Some General Results About Proof Normalization

Marc Aiguier, Longuet Delphine

Some general results about proof normalization

Marc Aiguier and Delphine Longuet

Abstract. In this paper, we provide a general setting under which results of normalization of proof trees such as, for instance, the logicality result in equational reasoning and the cut-elimination property in sequent or natural deduction calculi, can be unified and generalized. This is achieved by giving simple conditions which are sufficient to ensure that such normalization results hold, and which can be automatically checked since they are syntactical. These conditions are based on basic properties of elementary combinations of inference rules which ensure that the induced global proof tree transformation processes do terminate.

Keywords. Formal systems, proof tree transformation, weak and strong normalization, cut-elimination, rewriting.

1. Introduction

Motivation. In many situations in logic, to facilitate the use of a logical system, or to obtain consistency results for proof systems, or else to automatically prove theorems, one often uses proof search (or proof construction) strategies. These strategies enable one to bound the search space for proofs to a given class of trees having a specific structure. One of the interests is to reduce proof search space (or even to make proof search feasible). This has been devised in various different contexts. Here are a few examples: the so-called "logicality result", which establishes a correspondence between derivability and convertibility in rewriting for many equational logics (sub-equational [58], mono-sorted [11], multi-sorted, conditional [40], partial [4], *etc.*); the cut-elimination result which shows that the cut rule is redundant for sequent and natural deduction calculi of many logics (classical first order [33], intuitionistic first order [41], some modal logics [59], linear [34], deduction modulo [28], *etc.*); the confluence property of rewrite systems which establishes that derivability can be proved by "valleys" for many logics dealing with

transitive relations (equational [20, 8], preorder [42, 52], special relations [51]),[1] or by using the Curry-Howard's isomorphism, normalization results in typed λ-calculi as initiated by D. Prawitz [48]. More recently, the authors also used such proof normalization results to show the correctness of procedures based on axiom unfolding which enable us to generate test data sets from various specification formalisms [1, 2, 7, 43, 45, 44].

In all these cases, the main difficulty is to show that the full derivability (i.e. without any specific proof strategy) coincides with the derivability restricted to a given class of proofs (i.e. with a specific proof strategy). Soundness of a given strategy of proof search, which means that the restricted derivability is included into the full one, is obvious, because proofs resulting of such a strategy are particular instances of the general class of proofs. Completeness, which is expressed by the converse inclusion, is much more difficult. Indeed, it requires that for any theorem proved by a tree in the general class, there exists a proof in the restricted one (i.e. built according to the considered strategy). In most logical systems, completeness is the consequence of a stronger result which consists in defining basic transformations to rewrite elementary combinations of inference rules (possibly by duplicating a sub-derivation), and showing that the global proof tree transformation which is naturally induced by these basic transformations is normalizing.[2] For instance, concerning the logicality result for the classical equational logic, the basic transformations consist in "distributing" substitution and context rules above transitivity, and removing all rules under reflexivity. In sequent calculus, the basic transformations consist, roughly speaking, in erasing the cut rule under axioms and moving it above all other rules, in order to eliminate it.

These basic transformations are usually tedious but easy to define. The difficulty is to show that the induced *global* proof tree transformation is indeed normalizing. For instance, proving the result of cut elimination often requires complex nested inductions, because it is not obvious that the process to "move up" the cut rule (in order to eliminate it) is terminating [32]. Each time, such results of normalization of proof trees are established for each underlying logical system in an *ad-hoc* way.

Contribution. In this article, we unify and generalize the method of normalization of proof trees used in sequent calculus, natural deduction, equational reasoning, or term rewriting into an abstract setting of arbitrary logical systems. Abstraction is obtained by studying proof tree normalization in the abstract framework of *formal systems*. This enables one to be logical system independent. Indeed, in computer science and mathematics, many logics have been (will be) defined (in the future) to answer better, in a more suitable way, some specific aspect related to the activity

[1] Besides, in the equational setting, G. Dowek has shown that the confluence of a rewrite system can be defined as the cut elimination property of a proof system associated to this rewrite system: asymmetric deduction modulo [25].

[2] Our paper does not aim to generalize normalization by evaluation approach, the main idea of which is to use the semantics for the purpose of normalization as in [10].

of formalization. By "to answer better", we mean either the impossibility for existing formalisms to describe some problems (capabilities) or their unsuitability to describe problems clearly and concisely enough to be usable (expressivity). What is observed is that, most of the time, beside the original idea underlying a new logic, the authors have to develop a lot of inevitable formal results which generalize (to the new framework) some well-known classical results. It is then very useful to provide a general framework allowing one to generalize these results as this is done in this paper for proof trees normalization results.

Related work. Up to now, this kind of approach —i.e. the study of some properties in the paradigm of "logical-system independency"— has been widely applied to generalize results in specification theory [13, 24, 36, 56], model theory [5, 6, 23, 49, 50, 54, 55] and theorem deduction [31, 46]. But as far as we know, operational aspects of logics (here represented by normalization results on proofs) have not received as much attention at this abstract level. We can cite [3] where an abstract framework of rewriting is defined and standard results such as logicality, Newman's lemma and Knuth-Bendix completion are generalized, or [21, 22] where an abstract form of the completion process is given.

Nevertheless, some general theorems of strong normalization have been established in the case of term rewriting. In this framework, we can cite [17, 18] where the authors give general theorems about the well-foundedness of a certain ordering on first-order terms that entails the strong normalization of rewrite systems. These theorems subsume the strong normalization result we present in this paper (see Theorem 4.1). However, the sufficient conditions used to prove the well-foundedness of the underlying ordering require nontrivial proofs when applying to some concrete cases of rewrite systems, whereas our conditions are syntactical, so they can be automatically checked (see Condition 1 in Section 4). Moreover, our general theorem dealing with the weak normalization of proof trees, which is the main result of this paper, is not taken into account by the theorems established in [17, 18].

Structure of the paper. The article is organized as follows. In Section 2, we recall standard definitions and notations about formal systems, deductions and proof trees. In Section 3, we introduce the class of (proof tree) transformation procedures for which we will express the crucial (shared) arguments of many different normalization results as generic conditions on allowed transitions. In Section 4, we formalize a first set of conditions that enables us to prove a strong normalization result by using standard rewriting techniques. We show that the cut-elimination procedure for the sequent calculus **LK** where cut rule premises have the same context does not satisfy this first set of conditions, and then the underlying cut-elimination procedure is not strongly normalizing.[3] A thorough study of the reasons why such a strong normalization result fails enables us to set off the limits of standard rewriting techniques for proving termination such as RPO. Then, we

[3]While the cut-elimination procedure is strongly normalizing when different contexts are allowed in the premisses of the cut rule (see Section 5.3).

give a new set of sufficient conditions that enable us to prove a weak normalization result. In Section 5, we present three instances of our approach: equational reasoning and its logicality result, Newman's lemma that enables to transform any proof (written as series of peaks) into a valley, and (a version of[4]) sequent calculus **LK** for classical first-order logic and its cut-elimination property. We establish a strong normalization result for the two first and both a weak and a strong normalization result for the third one. Finally, in Section 6, as an application of our generalization, we present a cut elimination algorithm for the sequent calculus modulo developed by Dowek, Hardin and Kirchner [26]. This result differs from the original one developed in [38, 39] which gives a semantic proof of cut elimination whereas a syntactical proof is given in this paper.

2. Preliminaries

We recall here basic notions of proof theory [8], and we introduce the notion of deductive system to abstractly deal with any calculus in any logical system. A deductive system simply is a set of inference rules over a set of formulas.

Definition 2.1 (Deductive System). *A* deductive system *(a so-called* calculus*)* $\mathcal{S} = (F, R)$ *consists of:*

- *a set* F *whose elements are called* formulas[5]
- *a set* R *of* n*-ary relations on* F*, called* inference rules*.*

Given a deductive system, an inference rule defines a set of elementary proof steps: a rule of arity n ($n \geq 1$) is a set of tuples $(\varphi_1, \ldots, \varphi_n)$ of formulas of F. Each sequence $(\varphi_1, \ldots, \varphi_n)$ belonging to a rule r of R is called an *instance* of that rule, where $\varphi_1, \ldots, \varphi_{n-1}$ are called its *premises* and φ_n its *conclusion*. An instance of a rule $r \in R$ is usually written $\dfrac{\varphi_1 \ \cdots \ \varphi_{n-1}}{\varphi_n} r$. If $n = 1$, the instance is called an *axiom* and is written $\dfrac{}{\varphi_1} r$. In the following, we will denote by $\mathcal{I} = \bigcup_{r \in R} r$ the set of rule instances.

Instances of inference rules can be composed to build *proof trees* and *proofs*.

Definition 2.2 (Proof Tree and Proof). *A* proof tree π *in a deductive system* $\mathcal{S}$ *is a finite tree whose nodes are labelled by formulas of* F *such that if an internal node is labelled by* φ_n *and its predecessor nodes are labelled by* $\varphi_1, \ldots, \varphi_{n-1}$ *from left to right, then* $\dfrac{\varphi_1 \ \cdots \ \varphi_{n-1}}{\varphi_n} r$ *is an instance of a rule* $r \in R$*.*

A proof *is a proof tree whose leaves are axioms.*

Let us notice that we distinguish between proof trees and proofs, proof trees being only parts of proofs. Therefore, a formula is a proof tree while it is not a proof.

[4] This is a version where structural rules are implicit.

[5] Classically, F is a subset of the set A^* of all finite words on the alphabet A. In this paper, this condition will never be used. Therefore, it will not be considered.

Definition 2.3 (Theorem). *A* theorem *in $\mathcal{S}$ is a formula φ such that there exists a proof in $\mathcal{S}$ whose conclusion is φ. The existence of such a proof is usually denoted by $\mathcal{S} \vdash \varphi$.*

Notations.

- Given a proof tree π, let $\mathcal{LM}(\pi)$ (resp. $\mathcal{LS}(\pi)$) denote the multiset[6] (resp. the set[7]) of leaves of π.[8]
- We will use the notation $\{\{a_1, a_2, \ldots, a_n\}\}$ to denote a finite multiset.
- A proof tree π of conclusion φ is denoted by $\pi : \varphi$.
- We write $\pi = (\pi_1, \ldots, \pi_n, \varphi)_\iota$, with $n \in \mathbb{N}$, the proof tree whose last instance of an inference rule is $\iota = \dfrac{\varphi_1, \ldots, \varphi_n}{\varphi} r$ with $r \in R$ and such that, for every i, $1 \leq i \leq n$, π_i is the subtree of π of conclusion φ_i.

Using a standard numbering of tree nodes by natural number strings, we can refer to positions in a proof tree. Thus, given a proof tree π, a *position* of π is a string $\omega \in \mathbb{N}^*$ which represents the path from the root of π to the subtree whose conclusion occurs at that position. This subtree is denoted by $\pi_{|_\omega}$. If π' is a proof tree, $\pi[\pi']_\omega$ is the proof tree obtained from π by replacing the subtree $\pi_{|_\omega}$ by π'. The trees $\pi_{|_\omega}$ and π' necessarily have the same root. If π and $\pi' : \varphi$ are two proof trees and ω is a leaf position of π such that $\pi_{|_\omega} = \varphi$, then we use the expression $\pi \cdot_\omega \pi'$ rather than $\pi[\pi']_\omega$. This operation is called *composition* of π and π' on leaf position ω.

3. Proof Transformation Procedure

In all logical calculi where proof search strategies have been applied, the completeness of restricted derivability with respect to the full one is obtained by defining normalizing proof transformation procedures based on elementary proof tree transformations. When we study most of these procedures, we can observe that they consist in replacing in proofs some basic patterns, that we will call *basic proof trees*, of the form $(\iota_1, \ldots, \iota_n, \varphi)_\iota$ where each ι_i $(1 \leq i \leq n)$ is either an instance of a rule in R or a formula in F, with proof trees in normal form (i.e. trees that are not reducible by other proof tree transformations). These basic patterns describe critical situations that do not respect the strategy because some of the rule instances ι_i $(1 \leq i \leq n)$ should not occur over ι. For instance, in the strategy underlying the logicality result for the equational logic (see Section 5), the substitution rule must always be applied before the transitivity rule. We then have the following transformation rule, which allows to transform any proof tree which would not

[6] In $\mathcal{LM}(\pi)$, all leaves of π are considered. We then have a multiset because several occurrences of a same formula can occur in π.

[7] Here, when a formula φ of π has several occurrences, only one is considered in $\mathcal{LS}(\pi)$.

