INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Project-Team Calligramme

Linear Logic, Proof Nets and Categorial Grammars

Lorraine

THEME SYM

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, (until September 29 2006)]
Céline Simon [INRIA, (since October 2 2006)]

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

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

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

Staff member Université Nancy 2
Guy Perrier [Professor, University Nancy 2, HdR]

Engineer
Karën Fort [INRIA, (since September 11 2006)]

Post-doctoral fellows
Kristofer Johannisson [INRIA postdoctoral fellow (until September 8)]
Ryo Yoshinaka [INRIA postdoctoral fellow (since October 1st)]

Ph. D. Students
Robert Hein [INRIA CORDI fellow (since September 18), defense planned in 2009]
Matthieu Kaczmarek [BDI CNRS fellow, defense planned in 2008]
Joseph Le Roux [MESR fellow, defense planned in 2007]
Sarah Maarek [MESR fellow (since October 1st), defense planned in 2009]
Jonathan Marchand [MESR fellow (since October 1st), defense planned in 2009]
Romain Péchoux [MESR fellow, defense planned in 2007]

2. Overall Objectives

2.1. Overall Objectives

Keywords: *categorial grammar*, *implicit complexity*, *lambda calculus*, *linear logic*, *proof nets*, *semantics of natural languages*, *sequent calculus*, *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 Jean-Yves Girard [31] results from a finer analysis of the part played by structural rules in Gentzen's sequent calculus [29]. 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 linear 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 [27] 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 [40].

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*.

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 a 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 [32].

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

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

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 sensitivity. 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 assignment:

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

where the slash ($/$) (resp. the backslash ($\setminus$)) are interpreted as fraction pairings that simplify to the right (resp. to the left). However we notice very soon that this simplification scheme, which is the basis of Bar-Hillel grammars [25], is not sufficient.

Lambek solves this problem by suggesting the interpretation of slashes and backslashes as implicative connectors [34], [35]. 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:

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

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 [37].

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 [39].

- Contrary to TAG where tree fragments can only be inserted, Interaction Grammars admit models where the interpretations of description fragments may overlap.

3.4. Implicit Complexity of Computations

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

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 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 program 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 programming 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 computability, 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 sets of concrete forms generated from these abstract structures. Consequently, one has a direct control on the parse structures of the grammar.
- The languages generated by the ACGs are sets of linear λ-terms. This may be seen as a generalization of both string-languages and tree-languages.
- 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 Intuitionistic Linear Logic (which have very specific properties) in a model-theoretical framework [37] and it was at first designed for modelling the syntax of natural languages [38].

4.1.3. Grammatical and lexical resources for French

The relevance of new linguistic formalisms needs to be proved by experiments on real corpora. Parsing real corpora requires large scale grammars and lexicons. There is a crucial lack of such resources for French and all researchers committed in natural language processing (NLP) projects for French based on different formalisms are confronted with the same problem. Now, building large scale grammars and lexicons for French demands a lot of time and human resources and it is crucial to overcome the multiplicity of existing formalisms by developing common and reusable tools and data. This is the sense of two directions of research:

1. The modular organization of formal grammars in a hierarchy of classes allows the expression of linguistic generalizations and it makes their development and their maintenance on a large scale possible. To be used in NLP applications such modular grammars have to be compiled into operational grammars. By comparison with the area of programming languages, we write source grammars in a language with a high abstraction level and then we compile them automatically to object grammars, directly usable by NLP applications.

Considering the multiplicity of linguistic formalisms, it would be interesting to express the various source grammars that can be written in different formalisms, in a common abstract language and to compile them with the same tool associated to this language. XMG is a first experiment in this direction: for the moment, it allows the edition and the compilation of source grammars for TAGs and IGs. Moreover, we can hope that the use of a common language of syntactic description with a high level of abstraction makes easier the reusability of some parts of grammars from one formalism to another.

2. With the same preoccupation of reusability, it is important to develop syntactic and semantic lexicons which contain only purely linguistic information and which are independent of the different existing grammatical formalisms. Now, a mechanism must be foreseen to combine these lexicons with the grammars built in the various formalisms. A convenient way of doing this is to design the entries of such lexicons in the form of feature structures and to associate also feature structures with the elementary constructions of the grammars. Then, their anchoring in the lexicons is realized by unification of the two kinds of feature structures. The construction of a syntactic and a semantic lexicon for French can be envisaged either by acquisition from corpora or by re-use of existing lexical information.

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 Cristal and Mimosa project-teams.

5. Software

5.1. Leopar

Keywords: *Interaction Grammar*, *parsing*.

Participants: Bruno Guillaume [correspondant], Guy Perrier, Guillaume Bonfante, Sylvain Pogodalla, Joseph Le Roux, Jonathan Marchand.

