combinatorial properties) and a compositional semantics. Classical categorial grammars, known as Ajdukiewicz/Bar-Hillel (AB) grammars, are weakly equivalent to context-free phrase structure grammars (CF-PSG). In the latter, the information encoded

Language and Grammar.
C. Casadio, P.J. Scott, R.A.G. Seely (eds.).

by the PS-rules is directly expressed in terms of the relations holding between basic and functorial category symbols listed in the categorial lexicon. These models are inadequate for analyzing many relevant linguistic facts, such as long distance dependencies, generalized conjunction or elliptical constructions. These are the same inadequacies shared by all immediate constituent systems.

Traditional categorial grammars have been extended along two main lines: (i) allowing a finer definition of the category symbols as complex signs consisting, e.g., of sets of features and of level indices; (ii) allowing a richer and more flexible set of categorial rules. In the standard model these latter are limited to the rule of functional application of a functor category to its argument(s) and the possible specification of the direction of combination (AB grammar). Starting from Ades and Steedman (1982), the properties of many types of combinatory rules have been studied, allowing the treatment of a wide range of unbounded dependencies and constituent conjunction constructions, as shown in Oehrle *et al.* (1988) and Moortgat (1988).

However, a more flexible approach to categorial grammar was already available in Bar-Hillel's time, defining the set of categorial rules (including *functional application* and *composition*, *type raising*, and *Geach's rule*) within an elegant algebraic theory. This is the calculus of syntactic types developed by Joachim Lambek (Lambek 1958, 1961). This calculus, widely known as the Lambek calculus, has been studied from many points of view: logical, linguistic, mathematical, computational. It has been investigated as a particular kind of resource-sensitive substructural logic and, as such, forms a distinguished fragment of intuitionistic non-commutative linear logic (or bilinear logic, as defined by Lambek himself).

Non-commutative linear logic is an attractive field in its own right, both from the theoretical viewpoint and for its linguistic applications. The properties of classical non-commutative linear logic have been studied in a number of papers (*e.g.* Abrusci 1991, 1995, 1996), while intuitionistic non-commutative linear logic, restricted to the multiplicative fragment, was first introduced in Lambek's syntactic calculus (SC). In fact, the basic connectives of SC, the tensor product "$\bullet$" , the left division "$\backslash$" , and the right division "/" , respectively correspond to the multiplicative conjunction and the two implications of non-commutative intuitionistic linear logic (NILL). The two implications: "$\multimap$" (linear post-implication) and "$\circ\!\!-$" (linear retro-implication), are the result of the exclusion of Gentzen's structural rule of exchange. This restriction is needed in the analysis of processes in which, as in ordinary linguistic communication, the order of the premises and of the conclusions is

relevant. In particular, SC corresponds to the multiplicative fragment of NILL.

It is worth noting that Lambek (1999, 2000, 2001) recently developed a new system, called *compact bilinear logic*, generated by an ordered monoid (a *pregroup* in the sense of Grishin 1983) in which each element has both a left and a right adjoint, whose properties are similar to the two negations of classical non-commutative linear logic. The reader may refer to Casadio and Lambek (2002) for a detailed comparison of the syntactic calculus with classical non-commutative linear logic (or classical bilinear logic) and compact bilinear logic.

0.1 The Syntactic Calculus

Introduced at the beginning of the 1960's[1], Lambek's syntactic calculus was crucial for the development of the algebraic side of Ajdukiewicz and Bar-Hillel (AB)-grammars. The calculus gives an effective procedure, or algorithm, to distinguish sentences from nonsentences in both formal as well as natural languages. Lambek's system is particularly suited to the formal analysis of natural languages and may be applied to several fields: (i) *linguistics*, in which it provides a rigorous formulation of the grammar of a given language, based on purely logical-mathematical properties. In particular, it allows an *effective* definition of the notion of *well-formed formulas* (grammatical sentences); (ii) *computation*, in which it provides a theory for parsing and diagrammatic strategies; (iii) *automatic translation*, in which it offers an effective analytical procedure such that, starting with the source language as input, will yield as output grammatical strings of the target language.

The syntactic calculus is a *formal deductive system* **L** consisting of a set of types, or categories[2], closed under the three binary operations

[1] Lambek presented his calculus in two well-known papers: *The mathematics of sentence structure* (Lambek 1958), and *On the calculus of syntactic types* (Lambek 1961). As pointed out in Lambek (2002), the arithmetic notation, connecting the tensor *product* with a *right* and a *left division*, resulted from discussions with the mathematician G. Findlay, during his post-doctoral stay at McGill University. The calculus, with the inclusion of the two basic types **n** and **s**, was also influenced by Church's theory of types (Church 1940), as an alternative to the "heavy" type hierarchy of *Principia Mathematica*. A further incentive in applying category theory to linguistic analysis came from the discovery of Bar-Hillel's paper in *Language* (Bar-Hillel 1953), who independently developed similar ideas, and used the two divisions, although not the product. See Lambek (1988), (2002).

[2] "It appeared that the word *categorical* should only refer to the categories introduced by Einlenberg and Mac Lane in their pioneering article (Eilenberg and Mac Lane, 1945), while *categorial* refers to the categories in the sense of types. Sam Eilenberg assures me that their choice of the words *category* and *functor* was not influenced by the occurrence of these words in the Polish school of logic. To avoid

$A \bullet B$, C/B, $A\backslash C$ (to be read as: A *times* B, C *over* B, A *under* C), together with certain axioms and rules of inference to be described below.

The syntactic calculus generates certain relations (called *sequents*) between linguistic expressions. Formally, a sequent is written $X \to Y$, where X and Y are types. Intuitively, those sequents derivable within the calculus represent a kind of syntactic transformation which, starting from a linguistic expression of type X produces an expression of type Y. Alternatively, we may view $X \to Y$ as saying "type Y is derivable from type X", where we now think of the types themselves as kinds of formulas of a logical system.