[8] Let us notice that when a proof tree π is a formula φ, then $\mathcal{LM}(\pi) = \mathcal{LS}(\pi) = \{\varphi\}$, while when π is an axiom $\frac{}{\varphi} r$ (and then a proof), $\mathcal{LM}(\pi) = \mathcal{LS}(\pi) = \emptyset$.

respect the strategy into one of the right form:

$$\dfrac{\dfrac{t = t'' \quad t'' = t'}{t = t'}\,\textit{Trans.}}{\sigma(t) = \sigma(t')}\,\textit{Sub.} \quad \rightsquigarrow \quad \dfrac{\dfrac{t = t''}{\sigma(t) = \sigma(t'')}\,\textit{Sub.} \quad \dfrac{t'' = t'}{\sigma(t'') = \sigma(t')}\,\textit{Sub.}}{\sigma(t) = \sigma(t')}\,\textit{Trans.}$$

As another example, we have the following transformation rule in the cut-elimination result for sequent calculi, since we want the cut rule to move up in the proof in order to eliminate it:[9]

$$\dfrac{\dfrac{\Gamma, \varphi \Rightarrow \Delta \quad \Gamma, \varphi' \Rightarrow \Delta}{\Gamma, \varphi \vee \varphi' \Rightarrow \Delta}\vee\text{Left} \quad \dfrac{\Gamma' \Rightarrow \Delta', \varphi, \varphi'}{\Gamma' \Rightarrow \Delta', \varphi \vee \varphi'}\vee\text{Right}}{\Gamma, \Gamma' \Rightarrow \Delta, \Delta'}\text{Cut} \quad \rightsquigarrow$$

$$\dfrac{\dfrac{\Gamma, \varphi \Rightarrow \Delta \quad \Gamma' \Rightarrow \Delta', \varphi, \varphi'}{\Gamma, \Gamma' \Rightarrow \Delta, \Delta', \varphi'}\text{Cut} \quad \Gamma, \varphi' \Rightarrow \Delta}{\Gamma, \Gamma' \Rightarrow \Delta, \Delta'}\text{Cut}$$

And, in the cut-elimination result for natural deduction, we have for instance the following transformation rule:

$$\dfrac{\dfrac{\Gamma \vdash \varphi \qquad \Gamma \vdash \psi}{\Gamma \vdash \varphi \wedge \psi}\wedge\textbf{-intro}}{\Gamma \vdash \varphi}\wedge\textbf{-elim} \quad \rightsquigarrow \quad \Gamma \vdash \varphi$$

Definition 3.1 (Basic Proof Tree). *Let $\mathcal{S} = (F, R)$ be a deductive system. A* basic proof tree *is a proof tree of the form $(\iota_1, \ldots \iota_n, \varphi)_\iota$ such that for every i, $1 \leq i \leq n$, either $\iota_i \in F$ or there exists $r \in R$ with $\iota_i \in r$.*

Note that by definition of a proof tree, ι is an instance of a rule in R. A basic proof tree is then either only a rule instance, or a rule instance whose some premisses are themselves conclusions of rule instances:

$$\iota = \dfrac{\varphi_1 \ldots \varphi_n}{\varphi} \qquad \text{or} \qquad \iota = \dfrac{\ldots \dfrac{\psi_1 \ldots \psi_{m_i}}{\varphi_i} \ldots \varphi_j \ldots}{\varphi}$$

Definition 3.2 (Transformation Procedure). *Let $\mathcal{S} = (F, R)$ be a deductive system. A* (proof tree) transformation rule for $\mathcal{S}$ *is a couple $\pi \rightsquigarrow \pi'$ of proof trees such that π is a basic proof tree, π and π' have the same root, and $\mathcal{LS}(\pi') \subseteq \mathcal{LS}(\pi)$.*

A (proof tree) transformation procedure $\rightsquigarrow$ *is a set of transformation rules such that for each proof tree π' occurring on the right-hand side of a rule, no transformation rule can be applied to it.*

The transformation of a proof tree consists in applying the rules of the transformation procedure to parts of this tree that match the basic proof trees at the left-hand side of the rules. Formally, this gives rise to the following definition:

[9] See Section 5 and [35] for a complete presentation of this result.

Definition 3.3 (Reductions). *Let $\rightsquigarrow$ be a (proof tree) transformation procedure for a deductive system $\mathcal{S}$. A proof π can be* reduced to *a proof π', denoted by $\pi \rightsquigarrow_{\mathcal{S}} \pi'$, if and only if there exists:*

- *a rule $\pi_1 \rightsquigarrow \pi_2$,*
- *a position ω in π, and*
- *n proofs $\pi'_i : \varphi_i$ such that $\pi_{|_\omega} = (\ldots(\pi_1 \cdot_{\omega_1} \pi'_1) \cdot_{\omega_2} \ldots) \cdot_{\omega_n} \pi'_n$ where for $\mathcal{LM}(\pi_1) = \{\{\varphi_1, \ldots, \varphi_n\}\}$, $\omega_1, \ldots, \omega_n$ are positions of leaf occurrences of $\varphi_1, \ldots, \varphi_n$ in π_1,*

such that $\pi' = \pi[(\ldots(\pi_2 \cdot_{\omega'_{i_1}} \pi'_{i_1}) \cdot_{\omega'_{i_2}} \ldots) \cdot_{\omega'_{i_k}} \pi'_{i_k}]_\omega$ where $\mathcal{LM}(\pi_2) = \{\{\varphi_{i_1}, \ldots, \varphi_{i_k}\}\}$, $1 \leq i_1 < \ldots < i_k \leq n$ and $\omega'_{i_1}, \ldots, \omega'_{i_k}$ are positions of leaf occurrences of $\varphi_{i_1}, \ldots, \varphi_{i_k}$ in π_2.

A schematic example of a transformation rule is the following:

$$\dfrac{\dfrac{\dfrac{\varphi_1 \quad \varphi_2}{\psi}\, r \qquad \varphi_3}{\varphi}\, s}{\gamma} \quad \rightsquigarrow \quad \dfrac{\varphi_1 \qquad \dfrac{\varphi_2 \quad \varphi_3}{\chi}\, s}{\varphi}\, r}{\gamma}$$

It can be applied to any proof tree containing the pattern given by the left-hand side of the rule, for instance:[10]

$$\dfrac{\dfrac{\dfrac{\pi_1 : \varphi_1 \quad \pi_2 : \varphi_2}{\psi}\, r \qquad \pi_3 : \varphi_3}{\varphi}\, s \qquad \pi_4}{\gamma}\, t \quad \rightsquigarrow \quad \dfrac{\pi_1 : \varphi_1 \qquad \dfrac{\pi_2 : \varphi_2 \quad \pi_3 : \varphi_3}{\chi}\, s}{\varphi}\, r \qquad \pi_4}{\gamma}\, t$$

We can easily show that a transformation procedure can be considered as a term rewrite system, thus allowing us to use the standard rewriting techniques to reason about proof tree transformation procedures (see the proof of Theorem 4.1). As a matter of fact, with every deductive system $\mathcal{S} = (F, R)$, we can associate the following multi-sorted signature $\Sigma_{\mathcal{S}} = (S, Op)$ where:

- $S = \{s_\varphi \mid \varphi \in F\}$
- $Op = \{f_\iota : s_{\varphi_1} \times \ldots \times s_{\varphi_{n-1}} \to s_{\varphi_n} \mid \iota = \frac{\varphi_1 \ldots \varphi_{n-1}}{\varphi_n} r \in \mathcal{I}\}$

Over the signature $\Sigma_{\mathcal{S}}$, proofs then are ground terms and proof trees are terms with variables in an S-indexed set V. We inductively define on the structure of proof trees a mapping $t : \mathit{ProofTree}_{\mathcal{S}} \to T_{\Sigma_{\mathcal{S}}}(V)$ where $T_\Sigma(V)$ is the S-indexed set of terms with variables over Σ and $\mathit{ProofTree}_{\mathcal{S}}$ is the whole set of proof trees over $\mathcal{S}$, as follows:

- if π is an axiom $\dfrac{}{\varphi} r$, then $t(\pi) = f_{\frac{}{\varphi} r}$,
- if π is a formula φ, then $t(\pi) = x_\varphi$ where $x_\varphi \in V_{s_\varphi}$, and
- if $\pi = (\pi_1, \ldots, \pi_n, \varphi)_\iota$, then $t(\pi) = f_\iota(t(\pi_1), \ldots, t(\pi_n))$.

[10] We recall that $\pi : \varphi$ denotes a proof tree π of conclusion φ.

Hence, each proof tree transformation rule $\pi \rightsquigarrow \pi'$ can be transformed into a term rewrite rule $t(\pi) \rightsquigarrow t(\pi')$, and Definition 3.3 is similar to the definition of rewriting steps via the transfomation along t. Therefore, by using the standard terminology available in rewriting, we say that a proof tree to which no transformation rule can be applied is in *normal form*. The global transformation $\rightsquigarrow_{\mathcal{S}}$ is *strongly normalizing* (or terminating) if every reduction sequence is finite, and *weakly normalizing* if every proof has a normal form.

Normalization of the global transformation procedure obtained by repeated application of transformation rules cannot be ensured without supplementary conditions. Moreover, there are some well-known examples of transformation procedures that meet all the requirements of Definition 3.2 and are strongly normalizing while some others are weakly normalizing (see the next section). In order to take into account the larger family of transformation procedures, in the next section, we will study conditions on basic transformation rules which are easy to check, yet powerful enough to ensure normalization.

4. Abstract Normalization Theorems

This section is devoted to (strong and weak) proof trees normalization theorems. Their proofs are abstract in the sense that they proceed on the structure of proof trees of any deductive system.

A basic and powerful idea underlying most of termination methods such as recursive path ordering (RPO) consists in comparing terms of rewrite rules by first comparing their root symbols, and recursively comparing the collections of their immediate subterms [19].

Therefore, start by assuming a well-founded ordering on atomic elements of proof trees (i.e. rule instances). Hence, given a transformation procedure $\rightsquigarrow$, we assume that a well-founded ordering $\preceq$ on $\mathcal{I}$ is supplied. This order is often simple to define for most pairs of rule instances. Indeed, it is obvious to impose that every instance ι of some rule that must not occur under some instances ι' of an other rule in proof trees has a greater weight for this order than ι'. Sometimes, this order is defined in a more ad-hoc way for the instances of a same inference rule. For instance, in the proof of the termination for the cut-elimination result, this order is defined as follows:

$$\forall @ \in \{\vee, \neg, \exists\},\ \text{Cut} \ \succ\ @\text{Right} \sim @\text{Left} \sim \text{Axiom}$$

and

$$\frac{\Gamma_1 \Rightarrow \Delta_1, \varphi_1 \quad \Gamma'_1, \varphi_1 \Rightarrow \Delta'_1}{\Gamma_1, \Gamma'_1 \Rightarrow \Delta_1, \Delta'_1} \succ \frac{\Gamma_2 \Rightarrow \Delta_2, \varphi_2 \quad \Gamma'_2, \varphi_2 \Rightarrow \Delta'_2}{\Gamma_2, \Gamma'_2 \Rightarrow \Delta_2, \Delta'_2} \quad \Leftrightarrow \quad |\varphi_2| < |\varphi_1|$$

where $|\varphi|$ is the depth of a formula defined to be $sup_k(1 + |\varphi_k|)$ if the φ_i are the direct sub-formulas of φ.

Given a well-founded ordering $\preceq$ on rule instances, we denote by $\prec$ its strict part and by $\sim$ its symmetric closure: $\iota \sim \iota' \Leftrightarrow \iota \preceq \iota' \wedge \iota \succeq \iota'$.

In most examples of strong normalization results, transformation procedures $\rightsquigarrow$ consist more or less in "distributing" some rules over others in order to respect the proof search strategy (see for instance the above case of the rule *Sub.* over the rule *Trans.*). Condition 1 generalizes this notion of distributivity of some rules over others.

Condition 1. *Let* $(\iota_1, \ldots, \iota_n, \varphi)_\iota \rightsquigarrow \pi$ *be a transformation rule. For each* $(\pi'_1, \ldots, \pi'_m, \varphi')_{\iota'}$ *subtree of* π*, the two following conditions must be satisfied:*

1. $\iota \succeq \iota'$ *if each* $\pi'_i \in F \cup \mathcal{I}$
   $\iota \succ \iota'$ *otherwise*
2. *if* $\iota \sim \iota'$*, then* $\{\{\iota_1, \ldots, \iota_n\}\} \succcurlyeq \{\{\iota'_1, \ldots, \iota'_m\}\}$ *where* ι'_i *is the last rule instance in* π'_i *and where* $\succcurlyeq$ *extends* $\succ$ *to multisets on* $F \cup \mathcal{I}$*.*

Notice that Condition 1 is a particular case of RPO, and then we have the following obvious result:

Theorem 4.1. *If every transformation rule in a transformation procedure* $\rightsquigarrow$ *satisfies Condition* 1*, then* $\rightsquigarrow_S$ *is strongly normalizing.*

Proof. By using our transformation mapping t defined in the previous section, with a recursive path ordering $\succ^{rpo}$ to order proofs induced by the well-founded relation (precedence) $\succeq$ on rule instances, we show that $t(\rightsquigarrow) \subseteq \succ^{rpo}$. Let $(\iota_1, \ldots, \iota_n, \varphi)_\iota \rightsquigarrow \pi$ be a transformation rule. By mathematical induction on π let us show that $t((\iota_1, \ldots, \iota_n, \varphi)_\iota) \succ^{rpo} t(\pi)$.

