From Axioms to Rewriting Rules

Guillaume Burel

ÉNSIIE/Cédric
1 square de la résistance, 91025 Évry cedex, France
guillaume.burel@ensiie.fr
http://www.ensiie.fr/~guillaume.burel/

Abstract

Deduction modulo is a generic framework to describe proofs in a theory better than using raw axioms. This is done by presenting the theory through a congruence over propositions that is most often defined by means of rules rewriting terms and propositions. It has been shown that such representations of theories preserve good properties of axiom-free deductive systems, can lead to theoretical proof-length speed-ups and actually improve automated proof search. In this paper, we positively answer the theoretical question whether all first-order theories can be represented by such rewriting systems, while preserving a crucial proof-theoretical property, namely cut admissibility, equivalent to the completeness of proof search. We also perform experiments to compare several techniques to orient axioms into rewriting systems, some of them being complete, some being based on heuristics. These experiments confirm the practical interest of using rewriting rules instead of axioms.

Keywords and phrases automated deduction, proof theory, theory reasoning, rewriting, refinements of resolution

Digital Object Identifier 10.4230/LIPIcs.xxx.yyy.p

1 Introduction

Proofs are rarely built without context: mathematical theorems are proved for instance in set theory, or in arithmetic; program correctness may use pointer arithmetic or the theories associated to the data structures of the program (chained lists, arrays, etc.); theories can also model characteristics of encryption functions to prove security properties. Some proving contexts can also be seen as theories. For instance, the Sledgehammer tactic of Isabelle sends to automated provers the goal to be proved with a number of related lemmas that can be used as axioms, and that form therefore a specific theory in which the goal must be proved. Therefore, it is essential to develop methods that are adapted to search for proofs in theories. For instance, SMT provers provide efficient tools. Nevertheless, they are restricted to some particular theories, such as linear arithmetic or arrays. We would like to have a generic and automated way of obtaining efficient methods for a given theory, provided it is consistent. A naive idea is to use an axiomatic presentation of the theory, but it is now folklore that this is not efficient enough. The theory should therefore be presented in a more effective manner. One solution is, starting from the axiomatic presentation, to automatically design a deductive system that is adapted to the theory. In [27], Negri and von Plato turn variable-free axioms into non-logical deduction rules that are added to a sequent calculus. Similarly, [12] transforms a large class of axioms into inference rules in sequent and hypersequent calculi. Deduction modulo [18] is a bit different: it presents the theory as computation, by means of a rewriting system, and the inference rules of an existing deductive system (natural deduction, sequent calculus, etc.) are applied modulo the congruence associated with this rewriting system. Deduction modulo can theoretically lead to unbounded proof-length speed-ups [7],

and we have shown in [8] that presenting theories as rewriting systems improves indeed the search for proofs in those theory.

If one wants these presentations to behave well, they should have the following proof-theoretical property: the cut rule must be admissible. Indeed, in the usual setting, cut admissibility implies the consistency of the theory, the subformula property (to find a proof, one can restrict oneself to the subformulas of the formula to be proved), the existence of proof normal forms, etc. Furthermore, in deduction modulo, this property is equivalent to the completeness of the various derived proof-search methods [18, 4, 16, 6]. For all systems produced by [27, 12], because of restrictions on the form of the theories, the cut admissibility holds. However, in deduction modulo, it depends on the considered rewriting system. The questions are: knowing that the theory is consistent, is it possible to present it as a rewriting system such that cut admissibility holds in deduction modulo? And can this transformation into a rewriting system be automated? A presentation as a rewriting system with cut admissibility was designed specially for particular theories, such as Peano arithmetic [20], simple type theory [17], and Zermelo set theory [19]. When we experimented with our integration of a proof search method based on deduction modulo into an existing prover [8], we had to design such a rewriting system by hand for each theory we considered, which led us to restrict ourselves to only five theories. Dowek designed a systematic way of transforming a consistent *propositional* theory into such a rewriting system, using a model of the theory. In [11], we gave a semi-algorithm that can handle any first-order theory: first, it produces a rewriting system that corresponds to the theory; second, it completes the rewriting system to ensure cut admissibility. It is the second part that may not terminate. In this paper, we show for the first time how any first-order theory can always be presented as a rewriting system with cut admissibility. This is done by developing a characterization [10] of an extension of the resolution method based on deduction modulo as a combination of the set-of-support strategy [32] and selection of literals.

The method that we introduce is a theoretical answer to the question of presenting first-order theories as rewriting systems with cut admissibility. However, in practice, the resulting rewriting system could be not much better than the use of axioms, in particular because it contains too many rules. We compare several ways to orient axioms into rewriting rules. Some of them are based on the theoretical work that we introduce here, and are therefore complete, in the sense that resolution modulo the resulting resulting rewrite systems is complete. Some others are based on heuristics. We have performed an experiment to compare them, using our integration of deduction modulo in iProver on problems from the TPTP database [30].

