INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Project-Team CALLIGRAMME

Linear Logic, Proof Nets and Categorial Grammars

Lorraine

THEME SYM

Activity Report

2004

Table of contents

1. Team

Head of project-team
Philippe de Groote [DR INRIA]

Vice-Head of project-team
François Lamarche [DR INRIA]

Administrative assistant
Laurence Benini [INRIA]

Staff members INRIA
Bruno Guillaume [CR]
Sylvain Pogodalla [CR]

Staff member Université Henri Poincaré-Nancy 1
Adam Cichon [Professor]

Staff members Institut National Polytechnique de Lorraine
Jean-Yves Marion [Professor, École des Mines de Nancy]
Guillaume Bonfante [Lecturer, École des Mines de Nancy]

Staff member Université Nancy 2
Guy Perrier [Lecturer]

Visiting scientist
Denys Duchier [INRIA Visiting scientist]

Post-doctoral fellows
Jérôme Besombes [Université Henri Poincaré, teaching assistant at ESIAL]
Jean-Yves Moyen [Teaching assistant at Université Henri Poincaré since Sept. 1, former ENS fellow]
Lutz Straßburger [INRIA postdoctoral fellow]

Ph. D. Students
Emmanuel Hainry [ENS fellow, joint thesis with project-team Protheo]
Joseph Leroux [MESR fellow]
Paulin Jacobé de Naurois [ENS fellow, joint thesis with project-team Protheo and the City University of Hong Kong (F. Cucker)]
Romain Péchoux [MESR fellow, since Sept. 1]
Sylvain Salvati [MESR fellow, INPL]

2. Overall Objectives

Keywords: *categorial grammar*, *implicit complexity*, *lambda calculus*, *linear logic sequent calculus*, *proof nets*, *semantics of natural languages*, *syntactic analysis of natural languages*, *type theory*.

Project-team Calligramme's aim is the development of tools and methods that stem from proof theory, and in particular, linear logic. Two fields of application are emphasized: in the area of computational linguistics, the modelling of the syntax and semantics of natural languages; in the area of software engineering the study of the termination and complexity of programs.

3. Scientific Foundations

3.1. Introduction

Project-team Calligramme's research is conducted at the juncture of mathematical logic and computer science. The scientific domains that base our investigations are proof theory and the λ-calculus, more specifically linear logic. This latter theory, the brainchild of J.-Y. Girard [41] results from a finer analysis of the part played by structural rules in Gentzen's sequent calculus [40]. These rules, traditionally considered as secondary, specify that the sequences of formulas that appear in sequents can be treated as (multi) sets. In the case of intuitionistic logic, there are three of them:

$$\frac{\Gamma \vdash C}{\Gamma, A \vdash C}\ \text{(Weakening)} \qquad \frac{\Gamma, A, A \vdash C}{\Gamma, A \vdash C}\ \text{(Contraction)} \qquad \frac{\Gamma, A, B, \Delta \vdash C}{\Gamma, B, A, \Delta \vdash C}\ \text{(Exchange)}$$

These rules have important logical weight: the weakening rule embodies the fact that some hypotheses may be dropped during a derivation; in a similar fashion the contraction rule specifies that any hypothesis can be used an unlimited number of times; as for the exchange rule it stipulates that no order of priority holds between hypotheses. Thus, the presence of the structural rules in the ordinary sequent calculus strongly conditions the properties of the logic that results. For example, in the Gentzen-style formulations of classical or intuitionistic logic, the contraction rule by itself entails the undecidability of the predicate calculus. In the same manner, the use of the weakening and contraction rules in the right half of the sequent in classical logic is responsible for the latter's non-constructive aspects.

According to this analysis, linear logic can be understood as a system that conciliates the constructivist aspect of intuitionistic logic and the symmetry of classical logic. As in intuitionistic logic the constructive character comes from the banning of the weakening and contraction rules in the right part of the sequent. But simultaneously, in order to preserve symmetry in the system, the same rules are also rejected in the other half.

	Propositional linear logic			
	Rudimentary linear logic			
	Negation	Multiplicatives	Additives	Exponentials
Negation	$A^{\perp}$			
Conjunction		$A \otimes B$	$A \& B$	
Disjunction		$A \parr B$	$A \oplus B$	
Implication		$A \multimap B$		
Constants		$1, \perp$	$\top, 0$	
Modalities				!A, ?A

The resulting system, called *rudimentary linear logic*, presents many interesting properties. It is endowed with four logical connectors (two conjunctions and two disjunctions) and the four constants that are their corresponding units. It is completely symmetrical, although constructive, and equipped with an involutive negation. As a consequence, rules similar to De Morgan's law hold in it.

In rudimentary linear logic, any hypothesis must be used once and only once during a derivation. This property, that allows linar logic to be considered as a resource calculus, is due, as we have seen, to the rejection of structural rules. But their total absence also implies that rudimentary linear logic is a much weaker system than intuitionistic or classical logic. Therefore, in order to restore its strength it is necessary to augment the system with operators that recover the logical power of the weakening and contraction rules. This is done via two modalities that give tightly controlled access to the structural rules. Thus, linear logic does not question the usefulness of the structural rules, but instead, emphasizes their logical importance. In fact, it rejects them as epitheoretical rules [37] to incorporate them as logical rules that are embodied in new connectors. This original idea is what gives linear logic all its subtlety and power.

The finer decomposition that linear logic brings to traditional logic has another consequence: the Exchange rule, which so far has been left as is, is now in a quite different position, being the only one of the traditional structural rules that is left. A natural extension of Girard's original program is to investigate its meaning, in other words, to see what happens to the rest of the logic when Exchange is tampered with. Two standard algebraic laws are contained in it: commutativity and associativity. Relaxing these rules entails looking for non-commutative, and non-associative, variants of linear logic; there are now several examples of these. The natural outcome of this proliferation is a questioning of the nature of the structure that binds formulas together in a sequent: what is the natural general replacement of the notion of (multi) set, as applied to logic? Such questions are important for Calligramme and are addressed, for example, in [6].

The activities of project-team Calligramme are organized around three research actions:

- Proof nets, sequent calculus and typed λ-calculi;
- Grammatical formalisms;
- Implicit complexity of computations.

The first one of these is essentially theoretical, the other two, presenting both a theoretical and an applied character, are our privileged fields of application.

3.2. Proof Nets, Sequent Calculus and Typed Lambda Calculi

Keywords: *Curry-Howard isomorphism*, *denotational semantics*, *lambda calculus*, *proof nets*, *sequent calculus*, *type theory*.

Participants: Guillaume Bonfante, Philippe de Groote, Bruno Guillaume, François Lamarche, Guy Perrier, Sylvain Pogodalla, Lutz Straßburger.

The aim of this action is the development of the theoretical tools that we use in our other research actions. We are interested, in particular, in the notion of formal proof itself, as much from a syntactical point of view (sequential derivations, proof nets, λ-terms), as from a semantical point of view.

Proof nets are graphical representations (in the sense of graph theory) of proofs in linear logic. Their role is very similar to lambda terms for more traditional logics; as a matter of fact there are several back-and-forth translations that relate several classes of lambda terms with classes of proof nets. In addition to their strong geometric character, another difference between proof nets and lambda terms is that the proof net structure of a proof of formula T can be considered as structure which is *added* to T, as a coupling between the atomic formula nodes of the usual syntactic tree graph of T. Since not all couplings correspond to proofs of T there is a need to distinguish the ones that do actually correspond to proofs; this is called a *correctness criterion*.

The discovery of new correctness criteria remains an important research problem, as much for Girard's original linear logic as for the field of non-commutative logics. Some criteria are better adapted to some applications than others. In particular, in the case of automatic proof search, correctness criteria can be used as invariants during the inductive process of proof construction.

The theory of proof nets also presents a dynamic character: cut elimination. This embodies a notion of normalization (or evaluation) akin to β-reduction in the λ-calculus.

As we said above, until the invention of proof nets, the principal tool for representing proofs in constructive logics was the λ-calculus. This is due to the Curry-Howard isomorphism, which establishes a correspondence between natural deduction systems for intuitionistic logics and typed λ-calculi.

Although the Curry-Howard isomorphism owes its existence to the functional character of intuitionistic logic, it can be extended to fragments of classical logic. It turns out that some constructions that one meets in functional progamming languages, such as control operators, can presently only be explained by the use of deduction rules that are related to proof by contradiction [42]

This extension of the Curry-Howard isomorphism to classical logic and its applications has a perennial place as research field in the project.

