1 Context and positioning of the proposal

Proof theory is one of the central pillars of mathematical logic. In the early 20th century, Hilbert's programme proposed to establish the consistency of mathematics by finitary means. Proof-theoretic investigations in this tradition were largely concerned with questions of *provability*. However, it became clear that in order to answer questions about provability a more detailed study of the structure of proofs is necessary. These investigations led to the development of the field of *structural proof theory* that studies proofs as combinatorial objects in their own right, and is therefore concerned with questions about the form and structure of proofs. The earliest result of this field is Gentzen's seminal cut-elimination theorem [20] that led to the construction of *analytic* calculi for classical and intuitionistic logics. (An analytic calculus is one where a proof uses only subformulas, in a sense adapted to the particular logical calculus, of the proved theorem.) The existence of such a calculus for a logic has many important implications, primary among them the consistency of the logic, and analyticity is indispensable for proof search.

In structural proof theory we can distinguish two traditions. The first, with mathematical roots, is concerned with first-order logic, and encompasses such methods as Herbrand's theorem [27, 11], Hilbert's ε-calculus [28] or Gödel's Dialectica interpretation [22]. The other tradition derives its motivations from computer science, therefore coming later than the first tradition, and is concerned with the propositional structure of proofs. It uses methods such as the Curry-Howard isomorphism [29], algebraic semantics, linear logic, proof nets and deep inference. The French project partners are situated primarily in the latter tradition, and the Austrian project partners in the former tradition. The proposed project will thus play an important role in the cross-fertilization of these different subfields of structural proof theory.

In both traditions, analytic proofs and procedures for obtaining such proofs play a paramount role. The aim of the proposed project is to further deepen the understanding of the structural properties of analytic proofs. More specifically, we plan to investigate extensions of the classical notion of *analyticity*, by using deep-inference calculi, and by investigating analyticity-properties in non-classical logics. Furthermore, we will treat questions about the complexity of transformations between analytic and non-analytic proofs. We also plan to compare such procedures in terms of the proofs they generate.

Structural proof theory plays a crucial role in the foundations of programming language. At the *propositional level*, computational interpretations of propositional deduction systems provide the basic instructions of programming languages. The classic example is of intuitionistic propositional natural deduction, whose computational interpretation leads to environment machines for functional languages. Changing the logic or the deduction system changes the computational model. Several computational models have been constructed by playing with possibilities, in particular along the lines of proof nets. At the *first-order level*, structural proof theory is used to extract correct programs from proofs, and to extract concrete bounds from abstract mathematical proofs. One of the original elements of the present project is to connect these two lines of research, which are often kept separate.

The present project is, from the French side, a prolongation of the ANR INFER project *"Theory and Application of Deep Inference"* between LIX (Palaiseau), LORIA (Nancy), and PPS (Paris). INFER has provided the theoretical basis on deep inference that is needed for our proposed research in STRUCTURAL, in particular for the tasks 2 (*Computational Interpretations of Deep Inference*) and 3 (*General Analytic Calculi in Deep Inference*).

The collaboration between the Austrian and French teams has been preceded by two ÖAD / PHC exchange projects, which have precisely the aim of initiating collaborations

- ÖAD-PHC Amadeus project *"Logic-Based Analysis of Computation"* FR 10/2009 between PPS and Universität Innsbruck. Duration 1/1/2009–31/12/2010. The aim of this project is to develop

logical tools for analysing and proving the correctness of programs. This research is directly related to the project goals of the proposed project: the Curry-Howard isomorphism allows to prove properties of programs while staying in a logical framework. Moreover, during this project possible relation between rewriting and deep inference have been discovered.

- ÖAD-PHC Amadeus on *"The Realm of Cut Elimination"* between INRIA Saclay – Île-de-France, PPS, and Technische Universität Wien (Austria). Duration: 01/01/2007–31/12/2008. Even though this project ended already two years ago, it established a close relationship between the Viennese and the Parisian partner that we want to further explore with STRUCTURAL.