Initially we consider the following axiom schema and rules of inference for generating sequents; later we shall see several alternate presentations.

Axioms

$A \to A$
$(A \bullet B) \bullet C \to A \bullet (B \bullet C)$
$A \bullet (B \bullet C) \to (A \bullet B) \bullet C$

Rules of Inference

$A \bullet B \to C$ if and only if $A \to C/B$,
$A \bullet B \to C$ if and only if $B \to A\backslash C$,
if $A \to B$ and $B \to C$ then $A \to C$.

There are a number of sequents easily derived from these axioms *via* these inference rules, including the following.

1. $(A/B) \bullet B \to A$
2. $A \bullet (A\backslash B) \to B$
3. $B \to (A/B)\backslash A$
4. $B \to A/(B\backslash A)$
5. $(A\backslash B)/C \leftrightarrow A\backslash(B/C)$
6. $(A/B)/C \leftrightarrow A/(C \bullet B)$
7. $(A/B) \bullet (B/C) \to (A/C)$
8. $A/B \to (A/C)/(B/C)$

Models of the syntactic calculus may be constructed from any set $\mathbf{M}$ equipped with a multiplication operator (denoted by $\cdot$), by interpreting the types as sets of elements of $\mathbf{M}$ and interpreting the operators of $\mathbf{L}$ as follows.

---

any possibility of confusion, I shall here use *type* in place of *category* in the latter sense." Lambek (1988, 297-298).

$A \bullet B = \{x{\cdot}y \in \mathbf{M} \mid x \in A \wedge y \in B\}$
$C/B = \{x \in \mathbf{M} \mid \forall y \in B, x{\cdot}y \in C\}$
$A\backslash C = \{y \in \mathbf{M} \mid \forall x \in A, x{\cdot}y \in C\}$

By appropriately interpreting sequents, such models are both sound and complete for the system **L**. Thus we can discuss the Lambek calculus either in terms of derivations within the deductive system **L** (what logicians would call its "proof theory") or in terms of its models, *i.e.* multiplicative systems ($\mathbf{M}$, $\cdot$) satisfying the axioms and rules.

Notice that the second and third axioms above form the *associative law*, which plays a crucial role with respect to the generative capacity of the syntactic calculus. In fact, there is some room for variation here: we could suppose that the multiplicative system ($\mathbf{M}$, $\cdot$) need not necessarily be associative. If the associative law holds, $\mathbf{M}$ is a *semigroup* allowing unbracketed strings, characterizing a strong notion of constituency[3]; if associativity is dropped, then only bracketed strings are allowed corresponding to standard grammatical constituents, but the types then increase in number and complexity[4]. $\mathbf{M}$ is a *monoid* if it is a semigroup with a unit element 1; letting $I = \{1\}$ we also get a unit in the induced syntactic model of **L**, since both $I \bullet A \rightarrow A$ and $A \rightarrow A \bullet I$ are derivable. For the present discussion, we shall consider the associative syntactic calculus, taking advantage of the full set of theorems (1)–(8) above.

Theorems (1)–(2) correspond to the cancellation rules (*functional application*) of AB grammar. However, further operations can be performed within the syntactic calculus, such as (7) which allows one to combine two adjacent functorial categories. The content of this rule, known as *functional composition*, from the analogy with the mathematical operation, is that a function from A to B combines with a function from B to C, to give a function from A to C; the rule is given for the right implication A/B, but also holds in the other direction: $A\backslash B$. The theorems (3) and (4) introduce the (left and right) rules to expand monadic types known as *type raising* or *type lifting* rules, and the expansion rule in (8) corresponds to the rule for expanding implicational types A/B advocated by Geach (the same holds for the other implication $A\backslash B$). The rule (6) is derivable on the basis of (5), which states that, according to associativity, parentheses may be omitted. Associativity is required also for the rules (7) and (8).

As pointed out by Lambek[5], when applying this notation to a natural

[3]This is the associative syntactic calculus introduced in Lambek (1958).
[4]The non-associative syntactic calculus is introduced in Lambek (1961).
[5]See *e.g.* Lambek (1988), (1989), (1993).

language, such as English or Italian, we are thinking of the multiplicative system, semigroup or monoid, freely generated by the words of the language under concatenation (expressed by the *tensor product*). Which of these three kinds of algebraic systems is relevant depends on whether one is interested in bracketed strings (*i.e.* trees), or unbracketed strings, or whether one wants to take the empty string into consideration.

We may think of the syntactic calculus (as a free multiplicative system, or semigroup or monoid, generated by the words of a language) as a universal system of rules in which a language-specific dictionary is embedded. Sets of strings of words will be called (*syntactic*) *types*. We will write S for the type of declarative sentences, and N for the type of names such as *Mary*, *John*. Starting from these basic types and the rules of the syntactic calculus, we can derive the types of the other sentential constituents.

For instance, taking the set $\mathbf{B}$ of *basic* categories given in (a), we can deduce the complex types for the English expressions in (b)–(d), so that the expressions obey the theorems derived above.

a. $\mathbf{B} = [N,\, NP,\, S]$

b. if **Mary** $\to NP$ and **Mary works** $\to S$,
then **works** $\to NP\backslash S$;

c. if **apple** $\to N$ and **an apple** $\to NP$,
then **an** $\to NP/N$;

d. if **Mary ate an apple** $\to S$ and **Mary** $\to NP$,
then **ate an apple** $\to NP\backslash S$;
if **an apple** $\to NP$, then **ate** $\to (NP\backslash S)/NP$.

The syntactic calculus will assign the type S (sentence) to a given string of words if and only if the dictionary assigns a type B_i to each word and the sequent $B_1 \bullet \cdots \bullet B_n \to S$ is a theorem of the syntactic calculus; that is, the combination of the assigned types multiply to the single type S , on the basis of the axioms and inference rules of the system.