In the two next sections, we briefly present deduction modulo and refinements of resolution. Section 4 describes how a theory can be presented as a rewriting system, and why cut admissibility is implied by the consistency of the theory. We then describe in Section 5 the different practical algorithms that can be used to perform this task, and we perform an experiment to compare them. We conclude by discussing further works.

2 Deduction Modulo

We use standard definitions for terms, predicates, propositions (with connectives $\neg, \Rightarrow, \wedge, \vee$ and quantifiers $\forall, \exists$), sequents, substitutions, term rewriting rules and term rewriting, as can be found in [1, 21] . The substitution of a variable x by a term t in a term or a proposition A is denoted by $\{t/x\}A$, and more generally the application of a substitution σ in a term or a proposition A by σA. A term t can be narrowed into s using substitution σ at position

$\mathfrak{p}$ ($t \overset{\mathfrak{p},\sigma}{\rightsquigarrow} s$) if σt can be rewritten into s using substitution σ at position $\mathfrak{p}$. A literal is an atomic proposition or the negation of an atomic proposition. A proposition is in clausal form if it is the universal quantification of a disjunction of literals $\forall x_1, \ldots, x_n.\ L_1 \vee \ldots \vee L_p$ where $x_1, \ldots, x_n$ are the free variables of $L_1, \ldots, L_p$. In the following, we will often omit the quantifications, and we will identify propositions in clausal form with clauses (i.e. set of literals) as if $\vee$ were associative, commutative and idempotent. The symbol $\square$ represents the empty clause. The polarity of a position in a proposition can be defined as follows: the root is positive, and the polarity switches when going under a $\neg$ or on the left of a $\Rightarrow$.

In deduction modulo, term rewriting and narrowing is extended to propositions by congruence on the proposition structure. In addition, there are also proposition rewriting rules whose left-hand side is an atomic proposition and whose right-hand side can be any proposition. Such rules can also be applied to non-atomic propositions by congruence on the proposition structure. We call a rewriting system the combination of a term rewriting system and a proposition rewriting system. Given a rewriting system $\mathcal{R}$, we denote by $A \underset{\mathcal{R}}{\longrightarrow} B$ the fact that A is rewritten in one step into B, either by a term rewriting rule or by a proposition rewriting rule, and by $A \underset{\mathcal{R}}{\rightsquigarrow} B$ the fact that A is narrowed to B. $\underset{\mathcal{R}}{\overset{*}{\longrightarrow}}$ is the reflexive transitive closure of $\underset{\mathcal{R}}{\longrightarrow}$. Deduction modulo consists in applying the inference rules of an existing proof system modulo such a rewriting system. This leads for instance to the asymmetric sequent calculus modulo [15], some of whose rules are presented in Figure 1.

▶ **Example 1.** Consider the rewriting rule $A \subseteq B \to \forall x.\ x \in A \Rightarrow x \in B$. We can build the following proof of the transitivity of the inclusion in the asymmetric sequent calculus modulo this rule:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{x \in C \vdash x \in C}\,\widehat{\vdash} \quad \dfrac{}{x \in B \vdash x \in B}\,\widehat{\vdash}}{x \in B \Rightarrow x \in C, x \in B \vdash x \in C}\,\Rightarrow\vdash}{B \subseteq C, x \in B \vdash x \in C}\,\forall\vdash \quad \dfrac{}{x \in A \vdash x \in A}\,\widehat{\vdash}}{x \in A \Rightarrow x \in B, B \subseteq C, x \in A \vdash x \in C}\,\Rightarrow\vdash}{A \subseteq B, B \subseteq C, x \in A \vdash x \in C}\,\forall\vdash}{A \subseteq B, B \subseteq C \vdash x \in A \Rightarrow x \in C}\,\vdash\Rightarrow}{A \subseteq B, B \subseteq C \vdash A \subseteq C}\,\vdash\forall$$

Rewriting rules can be applied indifferently to the left- or the right-hand side of a sequent. Consequently, they can be considered semantically as an equivalence between their left- and right-hand sides. To be able to consider implications, a polarized version of deduction modulo was introduced [14]. Proposition rewriting rules are tagged with a polarity $+$ or $-$; they are then called polarized rewriting rules. A proposition A is rewritten positively into a proposition B ($A \longrightarrow^{+} B$) if it is rewritten by a positive rule at a positive position or by a negative rule at a negative position. It is rewritten negatively ($A \longrightarrow^{-} B$) if it is rewritten by a positive rule at a negative position or by a negative rule at a positive position. Intuitively, a positive rule $A \to^{+} B$ (resp. a negative rule $B \to^{-} A$) corresponds to an implication $B \Rightarrow A$. Term rewriting rules (but not proposition rewriting rules) are considered as both positive and negative. $\overset{*}{\longrightarrow}^{\pm}$ is the reflexive transitive closure of $\longrightarrow^{\pm}$. This gives the polarized sequent calculus modulo, some of whose rules are presented in Figure 2.