Basic case: Here, two cases have to be considered :

1. π is the formula φ. By Definition 3.2, we have $\mathcal{LS}(\pi) \subseteq \mathcal{LS}((\iota_1, \ldots, \iota_n, \varphi)_\iota)$. Therefore, because RPO has the subterm property, we conclude $t((\iota_1, \ldots, \iota_n, \varphi)_\iota) \succ x_\varphi$.
2. π is an axiom $\dfrac{}{\varphi} r$. By Condition 1.1, we have $\iota \succ \dfrac{}{\varphi} r$, and then $t((\iota_1, \ldots, \iota_n, \varphi)_\iota) \succ f_{\frac{}{\varphi} r}$

General case: π is of the form $(\pi_1, \ldots, \pi_n, \varphi)_{\iota'}$. Here two cases have to be considered:

1. $\iota \succ \iota'$. By the induction hypothesis, for every i, $1 \leq i \leq n$, $t((\iota_1, \ldots, \iota_n, \varphi)_\iota) \succ^{rpo} t(\pi_i)$, and then by RPO, $t((\iota_1, \ldots, \iota_n, \varphi)_\iota) \succ^{rpo} t(\pi)$.
2. $\iota \sim \iota'$. By Condition 1.1, each π_i is in $F \cup \mathcal{I}$ $(1 \leq i \leq n)$. By Condition 1.2, we have $\{\{\iota_1, \ldots, \iota_n\}\} \succcurlyeq \{\{\iota'_1, \ldots, \iota'_m\}\}$, and then $\{\{t(\iota_1), \ldots, t(\iota_n)\}\} \succcurlyeq^{rpo} \{\{t(\iota'_1), \ldots, t(\iota'_m)\}\}$. By RPO, we then conclude $t((\iota_1, \ldots, \iota_n, \varphi)_\iota) \succ^{rpo} t(\pi)$.

□

However, problems may occur with transformation rules of the form

$$(\iota_1, \ldots, \iota_n, \varphi)_\iota \rightsquigarrow \pi$$

such that there exists a subtree $(\iota'_1, \ldots, \iota'_m, \varphi')_{\iota'}$ of π with $\iota'_j \in F \cup \mathcal{I}$ for every j, $1 \leq j \leq m$, satisfying:

- $\iota \sim \iota'$, and
- $\mathcal{LM}(\iota'_j) = \mathcal{LM}((\iota_1, \ldots, \iota_n, \varphi)_\iota)$

Indeed, such transformation rules may prevent from using the standard techniques to prove termination such as RPO, and then from obtaining a strong normalization result.

An example of such a transformation rule is Rule **R1** defined in the cut-elimination procedure for the sequent calculus **LK** that considers the following cut rule:

$$\frac{\Gamma \Rightarrow \Delta, \varphi \quad \Gamma, \varphi \Rightarrow \Delta}{\Gamma \Rightarrow \Delta}\text{Cut}$$

Indeed, with such a cut rule, we need weakening as explicit rule in order to allow such basic transformations:

R1
$$\frac{\dfrac{\Gamma \Rightarrow \Delta, \varphi}{\Gamma' \Rightarrow \Delta', \varphi}\text{@Right} \quad \Gamma', \varphi \Rightarrow \Delta'}{\Gamma' \Rightarrow \Delta'}\text{Cut} \quad \rightsquigarrow$$

$$\frac{\dfrac{\dfrac{\Gamma \Rightarrow \Delta, \varphi}{\Gamma, \Gamma' \Rightarrow \Delta, \Delta', \varphi}\text{Weak.} \quad \dfrac{\Gamma', \varphi \Rightarrow \Delta'}{\Gamma, \Gamma', \varphi \Rightarrow \Delta, \Delta'}\text{Weak.}}{\Gamma, \Gamma' \Rightarrow \Delta, \Delta'}\text{Cut}}{\Gamma' \Rightarrow \Delta'}\text{@Right}$$

Here, the instance of Cut in π is equal to the instance ι with respect to $\preceq$, and its leaves are exactly the same as those of ι. The only measure that decreases with such a rule is the number of occurrences of rules @Left and @Right that occur above Cut in the right-hand side of the rule. Therefore by applying a strategy that, for instance, first eliminates in proof trees the critical situations (here an instance of @Left or @Right above an instance of Cut) whose instance of the cut rule is the heaviest (see above how is defined the weight of a cut rule), this kind of measure will be sufficient to obtain a result of weak normalization.

For a given transformation procedure, we define the set *Elim* of rule instances that have to move up in proof trees or to be eliminated in order to respect the underlying strategy. These rule instances are those occurring at the left-hand side of a transformation rule and correspond to critical situations.

Definition 4.2. *Let $\rightsquigarrow$ be a transformation procedure. Let us define the set Elim as follows:*

$$\mathit{Elim} = \{\iota \mid \iota \in \mathcal{I}, \exists \text{ a transformation rule } (\iota_1, \ldots, \iota_n, \varphi)_\iota \rightsquigarrow \pi\}$$

For the cut elimination procedure for instance, *Elim* contains all the rule instances of Cut, Weak, and Sub.

Now, we resume the above discussion by the following condition expressed in our abstract framework. Before giving this condition, let us define the notion of proof tree length.

Definition 4.3 (Proof tree length). *The length of a proof tree* $\pi = (\pi_1, \ldots, \pi_n, \varphi)_\iota$, *denoted by* $|\pi|$, *is inductively defined as follows:*

$$|\pi| = \begin{cases} \sum_i |\pi_i| & \text{if } \iota \in \mathit{Elim} \\ \sum_i |\pi_i| + 1 & \text{otherwise} \end{cases}$$

Hence, the length of a proof π is the number of rule instances that do not belong to *Elim*.

Condition 2. *Let* $(\iota_1, \ldots, \iota_n, \varphi)_\iota \rightsquigarrow \pi$ *be a transformation rule. The four following conditions must be satisfied:*

1. *there exists* i, $1 \leq i \leq n$, *such that* $\iota_i \in \mathcal{I} \setminus \mathit{Elim}$
2. $\mathcal{LM}(\pi) \subseteq \mathcal{LM}((\iota_1, \ldots, \iota_n, \varphi)_\iota)$
3. *for every rule instance* ι' *occurring in* π, $\iota \succeq \iota'$
4. *for every subproof* $\pi' = (\pi'_1, \ldots, \pi'_k, \varphi')_{\iota'}$ *of* π *with* $\iota' \in \mathit{Elim}$, $|\pi'| < |(\iota_1, \ldots, \iota_n, \varphi)_\iota|$.

Condition 2 being purely syntactical, it can be automatically checked, given a transformation procedure and a well-founded ordering on rule instances.

Now we arrive at the main theorem of this paper.

Theorem 4.4. *If every transformation rule of a transformation procedure* $\rightsquigarrow_S$ *satisfies Condition* 2, *then* $\rightsquigarrow_S$ *is weakly normalizing.*

Proof. The theorem is obtained by generalizations of Tait's proof [53].

It is well-known that all well-founded sets are isomorphic to a unique ordinal. Note $d : (\mathcal{I}, \preceq) \to \alpha$, where α is an ordinal, this isomorphism.

Let $\pi = (\pi_1, \ldots, \pi_n, \varphi)_\iota$ be a proof. Then, we define the *degree* of π, denoted by $\partial(\pi)$, to be $d(\iota) + |\pi|$.

Let $\pi = (\pi_1, \ldots, \pi_n, \varphi)_\iota$ be a proof. Then, we define the *measure* of π, denoted by $m(\pi)$, to be the multiset:

$$\{\{\partial(\pi') \mid \pi' : \text{subproof of } \pi \text{ whose last inference rule belongs to } \mathit{Elim}\}\}$$

To prove the theorem, we start by proving the two following lemmas.

Lemma 4.5. *Let* $(\iota_1, \ldots, \iota_n, \varphi)_\iota \rightsquigarrow \pi$ *be a transformation rule. Let* $\pi_1, \ldots, \pi_n$ *be* n *proofs such that for every* i, $1 \leq i \leq n$, $\pi_i : \iota_i$ *if* $\iota_i \in F$ *or* $\pi_i = (\pi_{i_1}, \ldots, \pi_{i_{m_i}}, \varphi_i)_{\iota_i}$ *otherwise (i.e.* $\exists r \in R$, $\iota_i \in r$*). Let* $\overline{\pi}'$ *be the proof obtained by applying the transformation rule to* $\pi' = (\pi_1, \ldots, \pi_n, \varphi)_\iota$. *If for every* i, $1 \leq i \leq n$, $\partial(\pi_i) < \partial(\pi')$, *then for every subproof* π'' *of* $\overline{\pi}'$ *whose last inference rule belongs to Elim,* $\partial(\pi'') < \partial(\pi')$.

Proof. By Conditions 2.2 and 2.4, we know that no rule instance of $\mathcal{I} \setminus \mathit{Elim}$ has been introduced in π'', and by Condition 2.1, that one of them has been removed. Therefore, we have $|\pi''| < |\pi'|$. Now, by Condition 2.3 and the fact that for every i, $1 \leq i \leq n$, $\partial(\pi_i) < \partial(\pi')$, we can conclude that $\partial(\pi'') < \partial(\pi')$. □

Lemma 4.6. *For every proof $\pi : \varphi$ which is not in normal form, there exists a proof $\overline{\pi} : \varphi$ such that $m(\overline{\pi}) \ll m(\pi)$.*[11]

Proof. As π is not in normal form, there exists a subproof π' of π that can be transformed by applying a transformation rule. Let us call such a subproof a *reducible proof*. Let π' be such a reducible proof with the greatest degree. By Lemma 4.5, we know that for every subproof π'' of $\overline{\pi}'$ (the proof obtained after applying the transformation to π') whose last inference rule belongs to *Elim*, we have $\partial(\pi'') < \partial(\pi')$. Among these subproofs, there are all the subproofs $\pi''_1, \ldots, \pi''_k$ that have been introduced by the right-hand side of the transformation rule. Therefore, the measure of $\overline{\pi}$ (the proof obtained from π by replacing π' with $\overline{\pi}'$) can be obtained by replacing the degree of π' with the degrees of $\pi''_1, \ldots, \pi''_k$ which are all less than $\partial(\pi')$, and then $m(\overline{\pi}) \ll m(\pi)$. □

The proof of Theorem 4.4 is then obtained by iterating Lemma 4.6. □

5. Applications

In this section, in order to illustrate our approach with some concrete examples, we apply our general method to classical results, which are, respectively, the logicality result in equational logic, Newman's lemma in rewriting theory and Gentzen cut-elimination result for a sequent calculus for classical first-order logic where structural rules are implicit (which is a quite standard formulation of the calculus). For each of them, we prove normalization results, respectively strong for the two first and both strong and weak for the last one. In the literature, each of them have been extended to many other logic calculi but, each time the underlying (meta)proof methodology is pretty much the same and it can be seen as a particular instance of our general abstract approach.

5.1. Logicality

Before stating the logicality result, let us first give the deductive system for equational logic. Let Σ be a signature, that is a set of function names, each one equipped with an arity in $\mathbb{N}$. Let V be a set of variables. We denote by $T_\Sigma(V)$ the set of terms with variables in V. Let Γ be a set of equations, that are sentences of the form $t = t'$ with $t, t' \in T_\Sigma(V)$. Therefore, the deductive system associated to Σ and Γ is given by the pair $\mathcal{S} = (F, R)$ where F is the set of all equations of the form $t = t'$ with $t, t' \in T_\Sigma(V)$ and R is the set of rule instances generated by the standard rule schemes recalled below.

[11] $\ll$ is the extension to multisets of the order on proof degrees. As the order on proof degrees is well-founded, then it is known that so is $\ll$.