0.2 Linguistic limitations: unbounded dependencies

Lambek grammar allows a relatively powerful system of rules, but several questions are still open in particular with reference to the treatment of *long distance dependencies* and *constituent conjunction*. These

phenomena may involve an inversion or *permutation* of the functor-argument order, a fact that does not follow directly from the syntactic calculus and has to be postulated as an additional axiom or rule. Permutations of the kind[6]:

$$X/Y\ Y \rightarrow Y\ X/Y \quad \textit{Permutation I: /}$$
$$X\ X\backslash Y \rightarrow X\backslash Y\ X \quad \textit{Permutation II: } \backslash$$

are in fact excluded by the axioms and inference rules of the syntactic calculus, which only allows the derivation of the (left and right)-oriented constituent structures: $X/Y\ Y$ and $X\ X\backslash Y$; to obtain the required inversion it is necessary to explicitly add a new inference rule of *permutation*[7]. As an example involving a simple sentence, consider Figure 1 where the locative PP occurs between the subject and the predicate:

a. Maria sul tavolo mette una tovaglia nuova.
Mary, on the table, is putting a new table cloth.

b. **(sul tavolo) mette (una tovaglia nuova)**

$$\dfrac{PP \qquad \dfrac{(VP/PP)/NP \qquad NP}{VP/PP}\ /}{VP}\ ?$$

FIGURE 1 Syntactic calculus: *Permutation*

(In this and subsequent figures, we have represented the evident derivation by a corresponding natural deduction style proof tree. We shall leave to the reader the simple exercise of translating this to the sequent style presentation of the syntactic calculus given earlier.) At the stage of the derivation in which the argument PP must combine with the corresponding predicate we are faced with two options, both implying an extension going beyond the expressive power of the syntactic calculus: either we need a rule that if $PP \bullet VP/PP \rightarrow VP$, then $PP \rightarrow VP/(VP/PP)$, or we need an axiom $PP \bullet VP/PP \rightarrow VP$ (under permutation).

[6] We frequently drop evident occurrences of the "times" $\bullet$, treating a string of types as their 'product".

[7] See van Benthem (1988), (1991), Moortgat (1988), (1997); the syntactic calculus is in fact non-commutative.

In the first case, a complex type is derived from the basic type PP which, inverting the functor-argument order, allows the combination with the predicate. Such a strategy is costly since the number of (left or right)-expanded types associated to the initial categories can indefinitely increase. Moreover, we lose the important relation holding between a *functional head*, such as a *ditransitive verb* $V = (VP/PP)/NP$ and its subcategorized arguments (PP and NP). The second solution consists in admitting some kind of permutation rule within the **L** grammar. Since this solution increases the generative power of the grammar, such rules need to be restricted to specific structural positions.

A typical environment in which we are faced with the difficulty mentioned above is represented by the structural configurations known as *unbounded dependencies*[8]. Consider the examples below and the derivations presented in Figure 2 and Figure 3:

a. Gianni ha detto che Maria ha perso il treno.
Gianni said that Mary had missed the train.

b. Che cosa hai detto che Maria ha perso?
What did you say that Mary had missed?

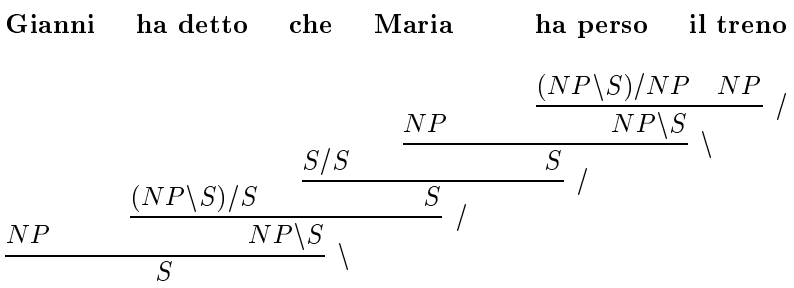

FIGURE 2 Syntactic calculus: *Unbounded dependency I*

The direct object of the embedded clause in the declarative sentence (a) occurs in initial sentence position within the corresponding *Wh*-question (b). Generative grammar analyzes this contrast as an effect of the transformation of *Wh*-movement[9]. From the point of view of

[8] See Gazdar and Pullum (1982), Gazdar et al. (1985), and Morrill (1994).
[9] See *e.g.* Chomsky (1965), (1973).

Che cosa Gianni ha detto che Maria ha perso?

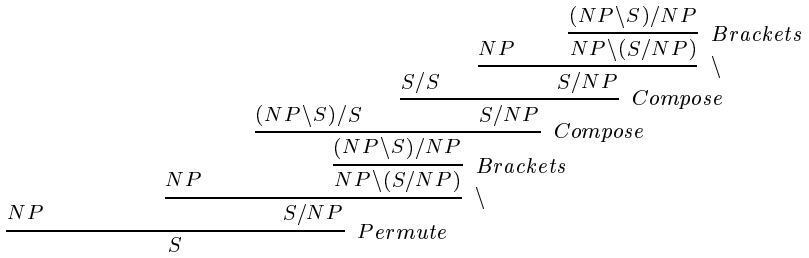

FIGURE 3 Syntactic calculus: *Unbounded dependency II*

categorial grammar, the direct combination and easy computation displayed in Figure 2, obtained by applying the rule of functional application in the two directions, is missed in the derivation of the unbounded dependency. As shown in Figure 3, four different types of rules are needed: *Brackets*, following from *Associativity*, *Composition*, to generate the complex verbal constituent with a missing NP-argument, (left)-*Application*, to combine with the embedded and the main subject NP, and *Permutation*, to 'close' the dependency and combine the leftmost NP argument, a *Wh*-pronoun in this case, with the complex predicate of type S/NP.