3.3. Categorial Grammars and Dependency Grammars

Keywords: *Montague semantics*, *categorial grammar*, *dependency grammar*, *semantics of natural languages*, *syntactic analysis of natural languages*, *syntactic inference*, *tree description*.

Participants: Jérôme Besombes, Guillaume Bonfante, Denys Duchier, Philippe de Groote, Bruno Guillaume, François Lamarche, Joseph Leroux, Jean-Yves Marion, Guy Perrier, Sylvain Pogodalla, Sylvain Salvati, Lutz Straßburger.

Lambek's syntactic calculus, which plays a central part in the theory of categorial grammars, can be seen a posteriori as a fragment of linear logic. As a matter of fact it introduces a mathematical framework that enables extensions of Lambek's original calculus as well as extensions of categorial grammars in general. The aim of this work is the development of a model, in the sense of computational linguistics, which is more flexible and efficient than the presently existing categorial models.

The relevance of linear logic for natural language processing is due to the notion of resource sensivity. A language (natural or formal) can indeed be interpreted as a system of resources. For example a sentence like *The man that Mary saw Peter slept* is incorrect because it violates an underlying principle of natural languages, according to which verbal valencies must be realized once and only once. Categorial grammars formalize this idea by specifying that a verb such as saw is a resource which will give a sentence S in the presence of a nominal subject phrase, NP, and only one direct object NP. This gives rise to the following type assigment:

Mary, Peter:		NP
saw		$(NP \setminus S)/NP$

where the slash ($/$) and the backslash ($\setminus$) are interpreted respectively as fraction pairings that simplify to the right and to the left, respectively. However we notice very soon that this simplification scheme, which is the basis of Bar-Hillel grammars [32] is not sufficient.

Lambek solves this problem by suggesting the interpretation of slashes and backslashes as implicative connectors [43][44]. Then not only do they obey the *modus ponens* law which turns out to be Bar-Hillel's simplification scheme

$$\frac{\Gamma \vdash A \quad \Delta \vdash A\backslash B}{\Gamma, \Delta \vdash B} \text{ (modus ponens)} \qquad \frac{\Gamma \vdash B/A \quad \Delta \vdash A}{\Gamma, \Delta \vdash B} \text{ (modus ponens)}$$

but also the introduction rules:

$$\frac{A, \Gamma \vdash B}{\Gamma \vdash A\backslash B} \setminus\text{-intro} \qquad \frac{\Gamma, A \vdash B}{\Gamma \vdash B/A} /\text{-intro}$$

The Lambek calculus does have its own limitations. Among other things it cannot treat syntactical phenomena like medial extraction and crossed dependencies. Thus the question arises: how can we extend the Lambek calculus to treat these and related problems? This is where linear logic comes into play, by offering an adequate mathematical framework for attacking this question. In particular proof nets appear as the best adapted approach to syntactical structure in the categorial framework.

Proof nets offer a geometrical interpretation of proof construction. Premises are represented by proof net fragments with inputs and outputs which respectively model needed and offered resources. These fragments must then be combined by pairing inputs and outputs according to their types. This process can also be interpreted in a model-theoretical fashion where fragments are regarded as descriptions for certain class of models: the intuitionistic multiplicative fragment of linear logic can be interpreted on directed acyclic graphs, while for the implicative fragment, trees suffice [48].

This perspective shift from proof theory to model theory remains founded on the notion of resource sensitivity (e.g. in the form of polarities and their neutralization) but affords us the freedom to interpret these ideas in richer classes of models and leads to the formalism of Interaction Grammars. For example:

- where previously we only considered simple categories with polarities, we can now consider complex categories with polarized features.
- we can also adopt more expressive tree description languages that allow us to speak about dominance and precedence relations between nodes. In this fashion we espouse and generalize the monotonic version of Tree Adjoining Grammars (TAG) as proposed by Vijay-Shanker [53].
- contrary to TAG where tree fragments can only be inserted, Interaction Grammars admit models where the interpretations of description fragments may overlap.

Another grammatical framework which embraces both the notion of resource sensitivity and the interpretational perspective of model theory is dependency grammar.

Dependency grammar is predicated on the notion of an asymmetrical relation of (syntactic or semantic) dependency. This analytical idea has a very long history dating back at least to Panini (450 BC) and the ancient logicians and philosophers, and made its way into european medieval linguistics under the spreading influence of the Arabic linguistic tradition. The modern notion of dependency grammar is usually attributed to Tesnière [52] and has been further developed in such stratificational formalizations as Functional Generative Description (FGD) [51] and Meaning-Text Theory [46].

The main formal notions of dependency grammar that reflect and embody its sensitivity to resources are subcategorization (sensitivity to syntactic resources) and valency (sensitivity to semantic resources). It should be noted that these core DG concepts of head/dependent asymmetry and subcategorization/valency have been adopted by other grammatical formalisms. In the categorial grammar tradition, these notions appear respectively as directional functional application and categorial types. In HPSG, they give rise to the notion of headed structures, head daughters, and SUBCAT lists.

Dependency grammar permits non-projective analyses, i.e. where branches may cross. For this reason, it holds a special appeal for languages with free or freer word-order than French or English, such as German, Russian, Czech... and is certainly one reason for the strong renewal of interest in DG in recent years.

Duchier [39] proposed a new formulation of DG with a model theoretic interpretation that has a natural reading as a concurrent constraint program. This approach, based on set constraints, offers an efficient treatment of both lexical and structural ambiguity, and produces parsers that take full effective advantage of constraint propagation to achieve very good practical performance.

Extending this approach, Duchier and Debusmann [38] proposed Topological Dependency Grammar (TDG) as a means to equip DG with a theory of word-order. TDG adopts the ID/LP[1] perspective and explains word-order phenomena as arising through the interactions of a non-ordered ID tree of syntactic dependencies and an ordered and projective LP tree of topological dependencies. They provided a detailed, yet simple, account of the challenging word-order phenomena in the verbal complex of German verb final sentences. This extension retains the computational advantages of model elimination through powerful constraint propagation.

3.4. Implicit Complexity of Computations

Keywords: *Complexity theory*, *Curry-Howard isomorphism*, *lambda calculus*, *termination orders*, *theory of programming*, *types*.

Participants: Guillaume Bonfante, Adam Cichon, Paulin Jacobé de Naurois, Jean-Yves Marion, Jean-Yves Moyen, Romain Péchoux.

The construction of software which is certified with respect to its specifications is more than ever a great necessity. It is crucial to ensure, while developing a certified program, the quality of the implementation in terms of efficiency and computational resources. Implicit complexity is an approach to the analysis of the resources that are used by a program. Its tools come essentially from proof theory. The aim is to compile a program while certifying its complexity.

The meta-theory of programming traditionally answers questions with respect to a specification, like termination. These properties all happen to be *extensional*, that is, described purely in terms of the relation between the input of the program and its output. However, other properties, like the efficiency of a program and the resources that are used to effect a computation, are excluded from this methodology. The reason for this is inherent to the nature of the questions that are posed. In the first case we are treating extensional properties, while in the second case we are inquiring about the manner in which a computation is effected. Thus, we are interested in *intensional* properties of programs.

The complexity of a program is a measure of the resources that are necessary for its execution. The resources taken into account are usually time and space. The theory of complexity studies the problems and the functions

[1]Immediate Dependence / Linear Precedence

that are computable given a certain amount of resources. One should not identify the complexity of functions with the complexity of programs, since a function can be implemented by several programs. Some are efficient, others are not.

One achievement of complexity theory is the ability to tell the "programming expert" the limits of his art, whatever the amount of gigabytes and megaflops that are available to him. Another achievement is the development of a mathematical model of algorithmic complexity. But when facing these models the programming expert is often flabbergasted. There are several reasons for this; let us illustrate the problem with two examples. The linear acceleration theorem states that any program which can be executed in time $T(n)$ (where n is the size of the input) can be transformed into an equivalent problem that can be executed in time $\epsilon T(n)$, where ϵ is "as small as we want". It turns that this result has no counterpart in real life. On the other hand a function is feasible if it can be calculated by a program whose complexity is acceptable. The class of feasible functions is often identified with the class Ptime of functions that are calculable in polynomial time. A typical kind of result is the definition of a progamming language LPL and the proof that the class of functions represented by that language is exactly the class Ptime. This type of result does not answer the programming expert's needs because the programming language LPL does not allow the "right algorithms", the ones he uses daily. The gulf between the two disciplines is also explained by differences in points of view. The theory of complexity, daughter of the theory of computatibility, has conserved an extensional point of view in its modelling practices, while the theory of programming is intrinsically intensional.