Let C be any term of $T_\Sigma(V \cup \{\Box\})$ with a unique occurrence of $\Box$.[12] Let $C[t]$ be the result of replacing in C the occurrence of $\Box$ by $t \in T_\Sigma(V)$, and let $\sigma : V \to T_\Sigma(V)$ be a substitution. The considered rule schemes are:

$$\mathit{Axiom}\frac{t = t' \in \Gamma}{t = t'} \qquad \mathit{Ref.}\frac{}{t = t} \qquad \mathit{Sym.}\frac{t = t'}{t' = t}$$

$$\mathit{Cont.}\frac{t = t'}{C[t] = C[t']} \qquad \mathit{Sub.}\frac{t = t'}{\sigma(t) = \sigma(t')} \qquad \mathit{Trans.}\frac{t = t' \quad t' = t''}{t = t''}$$

$\mathcal{S} \vdash t = t'$ is usually written $\Gamma \vdash t = t'$.

A classical result due to G. Birkhoff [11] ensures that equational reasoning (given by derivability according to the rules above) coincides with convertibility in rewriting for equational logic. This property, named *logicality* [60], is expressed by:

$$\Gamma \vdash t = t' \iff t \overset{*}{\leftrightarrow}_\Gamma t' \qquad (\Gamma\text{: set of equations})$$

where $\leftrightarrow_\Gamma$ is the convertibility relation defined as the closure of equations of Γ under symmetry (*Sym.*), substitution (*Sub.*) and context (*Cont.*), and $\overset{*}{\leftrightarrow}_\Gamma$ is the closure of $\leftrightarrow_\Gamma$ under reflexivity (*Ref.*) and transitivity (*Trans.*). Hence, $\overset{*}{\leftrightarrow}_\Gamma$ actually defines proof search (building) strategies which select proof trees π composed of instances in R such that no instance of *Trans.* occurs over instances of *Sym.*, *Sub.* and *Cont.*

Completeness of such strategies can be shown by using the following basic proof tree transformations:

$$\mathit{Sub.}\frac{\mathit{Trans.}\dfrac{t = t'' \quad t'' = t'}{t = t'}}{\sigma(t) = \sigma(t')} \qquad \rightsquigarrow \qquad \mathit{Trans.}\frac{\mathit{Sub.}\dfrac{t = t''}{\sigma(t) = \sigma(t'')} \quad \mathit{Sub.}\dfrac{t'' = t'}{\sigma(t'') = \sigma(t')}}{\sigma(t) = \sigma(t')}$$

[12] Intuitively, the special variable $\Box$ represents a "hole", so that C can be seen as a *context*.

The cases of *Sym.* and *Cont.* correspond to a similar transformation, that is a distribution of *Sym.* and *Cont.* over *Trans.*

$$Sub.\dfrac{Ref.\dfrac{}{t=t}}{\sigma(t)=\sigma(t)} \quad \rightsquigarrow \quad Ref.\dfrac{}{\sigma(t)=\sigma(t)}$$

$$Trans.\dfrac{Ref.\dfrac{}{t=t} \quad \dfrac{\ldots}{t=t'}}{t=t'} \quad \rightsquigarrow \quad \dfrac{\ldots}{t=t'}$$

$$Sym.\dfrac{Ref.\dfrac{}{t=t}}{t=t} \quad \rightsquigarrow \quad Ref.\dfrac{}{t=t}$$

$$Trans.\dfrac{\dfrac{\ldots}{t=t'} \quad Ref.\dfrac{}{t'=t'}}{t=t'} \quad \rightsquigarrow \quad \dfrac{\ldots}{t=t'}$$

$$Cont.\dfrac{Ref.\dfrac{}{t=t}}{C(t)=C(t)} \quad \rightsquigarrow \quad Ref.\dfrac{}{C(t)=C(t)}$$

Let $\succ$ be the well-founded ordering defined by:

$$Sym. \sim Sub. \sim Cont. \succ Trans. \succ Ref. \sim Axiom$$

As we have already observed, the transformation rules consist in distributing instances of *Sym.*, *Sub.* and *Cont.* over instances of *Trans.*, and erasing all rule instances occurring under *Ref.*'s ones. Therefore, all transformation rules are:

- either of the form
$$\iota\dfrac{Trans.}{\varphi} \rightsquigarrow Trans.\dfrac{\iota_1 \ \iota_2}{\varphi}$$
with $\iota, \iota_1, \iota_2 \in r$ and $r \in \{Sub., Cont., Sym.\}$
- or of the form
$$\iota\dfrac{Ref.}{\varphi} \rightsquigarrow Ref.\dfrac{}{\varphi}$$

In the first case, $\iota \succ Trans.$, $\iota \sim \iota_j$ and $\mathcal{LS}(\iota_j) \subseteq \mathcal{LS}(Trans.)$ for $j = 1, 2$, while in the second case, $\iota \succ Ref.$ Therefore, all transformation rules satisfy Condition 1, and then by Theorem 4.1 the logicality procedure is strongly normalizing.

5.2. Newman's Lemma

It is well-known that from any locally confluent and terminating rewrite system $\mathcal{R}$, all proofs written as series of peaks can be transformed into valleys by removing and replacing, step by step, local peaks by equivalent valleys. Although there exists a more direct proof of Newman's lemma, to show that our results can be applied to a large family of normalization results, we are going to formalize the process that underlies Newman's lemma by a terminating transformation procedure that meets all the requirements given in Section 4. First, let us introduce the underlying deductive system. Let $\mathcal{R}$ be a rewrite system, that is a binary relation

$\rightarrow \subseteq T_\Sigma(V) \times T_\Sigma(V)$, which is terminating and locally confluent. The associated deductive system $\mathcal{S} = (F, R)$ is then defined as follows:[13]

- F is the set of all sentences of the form $u \rightarrow_\mathcal{R} v$, $u \overset{*}{\rightarrow}_\mathcal{R} v$ and $u \overset{*}{\leftrightarrow}_\mathcal{R} v$
- R is the set of inference rules

$$Axiom\frac{u \rightarrow v \in \mathcal{R}}{u \rightarrow_\mathcal{R} v} \qquad Rew.\frac{u \rightarrow_\mathcal{R} v}{u \overset{*}{\rightarrow}_\mathcal{R} v} \qquad Der/Pr.\frac{u \overset{*}{\rightarrow}_\mathcal{R} v}{u \overset{*}{\leftrightarrow}_\mathcal{R} v}$$

$$Cont.\frac{u \overset{*}{\rightarrow}_\mathcal{R} v}{C[u] \overset{*}{\rightarrow}_\mathcal{R} C[v]} \qquad Sub.\frac{u \overset{*}{\rightarrow}_\mathcal{R} v}{\sigma(u) \overset{*}{\rightarrow}_\mathcal{R} \sigma(v)}$$

$$Peak\frac{u \ _\mathcal{R}\overset{*}{\leftarrow} w \overset{*}{\rightarrow}_\mathcal{R} v}{u \overset{*}{\leftrightarrow}_\mathcal{R} v} \qquad Valley\frac{u \overset{*}{\rightarrow}_\mathcal{R} w \ _\mathcal{R}\overset{*}{\leftarrow} v}{u \overset{*}{\leftrightarrow}_\mathcal{R} v}$$

$$Proof\frac{u \overset{*}{\leftrightarrow}_\mathcal{R} w \overset{*}{\leftrightarrow}_\mathcal{R} v}{u \overset{*}{\leftrightarrow}_\mathcal{R} v} \qquad Deriv.\frac{u \overset{*}{\rightarrow}_\mathcal{R} w \overset{*}{\rightarrow}_\mathcal{R} v}{u \overset{*}{\rightarrow}_\mathcal{R} v}$$

Basic transformation rules then are the following:

$$Peak\frac{Deriv.\dfrac{u \ _\mathcal{R}\overset{*}{\leftarrow} u' \ _\mathcal{R}\overset{*}{\leftarrow} w}{u \ _\mathcal{R}\overset{*}{\leftarrow} w} \qquad w \overset{*}{\rightarrow}_\mathcal{R} v}{u \overset{*}{\leftrightarrow}_\mathcal{R} v} \quad \rightsquigarrow$$

$$Proof\frac{Der/Pr\dfrac{u \ _\mathcal{R}\overset{*}{\leftarrow} u'}{u \overset{*}{\leftrightarrow}_\mathcal{R} u'} \quad Peak\dfrac{u' \ _\mathcal{R}\overset{*}{\leftarrow} w \overset{*}{\rightarrow}_\mathcal{R} v}{u' \overset{*}{\leftrightarrow}_\mathcal{R} v}}{u \overset{*}{\leftrightarrow}_\mathcal{R} v}$$

$$Peak\frac{u \ _\mathcal{R}\overset{*}{\leftarrow} w \qquad Deriv.\dfrac{w \overset{*}{\rightarrow}_\mathcal{R} v' \overset{*}{\rightarrow}_\mathcal{R} v}{w \overset{*}{\rightarrow}_\mathcal{R} v}}{u \overset{*}{\leftrightarrow}_\mathcal{R} v} \quad \rightsquigarrow$$

$$Proof\frac{Peak\dfrac{u \ _\mathcal{R}\overset{*}{\leftarrow} w \overset{*}{\rightarrow}_\mathcal{R} v'}{u \overset{*}{\leftrightarrow}_\mathcal{R} v'} \quad Der/Pr\dfrac{v' \overset{*}{\rightarrow}_\mathcal{R} v}{v' \overset{*}{\leftrightarrow}_\mathcal{R} v}}{u \overset{*}{\leftrightarrow}_\mathcal{R} v}$$

$$Peak\frac{Rew.\dfrac{u \ _\mathcal{R} \leftarrow w \rightarrow_\mathcal{R} v}{u \ _\mathcal{R}\overset{*}{\leftarrow} w \ _\mathcal{R}\overset{*}{\rightarrow} v}}{u \overset{*}{\leftrightarrow}_\mathcal{R} v} \quad \rightsquigarrow \quad Valley\frac{u \overset{*}{\rightarrow}_\mathcal{R} x \ _\mathcal{R}\overset{*}{\leftarrow} v}{u \overset{*}{\leftrightarrow}_\mathcal{R} v}$$[14]

[13] We suppose that the rewrite system $\mathcal{R}$ does not contain trivial rules of the form $u \rightarrow u$.

[14] Such x exists and can be computed because $\mathcal{R}$ is locally confluent. Moreover, it can be computed because $\mathcal{R}$ is terminating.

$$\mathit{Proof}\dfrac{\mathit{Valley}\dfrac{u \overset{*}{\to}_{\mathcal{R}} x \;\; \mathcal{R} \;\; x \overset{*}{\leftarrow} w}{u \overset{*}{\leftrightarrow}_{\mathcal{R}} w} \quad \mathit{Der/Pr}\dfrac{w \overset{*}{\to}_{\mathcal{R}} v}{w \overset{*}{\leftrightarrow}_{\mathcal{R}} v}}{u \overset{*}{\leftrightarrow}_{\mathcal{R}} v} \quad \rightsquigarrow$$

$$\mathit{Proof}\dfrac{\mathit{Der/Pr}\dfrac{u \overset{*}{\to}_{\mathcal{R}} x}{u \overset{*}{\leftrightarrow}_{\mathcal{R}} x} \quad \mathit{Der/Pr}\dfrac{x \;\; \mathcal{R} \;\; x \overset{*}{\leftarrow} w \overset{*}{\to}_{\mathcal{R}} v}{x \overset{*}{\leftrightarrow}_{\mathcal{R}} v}}{u \overset{*}{\leftrightarrow}_{\mathcal{R}} v}$$

$$\mathit{Proof}\dfrac{\mathit{Der/Pr}\dfrac{u \;\; \mathcal{R} \;\; u' \overset{*}{\leftarrow} v}{u \overset{*}{\leftrightarrow}_{\mathcal{R}} u'} \quad \mathit{Valley}\dfrac{u' \overset{*}{\to}_{\mathcal{R}} x \;\; \mathcal{R} \;\; x \overset{*}{\leftarrow} v}{u' \overset{*}{\leftrightarrow}_{\mathcal{R}} v}}{u \overset{*}{\leftrightarrow}_{\mathcal{R}} v} \quad \rightsquigarrow$$

$$\mathit{Proof}\dfrac{\mathit{Peak}\dfrac{u \;\; \mathcal{R} \;\; u' \overset{*}{\leftarrow} v \overset{*}{\to}_{\mathcal{R}} x}{u \overset{*}{\leftrightarrow}_{\mathcal{R}} x} \quad \mathit{Der/Pr}\dfrac{x \overset{*}{\leftarrow}_{\mathcal{R}} v}{x \overset{*}{\leftrightarrow}_{\mathcal{R}} v}}{u \overset{*}{\leftrightarrow}_{\mathcal{R}} v}$$

The transformation procedure $\rightsquigarrow$ defined by the above basic transformations satisfies Condition 1 for the well-founded ordering $\preceq$ defined as follows. We denote by $>$ the well-founded ordering on $T_\Sigma(V)$ defined by $u > v \Leftrightarrow u \neq v \wedge u \overset{*}{\to}_{\mathcal{R}} v$.

$$\mathit{Peak},\ \mathit{Proof} \succ \mathit{Rew} \sim \mathit{Cont.} \sim \mathit{Der/Pr.} \sim \mathit{Sub.} \sim \mathit{Valley} \sim \mathit{Deriv.}$$

$$\iota\ \dfrac{u_1 \mathbin{@} u_2 \mathbin{@} u_3}{u_1 \overset{*}{\leftrightarrow}_{\mathcal{R}} u_3} \succ \iota'\ \dfrac{u'_1 \mathbin{@'} u'_2 \mathbin{@'} u'_3}{u'_1 \overset{*}{\leftrightarrow}_{\mathcal{R}} u'_3} \iff \begin{cases} (\forall i \in \{1,2,3\}, u_i \overset{*}{\to}_{\mathcal{R}} u'_i \\ \qquad \wedge\ \exists j \in \{1,2,3\}, u_j > u'_j) \\ \vee\ u'_1 \mathbin{@'} u'_2 \mathbin{@'} u'_3 \\ \qquad \text{subproof of } u_1 \mathbin{@} u_2 \mathbin{@} u_3 \end{cases}$$

with $\iota, \iota' \in \mathit{Proof} \cup \mathit{Peak}$ and $@, @' \in \{\overset{*}{\leftrightarrow}_{\mathcal{R}}, \overset{*}{\to}_{\mathcal{R}}\}$.

$$\mathit{Peak}\dfrac{u_1 \;\; \mathcal{R} \;\; w \overset{*}{\to}_{\mathcal{R}} u_2}{u \overset{*}{\leftrightarrow}_{\mathcal{R}} v} \;\succ\; u \overset{*}{\to}_{\mathcal{R}} v \iff \exists j \in \{1,2\}, u_j > u$$

All the transformation rules satisfy Condition 1, then the procedure is strongly normalizing.