2 Scientific and technical description

2.1 Background, state of the art

This project is about bringing together different aspects and developments in structural proof theory, namely *deep inference*, the *Curry-Howard correspondence*, *term rewriting*, and *Hilbert's ε-calculus*.

2.1.1 Deep Inference

Ever since Gentzen's seminal work [20], the main tools of structural proof theory have been the sequent calculus and natural deduction. In both formalisms, the main principle of deduction is to decompose the formula to be proven along their main connective, and continue to work on the subformulas. Only within the last decade, a new paradigm, called *deep inference* [26, 10] entered the scene, which is a radical departure from the use of principal (top-level) connectives used in traditional formalisms. Deep inference means that inference rules are allowed to modify formulas deep inside an arbitrary context. This change in the application of inference rules has drastic effects on the most basic proof theoretical properties of the systems, like *cut elimination*. Thus, much of the early research on deep inference went into reestablishing these fundamental results of logical systems. Now, *deep inference* is a mature paradigm, and for most logics studied in computer science, there is a deductive system using deep inference, for example classical logic [10], intuitionistic logic [6, 44], linear logic [26, 42], non-commutative logics [26, 18], and modal logics [40, 9]. And for these systems various techniques for proving cut-elimination have been presented, for example via rule permutation [42], via substitution [7], via splitting [23], or via atomic flows [24, 25].

The next step is to establish *new* proof theoretical results for deep inference systems that are not possible in shallow inference systems. The most remarkable results so far are decomposition theorems [42]. In general, a decomposition theorem says that a given system $\mathcal{S}$ can be divided into n pairwise disjoint subsystems $\mathcal{S}_1, \ldots, \mathcal{S}_n$ such that every derivation Δ in system $\mathcal{S}$ can be rearranged as composition of n derivations $\Delta_1, \ldots, \Delta_n$, where Δ_i uses only rules of $\mathcal{S}_i$, for every $1 \leq i \leq n$. Similar results are often obtained for term rewriting systems (see the next section), so a part of this project is to investigate their common relationships.

2.1.2 Term Rewriting

Term rewriting is a conceptually simple (but Turing-complete) model of computation. A *term rewrite system* (TRS) $\mathcal{R}$ is a set of rewrite rules $l \to r$ between terms (or term schemas) l and r. Let $\mathcal{T}$ denote the set of terms over some language. The *rewrite relation* $\to_{\mathcal{R}}$ is the least binary relation on $\mathcal{T}$ containing $\mathcal{R}$ such that (i) if $s \to_{\mathcal{R}} t$ and σ a substitution, then $s\sigma \to_{\mathcal{R}} t\sigma$ holds, and (ii) if $s \to_{\mathcal{R}} t$, then for all $f \in \mathcal{F}$: $f(\ldots, s, \ldots) \to_{\mathcal{R}} f(\ldots, t, \ldots)$ holds. The reflexive and transitive closure of $\to_{\mathcal{R}}$ is denoted as $\to^*_{\mathcal{R}}$.

Given a TRS $\mathcal{R}$, we can ask whether the relation $\rightarrow_{\mathcal{R}}$ is well-founded, that is whether $\mathcal{R}$ is *terminating*. A related problem is *reachability*: given terms s, t, does there exists a derivation $D\colon s \rightarrow_{\mathcal{R}}^{*} t$? Another well-studied property of TRS is *confluence* (also known as the *Church-Rosser* property): for all terms $s, t_1, t_2 \in \mathcal{T}$ with $s \rightarrow_{\mathcal{R}}^{*} t_1$ and $s \rightarrow_{\mathcal{R}}^{*} t_2$, does there exist a term t_3 such that $t_1 \rightarrow_{\mathcal{R}}^{*} t_3$ and $t_2 \rightarrow_{\mathcal{R}}^{*} t_3$? Recently, there has been a renewed focus on the *complexity* of a given TRS: for a derivation D (with respect to $\mathcal{R}$), what is the length of D? All these properties have been intensively studied in the last few decades, and for all questions a variety of automatable techniques are known.