5.1.1. Software description

LEOPAR is a parser for natural languages which is based on the formalism of Interaction Grammars (IG) [38]. It uses a parsing principle, called “electrostatic parsing” which consists in neutralizing opposite polarities. A positive polarity corresponds to an available linguistic feature and a negative one to an expected feature.

Parsing a sentence with an Interaction Grammar consists in first selecting a lexical entry for each of its words. A lexical entry is an underspecified syntactic tree, a tree description in other words. Then, all selected tree descriptions are combined by partial superposition guided by the aim of neutralizing polarities: two opposite polarities are neutralized by merging their support nodes. Parsing succeeds if the process ends with a minimal and neutral tree. As IGs are based on polarities and 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 nondeterminism 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 which nodes to merge is made. Polarities have shown their efficiency in pruning the search tree for the following two steps:

- In the first step (tagging the words of the sentence with tree descriptions), we forget the structure of descriptions, and only keep the bag of their features. In this case, parsing inside the formalism is greatly simplified because composition rules reduce to the neutralization of a negative feature-value pair $f \longleftarrow v$ by a dual positive feature-value pair $f \longrightarrow v$. As a consequence, parsing reduces to a counting of positive and negative polarities present in the selected tagging for every pair (f, v): every positive occurrence counts for +1 and every negative occurrence for –1, the sum must be 0.
- In the second step (node-merging phase), polarities are used to cut off parsing branches when their trees contain too many non neutral polarities.

5.1.2. Current state of the implementation

The current implementation started in 2004 (by Guillaume Bonfante, Bruno Guillaume, Guy Perrier and Sylvain Pogodalla). In 2006, Joseph Le Roux and Jonathan Marchand joined the development team.

This implementation (http://www.loria.fr/equipes/calligramme/leopar/) is a public project on the InriaGforge platform (http://gforge.inria.fr/projects/leopar/). It is freely available under the CECILL License (http://www.cecill.info). A release for a larger audience is planned for the beginning of 2007.

The main features of the current implementation are:

- Automatic parsing of a sentence or a set of sentences;
- Manual parsing (the user chooses the couple of nodes to merge);
- Visualization of grammars produced by XMG or of set of description trees associated to some French word;
- A graphical interface (using GTK) which is useful for debugging grammars.

The main features added this year:

- a new tabular parsing algorithm inspired by the CKY algorithm for context-free grammars,
- a new Earley-like parsing algorithm,
- a more modular management of the linguistic resources.

The current implementation comes with a middle-size coverage grammar for French (830 tree descriptions in the grammar produced with XMG). It also includes hand-made morphological and syntactical lexicons that cover the French examples of the TSNLP (Test Suite for Natural Language Processing) [36].

5.2. XMG

Keywords: *metagrammar*.

Participants: Joseph Le Roux [correspondant], Yannick Parmentier [Langue Et Dialogue].

The eXtensible MetaGrammar (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.

This software is based on two important concepts from logic programming, namely the Warren's Abstract Machine and constraints on finite sets. It has been developed by Benoît Crabbé, Yannick Parmentier, Denys Duchier and Joseph Le Roux. It is available at http://sourcesup.cru.fr/xmg. It is now maintained by Ph.D students Yannick Parmentier and Joseph Le Roux.

At current stage of implementation, XMG generates Tree Adjoining Grammars and Interaction Grammars but the underlying formalism is generic so it could be extended to others grammars like dependency grammars or lexical functional grammars, depending on users' requests.

Recent developments of XMG include:

- making XMG as generic as possible in order to generate target grammars in various formalisms[17];
- incorporating linguistic theories in the generated grammars [14].

XMG is used in the research field (by Guy Perrier, Claire Gardent and Owen Rambow) to design lexicalized grammars for NLP parsers and in computational linguistics teaching.

5.3. ACG support system

Keywords: *Abstract Categorial Grammars*.

Participants: Sylvain Pogodalla [correspondant], Philippe de Groote.

The current ACG development toolkit is being rewritten. It aims at providing support for the planned extension of the ACG type system and to offer a more modular architecture. A one-month trainee (Florent Pompigne, L3 ENS Cachan) worked on the parsing of grammar files.

5.4. Crocus

Participant: Guillaume Bonfante [correspondant].

CROCUS is a program that synthetizes quasi-interpretations. A quasi-interpretation is a way of proving the complexity of (functional programming style) programs. It does not provide termination but, associated to some termination orderings such as RPO, it gives polynomial bounds on the execution of programs.

CROCUS uses as input programs written in a specific first order language, called qi. But we have implemented a small tool that transforms (restricted form of) CAML programs into qi programs. This small tool is written in CAML.

We have currently implemented rather realistic algorithms (in CAML):

1. sorts (insertion sort, bubble sort);
2. Huffman coding trees (coding, decoding);
3. list algorithms (insertion, find, ...);
4. quantified boolean formulae;
5. longest common subsequence.