The need to reason on programs is a relevant issue in the process of software development. The certification of a program is an essential property, but it is not the only one. Showing the termination of a program that has exponential complexity does not make sense with respect to our reality. Thus arises the need to construct tools for reasoning on algorithms. The theory of implicit complexity of computations takes a vast project to task, namely the analysis of the complexity of algorithms.

4. Application Domains

4.1. Modelling the Syntax and Semantics of Natural Languages

4.1.1. Abstract Categorial Grammars

Abstract Categorial Grammars (ACGs) are a new categorial formalism based on Girard's linear logic. This formalism, which sticks to the spirit of current type-logical grammars, offers the following features:

- Any ACG generates two languages, an abstract language and an object language. The abstract language may be thought as a set of abstract grammatical structures, and the object language as the set of concrete forms generated from these abstract structures. Consequently, one has a direct control on the parse structures of the grammar.
- The langages generated by the ACGs are sets of linear λ-terms. This may be seen as a generalization of both string-langages and tree-langages.
- ACGs are based on a small set of mathematical primitives that combine via simple composition rules. Consequently, the ACG framework is rather flexible.

Abstract categorial grammars are not intended as yet another grammatical formalism that would compete with other established formalisms. It should rather be seen as the kernel of a grammatical framework in which other existing grammatical models may be encoded.

4.1.2. Interaction Grammars

Interaction Grammars (IGs) are a linguistic formalism that aims at modelling both the syntax and the semantics of natural languages according to the following principles:

- An IG is a monotonic system of constraints, as opposed to a derivational/transformational system, and this system is multidimensional: at the syntactic level, basic objects are tree descriptions and at the semantic level, basic objects are Directed Acyclic Graph descriptions.
- The synchronization between the syntactic and the semantic levels is realized in a flexible way by a partial function that maps syntactic nodes to semantic nodes.
- Much in the spirit of Categorial Grammars, the resource sensitivity of natural language is built-in in the formalism: syntactic composition is driven by an operation of cancellation between polarized morpho-syntactic features and in parallel, semantic composition is driven by a similar operation of cancellation between polarized semantic features.

The formalism of IG stems from a reformulation of proof nets of Intuitionisitc Linear Logic (which have very specific properties) in a model-theoretical framework [48] and it was at first designed for modelling the syntax of natural languages [49].

4.1.3. Dependency Grammars

Dependency grammar (DG) is a resource sensitive grammar formalism related to categorial grammar. An important appeal of DG is that it naturally permits non-projective analyses and is thus especially well suited for languages with free or freer word-order (e.g. Czech, German, Russian, etc...). However, free(r) word-order poses an especially thorny challenge for processing: all efficient parsing algorithms rely fundamentally on properties of adjacency and projectivity. Duchier [39] described a declarative formulation of valid DG analyses, and showed how effective model elimination using constraint programming technology could serve as a practical foundation for DG parsing.

Of course, not all languages have free(r) word-order, and even for those, word-order is not really entirely free. The next challenge for DG was to permit an account of word-order. While this had been attempted in the past, no proposal to date was especially appealing. Duchier and Debusmann [38] proposed Topological Dependency Grammar (TDG) which took inspiration in the topological field theory that is at the heart of traditional German descriptive syntax. In TDG, an analysis consists of two trees: a non-ordered ID tree of syntactic dependencies and an ordered and projective LP tree of topological dependencies. The ID tree is related to the LP tree through a process of emancipation. Thus, in TDG, macroscopic word-order phenomena are not modeled directly, rather they are seen as emerging from (1) simple lexical constraints and (2) from the mutually constraining interactions of the ID and LP trees.

4.1.4. Meta-grammars and lexical resources

Every lexicalized grammar formalism (tree adjoining grammar, categorial grammar, interaction grammar, dependency grammar) is faced with the problem of organizing and structuring the lexicon. On the one hand, there is the pragmatic consideration that the lexicon should be modularly organized so as to make it easier to develop, maintain and extend. On the other hand it should also be possible to express linguistic generalizations (e.g. passivisation schemata).

Marie-Hélène Candito proposed and developed a meta-grammatical approach for lexicalized TAGs (Tree Adjoining Grammars) that generated a lot of interest. Unfortunately, it also exhibited a number of infelicitous properties, in particular difficulties in managing named entities and implicit crossings.

4.2. Termination and complexity of programs

The theory of implicit complexity is quite new and there are still many things to do. So, it is really important to translate current theoretical tools into real applications; this should allow to validate and guide our hypotheses. In order to do so, three directions are being explored.

1. First order functional programming. A first prototype, called ICAR has been developed and should be integrated into ELAN (http://elan.loria.fr).
2. Extracting programs from proofs. Here, one should build logical theories in which programs extracted via the Curry-Howard isomorphism are efficient.
3. Application to mobile code system. This work starts in collaboration with the INRIA Crystal and Mimosa project-teams.

5. Software

5.1. Leopar

LEOPAR is a parser for natural languages which is based on the formalism of Interaction Grammars (IG) [49]. The first release has been developed by G. Bonfante, B. Guillaume and G. Perrier. G. Bonfante, B. Guillaume, G. Perrier and S. Pogodalla have worked on the current version. It uses a parsing principle, called "electrostatic parsing" which is based on neutralizing opposite polarities. A positive polarity corresponds to an available linguistic constituent and a negative one to an expected constituent.

Parsing a sentence with an Interaction Grammar (IG) consists in first selecting a lexical entry for each of its words, then in merging all selected descriptions—tree descriptions à la Vijay-Shanker—into a unique one which represents a syntactic description of the sentence. The criterion for success is that this ultimate description is a neutral tree description. As IG are based on under-specified trees, LEOPAR uses some specific and non-trivial data-structures and algorithms.

The electrostatic principle has been intensively considered in LEOPAR. The theoretical problem of parsing IGs is NP-complete; the indeterminism usually associated to NP-completeness is present at two levels: when a description for each word is selected from the lexicon, and when a choice of what nodes to merge is made. Polarities have shown their efficiency in pruning the search tree for these two steps.

For the first step (tagging the words of the sentence), we forget the structure of description, and only keep the feature structures. In this case, parsing inside the formalism is greatly simplified because composition rules reduce to the neutralization of two labels $+l$ and $-l$. As a consequence, parsing reduces to a counting of positive and negative polarities present in the selected tagging for every label l: every positive label counts for +1 and every negative label for –1, the sum must be 0.

for the second step (node-merging phase), polarities are used to cut off parsing branches whose trees contain too many uncancelled polarities.

LEOPAR's first release is available on the web under the GNU General Public License (http://www.loria.fr/equipes/calligramme/leopar/). The current implementation is provided with a small grammar for French and a lexicon suited for this grammar. The grammar contains 31 descriptions. Despite its small size, it covers several non trivial linguistic phenomena (relative clauses, negation, pied-piping in the relatives...). The lexicon contains 78 entries.

5.2. XMG

The eXtensible Metagrammar Generator [20] (XMG) is a tool for generating large coverage grammars from concise descriptions of linguistic phenomenena (the so-called *metagrammar*). This software is a Calligramme and Langue Et Dialogue joint work and was formerly known as The Metagrammar Workbench.

The generated grammars can be lexicalized tree adjoining grammars or interaction grammars, although XMG may easily be extended to any lexicalized formalism based on tree descriptions. Each lexical item is composed of both a tree description and a semantic dimension adapted to the hole semantics.

This software is based on two important concepts from logic programming, namely, Warren's Abstract Machine and constraints on finite sets. It was developed by Benoît Crabbé, Yannick Parmentier, Denys Duchier and Joseph Le Roux. The first release is available at http://sourcesup.cru.fr/xmg.

5.3. ACG related software

A development environment for ACGs is being developed by B. Guillaume, Ph. de Groote and S. Pogodalla. The main features are the abilities to read signatures and lexicons and to realize object terms from abstract ones. Parsing (to build abstract terms from object terms) and graphical interface are being developed.

6. New Results

6.1. Proof Nets, Sequent Calculus and Typed Lambda Calculi

6.1.1. Proof Nets for Multiplicative Constants

In [26] Lutz Straßburger and François Lamarche present a new solution to the problem of including multiplicative constants in proof nets for classical multiplicative linear logic. It is well known that the constant $\perp$ has to be attached somewhere, but that the correct choice of attachment point is very problematic. An example in [26] shows that there is no hope of having a normal form in the standard sense, and that a quotient by a suitable equivalence relation has to be taken. The point of the given approach is that it uses the ordinary technology of proof nets, so there is no need to introduce new special types of links, and ordinary correctness criteria can be used; but some of these ordinary links are used to “bunch” several formulas together and thus give a much better range of choices for attaching $\perp$s. This theory obeys the most stringent possible criterion of success for the problem of the units, in the sense that it is shown that the free *-autonomous category (with units) is obtained.