The set of rules of the associative syntactic calculus **L** are summarized in Table 1, where the last two lines include the permutation rules that can be added to extend the generative capacity of the system[10].

0.3 Decision procedure

The decision problem for the syntactic calculus consists in determining the set of strings that are theorems. By this we mean to find an effective algorithm for deciding whether a sentence, corresponding to a string of types, is deducible from the axioms and inference rules of the syntactic calculus.

As remarked by Lambek[11], in the systems of Ajdukiewicz and Bar-Hillel the corresponding question was easily answered, since there were

[10] However the addition of these to the axiomatic base has the critical effect of making the calculus **L** collapse into its permutation closed variant **LP**; see Moortgat (1988, 90-91).

[11] See Lambek (1988, 304), (2002).

(I)	$X/Y\ Y \to X$	Right application: /
(II)	$X\ X\backslash Y \to Y$	Left application: \
(III)	$X \to (Y/X)\backslash Y$	Expansion I: \
(IV)	$X \to Y/(X\backslash Y)$	Expansion I: /
(V)	$X\backslash Y \to (Z\backslash X)\backslash(Z\backslash Y)$	Expansion II: \
(VI)	$X/Y \to (X/Z)/(Y/Z)$	Expansion II: /
(VII)	$X/Y\ Y/Z \to X/Z$	Composition: /
(VIII)	$X\backslash Y\ Y\backslash Z \to X\backslash Z$	Composition: \
*(IX)	$X/Y\ Y \to Y\ X/Y$	Permutation I: /
*(X)	$X\ X\backslash Y \to X\backslash Y\ X$	Permutation II: \

TABLE 1 Rules of the syntactic calculus

only the contraction rules $(A/B) \bullet B \to A$ and $A \bullet (A\backslash B) \to B$, and of course the implicit *associativity*. The syntactic calculus, on the other hand, contains not only *contraction* rules, but also *expansion* rules, such as the *type lifting* rules and the *division* rules discussed above (III-VI in Table 1).

The decision problem was solved by Lambek by applying to the freely generated syntactic calculus an adaptation of the decision procedure discovered by Gentzen for intuitionistic propositional calculus[12]. It is not surprising that a similar procedure should work, since the syntactic calculus is essentially Heyting's intuitionistic propositional calculus from which the three *structural rules* of Gentzen: *weakening*, *contraction*, and *exchange* (see Table 2) have been excluded.

The formulation of axioms and inference rules of the syntactic calculus within Gentzen's two-sided sequent calculus is given in Table 3, where capital Greek letters stand for finite sequences $A_1, \ldots, A_n$[13]. In

[12] Gentzen (1934), Kleene (1952, XV).

[13] "All one has to do is to replace arrows $f\colon A \to B$ with *multi-arrows*

CONTRACTION	WEAKENING	EXCHANGE
$\dfrac{\Gamma, A, A, \Delta \ \to B}{\Gamma, A, \Delta \ \to B}$ (C)	$\dfrac{\Gamma, \Delta \ \to B}{\Gamma, A, \Delta \ \to B}$ (W)	$\dfrac{\Gamma, A, B, \Delta \ \to C}{\Gamma, B, A, \Delta \ \to C}$ (E)

TABLE 2 Gentzen's Structural Rules

a cut-free Gentzen-style system, it is clear whether a sequent $\Gamma \to S$ is derivable or not. But in general, formal proofs make use of the *cut* rule, which considerably complicates proof search. However, following Gentzen, one can prove that cuts can be eliminated[14]. "The proof of the cut elimination theorem for the syntactic calculus is easier and, in a sense, purer than the corresponding proof for the intuitionistic propositional calculus, because of the complete absence of Gentzen's so-called structural rules" (Lambek 1988, 306).

As an illustration, in Table 4 we show the proofs of the *Associative* rule (in one direction), and of the introduction of *Right Application*, presented in Lambek (1958), and the proofs of the *Identity* rule, with cut and without cut, given in Lambek (1988). In the proofs, the sequences Γ and Δ are assumed to be empty, and the tensor product is denoted, as we did earlier, by simple juxtaposition.

0.4 Generative capacity of Lambek grammars

The hierarchy of grammars defined by Chomsky suggests an interesting question: what is the status of the Lambek calculus within this hierarchy? Chomsky himself conjectured the *weak equivalence* of product-free

$f : A_1, A_2, \ldots, A_n \to B$. Leaving out the letter f we obtain what Gentzen calls a *sequent*." Lambek (1988, 304-305).

The equivalence of the full sequent calculus formulation of the syntactic calculus with the earlier presentation above arises by identifying a general sequent $A_1, A_2, \ldots, A_n \to B$ with the sequent $A_1 \bullet A_2 \bullet \ldots \bullet A_n \to B$.

[14]The proof of the *cut elimination theorem* can be done by induction on the occurrences of connectives within the principal formula of the cut rule. For systems without the symbol I, this was done in Lambek (1958). The inclusion of I was shown to cause no difficulty in Lambek (1969), where the proof was lifted to the categorical level, interpreting the sequents as multilinear operations. The proof is reconsidered in "Logic without Structural Rules. (Another Look at Cut Elimination)", Lambek (1993), and becomes more transparent by using terms, inductively defined, in place of operations.