Term rewriting systems are an important basis for theoretical computer science and are connected to proof theory in (at least) the following two ways. First, proof normalization can be understood as term rewriting in the lambda calculus or related calculi, where the proofs play the role of terms in the TRS. This correspondence, known as Curry-Howard isomorphism, is one of the pillars of programming language theory and proof theory in computer science. The second connection is that some deduction systems, like the Calculus of Structures, can be regarded as term rewriting systems for, *e.g.*, the equational theory of Boolean Algebras in the case of classical propositinal logic. In this setting, the formulas in a proof play the role of terms and the TRS that of the rules in the calculus. This second relation is less well investigated and will be an imporant aspect of the present project.

2.1.3 Curry-Howard correspondence

The Curry-Howard correspondence between proofs and programs is probably the most interesting, and also most puzzling, connection between mathematics and computer science. It was discovered in the 1960s, but its main development started in the 1980s. The basis of the correspondence is a correspondence between intuitionistic proofs and typed functional programs (written as terms of lambda-calculus). It has been extended in the beginning of the 1990s to classical proofs, where, surprisingly, different formal variants of the absurdity rule correspond to known variants of the control operators in programming languages. It is also the base of computing models, starting with proofs nets. Several models have been proposed in this direction, in particular interaction nets.

One of the guiding ideas behind the intensive research done in this field is that classical logic, or some variants of it, should provide new models for parallelism and concurrency. If this succeeds, it would provide a real logical control of these computing models, which are in pratice hard to master. However, so far there is still a gap between the computing level and the logical level. This is where Deep Inference could help to make some decisive advances. Correspondences between proofs and programs established so far use variants of natural deduction and sequent calculus, which are shallow inference systems. Deep inference provides new properties, not available in shallow deduction systems. Namely, full symetry and atomicity, which open new possibilities at the computing level. Enough theoretical work has now been done in Deep Inference to seriously consider these possibilities. This is one of the main tasks of this project.

2.1.4 Epsilon calculus

Hilbert's ε-calculus [28] is based on an extension of the language of predicate logic by a term-forming operator ε_x governed by the *critical axiom* schema $A(t) \supset A(\varepsilon_x A(x))$ for any term t. Within the ε-calculus, quantifiers become definable by $\exists x A(x) \equiv A(\varepsilon_x A(x))$ and $\forall x A(x) \equiv A(\varepsilon_x \neg A(x))$. Two fundamental results about the ε-calculus, the first and second ε-theorem, play a role similar to that which the cut-elimination theorem plays in sequent calculus. In particular, Herbrand's Theorem is a consequence of the ε-theorems.

One simple example of the merits of the ε-calculus is that, because quantifiers are encoded on the term-level, informal proofs are sometimes easier to formalize in the ε-calculus compared to *e.g.*, the

sequent calculus. The ε-calculus allows more condensed representation of proofs than standard sequent or natural deduction calculi. In particular, for the above mentioned ε-theorem, only knowledge of part of the proof is required: it suffices to know the set of critical axioms. Thus the propositional part of the proof can be largely ignored.

This potential of the ε-calculus as as supreme formalism for the formalization of informal proofs has never been properly exploited. Note that for the formalization of an informal proof it suffices to guess a suitable set of critical axioms and verify that this set entails the (translation of) the conclusion of the informal theorem at hand. Therefore a thorough complexity analysis of ε-calculus provides valuable information about informal proofs. Furthermore the possibility within the ε-calculus to ignore part of the proof seems to imply that this formalism should yield a unique and interesting computational interpretation. There is very little work in this direction.

It should be pointed out that the encoding of quantifiers on the term-level can come at the significant cost of complicating the term structure when transforming quantified formulas.

2.2 Rationale highlighting the originality and novelty of the proposal

One aspect of proof theory which has been remarkably stable during the 20th century is how deductive systems are formulated. There has not been any significant departure from Gentzen's seminal work, which introduced the sequent calculus and natural deduction in the mid-thirties (and Tableaux-systems are a variant of sequent calculus; and earlier Frege-Hilbert style systems are still used sporadically today).