5.3. Gentzen's Cut Elimination

First, let us introduce the deductive system for (a version of) the sequent calculus **LK** for classical first-order predicate logic. Here, we restrict ourselves to the logical connectives $\{\neg, \vee\}$ and the quantifier $\exists$. It is well-known that other classical connectives and quantifiers can be defined from this restricted set. First, we recall the basic notions and notations of first-order logic and sequent calculus.

In the case of first-order logic, a signature $\Sigma = (Op, P)$ contains two sets of operation names and predicate names, and each operation and predicate name is equipped with an arity in $\mathbb{N}$. Σ-atoms are formulas $p(t_1, \ldots, t_n)$ where $p \in P$ and every t_i $(1 \leq i \leq n)$ is a term in $T_\Sigma(V)$. Well-formed Σ-formulas are then either atoms or sentences of the form $\neg\varphi$, $\varphi \vee \psi$ and $\exists x.\varphi$ where φ and ψ are well-formed Σ-formulas and x is a variable in V. The notions of free and bound variable are

defined as usual, and we write $\varphi[x/t]$ for clash of variables-avoiding substitution of t for x in φ.

In the formulation of classical sequent calculus chosen here, a Σ-sequent is any pair $\Gamma \Rightarrow \Delta$ where Γ and Δ are two finite sets of well-formed Σ-formulas. In the following, we will write Γ, φ and Γ, Γ' to mean $\Gamma \cup \{\varphi\}$ and $\Gamma \cup \Gamma'$, respectively. Our choice to consider sets instead multisets (or even lists) of formulas in the definition of sequents eliminates contraction and exchange as explicit structural rules.

Given a signature $\Sigma = (Op, P)$, the deductive system $\mathcal{S} = (F, R)$ for Σ associated to Gentzen-style sequent calculi is defined as follows:

- F is the set of Σ-sequents[15]
- R contains all the instances of the following rule schemes:

$$\frac{}{\Gamma, \varphi \Rightarrow \Delta, \varphi}\text{Ax.}^{16}$$

$$\frac{\Gamma, \varphi \Rightarrow \Delta \quad \Gamma, \varphi' \Rightarrow \Delta}{\Gamma, \varphi \vee \varphi' \Rightarrow \Delta}\vee\text{Left} \qquad \frac{\Gamma \Rightarrow \Delta, \varphi, \varphi'}{\Gamma \Rightarrow \Delta, \varphi \vee \varphi'}\vee\text{Right}$$

$$\frac{\Gamma \Rightarrow \Delta, \varphi}{\Gamma, \neg\varphi \Rightarrow \Delta}\neg\text{Left} \qquad \frac{\Gamma, \varphi \Rightarrow \Delta}{\Gamma \Rightarrow \Delta, \neg\varphi}\neg\text{Right}$$

$$\frac{\Gamma, \varphi[x/y] \Rightarrow \Delta}{\Gamma, \exists x.\varphi \Rightarrow \Delta}\exists\text{Left} \qquad \frac{\Gamma \Rightarrow \Delta, \varphi[x/t]}{\Gamma \Rightarrow \Delta, \exists x.\varphi}\exists\text{Right}$$

$$\frac{\Gamma \Rightarrow \Delta, \varphi \quad \Gamma, \varphi \Rightarrow \Delta}{\Gamma \Rightarrow \Delta}\text{Cut}$$

where the $\exists$Left rule obeys the usual *eigenvariable* condition, stating that x is not free in Γ, Δ.

In the cut rule, φ is called the *cut formula*. Finally, we use the standard terminology of *principal formula* with respect to a rule r as follows:

- $\neg\varphi$ is principal with respect to $\neg$Left and $\neg$Right,
- $\varphi \vee \varphi'$ is principal with respect to $\vee$Left and $\vee$Right, and
- $\exists x.\varphi$ is principal with respect to $\exists$Left and $\exists$Right,

In [33], Gentzen showed that all instances of the cut rule can be eliminated from proof trees for a version of classical sequent calculus where structural rules are explicit. To achieve this purpose, he gave a constructive proof of the cut-elimination result, by defining an effective procedure to eliminate cut instances from a proof tree π whose conclusion is a tautology A and whose leaves are axioms, so as to obtain a cut-free proof tree π' proving A from axioms. This procedure is based on basic proof transformations which can be modeled in our abstract framework.

[15] Note that this definition is independent from the chosen structure for sequents, so that it can be generalized to other sequent calculi.

[16] It is well known that this formulation of the axioms makes the structural rule Weakening admissible in the calculus.

First, we need to add to the set R of inference rules the two following admissible rules:

$$\frac{\Gamma \Rightarrow \Delta}{\sigma(\Gamma) \Rightarrow \sigma(\Delta)}\text{Sub.}$$

where $\sigma : V \to T_\Sigma(V)$ is a substitution and $\sigma(\Gamma) = \sigma(\varphi_1), \ldots, \sigma(\varphi_n)$ when $\Gamma = \varphi_1, \ldots, \varphi_n$.

$$\frac{\Gamma \Rightarrow \Delta}{\Gamma' \Rightarrow \Delta'}\text{Weak}$$

where $\Gamma \subseteq \Gamma'$ and $\Delta \subseteq \Delta'$.

Let us call the resulting extended system of rules R'. The intuitive reason of such an extension of R to R', is that, in usual cut elimination proofs for calculi with implicit structural rules given in the literature [32, 41], these two supplementary rules are replaced by obvious intermediary lemmas, which are situated at a meta-theoretical level. Since we aim to see the cut-elimination procedure as a rewriting procedure, we do need to shift them at the object level. Note that, because of the admissibility of Sub. and Weak., there is a proof via R of a tautology A if and only if there is a proof of A via R'. Moreover, our method will be such that cut-free proofs with respect to R' are actually cut-free proofs using only rules in R.

Weak normalization result. Gentzen's result (and, in particular, the termination of the cut-elimination procedure, which is the difficult part) can be shown by using the transformation rules described below. Such a set of transformation rules can be organized in four cases:

1. Case where the cut formula is not principal with respect to at least one of the inferences leading immediately to the premises of the cut. Here, we give transformations for the case where right rules are applied so as to get the left premise of the cut rule, but one can generalize such transformations to the symmetric case in a standard way.

$$\frac{\dfrac{\Gamma \Rightarrow \Delta, \varphi}{\Gamma' \Rightarrow \Delta', \varphi}@\text{Right} \qquad \Gamma', \varphi \Rightarrow \Delta'}{\Gamma' \Rightarrow \Delta'}\text{Cut} \quad \rightsquigarrow$$

$$\frac{\dfrac{\dfrac{\Gamma \Rightarrow \Delta, \varphi}{\Gamma, \Gamma' \Rightarrow \Delta, \Delta', \varphi}\text{Weak.} \qquad \dfrac{\Gamma', \varphi \Rightarrow \Delta'}{\Gamma, \Gamma', \varphi \Rightarrow \Delta, \Delta'}\text{Weak.}}{\Gamma, \Gamma' \Rightarrow \Delta, \Delta'}\text{Cut}}{\Gamma' \Rightarrow \Delta'}@\text{Right}$$

   where $@ \in \{\vee, \neg, \exists\}$ and $\Gamma' = \Gamma$ except for the $\neg$Right rule, where $\Gamma' = \Gamma, \neg\psi$ for some formula ψ.
2. Case where the cut formula is principal with respect to both inferences leading immediately to the premises of the cut. In this case we have the following proof tree transformations:

$$\dfrac{\dfrac{\Gamma, \varphi \Rightarrow \Delta \quad \Gamma, \varphi' \Rightarrow \Delta}{\Gamma, \varphi \vee \varphi' \Rightarrow \Delta} \vee\text{Left} \qquad \dfrac{\Gamma \Rightarrow \Delta, \varphi, \varphi'}{\Gamma \Rightarrow \Delta, \varphi \vee \varphi'} \vee\text{Right}}{\Gamma \Rightarrow \Delta} \text{Cut} \quad \rightsquigarrow$$

$$\dfrac{\dfrac{\dfrac{\Gamma, \varphi \Rightarrow \Delta}{\Gamma, \varphi \Rightarrow \Delta, \varphi'} \text{Weak.} \qquad \Gamma \Rightarrow \Delta, \varphi, \varphi'}{\Gamma \Rightarrow \Delta, \varphi'} \text{Cut} \qquad \Gamma, \varphi' \Rightarrow \Delta}{\Gamma \Rightarrow \Delta} \text{Cut}$$

$$\dfrac{\dfrac{\Gamma \Rightarrow \Delta, \varphi}{\Gamma, \neg\varphi \Rightarrow \Delta} \neg\text{Left} \qquad \dfrac{\Gamma, \varphi \Rightarrow \Delta}{\Gamma \Rightarrow \Delta, \neg\varphi} \neg\text{Right}}{\Gamma \Rightarrow \Delta} \text{Cut} \quad \rightsquigarrow \quad \dfrac{\Gamma \Rightarrow \Delta, \varphi \quad \Gamma, \varphi \Rightarrow \Delta}{\Gamma \Rightarrow \Delta} \text{Cut}$$

$$\dfrac{\dfrac{\Gamma, \varphi[x/y] \Rightarrow \Delta}{\Gamma, \exists x.\varphi \Rightarrow \Delta} \exists\text{Left} \qquad \dfrac{\Gamma \Rightarrow \Delta, \varphi[x/t]}{\Gamma \Rightarrow \Delta, \exists x.\varphi} \exists\text{Right}}{\Gamma \Rightarrow \Delta} \text{Cut} \quad \rightsquigarrow$$

$$\dfrac{\dfrac{\Gamma, \varphi[x/y] \Rightarrow \Delta}{\Gamma, \varphi[x/y][y/t] \Rightarrow \Delta} \text{Sub.} \qquad \Gamma \Rightarrow \Delta, \varphi[x/t]}{\Gamma \Rightarrow \Delta} \text{Cut}$$

3. Case where at least one premise of the cut rule is an axiom.

$$\dfrac{\dfrac{}{\Gamma, \varphi \Rightarrow \Delta} \text{Ax.} \quad \Gamma \Rightarrow \Delta, \varphi}{\Gamma \Rightarrow \Delta} \text{Cut} \quad \rightsquigarrow \quad \dfrac{}{\Gamma \Rightarrow \Delta} \text{Ax.}$$

   The dual case (i.e. the rule Ax. is applied on the right premise of the cut rule) is handled in the same way.

4. Finally, the case remains where the rules Sub. and Weak. have a premise which is the conclusion of some instances of @Left and @Right for $@ \in \{\wedge, \vee, \exists\}$, or Ax. Here, we can easily show that Sub. and Weak. pass over rules in @Left and @Right rules where $@ \in \{\vee, \wedge, \exists\}$, and are canceled out when they are occurring under instances of Ax. The transformations are similar to the ones for logicality.