The full version of this work is [30]. This paper takes pains to make everything as elementary and explicit as possible, with no hand-waving whatsoever—in particular the algebraic calculations are given in full. This make the paper readable by beginners, who have only mastered the basics of the theory of proof nets and of category theory.

6.1.2. Proof Nets for Classical Logic

In [31] François Lamarche and Lutz Straßburger propose a theory of proof denotations for ordinary, Boolean propositional logic. Two full models are constructed, and they show characteristics of both syntax (proof nets) and denotational semantics. The proof net aspect is due to the fact that the “essence” of a proof is captured by the means of axiom links. In the first model, which does not count how many times a formula is used, a full completenes theorem is given, and thus it can be seen as a substitute for ordinary syntax (sequent calculs, $\lambda\mu$-calculus...) for denoting classical proofs. The second model, which does use natural numbers for counting, is not endowed with such a full completeness theorem, and thus is closer to an ordinary denotational semantics. But it shows definite promise from the point of view of complexity: it has a direct relevance to the NP vs. co-NP problem. Both models are deceptively simple; the fact that they have been discovered so late (although Andrew’s work of 1976 was a definite precursor) is perhaps more due to widely held ideological positions about the computational nature of cut-elimination than inherent technical difficulty. These models give a very satisfying solution to a well-known problem: it is very easy to find denotational semantics for proofs in intuitionistic logic, as any bi-cartesian closed category will do, and they abound in nature (e.g., the category of sets and functions, or posets and monotone maps). But adapting this naively to classical logic leads to an immediate collapse of a candidate category to a poset (a Boolean algebra, naturally). The present work shows very clearly what standard algebraic property has to be abandoned to avoid the collapse, and it is preservation of the weakening maps. The algebraic treatment is very clean, considerably simpler—but less general in a certain sense—than what is proposed by Fuhrmann and Pym.

6.1.3. Proof Nets without Links for Lambek Calculus

In a joint work, Christian Retoré (INRIA project Signes) and S. Pogodalla define proof nets without connective links for the Lambek calculus. Proof nets without links were introduced by C. Retoré. In the commutative version, both commutativity and associativity are interpreted as equality and proof structures are defined with graph theoretical notions (series-parallel graphs, perfect matching). To cope with cyclicity

(Lambek non commutativity), they also use the notions of path and cyclic order to define proof structures and correctness criterion for the Lambek calculus.

6.2. Categorial Grammars and Dependency Grammars

6.2.1. Lexical disambiguation

G. Bonfante, B. Guillaume and G. Perrier have developed original parsing methods for interaction grammars. Recent developments in this area have led them to generalize these methods in order to deal with other well-known formalisms. These methods exploit the fact that interaction grammars are a polarized formalism. Their parsing procedure is effected in two steps: the first achieves lexical disambiguation by a global counting of polarities; the second step is the parsing process itself. It is that first step of lexical disambiguation that has been adapted to other formalisms.

Polarization in interaction grammars reflects the fact that many syntactic constituents can be viewed as consumable resources. In some other formalisms, like tree adjoining grammars (TAG) or Lambek grammars (LG) for instance, the resource sensitivity is hidden in the syntactic composition rules. It is possible to make the resource sensitivity explicit by adding polarities and then specific methods based on these polarities can be applied to the polarized versions.

In [16], this idea is formalized. A general notion of grammatical formalism is given, followed by a notion of morphism between grammatical formalisms. This notion of morphism is a very general framework for linking two grammatical formalisms. It can be used both for formalizing the polarization of an arbitray grammatical formalism, and for lexical disambiguation.

For TAGs the polarization morphism is defined as follows: every root of an elementary tree carries a positive polarity and every substitution node carries a negative polarity. Then, a substitution operation can be viewed as the neutralization of two polarities. For dealing with adjunctions, two dual polarities are put on every node where an adjunction could occur and on the root and foot node of an adjunction tree. For LG, the polarization morphism is easier to define: it is enough to polarize positively (resp. negatively) every output (resp. input) formula.

For lexical disambiguation, the key idea of our method is the definition of an abstraction morphism from the considered formalism to a simpler one. This morphism ensures that parsing in the simpler formalism is equivalent to lexical disambiguation in the former one.

In the paper, several methods for lexical disambiguation of polarized formalisms are given. The most general one consists in forgetting everything but the polarities in every syntactic structure. Hence, an elementary syntactic structure is a multiset of polarities. In a polarized formalism, a successful parsing is globally neutral. In the case of multisets of polarities, we use automata-based techniques to select globally neutral taggings for a sentence. Then this method is applied to TAG and LG.

In order to improve the performance of lexical disambiguation, we define some other methods specific to a formalism. For instance, some methods use either projectivity of the formalism (for LG) or different polarities for substitution and adjunction (for TAG). In these cases, a bottom-up parsing algorithm is used.

6.2.2. Interaction Grammars

In their original version [4] as well as their implementation via LEOPAR 5.1, Interaction Grammars (IGs) are focused on the syntax of natural languages—but syntax is only a way of accessing semantics and a linguistic formalism cannot really process natural languages at the syntactic level without relating it to the semantic one. In the classical presentation of grammatical formalisms, like TAGs and Categorial Grammars, the semantic representation of sentences is a simple projection of their syntactic derivation tree. Such a representation is too rigid to express some noncompositional phenomena—relations between quantification scopes for instance. G. Perrier has extended IGs to semantics with a more flexible approach of the syntax-semantics interface [22]. He has added a new level to the syntactic one, at which Directed Acyclic Graph descriptions are used to represent underspecified logical forms, and the same mechanism—feature neutralization—as for the syntactic level is used for composing the semantic representation of utterances. The linking between the syntactic and the semantic levels is performed by a partial function from the nodes of syntactic descriptions to the nodes of the associated semantic descriptions.

6.2.3. Abstract Categorial Grammars

Ph. de Groote and S. Pogodalla describe how to express various context-free formalisms with ACGs. It concerns context-free string grammars, linear context-free tree grammars and linear context-free rewriting systems. This proves ACGs to be able to cover important (w.r.t. natural language modeling) classes of languages such as multi-component tree adjoining grammars [54], multiple context-free grammars [50] or minimalist grammars [47].

Besides the study of the expressive power of ACGs, S. Pogodalla has extended them with some non-linear languages so that the decidability properties of the related problems (parsing and generation) remain. Non-linearity is needed for semantic representation languages and is obtained in [24] through restrictions on the lexicon between the abstract language (which remains linear) and the object one (which can be non-linear).

With that extension, S. Pogodalla [25][23] shows how to address some problems in building semantic representations for TAGs from the derivation tree (represented as abstract terms), when dealing with quantification, opaque adverbs and intersective or subsective adjectives, verbs with phrasal arguments or wh-questions.

Sylvain Salvati has developped a formal system, the calculus of syntactic descriptions. It formalizes the notion of index used in Earley algorithms for Context Free Grammars and Tree Adjoining Grammars and extends it to the linear λ-calculus. This formalization led him to propose a formal system for solving linear matching equations in the linear λ-calculus and another one for parsing Abstract Categorial Grammars whose abstract language is built on a second order signature. Those systems have made possible, for both problem, the design of an algorithm which uses tabulation technics.

In the perspective of extending Abstract Categorial Grammars with the other connectives of linear logic, Philippe de Groote and Sylvain Salvati studied the complexity of higher-order matching in the calculus associated, through the Curry-Howard correspondance, to the Multiplicative and Additive fragment of Intuitionnistic Linear Logic. They proved that the problem is NP-complete provided that the left member of the equation is given in normal form [28].

In order to study the expressive power of the Abstract Categorial Grammars, Ph. de Groote, B. Guillaume and S. Salvati have introduced the notion of *Vector Addition Tree Automaton*. They proved that the reachability problem for these automata, which corresponds to the decidability of emptiness for the Abstract Categorial Grammars, is equivalent to decidability of exponential multiplicative linear logic [27].

6.2.4. Grammatical inference

The study of exact inference algorithms for learning languages gives ones improved insights on the problem of language acquisition. Indeed, one can expect to have mathematical models of grammar acquisition and thus have a finer organization of data compared to standard stochastic approaches. Such studies have been made by Jérôme Besombes and Jean-Yves Marion. Their main contribution is certainly, starting with structural examples, the switch from learning ordinary grammars to learning tree languages. Learning from structural languages was suggested by Sakkakibara and and taken up again by Kanasawa recently.