AXIOM & CUT

$$A \to A \quad \text{(Axiom)} \qquad \frac{\Gamma \to A \quad \Delta, A, \Delta' \to B}{\Delta, \Gamma, \Delta' \to B}\ (\text{CUT})$$

• RULES

$$\frac{\Gamma, A, B, \Gamma' \to C}{\Gamma, A \bullet B, \Gamma' \to C}\ (L, \bullet) \qquad \frac{\Gamma \to A \quad \Delta \to B}{\Gamma, \Delta \to A \bullet B}\ (R, \bullet)$$

/ RULES

$$\frac{\Gamma \to B \quad \Delta, A, \Delta', \to C}{\Delta, A/B, \Gamma, \Delta' \to C}\ (L, /) \qquad \frac{\Gamma, B \to A}{\Gamma \to A/B}\ (R, /)$$

\ RULES

$$\frac{\Gamma \to A \quad \Delta, B, \Delta' \to C}{\Delta, \Gamma, A\backslash B, \Delta' \to C}\ (L, \backslash) \qquad \frac{A, \Gamma \to B}{\Gamma \to A\backslash B}\ (R, \backslash)$$

I RULES

$$\frac{\Gamma, \Delta \to A}{\Gamma, I, \Delta, \to A}\ (I) \qquad \to I$$

TABLE 3 Syntactic calculus : Gentzen Presentation

Lambek grammars and context-free grammars[15]. Before this, an important result was obtained by Gaifman, who proved the weak equivalence of Bar-Hillel categorial grammar (AB grammar) with context free grammars (CFG)[16]. Following Chomsky's suggestion, Cohen[17] gave a proof of the equivalence of product-free Lambek grammars and CFG's, but, as shown by Zielonka and Buszkowski, his reasoning was in fact incorrect. A new proof of the weak equivalence of the left-division and product-free Lambek grammar with CFG's was then given by

[15] Chomsky (1963); the term Lambek grammar is used here for syntactic calculus.
[16] Bar-Hillel, Gaifman and Shamir (1960).
[17] Cohen (1967).

PROOF OF ASSOCIATIVITY

$$\dfrac{\dfrac{A \to A \quad \dfrac{B \to B \quad C \to C}{B,\ C \to BC}}{A, B, C \to A(BC)}}{\dfrac{AB,\ C \to A(BC)}{(AB)C \to A(BC)}}$$

PROOF OF RIGHT APPLICATION

$$\dfrac{\dfrac{A \to A \quad B \to B}{A,\ B \to AB}\ /\ \dfrac{B \to B \quad AB \to C}{(AB)/C,\ B \to C}\ /_{\text{CUT}}}{\dfrac{A,\ B \to C}{A \to C/B}}$$

PROOF OF IDENTITY WITH CUT

$$\dfrac{\dfrac{A \to A}{A,\ I \to A}\quad \dfrac{\to I \quad A \to A}{A \to I\ A}}{A\ I \to I\ A}\ \text{CUT}$$

CUT-FREE PROOF OF IDENTITY

$$\dfrac{\dfrac{A \to A}{A,\ I \to A}}{\dfrac{\to I \quad A\ I \to A}{A\ I \to I\ A}}$$

TABLE 4 Proofs in *Intuitionistic Sequent Calculus*

Buszkowski[18].

On the basis of these results we know that context-free grammars are weakly equivalent to Lambek grammars, but the converse, corresponding to Chomsky's conjecture, is not trivial. A definitive answer was given by Pentus[19], who proved that *the languages generated by Lambek grammars are exactly the context-free languages.* We summarize the results concerning the generative power of the Lambek grammar in Table 5, where *L(G)* is the language of the grammar *G*.

As shown by Buszkowski (1993), the Gaifman theorem, establishing the equivalence of AB grammars (ABG) and CFG's, is the conjunction of two statements: (i) each ABG is equivalent to some CFG; (ii) each CFG is equivalent to some ABG, whose initial type assignment uses at most types of the form p, p/q, $(p/q)/r$ (where p, q, r are atomic types). The direction involved in each of these statements has been extended also to the Lambek grammars in several steps concluded by the Pentus theorem. The last line of the table introduces the important result of Kanazawa (1992), (1994), concerning the extensions of the Lambek calculus with additional connectives, in particular additive connectives.

[18] See Zielonka (1981), Buszkowski (1985), (1986); see also Buszkowski (1997).
[19] Pentus (1993), (1997). Pentus' proof is discussed in detail in Retoré (2000).

Two grammars G and H are *weakly equivalent* iff L(G) = L(H).

Gaifman theorem (1960) : The languages recognized by basic CGs (ABG) are the context-free languages; ABG's are weakly equivalent to CFG's .

Chomsky conjecture (1963) : The languages recognized by Lambek CG are the context-free languages; Lambek CGs are (weakly) equivalent to CFG's .

Buszkowski proof (1985), (1988) : Unidirectional Lambek CGs are (weakly) equivalent to context-free grammars .

Kandulski proof (1988) : Equivalence of non-associative Lambek CGs and context-free grammars .

Pentus proof (1993), (1997) : Lambek CGs generate only context-free languages, thus, Lambek CGs are (weakly) equivalent to context free grammars and also to basic CGs.

Buszkowski proof (1993) : for any product-free Lambek CG there is an equivalent basic CG (ABG).

Kanazawa (1992) : CGs based on the Lambek calculus with additive conjunction surpass the recognizing power of context-free grammars .

TABLE 5 Lambek Grammars : Generative Capacity

0.5 Lambek calculus and non-commutative linear logic

> There is an interesting prefiguration of linear logic in the literature, namely Lambek's *syntactic calculus*, introduced in 1958 to cope with certain questions of linguistics ... This system is based on a non-commutative $\otimes$ which in turn induces two linear implications. There would be no problems to enrich the system with additives & and $\oplus$, but the expressive power remains extremely limited. The missing items are exponentials and negation ... (Girard 1995, 6) [20]

Lambek's syntactic calculus corresponds to the multiplicative fragment of non-commutative intuitionistic linear logic (NILL). In fact the basic connectives of the syntactic calculus, the tensor product

[20] The problem of adding exponentials to Lambek CG's is discussed in this volume by P. de Groote

"•", the left division "\" and the right division "/", correspond respectively to the multiplicative conjunction $\otimes$ and the two implications of (NILL), "$\multimap$" (linear post-implication) and "$\circ\!\!-$" (linear retro-implication) which result from the exclusion of the structural rule of *exchange*.