For example the deductive system that are associated with linear logic are formulated in these traditional terms, the more frequent one being the sequent calculus. Deep inference's innovation lies at this very level: how a formal system is presented. It furthers the importance of linearity by incorporating it at the level of deduction steps themselves. Furthermore, in the calculus of structures, inference rules do not work on the root connective of the formula (as we know it from tableaux, sequent calculus, or natural deduction) but are conducted as rewriting steps (as in ordinary term rewriting), potentially at any depth inside the formula. This why the name *deep inference* was given to the whole enterprise. From the viewpoint of structural proof theory, deep inference seems like a step back in history, because modern proof theory has been made possible only through Gentzen's idea of rigorously exploring the concept of the main connective. The dropping of this concept caused a breakdown of all proof theoretical techniques developed since then. Indeed, most of the early research on deep inference went into the development of new techniques for proving things like cut elimination.

However, the use of deep inference results in a much finer analysis of proofs than what traditional systems permit; in particular it allows the proof of more precise versions of the classical theorems of proof theory, like interpolation and Herbrand's theorem; it also gives a more synthetic view of a formal proof. Now that these *preliminary* results are established, one important goal of the present project is to move on to the next step, namely tackling problems that are directly relevant to computer science.

Another aspect of proof theory which has not changed much over the last century is that it is mainly concerned with *formal* proofs. But the notion of *informal* proof has been notoriously neglected, although it should be an important topic of proof theory to study the relation between formal and informal proofs. This is where the ε-calculus comes in. Its full potential as supreme formalism for the formalization of informal proofs has never been properly exploited: To formalize a non-formal proof it is sufficient to determine the substitutions in the proof represented by critical formulas and to check whether these critical formulas already imply the translation of the result—if not some substitutions have been overlooked. Therefore a thorough complexity analysis of ε-calculus renders valuable informations about non-formal proofs.

Deep Inference and ϵ-calculus are two original tools of high potential that will be used in this project.

But there are two other original aspects of our project that should be emphasized:
- it connects two traditions of structural proof theory, one directed towards mathemathical applications, the other one directed towards computer science;
- it connects two levels of structural proof theory, one dealing with the propositionnal part of logic, the other one dealing with the first order quantifiers.

Building on this original aspects, the project intends to develop fundamental tools and techniques of structural proof theory having a strong potential of applications in computer science, in particular at the level of the models of computation and the extraction of programs and effective bounds from proofs.

3 Scientific and technical programme, project management

3.1 Scientific programme, specific aims of the proposal

The projects is organized in 9 tasks which are developed in section Section 3.3

- Task 1: Foundational Issues in Deep Inference
- Task 2: Computational Interpretations of Deep Inference
- Task 3: General Analytic Calculi in Deep Inference
- Task 4: Deep Inference as Term Rewriting System
- Task 5: Deep Inference in Modal Logics
- Task 6: Proof Compression via Cut-Introduction and Tseitin Extension
- Task 7: Hilbert's Epsilon Calculus and Proof Complexity
- Task 8: Herbrand-disjunctions and computational interpretations of proofs
- Task 9: Gödel's Dialectica interpretation versus cut-elimination

3.2 Project management

Michel Parigot from PPS (Partner P) will in charge of the coordination. He will be assited by a steering committee consisting of himself, Lutz Straßburger (Partner X), Georg Moser (Partner I) and Matthias Baaz (Partner V).

Due to the innovative and fundamental nature of the research project, there no a priori fix planning of execution of the tasks: flexibility and adaptation are necessary and the role of the steering committee will be to regularly adapt the tasks to the already obtained results.

In order take into count the fact that there are 4 partners in two countries and the different background of the Austrian and French teams - which is key factor of the success of the proposed researches - we plane to organize a workshop each 6 months, alternatively in each country.

We also plane to have longer stay (up to one month) of some participant of one team in another team for working on specific question which need an intensive collaboration of two or three persons.