We have already noticed in Section 4 that the transformations given above satisfy Condition 2 for the well-founded ordering $\preceq$ defined as follows:

$$\forall @ \in \{\vee, \neg, \exists\},\ \text{Cut} \succ \text{Weak.} \sim \text{Sub.} \succ @\text{Left} \sim @\text{Right} \sim \text{Ax.}$$

$$\dfrac{\Gamma_1 \Rightarrow \Delta_1, \varphi_1 \quad \Gamma_1, \varphi_1 \Rightarrow \Delta_1}{\Gamma_1 \Rightarrow \Delta_1} \text{Cut} \quad \succ \quad \dfrac{\Gamma_2 \Rightarrow \Delta_2, \varphi_2 \quad \Gamma_2', \varphi_2 \Rightarrow \Delta_2'}{\Gamma_2, \Gamma_2' \Rightarrow \Delta_2, \Delta_2'} \text{Cut}$$

$$\Leftrightarrow$$

$$|\varphi_2| < |\varphi_1|$$

where $|\varphi|$ is the depth of formula φ defined to be $sup_k(1 + |\varphi_k|)$ if the φ_i are the direct sub-formulas of φ. Therefore, by Theorem 4.4, the cut-elimination procedure as presented here is weakly normalizing.

Strong normalization result. We can observe that all the transformation rules of Cases 2., 3., and 4. satisfy Condition 1. Only the transformation rules of Case 1. do not satisfy this condition. This is due to the fact that the cut rule has the form

$$\frac{\Gamma \Rightarrow \Delta, \varphi \qquad \Gamma, \varphi \Rightarrow \Delta}{\Gamma \Rightarrow \Delta}\text{Cut}$$

But if we consider the cut rule with the following form

$$\frac{\Gamma \Rightarrow \Delta, \varphi \qquad \Gamma', \varphi \Rightarrow \Delta'}{\Gamma, \Gamma' \Rightarrow \Delta, \Delta'}\text{Cut}$$

the transformation rules of Case 1. become

$$\frac{\dfrac{\Gamma \Rightarrow \Delta, \varphi}{\Gamma' \Rightarrow \Delta', \varphi}@\text{Right} \qquad \Gamma'', \varphi \Rightarrow \Delta''}{\Gamma', \Gamma'' \Rightarrow \Delta', \Delta''}\text{Cut} \quad \rightsquigarrow \quad \frac{\Gamma \Rightarrow \Delta, \varphi \qquad \dfrac{\Gamma'', \varphi \Rightarrow \Delta''}{\Gamma, \Gamma'' \Rightarrow \Delta, \Delta''}@\text{Right}}{\Gamma', \Gamma'' \Rightarrow \Delta', \Delta''}\text{Cut}$$

where $@ \in \{\vee, \neg, \exists\}$ and $\Gamma' = \Gamma$ except for the $\neg$Right rule, where $\Gamma' = \Gamma, \neg\psi$ for some formula ψ. The other rules of Case 2. are noticeably similar. Now, all of these transformation rules satisfy Condition 1, and then by Theorem 4.1, the cut-elimination procedure is strongly normalizing.

Sequent calculus with explicit structural rules. In cut-elimination procedures for sequent calculus with explicit structural rules (i.e. contraction and weakening, and then sequents are defined by lists rather than sets of formulas), it is well-known that infinite reduction sequences may happen. Indeed, in a proof tree, when contraction occurs over cuts, this leads to the following transformation:

R2
$$\frac{\dfrac{\Gamma \Rightarrow \Delta, \varphi, \varphi}{\Gamma \Rightarrow \Delta, \varphi}\text{Contr.} \qquad \Gamma, \varphi \Rightarrow \Delta}{\Gamma \Rightarrow \Delta}\text{Cut} \quad \rightsquigarrow$$

$$\frac{\dfrac{\Gamma \Rightarrow \Delta, \varphi, \varphi \qquad \dfrac{\Gamma, \varphi \Rightarrow \Delta}{\Gamma, \varphi \Rightarrow \Delta, \varphi}\text{Weak}}{\Gamma \Rightarrow \Delta, \varphi}\text{Cut} \qquad \Gamma, \varphi \Rightarrow \Delta}{\Gamma \Rightarrow \Delta}\text{Cut}$$

Notice that both Conditions 1 and 2 do not hold for **R2**. Condition 1(i) is not verified since Cut should be strictly greater than any rule instance in π whose premises are not only formulas or rule instances, which does not hold for the last instance of Cut. Condition 2(ii) is not verified since the sequent $\Gamma \Rightarrow \varphi, \Delta$ is duplicated. Moreover, Condition 2(iv) is not verified either since Weak $\notin$ *Elim*.[17]

For such sequent calculi, some normalizing cut-elimination procedures have been developed. The first proof of strong normalization has been given by Dragalin in 1979 [30]. More recently, other proofs of strong normalization of cut-elimination procedures have been given in [12, 14, 37, 57]. All of these strong results are

[17] We recall that the weakening rules is explicit.

obtained either by "camouflaging" contractions into introduction or cut rules, or by transforming the cut rule in order to prevent contractions from occurring over cut rules. For the latter, this is how E. Tahhan Bittar in [12] enters upon strong normalization proofs for cut-elimination. The cut rule in [12], called Mix rule, is defined as follows:

$$\frac{\Gamma \vdash \Delta, \varphi^n \quad \Gamma', \varphi^m \vdash \Delta'}{\Gamma, \Gamma' \vdash \Delta, \Delta'} \text{Mix}$$

where φ^n means $\underbrace{\varphi, \ldots, \varphi}_{n \ times}$. For such a calculus, Rule **R2** above is then transformed as follows:

$$\frac{\dfrac{\Gamma \vdash \Delta, \varphi, \varphi}{\Gamma \vdash \Delta, \varphi}\text{Contr.} \qquad \Gamma', \varphi \vdash \Delta'}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\text{Mix} \quad \rightsquigarrow \quad \frac{\Gamma \vdash \Delta, \varphi, \varphi \quad \Gamma', \varphi \vdash \Delta'}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\text{Mix}$$

By using the precedence order:

$$\forall @ \in \{\vee, \neg, \exists, \text{Contr.}, \text{Weak}\},\ \text{Mix} \succ @\text{Right} \sim @\text{Left} \sim \text{Axiom}$$

we easily show that this cut-elimination procedure satisfies Condition 1. Of course, considering the Mix rule with the form

$$\frac{\Gamma \vdash \Delta, \varphi^n \quad \Gamma, \varphi^m \vdash \Delta}{\Gamma \vdash \Delta}$$

leads to a weak normalization result.

6. Cut-Elimination in Deduction modulo

Recently, sequent calculi for first-order logic have been extended by G. Dowek, Th. Hardin and C. Kirchner [26] to some computations on formulas. This extension is based on the fact that some axioms can be successfully replaced by rewrite rules on formulas. This leads to a faster proof-search and more readable proofs. The cut-elimination result given here differs from Hermant's one [38, 39]. Indeed, Hermant gives a semantic proof of cut elimination whereas a syntactical proof is given in this paper by defining an algorithm that transforms a proof into a cut-free one.

6.1. Sequent Calculus Modulo

First, let us introduce the deductive system for the sequent calculus modulo for classical first-order predicate logic. We follow the same notations and conventions for first-order logic and sequents as in Subsection 5.3.

In the sequel, we suppose that the reader is accustomed with the elementary definitions of rewriting theory as found in the introductory chapters of textbooks on the subject [8, 20]. We only introduced the following definition and notations:

Definition 6.1. *Let Σ be a signature. A* formula rewrite rule *is a pair of formulas $\varphi \to \psi$ such that φ is a Σ-atom and all free variables of ψ occur in φ. A* rewrite system *$\mathcal{R}$ is a set of formula rewrite rules.*
For a rewrite system $\mathcal{R}$, we denote by $\to_{\mathcal{R}}$ the rewriting relation induced by $\mathcal{R}$,

$\xrightarrow{*}_{\mathcal{R}}$ *the transitive and reflexive closure of* $\rightarrow_{\mathcal{R}}$ *and* $=_{\mathcal{R}}$ *its transitive, reflexive and symmetric closure.*

Given a signature $\Sigma = (Op, P)$ and a rewrite system $\mathcal{R}$, the deductive system $\mathcal{S} = (F, R)$ for Σ associated to Gentzen-style sequent calculi modulo $\mathcal{R}$ is defined as follows:

- F is the set of Σ-sequents
- R contains all the instances of the following rule schemes:

$$\frac{}{\Gamma, \varphi \Rightarrow \Delta, \psi}\text{Ax.} \quad \text{if } \varphi =_{\mathcal{R}} \psi$$

$$\frac{\Gamma, \varphi \Rightarrow \Delta \quad \Gamma, \varphi' \Rightarrow \Delta}{\Gamma, \psi \Rightarrow \Delta}\vee\text{Left} \qquad \frac{\Gamma \Rightarrow \Delta, \varphi, \varphi'}{\Gamma \Rightarrow \Delta, \psi}\vee\text{Right} \quad \text{if } \varphi \vee \varphi' =_{\mathcal{R}} \psi$$

$$\frac{\Gamma \Rightarrow \Delta, \varphi}{\Gamma, \psi \Rightarrow \Delta}\neg\text{Left} \qquad \frac{\Gamma, \varphi \Rightarrow \Delta}{\Gamma \Rightarrow \Delta, \psi}\neg\text{Right} \quad \text{if } \neg\varphi =_{\mathcal{R}} \psi$$

$$\frac{\Gamma, \varphi[x/y] \Rightarrow \Delta}{\Gamma, \psi \Rightarrow \Delta}\exists\text{Left} \qquad \frac{\Gamma \Rightarrow \Delta, \varphi[x/t]}{\Gamma \Rightarrow \Delta, \psi}\exists\text{Right} \quad \text{if } \exists x.\varphi =_{\mathcal{R}} \psi$$

$$\frac{\Gamma \Rightarrow \Delta, \varphi \quad \Gamma, \psi \Rightarrow \Delta}{\Gamma \Rightarrow \Delta}\text{Cut} \quad \text{if } \varphi =_{\mathcal{R}} \psi$$

where the $\exists$Left rule obeys the usual *eigenvariable* condition, stating that x is not free in Γ, Δ.

In the cut rule, the pair (φ, ψ) is called the *cut formula*. Finally, we use the standard terminology of *principal formula* with respect to a rule r as follows: any ψ such that

- $\psi =_{\mathcal{R}} \neg\varphi$ is principal with respect to $\neg$Left and $\neg$Right,
- $\psi =_{\mathcal{R}} \varphi \vee \varphi'$ is principal with respect to $\vee$Left and $\vee$Right, and
- $\psi =_{\mathcal{R}} \exists x.\varphi$ is principal with respect to $\exists$Left and $\exists$Right,

6.2. Cut-Elimination Algorithm

It has been shown in [29, 39] that the cut elimination property for the sequent calculus modulo for the first-order logic[18] depends on the rewrite system considered. Indeed, rewrite systems with rewrite rules on formulas define theories, and it is well-known that cut-elimination is not satisfied in general for theories. For instance, suppose the signature Σ is only composed of the constant R and the binary predicate $\in$. Then, consider the theory defined by the confluent rewrite system $\mathcal{R}$ composed of the unique rewrite rule: $R \in R \rightarrow \neg R \in R$ which is Russell's paradox. Obviously, we can prove the sequent $\vdash$ (which is a tautology for the sequent

[18] This is also true for the propositional and intuitionistic fragment.

calculus modulo $\mathcal{R}$):

$$\dfrac{\dfrac{\dfrac{\dfrac{}{R \in R \vdash R \in R}\text{Axiom}}{R \in R, R \in R \vdash}\neg\text{-R}}{R \in R \vdash}\text{C-L} \qquad \dfrac{\dfrac{\dfrac{}{R \in R \vdash R \in R}\text{Axiom}}{\vdash R \in R, R \in R}\neg\text{-L}}{\vdash R \in R}\text{C-R}}{\vdash}\text{Cut}$$

Obviously, $\mathcal{R}$ is not terminating. However, it has been shown that termination of rewrite systems is not sufficient to get the cut-elimination property. Indeed, for the same signature as above, if we consider the theory defined by the confluent and terminating rewrite system

$$R \in R \to \forall y((\forall x(\neg x \in R \Rightarrow \neg x \in y)) \Rightarrow \neg R \in y)$$

it has been shown in [27] that the empty sequent $\vdash$ can also be proved and then the theory does not have the cut elimination property. Hermant in [39] then showed by semantic arguments that for every confluent rewrite system compatible with a well-founded order $>$ on formulas (and then terminating) which satisfies the subformula property, the cut elimination property holds. Here, we show that this condition also algorithmically entails the cut elimination property. We then assume a well-founded order $>$ on formulas that has the subformula property:

- $\varphi_1 \vee \varphi_2 > \varphi_i$
- $\neg\varphi > \varphi$
- $\exists x.\varphi > \varphi[x/t]$

Definition 6.2. *A rewrite system $\mathcal{R}$ is said* compatible *with $>$ if and only if $\to_{\mathcal{R}} \subseteq >$.*

Therefore, a confluent rewrite system $\mathcal{R}$ compatible with $>$ is strongly normalizing, and then the normal form of a formula φ exists and is unique. In the following, we will denote the normal form of φ by $\varphi_{\downarrow}$.