6. New Results

6.1. Proof Nets, Sequent Calculus and Typed Lambda Calculi

Keywords: *linear logic*, *proof nets*, *sequent calculus*.

Participant: François Lamarche.

6.1.1. Proof Nets for Units in Linear Logic

In [9] François Lamarche and Lutz Straßburger present a complete theory of proof nets and multiplicative units in linear logic, and then prove that the category of proof nets thus obtained is indeed the free *-autonomous category. One important aspect of the theory is that it uses only standard components like axiom links, tensor links... including an ordinary correctness criterion like Danos-Regnier. The difference is that some of these links are used in a non-standard way, as a structure describing in fine details how the bottoms are attached to the rest of the net. As is expected a theory of how to quotient these nets is needed and provided. It is the only complete (and self-contained) proof of the construction of the free *-autonomous category available in the literature.

6.1.2. Denotational Semantics of Classical Logic

François Lamarche begins the paper [22] by giving a full categorical axiomatization of the Medial rule, which is an essential component for the presentation of classical logic by the means of deep inference. It takes the form of additional structure on a *-autonomous category (so the symmetry of classical logic is kept), which can be expressed in the language of monoidal functors. He then shows abstract conditions that are necessary for a class of "bimonoids" in such a category to become a model of classical logic. The reason a model of this kind does not fall prey to "Joyal's paradox" (i.e., collapse to a poset) is that the full bimonoid structure is not preserved by every linear map between bimonoids.

Then he shows that such things actually exist in nature, by first exhibiting an example of Medial structure in a modified version of the category of coherence spaces and linear maps, made famous by Jean-Yves Girard. The final step is the extraction of classes of bimonoids in that category that obey the necessary "intrinsicness" condition to become models of classical logic. Several such classes are exhibited. One of them is very natural, but it does not have the ability to count how many times a given axiom link is reused by means of contraction. The final model has this ability (up to a finite, but arbitrary number of reuses), but its construction is much more involved, and it is the first such model ever constructed (proof net categories do not have that property). One interesting aspect of these semantics is that they contain additive counterparts to conjunction and disjunction, which are equivalent to the traditional connectives from the point of view of provability, but not from the point of view of naming proofs.

6.2. Categorial Grammars

Keywords: *Abstract Categorial Grammars*, *Earley algorithm*, *Interaction Grammars*, *discourse dynamics*, *scope ambiguity*.

Participants: Philippe de Groote, Guy Perrier, Sylvain Pogodalla, Bruno Guillaume, Joseph Le Roux, Jonathan Marchand.

6.2.1. Abstract Categorial Grammars

In collaboration with Makoto Kanazawa's team (NII, Tokyo), we have characterized the expressive power of several fragments of the Abstract Categorial Grammar hierarchy. In particular, we have obtained a complete characterization of the second-order fragment in the case of string languages. The successive layers of this fragment correspond respectively to regular languages, context-free languages, yields of linear context-free languages (which coincide, in the monadic case, with the languages generated by Tree Adjoining Grammars), and mildly context-sensitive languages. Then, the hierarchy collapses. We have also generalized these results to the case of tree languages. A joint journal paper is under preparation.

In order to increase the modeling capacity of the abstract categorial framework, we have proposed an extension of its type system[1]. This includes enumerated types, records, variants (in order to define feature matrices), and dependent products (in order to define parametric syntactic categories).

We have shown how to allow for discourse dynamics in a categorial framework. The idea consists in providing Montague semantics with an appropriate notion of context. The resulting framework subsumes Discourse Representation Theory without appealing to any ad hoc definition. It is based on Church's simply typed λ-calculus, and the notions of free and bound variables are as usual. In particular, there is no need for any kind of variable renaming other than the standard notion of α-conversion [21].

We also proposed a modeling of scope ambiguity in the framework of ACGs [19]. This is a generalization of the type-theoretic approach of type-logical grammars that enables to associate many semantic representations to a unique syntactic representation.

[1] http://research.nii.ac.jp/~kanazawa/lcfg2/acg-ext.pdf

6.2.2. Interaction Grammars

We developed original techniques for lexical selection –a major issue with strongly lexicalized formalisms– based on the notion of polarity which is the heart of Interaction Grammars. This appoach is twofold:

1. at the sentence level, we check the global neutrality of a selection, which is needed for correct parsing;
2. at the constituent level, we take advantage of the syntactical modelisation of some phenomena (e.g., coordination) to refine the global neutrality constraint by checking polarities before and after distinguished tree descriptions (e.g., tree description corresponding to *and*).

This method gives very good results as stated in [13].

Jonathan Marchand and Joseph Le Roux have designed an Earley-like parsing algorithm for Interaction Grammars [23]. This algorithm has been implemented in LEOPAR. Although it is still under development, first tests on our current corpora show encouraging results and we will be soon able to measure the actual improvement of the algorithm.