3.3 Detailed description of the work organised in tasks

3.3.1 Task 1: Foundational Issues in Deep Inference

Deep inference is a recently developed paradigm for presenting formal systems. While traditional formalisms like natural deduction or the sequent calculus decompose formulas along their main connectives when progressing in the proof, deep inference formalisms, like the calculus of structures, allow the inference rules to rewrite parts deep inside the formulas.

In this project we want to continue the foundational research on deep inference. In particular, we want to investigate the combinatorics of deep inference derivations, as it has been stated in [41], which gives a sound an complete correctness criterion for derivations containing only the *medial* rule, which is central in the deep inference presentation for classical logic [10]. From the well-known Danos-Regnier correctness criterion [16] for linear logic, one can derive such a criterion also for the *switch* rule. It is however completely open how a combinatorial correctness criterion for a whole logical system could look like.

Participants: Partner P (Parigot, Herbelin, Roziere, Gimenez, Gundersen), Partner X (Straßburger, Miller, Chaudhuri, Houtmann, Guenot), Partner I (Moser), Partner V (Fermüller, Ciabattoni)

3.3.2 Task 2: Computational Interpretations of Deep Inference

Intuitionistic logic provides a foundation for functional programming though the Curry-Howard correspondence between intuitionistic proofs and typed functional programs (written as terms of the lambda calculus). This correspondence has served as the basis of an original computing model, the proof nets of linear logic, in the mid 1980s. This correspondence has been extended to classical logic in the beginning of the 1990s with, in particular, the lambda-mu-calculus introduced by M. Parigot. There have been numerous developments since then in the computational interpretation of classical logic. For instance, Curien and Herbelin showed the duality of call-by-name and call-by-value.

However, the computational interpretations that have so far been proposed for classical logic rely on variants of the deduction systems due to Gentzen in 1930s, natural deduction and sequent calculus. Deep inference opens a new and promising field of posssibilies. Deep inference is now mature enough, thanks in particular to the work done in the ANR INFER project, to give birth to new computational models exploiting its basic properties not available in sequent calculus: deep applications of inference rules, full symmetry, and atomicity. The recent work by Guglielmi, Gundersen, Straßburger and Parigot on atomic flows provide a good starting point and raise very interesting questions. For instance, what is the role of cocontractions and their relation to sharing? Cocontraction gives a quasipolynomial-time cut-elimination procedure in propositional deep inference, while all the known cut-eliminations procedures in sequent calculus are exponential. Connections with the differential lambda-calculus introduced by Ehrhard and Regnier, which uses cocontractions, also seem likely. Lastly, full symetry and atomicity suggest new possibilities for concurrency.

Participants: Partner P (Parigot, Ehrhard, Herbelin, Joly, Gimenez, Gundersen), Partner X (Straßburger, Houtmann, Guenot), Partner I (Moser), Partner V (Fermüller)

3.3.3 Task 3: General Analytic Calculi in Deep Inference

Since the axiomatisation of classical propositional logic by Hilbert in 1922, such axiomatic descriptions (nowadays called *Hilbert-systems*) have been successfully used to introduce and characterize logics. Ever since Gentzen's seminal work it has been an important task for proof theorists to design for these logics deductive systems that admit cut-elimination. The admissibility of cut is crucial to

establish important properties of corresponding logics such as consistency, decidability, conservativity, interpolation, and is also the key to make a system suitable for proof search. As designers of deductive systems could never keep pace with the speed of logicians and practitioners coming up with new logics, general tools to automate this design process and extract suitable rules out of axioms would be very desirable. Work in this direction are e.g. [31, 38, 36].