Lemma 6.3.

1. $(\varphi_1 \vee \varphi_2)_{\downarrow} = \varphi_{1\downarrow} \vee \varphi_{2\downarrow}$
2. $(@\ \varphi)_{\downarrow} = @\ \varphi_{\downarrow}$ *where* $@ \in \{\exists x\}_{x \in V} \cup \{\neg\}$.

Proof. Results from the form of formula rewrite rules whose left-hand side is a Σ-atom. □

As for the classical sequent calculus (see Section 5.3), when considering the cut rule with the above form, to give a syntactical proof of the cut-elimination property, we need to add to the set R of inference rules the three following admissible rules:

$$\dfrac{\Gamma \Rightarrow \Delta}{\sigma(\Gamma) \Rightarrow \sigma(\Delta)}\text{Sub.}$$

where $\sigma : V \to T_\Sigma(V)$ is a substitution and $\sigma(\Gamma) = \sigma(\varphi_1), \ldots, \sigma(\varphi_n)$ when $\Gamma = \varphi_1, \ldots, \varphi_n$.

$$\frac{\Gamma \Rightarrow \Delta}{\Gamma' \Rightarrow \Delta'}\text{Weak}$$

where $\Gamma \subseteq \Gamma'$ and $\Delta \subseteq \Delta'$.

$$\frac{\Gamma, \varphi \vdash \Delta}{\Gamma, \varphi_{\downarrow} \vdash \Delta}\text{-}_{\downarrow}\text{-L} \quad \frac{\Gamma \vdash \Delta, \varphi}{\Gamma \vdash \Delta, \varphi_{\downarrow}}\text{-}_{\downarrow}\text{-R}$$

We already explained the reason for the two first rules in Subsection 5.3. The third rule comes from the fact that we are dealing with confluent rewrite systems compatible with a well-founded order $>$ on formulas. Let us call the resulting extended system of rules R'. We will see just below that our method is such that cut-free proofs with respect to R' are actually cut-free proofs using only rules in R.

Lemma 6.4. $\varphi[x/t]_{\downarrow} = \varphi_{\downarrow}[x/t]$

Proof. By induction on the structure of the formula φ. □

Therefore, Gentzen's result (and, in particular, the termination of the cut-elimination procedure, which is the difficult part) can be shown by using the basic transformation rules described below, that are noticeably similar to the ones given in Section 5.3. Therefore, we have the following transformation rules that can also be organized in 4 cases.

1. Case where the cut formula is not principal with respect to at least one of the inferences leading immediately to the premises of the cut. Here, we give the transformations for the case where right rules are applied so as to get the left premise of the cut rule, but one can generalize such transformations to the symmetric case in a standard way.

$$\dfrac{\dfrac{\Gamma \Rightarrow \Delta, \varphi}{\Gamma' \Rightarrow \Delta', \varphi}\text{@Right} \qquad \Gamma', \varphi' \Rightarrow \Delta'}{\Gamma' \Rightarrow \Delta'}\text{Cut} \quad \rightsquigarrow$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\Gamma \Rightarrow \Delta, \varphi}{\vdots}\text{Weak.}}{\Gamma, \Gamma' \Rightarrow \Delta, \Delta', \varphi}\text{Weak.} \qquad \dfrac{\dfrac{\Gamma', \varphi' \Rightarrow \Delta'}{\vdots}\text{Weak.}}{\Gamma, \Gamma', \varphi' \Rightarrow \Delta, \Delta'}\text{Weak.}}{\Gamma, \Gamma' \Rightarrow \Delta, \Delta'}\text{Cut}}{\Gamma' \Rightarrow \Delta'}\text{@Right}$$

   where $@ \in \{\vee, \neg, \exists\}$ and $\Gamma' = \Gamma$ except for the $\neg$Right rule, where $\Gamma' = \Gamma, \neg\psi$ for some formula ψ.
2. Case where the cut formula is principal with respect to both inferences leading immediately to the premises of the cut. In this case we have the following proof tree transformation:

$$\dfrac{\dfrac{\Gamma, \varphi_1 \Rightarrow \Delta \quad \Gamma, \varphi_1' \Rightarrow \Delta}{\Gamma, \psi_1 \Rightarrow \Delta} \vee\text{Left} \qquad \dfrac{\Gamma \Rightarrow \Delta, \varphi_2, \varphi_2'}{\Gamma \Rightarrow \Delta, \psi_2} \vee\text{Right}}{\Gamma \Rightarrow \Delta} \text{Cut} \quad \rightsquigarrow$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\Gamma, \varphi_1 \Rightarrow \Delta}{\Gamma, \varphi_{1\downarrow} \Rightarrow \Delta} \downarrow\text{Left}}{\Gamma, \varphi_{1\downarrow} \Rightarrow \Delta, \varphi_{2\downarrow}'} \text{Weak.} \qquad \dfrac{\dfrac{\Gamma \Rightarrow \Delta, \varphi_2, \varphi_2'}{\Gamma \Rightarrow \Delta, \varphi_{2\downarrow}, \varphi_2'} \downarrow\text{Right}}{\Gamma \Rightarrow \Delta, \varphi_{2\downarrow}, \varphi_{2\downarrow}'} \downarrow\text{Right}}{\Gamma \Rightarrow \Delta, \varphi_{2\downarrow}'} \text{Cut} \qquad \dfrac{\dfrac{\Gamma, \varphi_1' \Rightarrow \Delta}{\Gamma, \varphi_{1\downarrow}' \Rightarrow \Delta} \downarrow\text{Left}}{\Gamma \Rightarrow \Delta} \text{Cut}$$

This transformation is correct because we have respectively $\psi_1 =_{\mathcal{R}} \psi_2$, $\psi_1 =_{\mathcal{R}} \varphi_1 \vee \varphi_1'$ and $\psi_2 =_{\mathcal{R}} \varphi_2 \vee \varphi_2'$. Therefore, we have $\varphi_1 \vee \varphi_1' =_{\mathcal{R}} \varphi_2 \vee \varphi_2'$, and then by Lemma 6.3 $\varphi_{1\downarrow} \vee \varphi_{1\downarrow}' = \varphi_{2\downarrow} \vee \varphi_{2\downarrow}'$, hence we conclude that $\varphi_{1\downarrow} = \varphi_{2\downarrow}$ and $\varphi_{1\downarrow}' = \varphi_{2\downarrow}'$.

For similar reasons to the first transformation, the two following ones are also correct. Moreover, in the last transformation, the way we use the rules $\downarrow$Left and $\downarrow$Right is correct by Lemma 6.4.

$$\dfrac{\dfrac{\Gamma \Rightarrow \Delta, \varphi}{\Gamma, \psi \Rightarrow \Delta} \neg\text{Left} \qquad \dfrac{\Gamma, \varphi' \Rightarrow \Delta}{\Gamma \Rightarrow \Delta, \psi'} \neg\text{Right}}{\Gamma \Rightarrow \Delta} \text{Cut} \quad \rightsquigarrow$$

$$\dfrac{\dfrac{\Gamma \Rightarrow \Delta, \varphi}{\Gamma \Rightarrow \Delta, \varphi_\downarrow} \downarrow\text{Right} \qquad \dfrac{\Gamma, \varphi' \Rightarrow \Delta}{\Gamma, \varphi_\downarrow' \Rightarrow \Delta} \downarrow\text{Left}}{\Gamma \Rightarrow \Delta} \text{Cut}$$

$$\dfrac{\dfrac{\Gamma, \varphi[x/y] \Rightarrow \Delta}{\Gamma, \psi \Rightarrow \Delta} \exists\text{Left} \qquad \dfrac{\Gamma \Rightarrow \Delta, \varphi'[x/t]}{\Gamma \Rightarrow \Delta, \psi'} \exists\text{Right}}{\Gamma \Rightarrow \Delta} \text{Cut} \quad \rightsquigarrow$$

$$\dfrac{\dfrac{\dfrac{\Gamma, \varphi[x/y] \Rightarrow \Delta}{\Gamma, \varphi_\downarrow[x/y] \Rightarrow \Delta} \downarrow\text{Left}}{\Gamma, \varphi_\downarrow[x/y][y/t] \Rightarrow \Delta} \text{Sub.} \qquad \dfrac{\Gamma \Rightarrow \Delta, \varphi'[x/t]}{\Gamma \Rightarrow \Delta, \varphi_\downarrow'[x/t]} \downarrow\text{Right}}{\Gamma \Rightarrow \Delta} \text{Cut}$$

3. Case where at least one premise of the cut rule is an axiom.

$$\dfrac{\dfrac{}{\Gamma, \varphi \Rightarrow \Delta} \text{Ax.} \qquad \Gamma \Rightarrow \Delta, \varphi}{\Gamma \Rightarrow \Delta} \text{Cut} \quad \rightsquigarrow \quad \Gamma \Rightarrow \Delta$$

   The dual case (i.e. the rule Ax. is applied on the right premise of the cut rule) is handled in the same way.

4. Finally, the case remains where the rules Sub. Weak. and $\downarrow$Left and $\downarrow$Right have a premise which is the conclusion of some instances of @Left and @Right for $@ \in \{\neg, \vee, \exists\}$, or Ax. Here, we can easily show that these three rules can move over rules in @Left and @Right rules where $@ \in \{\vee, \neg, \exists\}$, and are canceled out when they occur under instances of Ax. The basic transformations are similar to the ones of the logicality result for the equational logic. For

instance, we have the following transformation:

$$\dfrac{\dfrac{\Gamma, \varphi \Rightarrow \Delta \quad \Gamma, \varphi' \Rightarrow \Delta}{\Gamma, \psi \Rightarrow \Delta}\vee\text{Left}}{\Gamma, \psi_{\downarrow} \Rightarrow \Delta}\downarrow\text{Left} \quad \rightsquigarrow \quad \dfrac{\dfrac{\Gamma, \varphi \Rightarrow \Delta}{\Gamma, \varphi_{\downarrow} \Rightarrow \Delta}\downarrow\text{Left} \quad \dfrac{\Gamma, \varphi' \Rightarrow \Delta}{\Gamma, \varphi'_{\downarrow} \Rightarrow \Delta}\downarrow\text{Left}}{\Gamma, \psi_{\downarrow} \Rightarrow \Delta}\vee\text{Left}$$

This transformation is correct by Lemma 6.3.

Let us consider the well-founded order $\geq$ on rule instances defined as follows:

$$\text{Cut} \succ \text{Weak.} \sim \text{Sub.} \sim \downarrow\text{Left} \sim \downarrow\text{Right} \succ @\text{Left} \sim @\text{Right} \sim \text{Ax.}$$

where $@ \in \{\vee, \neg, \exists\}$, and

$$\dfrac{\Gamma_1, \varphi_1 \Rightarrow \Delta_1 \quad \Gamma_1 \Rightarrow \Delta_1, \psi_1}{\Gamma_1 \Rightarrow \Delta_1}\text{Cut} \quad \succ \quad \dfrac{\Gamma_2, \varphi_2 \Rightarrow \Delta_2 \quad \Gamma_2 \Rightarrow \Delta_2, \psi_2}{\Gamma_2 \Rightarrow \Delta_2}\text{Cut}$$

$$\Longleftrightarrow$$

$$\{\varphi_1, \psi_1\} \gg \{\varphi_2, \psi_2\}$$

where $\gg$ is the extension of the order on formulas to multisets of formulas.

By this well-founded order, we can easily check that the above transformation rules satisfy Condition 2. Therefore, by Theorem 4.4, the syntactical cut-elimination procedure presented here is weakly normalizing. Considering the cut rule with the form

$$\dfrac{\Gamma, \varphi \Rightarrow \Delta \quad \Gamma' \Rightarrow \Delta', \psi}{\Gamma, \Gamma' \Rightarrow \Delta, \Delta'}\text{Cut}$$

leads to a strong normalization result.