Here are some examples of the logical types of the syntactic calculus and of their translations into non-commutative intuitionistic linear logic, starting from the basic types NP (noun phrase) and S (sentence):

Intransitive verb:	IV	$NP\backslash S$	$NP \multimap S$
Transitive verb:	TV	$(NP\backslash S)/NP$	$(NP \multimap S) \circ\!\!- NP$
Ditransitive verb:	DTV	$((NP\backslash S)/PP)/NP$	$((NP \multimap S) \circ\!\!- PP) \circ\!\!- NP$
Determiner phrase[21]:	Q_1	$S/(NP\backslash S)$	$S \circ\!\!- (NP \multimap S)$
Determiner phrase:	Q_2	$(S/NP)\backslash S$	$(S \circ\!\!- NP) \multimap S)$
Intransitive adverb:	IVA	$(NP\backslash S)\backslash(NP\backslash S)$	$(NP \multimap S) \multimap (NP \multimap S)$
Preposition:	P	PP/NP	$PP \circ\!\!- NP$

In the framework of intuitionistic non-commutative linear logic, the fact that a string of words of a language is a sentence of that language may be represented by writing a sequent with conclusion[22] S; *e.g.* the string of English words (*the, children, eat, apples*) corresponds to the sequent:

$$\textit{The children}\colon NP,\ \textit{eat}\colon TV,\ \textit{apples}\colon NP \ \rightarrow\ S$$

that formalizes the empirical observation that a transitive verb gives a sentence, when it is preceded and followed by a noun phrase. The corresponding type for the transitive verb is derivable by the logical rules of introduction of the two implications:

$$\textit{eat}\colon (NP \multimap S) \circ\!\!- NP$$

From this simple example it is easy to see that non-commutative linear logic is basically resource sensitive: the number of the linguistic resources and the order in which they are produced or consumed is always taken into account. In fact, non-commutative linear logic pays attention to the direction of the linguistic processes and does not try to avoid the obstacles met in communication (e.g. changing the word order by means of suitable transformations), but finds new ways of communication by changing the points of view from which the linguistic environments are considered.

[21] Q_1 denotes determiner phrases in subject position and Q_2, determiner phrases in object position.

[22] Sequents of intuitionistic non-commutative linear logic allow just one formula in the conclusions.

0.6 The papers in this volume

This volume offers a series of contributions to the perspectives on linguistics arising from the Lambek program[23]. Its intended audience is specialists: researchers, graduate students who already know about type-logical approaches to grammar, computer scientists and logicians who know about type theory and are interested in high-quality natural language applications. Although we hope this introductory essay will help the reader place this material in context, it will be clear that the subsequent papers will require expertise beyond what we have covered here. Beginners new to this framework will want to consult the papers in the various bibliographies, and will probably find the material challenging, but worth further investigation.

At the core of the volume is the continuing mathematical research on the syntactic calculus and its extensions, and with the corresponding fragments of linear logic. This is pursued here in several papers which appear in Section 2, "Grammar". M. Pentus considers conditions characterizing atomicity (whether a type is equivalent to an atomic type) in the Lambek calculus (allowing empty premises) as well as in constant-free noncommutative multiplicative linear logic. W. Buszkowski studies the (associative and non-associative) Lambek calculi enriched with "non-logical axioms", and gives a (new) proof that the former are undecidable, whereas the latter are decidable in polynomial time, and generate context-free languages.

As was mentioned in the quotation above by J-Y Girard, in linear logic one may re-introduce the structural rules in a controlled manner *via* the "exponential" modalities **!** ("of course!") and **?** ("why not?"). P. de Groote considers the corresponding extension of the product-free associative Lambek calculus, and shows any recursively enumerable language can be described by a categorial grammar based on this system (which in particular shows the system to be undecidable). Next, the paper by M. Fadda and G. Morrill develops a calculus combining features of the two variants Lambek originally presented in 1958 and 1961, one based on expressions as strings (*i.e.* as flat trees), the other based on expressions as binary-bracketed strings (*i.e.* as binary trees). Their new calculus configures expressions as bracketed strings in general (*i.e.* as general trees); they give a completeness theorem for this system in terms of pre-ordered bracketed monoids, and a correctness theorem in terms of a notion of "bracketed" proof nets (similar to those of linear

[23] In order to give readers the historical background, as well as a survey of important research in the field, at the end of this section we have provided an extended list of references, not all of which are referred to in the introduction.

logic).

Recently Lambek has introduced the notion of pregroup as a vehicle for understanding the grammars of natural languages; Buszkowski has shown that languages based on free pregroups are context-free. We end this section with three papers dealing with these pregroups. First, a paper by Lambek introduces the notion with examples of its use in categorial grammar. A short paper by M. Barr studies a family of examples of these pregroups (this paper is entirely mathematical). And finally, Moortgat and Oehrle consider combining the simple computational properties of pregroup grammars with the linguistic expressiveness of type logical grammar, which usually is rather more complex computationally.

This mathematical core section is framed by two sections. The first section, "Language", consists of two papers which offer an exploration of the fundamental subjects involved in the formal analysis of language. J. van Benthem considers the "fit" between the logical and mathematical machinery and the natural languages it attempts to analyse. His conclusion that "the love match brokered by Lambek between categorial grammar and natural language still has some romance to it, even though there are lots of interesting twists, misunderstandings and subplots to go before the happy ending" is a theme of this entire volume.

E. Keenan studies some entailment patterns in English and their semantic generalizations, giving a formal framework for their analysis. This paper illustrates the spirit of the program: while it does not draw directly on Lambek's work, it is inspired by his mix of linguistic observation and mathematical formulation.