6.3. Development of linguistic resources

Keywords: *French formal grammar*, *Gross' grammar lexicon*, *lexicon*, *subcategorisation*.

Participants: Guy Perrier, Bruno Guillaume.

6.3.1. Extraction of a syntactical lexicon from Maurice Gross' grammar lexicon

For French, there exists to date no reference lexicon that would contain detailed extensive subcategorisation information (that is, information about the complements of natural language predicative items such as verbs, deverbal nouns and predicative adjectives).

In the paper [12], Claire Gardent (Langue Et Dialogue team), Bruno Guillaume, Guy Perrier and Ingrid Falk (ATILF) propose a method for producing such a syntactical lexicon from the LADL tables (Maurice Gross' grammar lexicon).

LADL tables provide a systematic description of the syntactic properties of the functors of French: verbs, predicative nouns and adjectives. Subcategorisation information contained in this lexicon is both detailed and extensive. Although the LADL tables are rich in content, their current format and structure make them difficult to use in NLP applications.

Hence, we propose a method for extracting from the LADL tables, an NLP-oriented syntactic lexicon. In essence, this method aims at making the table structure explicit and at translating the headings into standard practice feature structure notation. For each table, a graph is (manually) produced which represents the interpretation of the table. This graph makes the table structure explicit and translates the headings into path equations. Then the NLP lexicon is automatically produced.

6.3.2. Development of an interaction Grammar for French

Guy Perrier has developed an interaction grammar for French using XMG [18]. The methodology is inspired by Benoit Crabbé, who has developed a large French TAG [26].

The source grammar is composed of 449 classes organized in an inheritance hierarchy with two operators of conjunction and disjunction. The leaves of the hierarchy describe elementary phenomena of the grammar. Conjunctions and disjunctions express two ways of representing complex phenomena: for instance, a particular diathesis for a verb can result from the conjunction of classes representing specific realizations of its arguments and the realization of a particular predicate argument structure can be expressed by the disjunction of the classes representing the different diatheses.

The compiled grammar is composed of 830 tree descriptions mainly covering the following phenomena of the French syntax:

- most subcategorisation frames for verbs, predicative adjectives and nouns,
- active, passive, middle and reflexive diatheses combined with personal and impersonal subject constructions,
- grammatical words and related syntactic constructions (clitics, personal, relative and interrogative pronouns, complementizers, prepositions, negations, auxiliary verbs, ...),
- some hard to model phenomena such as: pied-piping in relative and interrogative clauses, islands for wh-extraction, long distance dependencies related to negative expressions (“ne...aucun”, “ne...personne”), past participle agreement in presence of the auxiliary “avoir”, control of the subject for the infinitives...

The grammar is in the process of being evaluated on the TSNLP [36] and Eurotra [28] test suites.

6.4. Implicit Complexity of Computation

Keywords: *computer virus*, *defense policy*, *first-order functional programming*, *implicit complexity*, *quasi-interpretation*, *self-reference*, *self-reproducing machines*.

Participants: Jean-Yves Marion, Guillaume Bonfante, Romain Péchoux, Matthieu Kaczmarek.

6.4.1. Implicit Complexity

New results have been obtained this year, from which the main ones are the following. First of all, Guillaume Bonfante, Jean-Yves Marion and Romain Péchoux have given a characterization of the parallel complexity class NC1 which is also ALOGTIME [20].

Guillaume Bonfante has contributed to the work of Neil Jones by refining the characterization of LOGSPACE, in particular some non-tail recursive programs may be in LOGSPACE. Guillaume Bonfante also proposed a new characterization of PTIME within the language WHILE of Jones where programs may be non-deterministic. It is an example of a language where a choice operator does not increase the extensional power of the language [10].

Romain Péchoux and Jean-Yves Marion proposed a new notion of sup-interpretation. These are more powerful than quasi-interpretations since they involve less constraints on the interpretations. Typically, algorithms such as divide and conquer algorithms are covered by this approach which is not the case of the approach with quasi-interpretations [16], [15].

6.4.2. Abstract Virology

Guillaume Bonfante, Jean-Yves Marion and Matthieu Kaczmarek have proposed a new definition of computer viruses. It is of crucial importance since this is fundamental in a perspective of defense. In the settings of Guillaume Bonfante, Jean-Yves Marion and Matthieu Kaczmarek, the detection remains undecidable but we have shown that in some particular cases, detection is possible. In particular, by means of information theory, a formal defense has been proposed against viruses.

Another contribution concerns the definition and the construction of polymorphic viruses. This is done in a very practical way, and we show how a virus compiler can be easily written [7].

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 specialized software for those tasks.
- Calligramme is part of the “Qualité et sûreté des Logiciels (QSL)” theme of the “contrat de plan État-Région”. Jean-Yves Marion is head of the QSL theme;

  The web page of this action is http://qsl.loria.fr.