▶ **Example 2.** Consider the polarized rewriting system

$$A \subseteq B \to^{-} \forall x.\ x \in A \Rightarrow x \in B$$
$$A \subseteq B \to^{+} \neg \mathit{diff}(A, B) \in A$$
$$A \subseteq B \to^{+} \mathit{diff}(A, B) \in B$$

$$\vdash\frown \frac{}{\Gamma, A \vdash B, \Delta}\ A \xrightarrow[\mathcal{R}]{*} C \xleftarrow[\mathcal{R}]{*} B \qquad \vdash\smile \frac{\Gamma, A \vdash \Delta \qquad \Gamma \vdash B, \Delta}{\Gamma \vdash \Delta}\ A \xleftarrow[\mathcal{R}]{*} C \xrightarrow[\mathcal{R}]{*} B$$

$$\Rightarrow\vdash \frac{\Gamma, B \vdash \Delta \qquad \Gamma \vdash A, \Delta}{\Gamma, C \vdash \Delta}\ C \xrightarrow[\mathcal{R}]{*} A \Rightarrow B \qquad \vdash\Rightarrow \frac{\Gamma, A \vdash B, \Delta}{\Gamma \vdash C, \Delta}\ C \xrightarrow[\mathcal{R}]{*} A \Rightarrow B$$

$$\forall\vdash \frac{\Gamma, \{t/x\}A \vdash \Delta}{\Gamma, B \vdash \Delta}\ B \xrightarrow[\mathcal{R}]{*} \forall x.\ A \qquad \vdash\forall \frac{\Gamma \vdash A, \Delta}{\Gamma \vdash B, \Delta}\ B \xrightarrow[\mathcal{R}]{*} \forall x.\ A \quad x \text{ not free in } \Gamma, \Delta$$

Figure 1 Some inference rules of the Asymmetric Sequent Calculus Modulo $\mathcal{R}$

$$\vdash\frown \frac{}{\Gamma, A \vdash B, \Delta}\ A \xrightarrow[\mathcal{R}]{*} {}^{-}C\,{}^{+} \xleftarrow[\mathcal{R}]{*} B \qquad \vdash\smile \frac{\Gamma, A \vdash \Delta \qquad \Gamma \vdash B, \Delta}{\Gamma \vdash \Delta}\ A\ {}^{-}\xleftarrow[\mathcal{R}]{*} C \xrightarrow[\mathcal{R}]{*} {}^{+}B$$

$$\Rightarrow\vdash \frac{\Gamma, B \vdash \Delta \qquad \Gamma \vdash A, \Delta}{\Gamma, C \vdash \Delta}\ C \xrightarrow[\mathcal{R}]{*} {}^{-}A \Rightarrow B \qquad \vdash\neg \frac{\Gamma, A \vdash \Delta}{\Gamma \vdash B, \Delta}\ B \xrightarrow[\mathcal{R}]{*} {}^{+}\neg A$$

$$\forall\vdash \frac{\Gamma, \{t/x\}A \vdash \Delta}{\Gamma, B \vdash \Delta}\ B \xrightarrow[\mathcal{R}]{*} {}^{-}\forall x.\ A \qquad \vdash\cdot\vee \frac{\Gamma \vdash A, B, \Delta}{\Gamma \vdash C, \Delta}\ \begin{array}{l} C \xrightarrow[\mathcal{R}]{*} {}^{+}A \\ C \xrightarrow[\mathcal{R}]{*} {}^{+}B \end{array}$$

Figure 2 Some inference rules of the Polarized Sequent Calculus Modulo $\mathcal{R}$