In this project we want to continue the development of a systematic and algebraic proof theory for non-classical logics that was recently launched in [13, 14]. The basic idea is to classify Hilbert axioms in the language of full Lambek calculus *FL* (*i.e.*, intuitionistic non-commutative linear logic without exponentials) into the *substructural hierarchy* $(\mathcal{P}_n, \mathcal{N}_n)_{n\in\mathbb{N}}$. The state of the art is that axioms up to level $\mathcal{N}_2$ can be translated into structural rules in the sequent calculus, and axioms up to the level $\mathcal{P}'_3$ (a subclass of $\mathcal{P}_3$) can be translated into structural rules in the hyper sequent calculus. Our conjecture is that deep inference can allow us to climb up the hierarchy even further, so to get a systematic proof theory for a larger class of non-classical logics.

Participants: Partner P (Hetzel), Partner X (Straßburger, Houtmann, Post-doc X), Partner V (Ciabattoni, PhD student V)

3.3.4 Task 4: Deep Inference as Term Rewriting System

Consider the following TRS $\mathcal{R}$

$$\begin{array}{lll} 1\colon & (x \bullet y) \circ (w \bullet z) \to (x \circ w) \bullet (y \circ z) & \qquad 3\colon \ \mathsf{a} \circ \mathsf{a} \to \mathsf{a} \\ 2\colon & x \bullet (y \circ z) \to (x \bullet y) \circ z & \qquad 4\colon \ x \to x \circ y \end{array}$$

This TRS represents a subsystem of the calculus of structurs, that is, a deep inference formalism: rule 1 is usually called *medial* rule 2 is the *switch* rule, and rules 3 and 4 are (atomic) contraction and weakening, respectively (see also [41]). One extends the above given TRS by an equational theory AC expressing that the function symbols $\bullet, \circ$ are associative and commutative. We write $s \to_{\mathcal{R}/\mathsf{AC}} t$, if $s =_{\mathsf{AC}} s' \to_{\mathcal{R}} t' =_{\mathsf{AC}} t$, for terms s', t'.

Thus, for example, proof search over the indicated inference rules can be expressed as the *reachability problem* with respect to rewriting modula AC, that is, with respect to the relation $\to_{\mathcal{R}/\mathsf{AC}}$. Moreover, proof complexity naturally translates to the *complexity of a TRS* (see [35] for further reading). Furthermore important conditions of consequnce of *confluence* can be successfully applied to show that the considered inference rules can be permuted freely.

Thus properties of the induced rewrite system yield structural properties of (deep inference) proofs. It is currently an open question to what extend techniques from rewriting developed for rewriting modulo AC (see for example [17, 30, 19, 37, 32, 43]) can be exploited to replace existing semantical arguments by purely combinatorial ones. The advantage of the later being that such arguments typically give rise to automated techniques.

Participants: Partner P (Parigot, Gundersen), Partner X (Straßburger, Houtmann, Guenot), Partner I (Moser, Avanzini, Schnabel, Post-doc I)

3.3.5 Task 5: Deep Inference in Modal Logics

Modal logics play an important role, not only in philosophy, but also in computer science. For example, temporal logics and epistemic logics are just special modal logics. However, traditional formalisms have, due to their attachment to the principle of the main connective (see Section 3.3.1), notorious difficulties with presenting deductive systems for modal logics. Consequently, modal logics are well

investigated only from the model theoretic point of view, but not from the proof theoretic point of view. Resent research has shown [40, 8, 9], that deep inference can be quite successful with giving cut-free deductive systems for modal logics. More precisely, [9] gives a completely modular account to the whole modal cube below the modal logic S5. That is to say, we have cut-free systems for the basic normal modal logics formed by any combination of the axioms d, t, b, 4, 5, such that each axiom has a corresponding rule and each combination of these rules is complete for the corresponding frame conditions. The natural continuation of this work is to extend the results to modal logics beyond S5, and possibly find a systematic way to translate modal axioms into structural rules, in a similar way as described for non-classical logics in Section 3.3.3.

Participants: Partner P (Hetzl), Partner X (Straßburger, Post-doc X), Partner V (Ciabattoni, Fasching)

3.3.6 Task 6: Proof Compression via Cut-Introduction and Tseitin Extension

One can safely say that the most important, and most fundamental, proof theoretical property of a deductive system is *cut elimination*. This is crucial for practical applications, since many computational logic systems, such as logic programming interpreters and model checkers, can be seen as producing cut-free proofs.