Jérome Besombes and Jean-Yves Marion [10][13] have defined a class of categorical grammars they call *discrete*. They show that the class of discrete classical categorial grammars is identifiable from positive structured examples. For this, they provide an original algorithm, which runs in quadratic time in the size of the examples. This work extends the previous results of Kanazawa. Indeed, in that work, several types can be associated to a word and the class is still identifiable in polynomial time. The relevance of the class of discrete classical categorial grammars is demonstrated by linguistic examples, like e.g. verbs with transitive and non transitive forms, homonymies.

In order to study the learning of regular tree languages as a model of natural language acquisition, Jérome Besombes and Jean-Yves Marion [14] studied a paradigm of exact learning from positive data and membership queries. A polynomial algorithm based on this paradigm has been constructed and its correctness proved. This learning paradigm is very natural in several situations, not only linguistics but also when a query is submitted to a server and the answer is the acceptance or not of the request. We have generalized this work to dependency grammar [11].

6.3. Implicit Complexity of Computation

6.3.1. Complexity in the Blum, Shub, Smale model of computation

In order to model discrete time computation over real numbers, Blum, Shub and Smale have introduced in 1989 a new model of computation, referred as the BSS machine or sometimes the real Turing machine. In this model, the complexity of a computational problem is given by the number of elementary arithmetical operations and comparisons needed to solve this problem, independently of the underlying representation of real numbers. This makes it very different from the recursive analysis model, and occurs to be a very natural way to describe computational problems over real numbers. The BSS model of computation has been later on extended to the notion of computation over arbitrary logical structure. Since classical complexity theory occurs to be the restriction of this general model to boolean structures, it gives a new insight on previous questions on classical complexity theory and its links with logic.

Paulin de Naurois' PhD thesis [7], jointly supervised by Olivier Bournez and Jean-Yves Marion, is focused on the study of complexity in this model. It provides new completeness results for geometrical problems when the arbitrary structure is specialized to the set of real numbers with addition and order. The range of these completeness results covers many of the most important complexity classes over this setting, which is one of the three major ones for complexity theory over arbitrary structures. He also provides several machine-independent characterizations of complexity classes over arbitrary structures. Also included is the extension of some results by Grädel, Gurevich and Meer in descriptive complexity, characterizing deterministic and non deterministic polynomial time decision problems in terms of logics over metafinite structures. In addition there can be found the extension of some results in implicit complexity by Bellantoni and Cook [34], that characterizes functions computable in sequential deterministic polynomial time, and by Leivant and Marion [45], that characterizes functions computable in parallel deterministic polynomial time in terms of algebras of recursive functions. In addition, there are characterizations of functions computable within the polynomial hierarchy, extending a result by Bellantoni [33], and in polynomial alternating time. These results have been published in 2003. [35][36]. Other related publications are [18][8].

6.3.2. Implicit Complexity

New results have been obtained this year, from which the main ones are the following. First of all, G. Bonfante, J.-Y. Marion and J.-Y. Moyen have shown that the synthesis of quasi-interpretation is decidable, more precisely, it can be done in double exponential time with respect to the size of the program. This is of major importance as one may automatize the process of certification. In mobile computing, the quasi-interpretation may be sent as a certificate with the code, such a certificate can be verified by the receiving device. For more restricted classes of quasi-interpretations, R. Péchoux has shown that synthesis of multilinear quasi-interpretations is NP-complete. This result slightly improves on Amadio's work because it applies to QI over real numbers and not over rationals.

Next, G. Bonfante, J.-Y Marion and J.-Y. Moyen have refined the complexity hierarchy, by introducing some new constraints on programs. This give rise to various complexity classes, among them: LOGSPACE, LINSPACE, PTIME, PSPACE.

7. Other Grants and Activities

7.1. Regional Actions

- Calligramme is part of the "Ingénierie des langues, du document, de l'information scientifique et culturelle" theme of the "contrat de plan État-Région". Calligramme's contributions range over the syntactical and semantical analysis of natural language, the building of wide coverage lexical resources, and the development of software specialized for those tasks.

- Calligramme, through Denys Duchier in particular, is a partner in the series of joint scientific meetings organized through the Lorraine-Saarland partnership.
- Calligramme is part of the "Qualité et sûreté des Logiciels (QSL) theme of the "contrat de plan État-Région". As a matter of fact the fact the head of the whole QSL theme is Jean-Yves Marion. Web page at http://qsl.loria.fr
- Jean-Yves Marion is head of the "Synthèse de code pour robots mobiles" (Sycomor) project, granted by the scientific council of the INPL;

7.2. National Actions

7.2.1. Action Concertée Incitative (ACI) Demonat

Calligramme is involved in the ACI DEMONAT, in section "Nouvelles interfaces des mathématiques", together with the "Logique" group of "Université de Savoie" and the "TALaNa" team of "Université Paris 7". The project concerns the parsing and the checking of mathematical proofs written in natural language.

7.2.2. Action Concertée Incitative (ACI) CRISS

Calligramme is involved in the ACI CRISS, in section "Sécurité informatique". Its purpose, which can be read from the full title, is "Contrôle de ressources et d'interfaces pour les systèmes synchrones". It is headed by Roberto Amadio at the University of Marseilles, and the co-ordinator on Calligramme's side is Jean-Yves Marion.

Web page at http://www.cmi.univ-mrs.fr/~amadio/Criss/criss.html

7.2.3. Action Concertée Incitative (ACI) Géocal

This "nouvelles interfaces des mathématiques" ACI regroups several research teams in both mathematics and computer science and is concerned, as its name implies, with the application to computer science of techniques developed for modern geometry. It is headed by Thomas Ehrhard at the CNRS in Marseilles, and the co-ordinator on Calligramme's side is Jean-Yves Marion.

Web page at http://iml.univ-mrs.fr/~ehrhard/geocal/geocal.html

7.2.4. Action Spécifique "Topologie Algébrique"

This is a smaller CNRS action whose full title is "Topologie algébrique pour l'étude des structures de calcul et notamment de la concurrence". It is headed by Éric Goubault at the Commissariat de l'Énergie Atomique, and many of its members are also part of the previous action. It ended in October.

Web page at http://www.di.ens.fr/~goubault/as.html

7.2.5. Réseau National des Technologies Logiciells (RNTL)

Calligramme, through Jean-Yves Marion, is a participant in the Ministry of Industry RNTL project Averroes.

Web page: http://www-verimag.imag.fr/AVERROES/

7.3. European Actions

- Calligramme is one half of a joint France-Germany (Procope) cooperation entitled "Structures and Deductions" with the Technische Universität Dresden, Germany.
- Calligramme is involved in the IST-2001-38957 european project, in the FET-Open program, entitled "Applied Semantics II". Jean-Yves Marion is the site leader of this project.
Web page: http://www.tcs.informatik.uni-muenchen.de/~mhofmann/appsem2
- Calligramme is involved in the european network CoLogNET (Computational Logic Network) on the themes: logic methodology and foundational tools, logic and natural language processing.

7.4. Visits and invitation of researchers

- Kai Brünller (University of Bern) visited the Calligramme project for three days in April and gave a talk at our seminar.
- William Funk of Vanderbilt University (Nashville, Tennessee) visited François Lamarche for a four-week period in May-June on an INRIA-Vanderbilt student grant.
- Noam Zeilberger (Ph.D. student, CMU) visited the Calligramme project from June 13 to August 18.
- François Toussenel (Lattice, University Paris 7) visited the Calligramme project in November (2 days).
- Aarne Ranta (Chalmers, Sweden) visited the Calligramme project in November (3 days).
- Ozan Kahramanogullari (TU Dresden) visited the Calligramme and Protheo teams during a month starting on November 17.
- Charles Stewart and Robert Heim (TU Dresden) visited Calligramme during the first week of December and both gave talks.
- Dale Miller (LIX and INRIA-Futurs) visited calligramme on December 6–7 and gave a talk.