The third section, "Language and Grammar", concludes the volume with some papers which we think reflect a synthesis of the concepts behind the program. A. Lecomte's paper represents a transition between Section 2 and Section 3 . He draws a connection between categorial grammar and minimalist grammar, in which syntax and semantics interact *via* two proof systems (one for each), so that syntactic operations are converted into semantical ones, and semantics acts as a correctness check for syntactic derivations.

B.S. Gillon offers a more traditionally presented paper on subject predicate order in classical Sanskrit. D. Bargelli gives a computational approach to the noun in Burushaski, using techniques developed by Lambek and his coauthors for other languages, from French to Arabic. M. Kandulski presents a survey of tree languages generated by categorial grammars, providing a context for many of the key technical results discussed in this introduction.

Finally G. Morrill illustrates the potential for geometry of language

and linguistic circuitry characterizing the syntactic structures of Lambek categorial grammar as proof nets of linear logic. With that paper, we end the volume, but clearly work within and around the Lambek program continues to enrich our understanding of language and grammar.

References

[1] Abrusci, V. M. 1991. Phase semantics and sequent calculus for pure Noncommutative Classical Linear Propositional Logic. *The Journal of Symbolic Logic*, 56:1403–1451.

[2] Abrusci, V. M. 1995. Noncommutative proof nets. In *Advances in Linear Logic*, eds. J. Y. Girard et al., 271–296. Cambridge: Cambridge University Press.

[3] Abrusci, V. M. 1996. Lambek Calculus, Cyclic Multiplicative-Additive Linear Logic, Noncommutative Multiplicative-Additive Linear Logic: language and sequent calculus. In *Proofs and Linguistic Categories. Applications of Logic to the Analysis and Implementation of Natural Language*, eds. V. M. Abrusci and C. Casadio, 21–48. Bologna: CLUEB.

[4] Ades, A. E. & M. J. Steedman. 1982. On the Order of Words, *Linguistics and Philosophy*, 4: 517-558.

[5] Ajdukiewicz, K. 1935. Die syntaktische Konnexität, *Studia Philosophica*, 1: 1–27. Eng. trans., Syntactic connexion. In *Polish Logic*, ed. S. McCall. Oxford: Clarendon Press, 1967.

[6] Bar-Hillel, Y. 1953. A quasi-arithmetical notation for syntactic description. *Language*, 29: 47–58. In *Language and Information Selected Essays on their Theory and Application*, ed. Y. Bar-Hillel, 61–74. Palo Alto: Addison-Wesley.

[7] Bar-Hillel, Y., C. Gaifman & E. Shamir 1960. On categorial and phrase structure grammars, *The Bulletin of the Research Council of Israel*, 1-16. In *Language and Information: Selected Essays on their Theory and Application*, ed. Y. Bar-Hillel, 99-115, 1964. Palo Alto: Addison-Wesley.

[8] Van Benthem, J. 1983. The semantics of variety in categorial grammar. Report 83-29, Simon Fraser University, Burnaby (B.C.), Canada. In *Categorial Grammar*, eds. W. Buszkowski, W. Marciszewski & J. van Benthem, 1988. Amsterdam: Benjamin.

[9] Van Benthem, J. 1988. *The Lambek Calculus.* In *Categorial Grammars and Natural Language Structures*, eds. R. T. Oehrle et al., 35-68. Dordrecht: Reidel.

[10] Van Benthem, J. 1991. *Language in Action. Categories, Lambdas, and Dynamic Logic.*, Amsterdam: North Holland.

[11] Buszkowski, W. 1985. The equivalence of unidirectional Lambek categorial grammars and context free grammars, *Zeitschr. f. math. Logik und Grund. d. Math.*, 31: 308–384.

[12] Buszkowski, W. 1986. Completeness results for Lambek Syntactic Calculus, *Zeitschr. f. math. Logik und Grund. d. Math.*, 32: 13-28.

[13] Buszkowski, W. 1988. Generative power of categorial grammars. In *Categorial Grammars and Natural Language Structures*, eds. R. T. Oehrle et al.. Dordrecht: Reidel.

[14] Buszkowski, W. 1993. On the equivalence of Lambek categorial grammar and basic categorial grammars, *ILLC prepublication Series*, vol. LP-93-07, ILLC, Amsterdam.

[15] Buszkowski, W. 1997. Mathematical linguistics and proof theory. In *Handbook of Logic and Language*, eds. J. van Benthem & A. ter Meulen, 683-736. Amsterdam: Elsevier.

[16] Casadio, C. & J. Lambek. 2001. An algebraic analysis of clitic pronouns in Italian. In *Logical Aspects of Computational Linguistics*, eds. P. de Groote, G. Morrill and C. Retoré, 110–124. Berlin: Springer-Verlag.

[17] Casadio, C. & J. Lambek. 2002. A tale of four grammars, *Studia Logica*, 71(2):315-329. Special Issue edited by W. Buszkowski.

[18] Chomsky, N. 1957. *Syntactic Structures*. The Hague: Mouton.

[19] Chomsky, N. 1959. On certain formal properties of grammars, *Information and Control*, 2: 137-167.

[20] Chomsky, N. 1963. Formal properties of grammars. In Luce R. D., R. R. Bush and E. Galanter (eds.)(1963), *Handbook of Mathematical Psychology*, J. Wiley and Sons, New York.

[21] Chomsky, N. 1965. *Aspects of the Theory of Syntax*. Cambridge (Mass.): The MIT Press.

[22] Chomsky, N. 1973. Conditions on transformations. In *A Festchrift for Morris Halle*, eds. Anderson S. & P. Kiparsky. New York: Holt, Rinehart and Winston.

[23] Church, A. 1940. A formulation of the simple theory of types, *The Journal of Symbolic Logic*, 5:56-69.