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. This year was the last one for this action.

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 Marseille, and the co-ordinator on Calligramme’s side is Jean-Yves Marion.

7.2.3. Action Concertée Incitative (ACI) Géocal

This “Nouvelles interfaces des mathématiques” ACI (2003-2006) 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 Marseille, and the co-ordinator on Calligramme’s side is Jean-Yves Marion.

The web page of this ACI is http://iml.univ-mrs.fr/~ehrhard/geocal/geocal.html.

7.2.4. Groupe de Recherche (GDR) Informatique et Mathématiques (IM) "Géometrie du calcul"

Several members of the Calligramme team belong to this group of researchers that benefit from a Ministry of Education grant to stage two meetings a year on the connections between computation and geometry. The first meeting was held in Lyon in late october.

the web page of this GDR is http://perso.ens-lyon.fr/daniel.hirschkoff/geocal/.

7.2.5. Agence Nationale de la Recherche (ANR) Inval

Headed by Éric Goubault at the CEA, this three-year “programme blanc” action (started in November 2005) aims at studying and developing algebraic invariants of computation, inspired by traditional homology and homotopy in algebraic topology. Two meetings have been held in 2006, one in Montpellier in May and one in Strasbourg in November. The co-ordinator for the Loria site is François Lamarche.

7.2.6. Agence Nationale de la Recherche (ANR) Infer

This three-year “programme blanc” action on the theoretical and applicative development of deep inference started in December. Two Loria teams, Calligramme and Protheo, are involved in it, along with teams at INRIA-Futurs and the PPS lab (Université Paris VII). The head of the project is Lutz Straßburger (Parsifal, INRIA-Futurs), and the local co-ordinator is François Lamarche.

7.2.7. Agence Nationale de la Recherche (ANR) Virus

The three-year ARA Sécurité, Systèmes embarqués et Intelligence ambiante "Virus" began on September 2006. It deals with the theory of computer viruses. It involves some members of Calligramme and the group of Éric Filiol at the ESAT-Rennes, a group leader in computer virology. The head of the Project is Jean-Yves Marion.

7.2.8. LexSynt project

Calligramme is involved in the LexSynt project. Thirteen French-speaking research teams work on this project. It aims at developing a syntactic lexicon with large coverage for French. In order to be usable in various NLP applications, this lexicon is independent of any grammatical formalism.

the web page of the project is http://lexsynt.inria.fr.

7.2.9. Action de Recherche Concertée (ARC) Mosaïque

Calligramme is involved in the Mosaïque INRIA ARC. Nine French-speaking research teams work on this project. It aims at developing a high level description language for the syntax of natural languages and a software environment for building large scale grammars, especially French grammars.

The web page of the project is http://mosaique.labri.fr.

7.2.10. Agence Nationale de la Recherche (ANR) Prelude

Calligramme is involved in the "programme blanc" action PRELUDE http://prevert.upmf-grenoble.fr/~alecomte/PRELUDE.htm. This action is starting and aims at giving a theory of pragmatics based on ludics [30] and continuations [21]. The partner teams are: Structures Formelles de la Langue[2](coordinator), Institut Mathématique de Luminy[3], the Signes INRIA project[4] and Calligramme.

7.3. Visits and invitation of researchers

- Philippe de Groote and Sylvain Pogodalla visited Makoto Kanazawa (NII, Tokyo) from January 30th to February 10th.
- Philippe de Groote visited Michael Moortgat (Utrecht Uniuversity) on March, 29th and 30th.
- Reinhard Muskens (Tilburg University) visited the Calligramme Project from July 1st to July 31st.
- Lutz Straßburger (Parsifal, Inria-Futurs) and François Lamarche met several times, both at the Loria and at École Polytechnique (where François Lamarche gave a talk at the Parsifal Seminar), and also at other workshops. One of these meetings at the Loria in October was the occasion for an impromptu workshop where they both gave talks, in addition to Yves Guiraud (Protheo).
- François Lamarche was invited by Christoph Benzmüller at the University of Saarbrücken in July, where he gave a talk on the semantics of dependent types.
- François Lamarche was invited to give a talk at the PPS seminar (Université Paris VII) on November 29, on the denotational semantics of classical logic.

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.