8. Dissemination

8.1. Activism within the scientific community

- Guillaume Bonfante is the vice president of the hiring committee, section 27, of the INPL, since April 2003.
- Guillaume Bonfante is an elected member of the scientific council of the INPL since July 2003.
- Guillaume Bonfante is a member of the engineering part of the Comipers hiring committee at LORIA.
- Adam Cichon was elected member of the “Conseil National des Universités” (CNU), section 27.
- Philippe de Groote is President of the INRIA-Lorraine Projects Committee (starting September 2004), and a member of INRIA’s evaluation board.
- Philippe de Groote is a member of the LORIA management board, and of the LORIA laboratory council.
- Philippe de Groote was president of the INRIA-Lorraine audience commission for Junior Researchers (CR2), June 2004.
- Philippe de Groote was a member of the INRIA admisssion jury for Junior Researchers (CR2) and Senior Researchers (DR2), June 2004.
- Philippe de Groote is a member of the editorial board of *Papers in Formal Linguistics and Logic* (Bulzoni, Roma), *Cahiers du Centre de Logique* (Academia-Bruylant, Louvain-la-Neuve).
- Philippe de Groote was member of the program committees of ESSLLI’04 (local co-chair), IJCAR’04, and FG’04.
- François Lamarche is member of the Bureau of the Département de Formation Doctorale of the IAEM doctoral school.
- François Lamarche heads the research (theses, postdocs and *ingénieurs spécialistes*) section of the Comipers hiring commitee at LORIA.
- Jean-Yves Marion is member of the steering committee of the International workshop on Logic and Computational Complexity (LCC).

- Jean-Yves Marion was the guest editor of Theoretical Computer Science on “Implicit computational complexity”, Volume 318, Number 1-2, 6 June 2004
- Jean-Yves Marion is member of the hiring committee(CS) at the University of Metz, section 27, since Sept. 2004.
- Jean-Yves Marion is member of the hiring committee at INPL (Professors and Lecturers), section 27, since February 2002.
- Jean-Yves Marion was elected to the scientific council of INPL in July 2003 and member of the board.
- Jean-Yves Marion initiated and organizes the monthly “Journées QSL” http://qsl.loria.fr
- Jean-Yves Marion organized the CRISS workshop in Nancy on June 23.

8.2. Teaching

- Adam Cichon, with Claude Kirchner, is in charge of the DEA course on “logique et démonstration automatique”.
- Adam Cichon teaches the course on Logic and Artificial Intelligence for the DESS “Compétences Complémentaires”. In 2004 this course took place in Casablanca, Morocco.
- Adam Cichon heads the “licence informatique”.
- François Lamarche is in charge of the DEA course on the semantics of natural languages, which he is giving this year with Philippe de Groote.
- François Lamarche gave a course at the 2004 ICCL Summer School on Proof Theory and Automated Theorem Proving in June in Dresden, with the title “Logic seen as a branch of geometry”.
- Jean-Yves Marion is in charge of the option “Ingénierie des systèmes informatiques” at École des Mines starting in September.
- Jean-Yves Marion is head of the proposal of the professionnal master “Conception sûre de systèmes embarqués et ambiants”.
- Jean-Yves Marion took part in the creation of the formation in computational biology at École des Mines and is in charge of the course on “Bases et banques de données”.
- Guy Perrier is in charge of the organization of the course on *algorithmics for the parsing of natural languages*, which he is teaching with Bertrand Gaiffe in the computer science DEA of Nancy.
- Guy Perrier with Bertrand Gaiffe gave a course on *basic parsing techniques for natural language* at the 16th European Summer School in Logic, Language and Information, which was organized in Nancy from the 9th to the 20th of August 2004.

8.3. Academic Supervision

- Philippe de Groote is supervising the thesis work of Sylvain Salvati.
- Denys Duchier is supervising Ralph Debusmann’s Ph.D. thesis (Saarbrücken).
- Guy Perrier is supervising the thesis work of J. Leroux.
- Jean-Yves Marion and Olivier Bournez (Project-team Protheo) are co-supervising the thesis work of Paulin Jacobé de Naurois and Emmanuel Hainry.
- Jean-Yves Marion is supervising the thesis work of Romain Péchoux from September 2004.

8.4. Thesis juries

- Ph. de Groote was referee and jury member for the Ph.D. of Mateu Villaret on June 30 (UPC, Barcelona; supervisor: Jordi Levy; thesis entitled "On Some Variants of Second-Order Unification").
- Ph. de Groote was referee and jury member for Olga Kouchnarenko's HDR on November 25 (LIFC, Besançon; supervisors: Françoise Bellegarde and Jacques Julliand; thesis entitled "Raffiner pour vérifier des propriétés de systèmes finis et infinis").
- J.-Y. Marion was thesis referee for Daniela Sofronie (supervisors: Rémi Gilleron (U. Lille 3) and Dumitriu Todoroi (U. Isai, Romania)), thesis entitled "Learning categorical grammars to simulate natural langugage acquisition with the help of semantic informations" on April, 13.
- J.-Y. Marion was president of the Ph.D. jury of Djame Seddah (supervisors: Jean-Marie Pierrel and Bertrand Gaiffe (Atilf/LORIA)), thesis titled "Synchronisation des connaissances syntaxiques et sémantiques pour l'analyse d'énoncés en langue naturelle à l'aide des grammaires d'arbres adjoints lexicalisés" on November, 5.
- J.-Y. Marion was president of the Ph.D. jury of Vincent Barreau (supervisors: Jean-Paul Haton, Dominique Fohr and Irina Illina (LORIA)), thesis titled "Reconnaissance automatique de la parole continue : compensation des bruits par transformation de la parole" on November 9.
- J.-Y. Marion was jury member for Ph. de Naurois' thesis, Hong Kong, December 15.
- J.-Y. Marion was thesis referee for Yannick Lenir (supervisor: Christian Rétoré (U. Bordeaux)), thesis entitled "Structure des analyses syntaxiques catégorielles. Application à l'inférence grammaticale" on December, 15.

8.5. Thesis defenses

- Paulin Jacobé de Naurois defended his thesis on December 15, 2004 (jury: O. Bournez, F. Cucker, Z. Ding-Xuan, M. Golin, P. Koiran, J.-Y. Marion, M. de Rougemont).

8.6. Participation to colloquia, seminars, invitations

- J. Besombes, G. Perrier and S. Pogodalla attended the Categorial Grammars meeting in Montpellier on June. In addition to a regular talk, S. Pogodalla presented a joint work with Christian Rétoré as an invited talk, and G. Perrier gave an invited talk.
- J. Besombes attended the ISS 2004 meeting in Zakopane (Poland) on May.
- J. Besombes visited the Symbiose group in Rennes and gave a talk.
- G. Besombes and J.-Y. Marion attended the ALT 2004 conference in Padova (Italy) on October.
- Three members of Calligramme (G. Bonfante, E. Hainry and J.-Y. Marion) attended the APPSEM 2004 meeting in Tallin (Estonia) on April and gave a talk on "On complexity analysis by Quasi-interpretations".
- G. Bonfante, F. Lamarche, J.-Y. Marion, J.-Y. Moyen and L. Straßburger attended the "Première rencontre de l'ACI GEOCAL" in Marseille in January and L. Straßburger gave a talk on "The calculus of structures", and François Lamarche on "the geometry of rewriting."
- G. Bonfante, J.-Y. Marion, J.-Y. Moyen and R. Péchoux attended the CRISS workshop at the INRIA Sophia, January 23. J.-Y. Moyen gave a talk on "Termination and ressource analysis of assembly-programs by Petri Nets".
- G. Bonfante, J.-Y. Marion, J.-Y. Moyen and R. Péchoux attended the CRISS workshop in Nancy in June and R. Péchoux gave a talk on "Synthèse et vérification de QI".