[24] Cohen, J. M. 1967. The equivalence of two concepts of categorial grammar, *Information and Control*, 10: 475-484.

[25] Eilenberg, S. & S. Mac Lane. 1945. General theory of natural equivalences, *Trans. Amer. Math. Soc.*, 58: 231-294.

[26] Gazdar G. & G. K. Pullum. 1982. *Generalized Phrase Structure Grammar: A Theoretical Synopsis*. Bloomington: I.U.L.C. .

[27] Gazdar, G., E. Klein, J. Pullum, & I. Sag. 1985. *Generalized Phrase Structure Grammar*. Oxford: Blackwell.

[28] Gentzen, G. 1934. Untersuchungen über das logische Schliessen, *Mathematische Zeitschrift*, 39, Eng. transl. in M. E. Szabo (ed.)(1969), *The Collected Papers of Gerhard Gentzen*, 68-131, North Holland, Amsterdam.

[29] Girard, J.-Y. 1995. Linear logic: its syntax and semantics. In *Advances in Linear Logic*, eds. J. -Y. Girard et al., 271–296. Cambridge: Cambridge University Press.

[30] Grishin, V. N. 1983. On a generalization of the Ajdukiewicz-Lambek system. In *Studies in Nonclassical Logics and Formal Systems*, 315-343. Moscow: Nauka. Eng. trans. by D. Cubric, edited by author. In *New Perspectives in Logic and Formal Linguistics*, eds. Abrusci V. M. & C. Casadio, 2002. Roma: Bulzoni Editore.

[31] Kanazawa, M. 1992. The Lambek calculus enriched with additional connectives, *Journal of Logic, Language and Information*, 1: 141-171.

[32] Kanazawa, M. 1994. *Learnable Classes of Categorial Grammars.* Ph.D. Dissertation, Stanford. Stanford, California: CSLI Publications/FoLLI, 1998.

[33] Kandulski, M. 1988. The equivalence of nonassociative Lambek Categorial Grammars and Context Free Grammars, *Zeit. Math. Logik und Grund. Math.*, 34:41-52.

[34] Kleene, S. C. 1952. *Introduction to Metamathematics.* New York: Van Nostrand & Amsterdam: North Holland.

[35] Lambek, J. 1958. The mathematics of sentence structure, *American Mathematical Monthly*, 65: 154–170.

[36] Lambek, J. 1961. On the calculus of syntactic types. In *Structure of Language and its Mathematical Aspects*, ed. R. Jacobson. AMS, Providence.

[37] Lambek, J. 1969. Deductive systems and categories II. Springer LNM 87, 76-122.

[38] Lambek, J. 1988. Categorial and categorical grammars. In *Categorial grammars and Natural Language Structures*, eds. R. T. Oehrle et al., 297-317. Dordrecht: Reidel.

[39] Lambek, J. 1989. On a connection between algebra, logic and linguistics, *Diagrammes*, 22: 59-75.

[40] Lambek, J. 1993. Logic without structural rules. (Another look at cut elimination). In *Substructural Logics*, eds. K. Došen & P. Schroeder-Heister, 179-206. Oxford: Oxford University Press.

[41] Lambek, J. 1999. Type grammars revisited. In *Logical Aspects of Computational Linguistics*, eds. A. Lecomte, F. Lamarche & G. Perrier, 1–27. Springer LNAI 1582.

[42] Lambek, J. 2000. Pregroups: a new algebraic approach to sentence structure. In *Recent Topics in Mathematical and Computational Linguistics*, eds. C. Martin-Vide & G. Paun. Bucharest: Editura Academici Române.

[43] Lambek, J. 2001. Type grammars as pregroups, *Grammars* 4: 21-39.

[44] Lambek, J. 2002. Réminescences Catégorielles. In *Les grammaires catgorielles. Histoire et perspectives contemporaines*, ed. B. Godart-Wendling,*Langages*, 146. Paris: Larousse.

[45] Moortgat, M. 1988. *Categorial Investigations. Logical and Linguistic Aspects of the Lambek Calculus*, Dordrecht: Foris.

[46] Moortgat, M. 1997. Categorial Type Logics. In *Handbook of Logic and Language*, eds. J. van Benthem & A. ter Meulen, 93-177. Amsterdam: Elsevier Science Publishers.

[47] Morrill, G. 1994. *Type Logical Grammar*. Dordrecht: Kluwer.

[48] Oehrle, R. T., E. Bach & D. Wheeler, eds. 1988. *Categorial Grammars and Natural Language Structures*. Dordrecht: Reidel.

[49] Pentus, M. 1993. Lambek grammars are context free. In *Proceedings Eighth LICS Conference*, 429–433, Montreal.

[50] Pentus, M. 1997. Product-free Lambek calculus and context-free grammars, *Journal of Symbolic Logic*, 62: 648–660.

[51] Retoré, C. 2000. *The Logic of Categorial Grammar*, Lecture Notes at ESSLLI 2000, Birmingham.

[52] Zielonka, W. 1981. Axiomatizability of Ajdukiewicz-Lambek calculus by means of cancellation schemes, *Zeitschr. f. math. Logik und Grund. d. Math.*, 27: 215-224.

Acknowledgments

The present collection would have not been possible without the scholarship of Joachim Lambek; he has been the inspiration for the volume and all the results it contains. We thank him for all that as well as for many specific pieces of advice and help he has provided during the preparation of this book. All who have worked with him will attest to the fact that he is a valued collegue and cherished friend; we are pleased and honoured to dedicate this book to him as a celebration of his contributions to mathematical linguistics.

We also wish to thank all the anonymous referees who in the course of two years have offered their help in reading and commenting on the papers collected in this volume. We believe all these works offer stimulating insight into the research field of type logical grammars and Lambek's theoretical contributions.