[2] UMR 7023, Paris 8, http://recherche.univ-paris8.fr/red_fich_equ.php?OrgaNum=48
[3] CNRS, http://iml.univ-mrs.fr/
[4] http://signes.labri.fr/

- 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, 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 is an associate editor of the journal *Higher-Order and Symbolic Computation*. He belongs to the editorial board of the series *Papers in Formal Linguistics and Logic* (Bulzoni, Roma), and *Cahiers du Centre de Logique* (Academia-Bruylant, Louvain-la-Neuve).
- Philippe de Groote was member of the program committees of LACL’05, and UNIF’05.
- François Lamarche was member of the Bureau of the Département de Formation Doctorale in Computer Science of the IAEM doctoral school, until September.
- François Lamarche heads the research section (theses, postdocs and *ingénieurs spécialistes*) of the Comipers hiring committee at the LORIA.
- Jean-Yves Marion is member of the steering committee of the International workshop on Logic and Computational Complexity (LCC).
- Jean-Yves Marion is member of the hiring committee (CS) at the University of Metz, section 27, since September 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 QSL seminars http://qsl.loria.fr. Every seminar gathers between 10 and 40 participants. There were 22 seminars since January 2003.
- Guy Perrier is a member of the editorial board of the journal *Traitement Automatique des Langues*.
- Guy Perrier is a member of the Program Committee of the conference TALN’2006.
- Guy Perrier is a member of the Bureau of the Département de Formation Doctorale in Computer Science of the IAEM doctoral school.

8.2. Teaching

- Jean-Yves Marion is in charge of the option “Ingénierie des systèmes informatiques” at École des Mines de Nancy starting in September.
- Jean-Yves Marion took part in the creation of the formation in computational biology at École des Mines de Nancy and is in charge of the course on “Bases et banques de données”.
- Guy Perrier heads the specialization “Traitement Automatique des Langues” which is common to the masters in computer science and cognitive sciences of the universities Nancy 2 and Henri Poincaré. This master specialization is involved in the Erasmus Mundus Program "Language and Communication Technologies", which has just been accepted by the European Union Commission.
- Guy Perrier is in charge of the organization of the course on *tools and algorithms for the parsing of natural languages*, which he is teaching with Bertrand Gaiffe in the master’s specialization “Traitement Automatique des Langues”.
- Guy Perrier is teaching the courses "Initiation au traitement automatique des langues (morphologie et syntaxe)" and "Bases de données lexicales" in the masters in computer science and cognitive sciences of the universities Nancy 2 and Henri Poincaré.

- Philippe de Groote is teaching the course “Sémantique computationnelle” of the Nancy master specialization “Traitement Automatique des Langues”.
- Philippe de Groote and Gérard Huet are teaching the course “Structures Informatiques et Logiques pour la Modélisation Linguistique” of the “Master Parisien de Recherche en Informatique”.
- Sylvain Pogodalla is teaching the course “Corpus linguistics and linguistics resources” of the Nancy computer science master, specialization “Traitement Automatique des Langues”.

8.3. Academic Supervision

- Philippe de Groote is supervising the thesis work of Sarah Maarek (starting from October 2006).
- François Lamarche is supervising the thesis work of Robert Hein (starting from September 2006).
- Guy Perrier is supervising the thesis work of Joseph Le Roux.
- Guy Perrier and Bruno Guillaume are co-supervising the thesis work of Jonathan Marchand (starting from October 2006).
- Jean-Yves Marion is supervising the thesis work of Romain Péchoux from September 2004.
- Jean-Yves Marion and Simona Ronchi Della Rocca (Torino university) are co-supervising the thesis work of Marco Gaboardi (Protheo team).
- Jean-Yves Marion and Guillaume Bonfante are co-supervising the thesis work of Matthieu Kaczmarek from September 2005.
- Bruno Guillaume and Guy Perrier advised a master student at the university Henri Poincaré, Mathieu Morey, for a two month internship devoted to the extraction of a lexicon usable in NLP systems from Maurice Gross’ grammar lexicon [33].

8.4. Thesis juries

- Philippe de Groote was referee and jury member for Patrick Thevenon’s and Saber Khelifa’s theses, Université de Savoie, December 5.
- Guy Perrier was jury member for Janna Khegai’s thesis, Göteborg, October, 13.

8.5. Participation to colloquia, seminars, invitations

- Sylvain Pogodalla and Philippe de Groote attended the Second Workshop on Lambda Calculus and Formal Grammar, Tokyo, February 2[5]. Sylvain Pogodalla gave an invited talk (“About Higher-Order in Semantic Representation for TAGs”)[6] and Philippe de Groote gave an invited talk (“Type-Theoretic Extensions of Abstract Categorial Grammars”)[7].
- Philippe de Groote and Guy Perrier attended the GEOCAL Workshop on logic and linguistics, Marseilles, February 13-17[8]. Philippe de Groote gave an invited talk (“Abstract categorial grammars: definition and formal properties”)[9] and Guy Perrier gave an invited talk (“Interaction Grammars and their implementation”) [10].
- Philippe de Groote attended the SALT conference, in Tokyo, March 22-24[11] and presented [21].
- Philippe de Groote and Sylvain Pogodalla attended and gave invited talks at the MathDIALOG workshop[12] at the Saarland University, March 14.