- D. Duchier, B. Guillaume and G. Perrier attended the COLING 2004 conference in Geneva (Switzerland) on August.
- D. Duchier and J. Le Roux attended the MOZ 2004 conference at Charleroi (Belgium) in October and presented a paper about the XMG design and implementation: “The Metagrammar Compiler: an NLP Application With a Multi-paradigm Architecture”.
- D. Duchier attended the CSLP 2004 conference in Copenhagen (Denmark) in September.
- Ph. de Groote, B. Guillaume and S. Salvati attended the Demonat workshop in Paris in November.
- Ph. de Groote and J.-Y. Marion attended the LICS 2004 conference on July. Ph. de Groote presented the paper “Vector Addition Tree Automata” [27].
- Ph. de Groote, S. Salvati and L. Straßburger attended the CSL 2004 conference in September, in Karpacz, Poland. L. Straßburger presented the paper “On proof nets for multiplicative linear logic with units” [26] and S. Salvati presented the paper “Higher-order Matching in the Linear λ-calculus with Pairing” [28].
- B. Guillaume, Ph. de Groote, S. Pogodalla and S. Salvati attended the Demonat workshop at the Université de Savoie, 5–6 May 2004;
- B. Guillaume, J. Le Roux, G. Perrier and S. Pogodalla attended a workshop organized by the Signes INRIA project to present Leopar and related NLP issues;
- B. Guillaume attended the 2nd international workshop Coq+rewriting at Palaiseau in September and gave a talk.
- F. Lamarche and L. Straßburger attended the ICCL Summer School 2004 on Proof Theory and Automated Theorem Proving, and the PCC Workshop 2004, in June in Dresden, Germany, and L. Straßburger gave a talk “On proof nets for classical propositional logic”.
- F. Lamarche and L. Straßburger attended the ICCL 2004 workshop in Dresden on September 25–26. Both gave a talk.
- J.-Y. Marion and J.-Y. Moyen attended the GeoCal meeting at LIPN (Université de Paris Nord) in September. J.-Y. Marion gave a talk “Resource analysis by Quasi-interpretations” and J.-Y. Moyen gave a talk “Resource termination – using Petri Nets to analyse programs”.
- J.-Y. Marion was twice invited speaker at CMAF in Lisbon (Portugal) in March and May.
- J.-Y. Marion attended the LCC 2004 workshop in July and gave an invited talk “Resource analysis by Quasi-interpretations”.
- J.-Y. Marion attended the ACI “Sécurité Informatique” meeting in Toulouse in November.
- J.-Y. Moyen visited the LCR group at LIPN (Université de Paris Nord) and gave a talk on “Ordres de teminaison et Quasi-interprétations”.
- J.-Y. Moyen visited the LogiCal group at LIX (Palaiseau) and gave a talk on “Terminaison par ressources”.
- J.-Y. Moyen visited the PPS group at Université Paris 7 and gave a talk on “Ordres de terminaison et Quasi-interprétations”.
- G. Perrier and S. Pogodalla each gave a talk at the 11th conference TALN 2004 that was held in Fès, Morocco, April 19-22.
- S. Pogodalla gave a talk at the 7th international workshop on Tree Adjoining Grammar and Related Formalisms (TAG+7), 20-22 May 2004, Vancouver, Canada.
- Lutz Straßburger visited the PPS group (CNRS UMR 7126) in Paris in February, and gave a talk on “The calculus of structures”.

- Lutz Straßburger visited Dale Miller at the LIX (École polytechnique) in Palaiseau in February and gave a talk on “Proof nets for multiplicative linear logic with units”.
- Lutz Straßburger attended the 9th Estonian Winter School in Computer Science, EWSCS 2004, in Palmse, Estonia, in March and gave a talk on “The calculus of structures”.
- Lutz Straßburger visited the IMMM (CNRS UMR 5149) in Montpellier in March and gave two talks: “Linear logic in the calculus of structures” and “On proof nets for multiplicative linear logic with units”.
- Lutz Straßburger visited the LaBRI (INRIA Futurs, CNRS UMR 5800) in Bordeaux in May and gave a talk “Le calcul des structures”.
- Lutz Straßburger visited the PPS group (CNRS UMR 7126) in Paris in June, and gave a talk “On proof nets for classical propositional logic”.
- Lutz Straßburger visited the Programming Systems Lab, Universität des Saarlandes, Saarbrücken, Germany, in September and gave a talk on “The calculus of structures”.

9. Bibliography

Major publications by the team in recent years

[1] J. BESOMBES, J.-Y. MARION. *Apprentissage des langages réguliers d’arbres et applications*, in "Traitement automatique de langues", vol. 44, n° 1, July 2003, p. 121–153.

[2] D. LEIVANT, J.-Y. MARION. *A characterization of alternating log time by ramified recurrence*, in "Theoretical Computer Science", vol. 236, n° 1-2, 2000, p. 192–208.

[3] G. PERRIER. *Interaction Grammars*, in "CoLing 2000, Sarrebrücken, Germany", International Committee on Computational Linguisitcs, August 2000.

[4] G. PERRIER. *Descriptions d’arbres avec polarités : les Grammaires d’Interaction*, in "9ième Conférence annuelle sur le Traitement Automatique des Langues Naturelles (TALN’02), Nancy, France", 2002, p. 297–306.

[5] P. DE GROOTE. *Towards abstract categorial grammars*, in "Association for Computational Linguistics, 39th Annual Meeting and 10th Conference of the European Chapter, Toulouse, France", July 2001, p. 148–155.

[6] P. DE GROOTE, F. LAMARCHE. *Classical Non Associative Lambek Calculus*, in "Studia Logica", vol. 71, n° 3, August 2002, p. 355–388.

Doctoral dissertations and Habilitation theses

[7] P. JACOBÉ DE NAUROIS. *Completeness Results and Syntactic Characterizations of Complexity Classes over Arbitrary Structures*, Thèse d’université, INPL, December 2004.

Articles in referred journals and book chapters

[8] O. BOURNEZ, F. CUCKER, P. JACOBÉ DE NAUROIS, J.-Y. MARION. *Implicit Complexity Over an Arbitrary Structure: Sequential and Parallel Polynomial Time*, in "Journal of Logic and Computation", November 2004, http://www.loria.fr/publications/2004/A04-R-285/A04-R-285.ps.

[9] J.-Y. MARION. *Editorial: Implicit Computational Complexity (ICC)*, in "Theoretical Computer Science", Editeur scientifique invité, vol. 318, n^o 1-2, June 2004, 1.

Publications in Conferences and Workshops

[10] J. BESOMBES, J.-Y. MARION. *Apprentissage des grammaires catégorielles à partir de structures*, in "Conférence Francophone d'Apprentissage - CAp'2004, Montpellier, France", Presse universitaire de Grenoble, June 2004, p. 315–330.

[11] J. BESOMBES, J.-Y. MARION. *Learning Dependency Languages from a Teacher*, in "9th conference on Formal Grammar, Nancy, France", August 2004.

[12] J. BESOMBES, J.-Y. MARION. *Learning regular trees with queries*, in "Intelligent Information Processing and Web Mining, Proceedings of the International IIS: IIPWM'04, Zakopane, Poland", M. A. KLOPOTEK, S. T. WIERZCHON, K. TROJANOWSKI (editors)., Advances in Soft Computing, Springer, May 2004, p. 181–190.

[13] J. BESOMBES, J.-Y. MARION. *Learning Reversible Categorial Grammars from Structures*, in "Categorial Grammars, Montpellier, France", June 2004.

[14] J. BESOMBES, J.-Y. MARION. *Learning Tree Languages from Positive Examples and Membership Queries*, in "15th international conference on Algorithmic Learning Theory - ALT'2004, Padova, Italy", S. BEN-DAVID, J. CASE, A. MARUOKA (editors)., Lecture notes in Artificial Intelligence, vol. 3244, Springer, October 2004, p. 440–453.

[15] M. BODIRSKY, D. DUCHIER, J. NIEHREN, S. MIELE. *A New Algorithm For Normal Dominance Constraints*, in "ACM-SIAM Symposium on Discrete Algorithms - SODA'2003, New Orleans, LA, USA", January 2004.

[16] G. BONFANTE, B. GUILLAUME, G. PERRIER. *Polarization and abstraction of grammatical formalisms as methods for lexical disambiguation*, in "20th Conference on Computational Linguistics - CoLing'2004, Geneva, Switzerland", August 2004, p. 303–309, http://www.loria.fr/publications/2004/A04-R-260/A04-R-260.ps.

[17] G. BONFANTE, J.-Y. MARION, J.-Y. MOYEN. *On complexity analysis by quasi-interpretation*, in "2nd Appsem II workshop - APPSEM'04, Tallinn, Estonia", April 2004, p. 85–95.

[18] O. BOURNEZ, F. CUCKER, P. JACOBÉ DE NAUROIS, J.-Y. MARION. *Tailoring Recursion to Characterize Non-Deterministic Complexity Classes Over Arbitrary Structures*, in "3rd IFIP International Conference on Theoretical Computer Science - TCS'2004, Toulouse, France", Kluwer Academic Press, August 2004, http://www.loria.fr/publications/2004/A04-R-287/A04-R-287.ps.

[19] O. BOURNEZ, F. CUCKER, P. JACOBÉ DE NAUROIS, J.-Y. MARION. *Tailoring Recursion to Characterize Non-Deterministic Complexity Classes Over Arbitrary Structures*, in "2nd APPSEM II Workshop - APPSEM'2004, Tallinn, Estonia", April 2004, http://www.loria.fr/publications/2004/A04-R-419/A04-R-419.ps.

[20] D. DUCHIER, J. LE ROUX, Y. PARMENTIER. *The MetaGrammar Compiler: An NLP Application with a Multi-paradigm Architecture*, in "Second International Mozart/Oz Conference - MOZ 2004, Charleroi, Belgium",

October 2004.