However, cut elimination usually comes at the cost of an exponential or hyper-exponential blow-up of the proofs, and indeed, in practice formal are usually large objects. Thus, it is reasonable to search for ways to compress them. In this project we want to investigate two such ways: cut-introduction, which can be seen as the reverse operation to cut-elimination, and the use of Tseitin extension, which is usually studied in the area of proof complexity.

- *Cut-Introduction:* Inside cut-free proofs there may be many sequents that are proved repeatedly. By using cut, it is possible to prove such a sequent once and then to have it assumed (as a lemma) so that any need to reprove that sequent is trivial. Such cuts are examples of "analytic cuts" since they only involve lemmas that are subformulas of the original theorem. Examples of using such analytic cuts were developed in [33] in the context of model checking optimized with "tabled deduction". However, this involved only atomic lemmas. The goal is to find more complex (non-analytic) lemmas, such that much greater compression should be possible. In general it is worth to discuss the possibilty to reconstruct proofs from their flow-graph descriptions, because this enables non-analytic cut introduction on a graph level (for first approaches cf. [3, 2]).
- *Tseitin extension:* Interestingly, so far, in structural proof theory usually only the *cut rule* is studied as source for proof compression. And the absence of cut is the cause of exponential blow-up in the size of proofs. However, in the area of proof complexity, one also studies Tseitin's *extension rule*. Unfortunately, this rule is only studied within Frege-Hilbert-systems [15], and is thus not available for structural proof theory. Naively eliminating extension also yields and exponential blow-up of the size of the proof. It is therefore a first goal of this project to use deep inference to provide a framework in which cut and extension can be studied independently and compared with respect to the proof compression that they provide. Once we have such a framework, we can start investigating the possibility of introducing extensions into a deduction in order to decrease its size.

Participants: Partner P (Hetzl, Gundersen), Partner X (Straßburger, Miller, Chaudhuri), Partner V (Baaz, Fermüller, Fasching)

3.3.7 Task 7: Hilbert's Epsilon Calculus and Proof Complexity

Hilbert's ε-calculus is of interest in the given context rests as this logical formalism gives rise to different bounds on the length of Herbrand disjunctions, than say the standard argument via the sequent calculus. The bound on the length of the Herbrand disjunction depends only on the number of critical axioms (aka the *critical count*) in the given proof, that is, essentially only on the number of quantifier inferences (see [34]). This is in contrast to the bound we would obtain by the more standard approach of cut-elimination and the mid-sequent theorem which depends on the length and cut complexity of the original proof (see, for example [12, 45]). It is open whether these results also extend to the first-order logic with equality. It remains to verify in what sense a bound on the length of Herbrand disjunction can be found in this case.

Furthermore, one ancillary practical benefit of the equivalence of $\exists x\, A(x)$ and $A(\varepsilon_x A(x))$ is that the right rule for $\exists$ in the sequent calculus can be given an invertible (also known as asynchronous or negative) reading, unlike the traditionally non-invertible (synchronous, positive) interpretation of $\exists$. Quantifier dependency is encoded in the nesting order of the εs, so both universal and existential variables are localized. The price paid for this interpretation of existential variables is that term unification, which is purely linear in first-order predicate logic, now becomes potentially undecidable because it might require checking the equivalence of arbitrary propositions. The practical experience of the Zenon theorem prover [5], which is a tableau prover using the ε-calculus, suggests that the additional cost of term unification is tolerable. The interaction of ε-terms with recent advances in the theory of the sequent calculus—such as focusing and polarity assignment—promises to be a fruitful source of new results.