[5] http://research.nii.ac.jp/~kanazawa/lcfg2.html
[6] http://research.nii.ac.jp/~kanazawa/lcfg2/nii-2006.pdf
[7] http://research.nii.ac.jp/~kanazawa/lcfg2/acg-ext.pdf
[8] http://iml.univ-mrs.fr/~mrd/linguistic_page.html
[9] http://www.cirm.univ-mrs.fr/videos/2006/exposes/w3%20linguistic/Degroote-slides.pdf
[10] http://www.cirm.univ-mrs.fr/videos/2006/exposes/w3%20linguistic/Perrier_slides.pps
[11] http://research.nii.ac.jp/salt16/index.html
[12] http://www.ags.uni-sb.de/~omega/workshops/MathDIALOG06/

- Guy Perrier gave a talk in the seminar of the LIPN (Laboratoire d'Informatique de Paris Nord), January, 10.
- Guy Perrier gave a talk in the seminar of the "Master Parisien de Recherche en Informatique", March, 20.
- Bruno Guillaume and Guy Perrier attended the TALN'2006 conference, in Leuven, Belgium. Bruno Guillaume presented [12].
- Bruno Guillaume, Philippe de Groote, Kristofer Johannisson and Sylvain Pogodalla attended the DEMONAT workshop at Chambery university, January 17-18.
- Karën Fort, Bruno Guillaume, Joseph Le Roux, Guy Perrier attended several LexSynt Worshops (January 19 in Nancy; March 9 in Bordeaux; April 28, September 28 and December 4 in Paris).
- Bruno Guillaume, Joseph Le Roux, Guy Perrier attended several Mosaïque Worshops (March 8 in Bordeaux; May 9, July 4 and December 1 in Paris).
- Guy Perrier attended the workshop on Large-scale Grammar Development and Grammar Engineering, Haïfa, Israël, in June.
- François Lamarche attended the two Inval workshops, the one in Montpellier in May and the one in Strasbourg in November. The second meeting was also attended by Robert Hein.
- François Lamarche attended several workhops related to deep inference, including one in Bath, UK, in June and one in Paris on December 1st-2, which was also attended by Robert Hein.

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^{o} 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^{o} 1-2, 2000, p. 192–208.

[3] G. PERRIER. *Interaction Grammars*, in "CoLing 2000, Sarrebrücken, Germany", International Committee on Computational Linguistics, August 2000, p. 600–606.

[4] G. PERRIER. *La sémantique dans les grammaires d'interaction*, in "Traitement Automatique des Langues", vol. 45, n^{o} 3, 2004, p. 123–144.

[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^{o} 3, August 2002, p. 355–388.

Year Publications

Articles in refereed journals and book chapters

[7] G. BONFANTE, M. KACZMAREK, J.-Y. MARION. *On Abstract Computer Virology from a Recursion Theoretic Perspective*, in "Journal in Computer Virology", vol. 1, March 2006, p. 45-54, http://dx.doi.org/10.1007/s11416-005-0007-4.

[8] O. BOURNEZ, F. CUCKER, P. JACOBÉ DE NAUROIS, J.-Y. MARION. *Implicit Complexity over an Arbitrary Structure: Quantifier Alternations*, in "Information and Computation", vol. 204, n° 2, 2006, p. 210–230, http://hal.inria.fr/inria-00000516/en/.

[9] F. LAMARCHE, L. STRASSBURGER. *From Proof nets to the Free *-Autonomous Categories*, in "Logical Methods in Computer Science", vol. 2, n° 4:3, 2006, p. 1–44, http://hal.inria.fr/inria-00099865/en/.

Publications in Conferences and Workshops

[10] G. BONFANTE. *Some programming languages for LOGSPACE and PTIME*, in "11th International Conference on Algebraic Methodology and Software Technology - AMAST'06, 07/2006, Kuresaare/Estonie", 2006, http://hal.inria.fr/inria-00105744/en/.

[11] P. FONTAINE, J.-Y. MARION, S. MERZ, L. PRENSA NIETO, A. TIU. *Expressiveness + Automation + Soundness: Towards Combining SMT Solvers and Interactive Proof Assistants*, in "12th International Conference on Tools and Algorithms for the Construction and Analysis of Systems - TACAS'06, 03/2006, Vienna/Austria", H. HERMANNS, J. PALSBERG (editors). , Lecture Notes in Computer Science, vol. 3920, Springer, 2006, p. 167-181, http://hal.inria.fr/inria-00001088/en/.

[12] C. GARDENT, B. GUILLAUME, G. PERRIER, I. FALK. *Extraction d'information de sous-catégorisation à partir des tables du LADL*, in "Traitement Automatique de la Langue Naturelle - TALN 2006, 04/2006, Leuven/Belgique", Actes de la 13ème conférence sur le Traitement Automatique de la Langue Naturelle, 2006, http://hal.inria.fr/inria-00103163/en/.