[21] J.-Y. MARION. *Resource Analysis by Quasi-Interpretations*, in "Sixth International Workshop on Logic and Computational Complexity, Turku, Finland", Neil Jones, July 2004, http://www.loria.fr/publications/2004/A04-R-349/A04-R-349.ps.

[22] G. PERRIER. *La sémantique dans les grammaires d'interaction*, in "11ième Conférence annuelle sur le Traitement Automatique des Langues Naturelles - TALN'2004, Fès, Morocco", April 2004, http://www.loria.fr/publications/2004/A04-R-259/A04-R-259.ps.

[23] S. POGODALLA. *Computing Semantic Representation: Towards ACG Abstract Terms as Derivation Trees*, in "Seventh International Workshop on Tree Adjoining Grammar and Related Formalisms - TAG+7, Vancouver, BC, Canada", May 2004, p. 64–71, http://www.loria.fr/publications/2004/A04-R-058/A04-R-058.ps.

[24] S. POGODALLA. *Using and Extending the ACG technology: Endowing Categorial Grammars with an Underspecified Semantic Representation*, in "Categorial Grammars, Montpellier, France", June 2004, p. 197–209, http://www.loria.fr/publications/2004/A04-R-155/A04-R-155.ps.

[25] S. POGODALLA. *Vers un statut de l'arbre de dérivation: exemples de construction de représentations sémantiques pour les Grammaires d'Arbres Adjoints*, in "Traitement Automatique des Langues Naturelles - TALN'04, Fès, Morocco", April 2004, http://www.loria.fr/publications/2004/A04-R-050/A04-R-050.ps.

[26] L. STRASSBURGER, F. LAMARCHE. *On Proof Nets for Multiplicative Linear Logic with Units*, in "18th International Workshop on Computer Science Logic - CSL'2004, Karpacz, Poland", J. MARCINKOWSKI, A. TARLECKI (editors)., Lecture Notes in Computer Science, 13th Annual Conference of the EACSL, vol. 3210, Springer-Verlag, September 2004, p. 145–159.

[27] P. DE GROOTE, B. GUILLAUME, S. SALVATI. *Vector Addition Tree Automata*, in "19th Annual IEEE Symposium on Logic in Computer Science - LICS'04, Turku, Finland", July 2004, p. 64–73.

[28] P. DE GROOTE, S. SALVATI. *Higher-order Matching in the Linear lambda-calculus with Pairing*, in "18th International Workshop on Computer Science Logic - CSL'2004, Karpacz, Poland", A. T. JERZY MARCINKOWSKI (editor)., Lecture notes in Computer Science, vol. 3210, Springer, September 2004, p. 220–234.

Internal Reports

[29] O. BOURNEZ, F. CUCKER, P. JACOBÉ DE NAUROIS, J.-Y. MARION. *Implicit Complexity over an Arbitrary Structure: Quantifier Alternations*, Rapport de recherche, Loria, October 2004, http://www.loria.fr/publications/2004/A04-R-300/A04-R-300.ps.

[30] F. LAMARCHE, L. STRASSBURGER. *From Proof nets to the Free *-Autonomous Categories*, Rapport de recherche, Loria, November 2004, http://www.loria.fr/publications/2004/A04-R-390/A04-R-390.ps.

[31] F. LAMARCHE, L. STRASSBURGER. *Naming Proofs in Classical Propositional Logic*, Rapport de recherche, Loria, October 2004, http://www.loria.fr/publications/2004/A04-R-391/A04-R-391.ps.

Bibliography in notes

[32] Y. BAR-HILLEL. *A quasi-arithmetical notation for syntactic description*, in "Language", vol. 29, 1950, p. 47-58.

[33] S. BELLANTONI. *Predicative recursion and the polytime hierarchy*, in "Feasible Mathematics II", P. CLOTE, J. REMMEL (editors)., Perspectives in Computer Science, Birkhäuser, 1994.

[34] S. BELLANTONI, S. COOK. *A new recursion-theoretic characterization of the poly-time functions*, in "Computational Complexity", vol. 2, 1992, p. 97–110.

[35] O. BOURNEZ, F. CUCKER, P. JACOBÉ DE NAUROIS, J.-Y. MARION. *Computability over an Arbitrary Structure. Sequential and Parallel Polynomial Time*, in "Foundations of Software Science and Computation Structures - FOSSACS'03, Warsaw, Poland", A. D. GORDON (editor)., Lecture Notes in Computer Science, vol. 2620, Springer, Apr 2003, p. 185-199.

[36] O. BOURNEZ, F. CUCKER, P. JACOBÉ DE NAUROIS, J.-Y. MARION. *Safe Recursion Over an Arbitrary Structure : PAR, PH and DPH*, in "Fifth International Workshop on Implicit Computational Complexity - ICC'2003, Ottawa, Canada", Electronic Notes in Computer Science, vol. 90, n^o 1, Jun 2003.

[37] H. CURRY. *Foundations of mathematical logic*, Dover Publications, 1977.

[38] D. DUCHIER, R. DEBUSMANN. *Topological Dependency Trees: A Constraint-based Account of Linear Precedence*, in "39th Annual Meeting of the Association for Computational Linguistics (ACL 2001), Toulouse, France", 9–11July 2001.

[39] D. DUCHIER. *Axiomatizing Dependency Parsing Using Set Constraints*, in "Sixth Meeting on Mathematics of Language, Orlando, Florida", July 1999, p. 115–126.

[40] G. GENTZEN. *Recherches sur la déduction logique (Untersuchungen über das logische schließen)*, Traduction et commentaire par R. Feys et J. Ladrière, Presses Universitaires de France, 1955.

[41] J.-Y. GIRARD. *Linear Logic*, in "Theoretical Computer Science", vol. 50, 1987, p. 1-102.

[42] T. G. GRIFFIN. *A formulae-as-types notion of control*, in "Conference record of the seventeenth annual ACM symposium on Principles of Programming Languages", 1990, p. 47-58.

[43] J. LAMBEK. *The mathematics of sentence structure*, in "Amer. Math. Monthly", vol. 65, 1958, p. 154-170.

[44] J. LAMBEK. *On the calculus of syntactic types*, in "Studies of Language and its Mathematical Aspects, Providence", Proc. of the 12th Symp. Appl. Math.., 1961, p. 166-178.

[45] D. LEIVANT, J.-Y. MARION. *Ramified recurrence and computational complexity II: substitution and poly-space*, in "Computer Science Logic, 8th Workshop, CSL '94, Kazimierz,Poland", L. PACHOLSKI, J. TIURYN (editors)., Lecture Notes in Computer Science, vol. 933, Springer, 1995, p. 486–500.

[46] I. MEL'ČUK. *Dependency Syntax: Theory and Practice*, State Univ. Press of New York, Albany/NY, 1988.

[47] J. MICHAELIS. *Transforming Linear Context-Free Rewriting Systems into Minimalist Grammars*, in "Proceedings of the conference Logical Aspects of Computational Linguistics (LACL '01)", LNCS/LNAI, vol. 2099, 2001.

[48] G. PERRIER. *Intuitionistic Multiplicative Proof Nets as Models of Directed Acyclic Graph Descriptions*, in "8th International Conference on Logic for Programming, Artificial Intelligence and Reasoning - LPAR 2001, Havana, Cuba", A. V. ROBERT NIEUWENHUIS (editor)., Lecture Notes in Artificial Intelligence, vol. 2250, Springer, Dec 2001, p. 233-248.

[49] G. PERRIER. *Descriptions d'arbres avec polarités : les Grammaires d'Interaction*, in "9ième Conférence annuelle sur le Traitement Automatique des Langues Naturelles - TALN'02, Nancy, France", Jun 2002, http://www.loria.fr/publications/2002/A02-R-123/A02-R-123.ps.

[50] H. SEKI, T. MATSUMURA, M. FUJII, T. KASAMI. *On Multiple Context-Free Grammars*, in "Theoretical Computer Science", vol. 223, 1991, p. 87–120.

[51] P. SGALL, L. NEBESKY, A. GORALCIKOVA, E. HAJICOVA. *A Functional Approach To Syntax In Generative Description Of Language*, 1969.

[52] L. TESNIÈRE. *Eléments de Syntaxe Structurale*, 1959.

[53] K. VIJAY-SHANKER. *Using Description of Trees in a Tree Adjoining Grammar*, in "Computational Linguistics", vol. 18, n^o 4, 1992, p. 481-517.

[54] D. J. WEIR. *Characterizing Mildly Context-Sensitive Grammar Formalisms*, Ph. D. Thesis, University of Pennsylvania, 1988.