Participants: Partner P (Parigot), Partner X (Chaudhuri), Partner I (Moser, Avanzini, Schnabel, Post-doc I), Partner V (Baaz)

3.3.8 Task 8: Herbrand-disjunctions and computational interpretations of proofs

We intend to develop an algorithm for the computation of Herbrand-disjunctions from proofs based on an approach that can be found [21]. There the authors adapt Gödel's Dialectica interpretation [22] to a setting of classical logic by adding case-distinction constants to the language of the lambda-calculus. The proof interpretation studied in [21] is based on Shoenfield's proof system [39] which has the disadvantage of not possessing a procedure for obtaining analytic proofs as cut-elimination in the sequent calculus and normalization in natural deduction. In order to compare term extraction mechanisms like the Dialectica interpretation with cut-elimination and normalization it is therefore important to develop an assignment of sequent calculus and natural deduction proofs in first-order classical logic to lambda terms that compute Herbrand-disjunctions. We plan to develop such an algorithm which is envisaged to use a lambda calculus enriched by case distinction constants as in [21]. This mapping should serve as a base for relating these two different approaches to the constructive content of a proof in classical logic.

Having laid a common formal framework for different approaches to the constructive content of a classical proof there, we will then conduct a comparison of these different approaches. In a first phase we will analyze examples, a particularly interesting one being the proof sequence described in [4] which exhibits a strongly non-confluent behaviour under cut-elimination. As term extraction mechanisms based on the lambda calculus are confluent the question arises how the (single) normal form extracted by such a mechanism compares to the set of normal forms produced by cut-elimination. In a second phase of this task we aim at general results relating the different Herbrand-disjunctions obtainable by these approaches.

In addition concepts of Herbrand disjunctions should be considered, which can be associated with

computational interpretations of number theoretic proofs (cf [1])

Participants: Partner P (Parigot, Hetzl), Partner X (Straßburger), Partner I (Moser), Partner V (Baaz, Fasching, Postdoc V)

3.3.9 Task 9: Gödel's Dialectica interpretation versus cut-elimination

As claimed by Georg Kreisel half century ago, general proof-theoretic techniques can be used to extract concrete values from abstract mathematical proof and calculate concrete bounds. Girard gave a beautiful example of this in 1981 with Van der Waerden theorem on progressions. He showed that by eliminating the cuts from the topological proof of this theorem due to Fürstenberg and Weiss, one can get the combinatorial proof of Van der Waerden with exactly the same bounds. This part of proof theory aimed at finding concrete mathematical applications remained inactive for a long time. But in the last decade, Kohlenbach started a proper research programme, called proof mining, aimed at extracting concrete bounds of classical mathematical theorem, in particular in functional analysis. The proof mining technique does not use cur elimination, as in the Girard case, but variant and extension of Gödel's Dialectica interpretation. This line of research has also been developed in the US with the work of Avigad on the Fürstenberg's proof of Szemereid using ergodic theory.

In this project, we do not intend to apply proof mining to extract concrete bounds of particular mathematical theorems, but only try to further analyze and develop the underlying proof theoretic techniques. Girard's technique has not been exploited yet, because, the way the cuts are eliminated is rather empirical. In order to make of it a general technique, one has to better understand how to guide cut elimination, and for this to better understand cut elimination itself. The relation between Girard's technique and proof mining needs also to be understood.

Participants: Partner P (Parigot, Herbelin, Roziere, Hetzl), Partner V (Baaz, Fasching, Postdoc V)

3.4 Planning of tasks, deliverables and milestones

The research involved, which is for most of the tasks, of a sophisticated theoretical nature, makes it impossible to give on serious grounds a precise schedule of the realisation of the tasks. In contrast to a fix planning of the tasks, what is needed for the success of the project is a flexible planning, that will be adapted to the course of the research depending on the results that will be obtained and which are not known in advance. For that reason we have planed a steering committee which will regularly discuss the status of the project.

For the same reason, we have not detailed out the time spent by each person on each task (required in the internet form in the French but not on the Austrian side), but only the time each person will spend globally.

4 Management of intellectual property, data management, data sharing, exploitation of results and communications

As usual in the academic community, we will present our results at major international conferences and publish in the relevant international journals.

Since this is a purely academic research project, there is no need of management of intellectual property.