[13] J. LE ROUX, G. BONFANTE, G. PERRIER. *Lexical disambiguation with polarities and automata*, in "11th International Conference on Implementation and Application of Automata, Taipei/Taiwan", 2006, http://hal.inria.fr/inria-00113369/en/.

[14] J. LE ROUX, B. CRABBÉ, Y. PARMENTIER. *A constraint driven metagrammar*, in "The Eighth International Workshop on Tree Adjoining Grammar and Related Formalisms (TAG+8), 15/07/2006, Sydney/Australie", 2006, http://hal.inria.fr/inria-00083550/en/.

[15] J.-Y. MARION, R. PÉCHOUX. *Quasi-friendly sup-interpretations*, in "8th International Workshop on Logic and Computational Complexity - LCC 2006 - LICS affiliated Workshop, 10/08/2006, Seattle/Etats-Unis", James Royer, 2006, http://hal.inria.fr/inria-00110245/en/.

[16] J.-Y. MARION, R. PÉCHOUX. *Resource analysis by sup-interpretation*, in "Eighth International Symposium on Functional and Logic Programming - FLOPS 2006, 24/04/2006, Fuji Susono/Japan", M. HAGIYA, P. WADLER (editors). , in: Lecture Notes in Computer Science, Functional and Logic Programming, n° 3945, Springer, 2006, p. 163–176, http://hal.inria.fr/inria-00000661/en/.

[17] Y. PARMENTIER, J. LE ROUX, B. CRABBÉ. *XMG - An expressive formalism for describing tree-based grammars*, in "11th Conference of the European Chapter of the Association for Computational Linguistics (Demo Session) - EACL, Trento/Italy", 2006, http://hal.inria.fr/inria-00001133/en/.

[18] G. PERRIER. *A French Interaction Grammar*, in "Workshop on Large-scale Grammar Development and Grammar Engineering, Haïfa, Israël", 2006, http://hal.inria.fr/inria-00111662/en/.

[19] S. POGODALLA. *Generalizing a Proof-Theoretic Account of Scope Ambiguity*, in "7th International Workshop on Computational Semantics, 01/2007, Tilburg/The Netherlands", 2006, http://hal.inria.fr/inria-00112898/en/.

[20] R. PÉCHOUX, J.-Y. MARION, G. BONFANTE. *A characterization of Alternating log time by first order functional programs*, in "13th International Conference on Logic for Programming Artificial Intelligence and Reasoning - LPAR-13, 13/11/2006, Phnom Penh/Cambodia", Logic for Programming, Artificial Intelligence, and Reasoning, Springer, 2006, http://hal.inria.fr/inria-00110014/en/.

[21] P. DE GROOTE. *Towards a Motagovian account of dynamics*, in "Proceedings of Semantics and Linguistic Theory XVI", 2006.

Internal Reports

[22] F. LAMARCHE. *Exploring the Gap between Linear and Classical Logic*, Submitted to Theory and Applications of Categories, Rapport de recherche, INRIA, 2006, http://hal.inria.fr/inria-00113785/en/.

[23] J. MARCHAND. *Algorithme de Earley pour les grammaires d'interaction*, Mémoire de Master, INRIA, 2006, http://hal.inria.fr/inria-00114130/en/.

Miscellaneous

[24] J.-Y. MARION, J.-Y. MOYEN. *Heap-size analysis for assembly programs*, ACI CRISS, 2006, http://hal.archives-ouvertes.fr/hal-00067838/en/.

References in notes

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

[26] B. CRABBÉ. *Computational representation of strongly lexicalised syntactic formalisms: application to Tree Adjoining Grammars*, thèse de doctorat, university Nancy 2, 2005.

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

[28] L. DANLOS, O. LAURENS. *Présentation du projet EUROTRA et des grammaires d'Eurotra-France*, Technical report, CNRS, Université Paris 7, 1991.

[29] 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.

[30] J.-Y. GIRARD. *Locus Solum*, in "Mathematical Structures in Computer Science", vol. 11, 2001, p. 301–506.

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

[32] 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.

[33] M. GROSS. *Méthodes en syntaxe*, Hermann, 1975.

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

[35] 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.

[36] S. LEHMANN, S. OEPEN, S. REGNIER-PROST, K. NETTER, V. LUX, J. KLEIN, K. FALKEDAL, F. FOUVRY, D. ESTIVAL, E. DAUPHIN, H. COMPAGNION, J. BAUR, L. BALKAN, D. ARNOLD. TSNLP — *Test Suites for Natural Language Processing*, in "Proceedings of COLING 1996, Kopenhagen", 1996.

[37] 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", R. NIEUWENHUIS, A. VORONKOV (editors). , Lecture Notes in Artificial Intelligence, vol. 2250, Springer, Dec 2001, p. 233-248.

[38] 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.

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

[40] P. DE GROOTE, F. LAMARCHE. *Classical Non Associative Lambek Calculus*, in "Studia Logica", 2000.