7. Conclusions and Perspectives

In this paper, we give a method to obtain results of normalization of proof trees independently of the underlying deductive system. This is achieved by generalizing the observation that some inference rules move over or are canceled out when they occur under any other rules. This generalization is expressed by a set of basic proof tree transformations which, when the induced global proof transformation is terminating, ensures the completeness of the corresponding proof strategy. However, termination cannot be ensured in general. Therefore, two conditions on the structure of basic proof tree transformations are given, which are sufficient to ensure such a result of termination. These two conditions allow us to obtain an abstract strong and an abstract weak normalization result, respectively. We would be able to increase the number of examples of already known normalization results such as cut-elimination for standard modal propositional sequent calculi for the logics *K*, *K4*, *T*, *S4*, for linear logic sequent calculus and/or proof-nets (when proof-nets are presented under the form of proof structures which are built according to the rules of linear sequent calculus as in [35]), or for display calculi [9, 16], and showing that all of them meet the conditions given in the paper.

In order to validate our approach, we plan to continue this work by submiting our "meta proofs" to a generic theorem proof assistant, as, for instance, Isabelle [47]

or Coq [15], as it is done for a specific proof of strong normalization for the case of display calculi in [16]. This enables one to certify, as a result, all existing and future normalization results. Indeed, it would be sufficient to check that the proof tree transformation rules which underlie the normalization result meet the sufficient conditions of Section 4.

References

[1] M. Aiguier, A. Arnould, C. Boin, P. Le Gall, and B. Marre. Testing from algebraic specifications: Test data set selection by unfolding axioms. In W. Grieskamp and C. Weise, editors, *Formal Approaches to Testing of Software (FATES)*, volume 3997 of *Lecture Notes in Computer Science*, pages 304–317. Springer-Verlag, 2005.

[2] M. Aiguier, A. Arnould, P. Le Gall, and D. Longuet. Test selection criteria for quantifier-free first-order specifications. In F. Arbab and M. Sirjani, editors, *International Symposium on Fundamentals of Software ENgineering (FSEN)*, volume 4767 of *Lecture Notes in Computer Science*, pages 144–160. Springer-Verlag, 2007.

[3] M. Aiguier and D. Bahrami. Structures for abstract rewriting. *Journal of Automated Reasoning*, 38(4):303–351, 2007.

[4] M. Aiguier, D. Bahrami, and C. Dubois. On a generalised logicality theorem. In *Artificial Intelligence, Automated Reasoning, and Symbolic Computation*, volume 2385 of *Lecture Notes in Artificial Intelligence*, pages 51–64. Springer Verlag, 2002.

[5] M. Aiguier and F. Barbier. An institution-independent proof of the Beth definability theorem. *Studia Logica*, 85(3):333–359, 2007.

[6] M. Aiguier and R. Diaconescu. Stratified institutions and elementary homomorphisms. *Information Processing Letters*, 103(1):5–13, 2007.

[7] M. Aiguier and D. Longuet. Test selection criteria for modal specifications of reactive systems. In *Joint IEEE/IFIP Symposium on Theoretical Aspects of Software Engineering (TASE)*, pages 159–170. IEEE Computer Society, 2007.

[8] F. Baader and T. Nipkow. *Term Rewriting and All That*. C.U. Press, 1998.

[9] N. D. Belnap. Display logic. *Journal of Philosophical Logic 11*, pages 375–417, 1982.

[10] U. Berger, M. Eberl, and H. Schwichtenberg. Term rewriting for normalization by evaluation. *Information and Computation*, 183:19–42, 2003.

[11] G. Birkhoff. On the structure of abstract algebras. In *Proceedings of The Cambridge Philosophical Society*, volume 31, pages 433–454, 1935.

[12] E. Tahhan Bittar. Strong normalization proofs for cut-elimination in Gentzen's sequent calculi. In *Proceedings of the Symposium on Algebra and Computer Science, Helena Rasiowa in Memoriam*, volume 46, pages 179–225. Banach Center Publication, 1999.

[13] T. Borzyszkowski. Logical systems for structured specifications. *Theoretical Computer Science*, 286:197–245, 2002.

[14] E.-A. Cichon, M. Rusinowitch, and S. Selhab. Cut elimination and rewriting: termination proofs. Technical report, INRIA-Lorraine, Nancy, France, 1996.

[15] C. Cornes, J. Courant, J.-C. Fillâtre, G. Huet, P. Manoury, C. Munoz, C. Murthy, C. Parent, Ch. Paulin-Mohring, A. Sabi, and B. Werner. The Coq proof assistant reference manual, version 5.10. Technical Report 177, INRIA, July 1995.

[16] J. Dawson and R. Gore. A new machine-checked proof of strong normalisation for display logic. In James Harland, editor, *Electronic Notes in Theoretical Computer Science*, volume 78. Elsevier, 2003.

[17] J.-E. Dawson and R. Goré. A general theorem on termination of rewriting. In *Computer Science Logic (CSL'04)*, volume 3210 of *Lecture Notes in Computer Science*, pages 100–114. Springer-verlag, 2004.

[18] J.-E. Dawson and R. Goré. Termination of abstract reduction systems computing. In *The Australian Theory Symposium, 2007 (CATS 2007)*, volume 65 of *Conferences in Research and Practice in Information Technology (CRPIT)*, pages 35–43. Australian Computer Society, 2007.

[19] N. Dershowitz. A note on simplification orderings. *Information Processing Letters*, 9(5):212–215, 1979.

[20] N. Dershowitz and J.-P. Jouannaud. Rewrite systems. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B: Formal Models and Semantics, pages 243–320. Elsevier, 1990.

[21] N. Dershowitz and C. Kirchner. Abstract saturation-based inference. In Kolaitis and Phokion, editors, *Proceedings of the annual Symposium on Logic in Computer Science*. IEEE Publiser Society, 2003.

[22] N. Dershowitz and C. Kirchner. Abstract canonical presentations. *Theoretical Computer Science*, 357(1-3):53–69, 2006.

[23] R. Diaconescu. *Institution-independent Model Theory*. Studies in Universal Logic. Birkhuser, 2008.

[24] R. Diaconescu, J. Goguen, and P. Stefaneas. Logical support for modularization. In G. Huet and G. Plotkin, editors, *Logical Environments*, pages 83–130, 1991. Proceedings of Workshop on Logical Frameworks.

[25] G. Dowek. Confluence as a cut-elimination property. In R. Nieuwenhuis, editor, *Rewriting Techniques and Applications*, volume 2706 of *Lecture Notes in Computer Science*, pages 2–14. Springer Verlag, 2003.

[26] G. Dowek, Th. Hardin, and C. Kirchner. Theorem proving modulo. *Journal of Automated Reasoning*, 31:33–72, 2003.

[27] G. Dowek and B. Werner. An inconsistent theory modulo defined by a confluent and terminating rewrite system. Available at www.lix.polytechnique.fr/∼dowek/.

[28] G. Dowek and B. Werner. Proof normalization modulo. In *International Workshop on Types for Proofs and Programs*, pages 62–77, London, UK, 1999. Springer Verlag.

[29] G. Dowek and B. Werner. Proof normalization modulo. *Journal of Symbolic Logic*, 68(4):1289–1316, 2003.

[30] A.-G Dragalin. Mathematical intuitionism, introduction to proof theory. *Translations of Mathematical Monographs*, 67:185–199, 1988. Translation of the russian original from 1979.

[31] J. Fiadeiro and A. Sernadas. Structuring theories on consequence. In *Recent Trends in Algebraic Development Techniques*, volume 332 of *Lecture Notes in Computer Science*, pages 44–72. Springer, 1988.

[32] J.-H. Gallier. *Logic for Computer Science*, volume Foundations of Automatic Theorem Proving. Harper & Row, 1986.

[33] G. Gentzen. Untersuchungen ber das logische schlieen. *Mathematische Zeitschrift*, 39(176-210):405–431, 1935. English translation in M.-E. Szabo, editor, The collected Papers of Gerhard Gentzen, pages 68–131, North-Holland, 1969.

[34] J.-Y. Girard. Linear logic. *Theoretical Computer Science*, 50:1–102, 1987.

[35] J.-Y. Girard, Y. Lafont, and P. Taylor. *Proofs and Types*. Cambridge University Press, 1989.

[36] J.A. Goguen and R.-M. Burstall. Institutions: Abstract model theory for specification and programming. *Journal of the ACM*, 39(1):95–146, 1992.

[37] H. Herbelin. A λ-calculus structure isomorphic to Gentzen-style sequent calculus structure. In L. Pacholski and J. Tiuryn, editors, *Proceedings of the 8th International Workshop on Computer Science Logic*, volume 933 of *Lecture Notes in Computer Science*, pages 61–75. Springer Verlag, 1995.

[38] O. Hermant. A model-based cut elimination proof. In *2nd St-Petersburg Days of Logic and Computability*, 2003.

[39] O. Hermant. Semantic cut elmination in the intuitionistic sequent calculus. In P. Urzyczyn, editor, *International conference on Typed Lambda Calculi and Applications (TLCA)*, volume 3461 of *Lecture Notes in Computer Science*, pages 221–233. Springer-Verlag, 2007.

[40] S. Kaplan. Conditional rewrite rules. *Theoretical Computer Science*, 33:175–193, 1984.

[41] S.-C. Kleene. *Introduction to Meta-mathematics*. North-Holland, 1952.

[42] J. Levy and J. Agustí. Bi-rewrite systems. *Journal of Symbolic Computation*, 22:279–314, 1996.

[43] D. Longuet and M. Aiguier. Specification-based testing for CoCasl's modal specifications. In T. Mossakowski, U. Montanari, and M. Haveraaen, editors, *Conference on Algebra and Coalgebra in Computer Science (CALCO)*, volume 4624 of *Lecture Notes in Computer Science*, pages 356–371. Springer-Verlag, 2007.

[44] D. Longuet and M. Aiguier. Integration testing from structured specifications via deduction modulo. In *International Colloquium on Theoretical Aspects of Computing*, 2009. To appear.

[45] D. Longuet, M. Aiguier, and P. Le Gall. Proof-guided test selection from first-order specifications with equality. *Journal of Automated Reasoning*, 2009. To appear.

[46] J. Meseguer. General logics. In *Logic Colloquium'87*, pages 275–329. Holland, 1989.

[47] T. Nipkow, L. Paulson, and M. Wenzel. Isabelle's logics: HOL. http://isabelle.in.tum.de/doc/logics-HOL.pdf.

[48] D. Prawitz. Natural deduction: A proof-theoretical study. *Almqvist and Wiksell, Stockholm*, 1965.

[49] A. Salibra and G. Scollo. Compactness and Löwenheim-Skolem properties in categories of pre-institutions. In C. Rauszer, editor, *Algebraic Methods in Logic and in Computer Science*, volume 28 of *Banach Center Publ.*, pages 67–94, 1993.

[50] A. Salibra and G. Scollo. Interpolation and compactness in categories of pre-institutions. *Mathematical Structures in Computer Science*, 6:261–286, 1996.

[51] W. Marco Schorlemmer. Term rewriting in a logic of special relations. In *7th International Conference on Algebraic Methodology and Software Technology*, volume 1548 of *Lecture Notes in Computer Science*, pages 178–195. Springer, 1998.

[52] G. Struth. Non-symmetric rewriting. Technical report, MPI für Informatik, 1996.

[53] W.-W. Tait. Normal derivability in classical logic. In J. Barwise, editor, *The Syntax and Semantics of Infinitary Languages*, pages 204–236. Springer Verlag, 1989.

[54] A. Tarlecki. On the existence of free models in abstract algebraic institutions. *Theoretical Computer Science*, 37:269–304, 1986.

[55] A. Tarlecki. Quasi-varieties in abstract algebraic institutions. *Journal of Computer and System Science*, 33(3):269–304, 1986.

[56] A. Tarlecki. *Algebraic Foundations of Systems Specification*, chapter Institutions: An abstract Framework for Formal Specifications, pages 105–131. IFIP State-of-the-Art Reports. Springer, 1999.

[57] C. Urban and G.-M. Bierman. Strong normalisation of cut-elimination in classical logic. *Acta Informaticae*, 45(1-2):123–155, 2001.

[58] V. van Oostrom. Sub-Birkhoff. In *7th International Symposium on Functional and Logic Programming*, volume 2998 of *Lecture Notes in Computer Science*, pages 180–195. Springer, 2004.

[59] L.-A. Wallen. *Automated Deduction for Non Classical Logics*. The MIT Press, 1990.

[60] T. Yamada, J. Avenhaus, C. Loria-Saenz, and A. Middeldorp. Logicality of conditional rewrite systems. *Theoretical Computer Science*, 236(1,2):209–232, 2000.

Marc Aiguier
MAS, École Centrale Paris (ECP), Châtenay-Malabry, France
e-mail: `marc.aiguier@ecp.fr`

Delphine Longuet
Univ. Paris-Sud, LRI UMR8623, 91405 Orsay, France
e-mail: `delphine.longuet@lri.fr`