(*diff* can be seen as the Skolem symbol introduced by the CNF transformation of the definition of the subset relation.) We can build the following proof of the transitivity of the inclusion in the polarized sequent calculus modulo this system:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\vdash\frown \dfrac{}{\mathit{diff}(A,C) \in C \vdash A \subseteq C} \qquad \vdash\frown \dfrac{}{\mathit{diff}(A,C) \in B \vdash \mathit{diff}(A,C) \in B}}{\Rightarrow\vdash \dfrac{\mathit{diff}(A,C) \in B \Rightarrow \mathit{diff}(A,C) \in C, \mathit{diff}(A,C) \in B \vdash A \subseteq C}{\mathit{diff}(A,C) \in B, \mathit{diff}(A,C) \in B \vdash A \subseteq C}}}{\forall\vdash \dfrac{\mathit{diff}(A,C) \in B, \mathit{diff}(A,C) \in B \vdash A \subseteq C}{B \subseteq C, \mathit{diff}(A,C) \in B \vdash A \subseteq C}} \qquad \vdash\frown \dfrac{}{\mathit{diff}(A,C) \in A \vdash \mathit{diff}(A,C) \in A}}{\Rightarrow\vdash \dfrac{\mathit{diff}(A,C) \in A \Rightarrow \mathit{diff}(A,C) \in B, B \subseteq C, \mathit{diff}(A,C) \in A \vdash A \subseteq C}{\mathit{diff}(A,C) \in A \Rightarrow \mathit{diff}(A,C) \in B, B \subseteq C, \mathit{diff}(A,C) \in A \vdash A \subseteq C}}}{\forall\vdash \dfrac{\mathit{diff}(A,C) \in A \Rightarrow \mathit{diff}(A,C) \in B, B \subseteq C, \mathit{diff}(A,C) \in A \vdash A \subseteq C}{A \subseteq B, B \subseteq C, \mathit{diff}(A,C) \in A \vdash A \subseteq C}}}{\vdash\neg \dfrac{A \subseteq B, B \subseteq C, \mathit{diff}(A,C) \in A \vdash A \subseteq C}{A \subseteq B, B \subseteq C \vdash A \subseteq C, A \subseteq C}}}{\vdash\cdot\vee \dfrac{A \subseteq B, B \subseteq C \vdash A \subseteq C, A \subseteq C}{A \subseteq B, B \subseteq C \vdash A \subseteq C}}$$

To a rewriting system $\mathcal{R}$ corresponds a theory, which is the set of formulas that can be proved in the sequent calculus modulo $\mathcal{R}$. It was proved that this theory can always be presented by a traditional set of axioms, which is then called a compatible presentation [18]. In this paper, we are concerned with the converse direction: is it possible to present any axiomatic first-order theory by a rewriting system? In [11, Corollary 25], we answered positively: it is possible to transform any first-order theory into a rewriting system. However, this rewriting system may not have all the good properties that ensure that deduction modulo behaves well, in particular the admissibility of the cut rule.

The cut rule is admissible in the sequent calculus modulo $\mathcal{R}$ if, whenever a sequent can be proved in it, then it can be proved without using the cut rule ($\vdash\smile$ in Figure 1 and 2). Abusing terminology, we say that a rewriting system $\mathcal{R}$ admits cut if the cut rule is admissible in the sequent calculus modulo $\mathcal{R}$. The admissibility of the cut rule has a strong proof-theoretical as well as practical importance: it involves that normal forms exist for proofs; it implies the consistency of the theory associated to $\mathcal{R}$; it is equivalent to the completeness of the proof search procedures based on deduction modulo $\mathcal{R}$ (such as ENAR [18], extending the resolution method, and TaMed [4], extending the tableau method); etc. Cut admissibility can

also be seen as the completeness of the cut-free sequent calculus w.r.t. the sequent calculus with cuts. In [11], to ensure the cut admissibility, we designed a procedure that completes the rewriting system. However, this procedure may not terminate (and produces too many rules in practice). In this paper, we propose another method to transform an axiomatic presentation of a theory into a cut-admitting rewriting system, that works for any finitely presented and consistent first-order theory.

3 Resolution Calculi

We briefly recall the resolution calculus and two refinements, namely the set-of-support strategy and ordered resolution with selection, before presenting the extension of resolution with deduction modulo. A derivation in resolution [28] tries to refute a set of clauses by inferring new clauses by means of the two following inference rules (where P and Q are atoms, whereas L and K are literals), until the empty clause is derived.

$$\textsf{Resolution}\ \frac{P \vee C \qquad \neg Q \vee D}{\sigma(C \vee D)}\ \sigma = mgu(P,Q) \qquad\qquad \textsf{Factoring}\ \frac{L \vee K \vee C}{\sigma(L \vee C)}\ \sigma = mgu(L,K)$$

3.1 Set-of-Support Strategy

The set-of-support strategy for resolution [32] consists in restricting the clauses on which resolution can be applied. The input set of clauses is separated into a theory Γ and a set of support Δ. At least one of the clauses on which resolution is applied must be in the set of support, and the generated clause is put into the set of support. If the theory Γ is assumed to be consistent, this strategy is complete: if Γ, Δ is a unsatisfiable set of clauses, the empty clause can be derived from it using the set-of-support strategy. The set-of-support strategy can therefore be seen as proving a formula $\neg\Delta$ in a theory Γ without trying to find a contradiction in Γ because Γ is assumed to be consistent. In the following, we say that a set of clause Δ is refuted by the set-of-support strategy for Γ if the empty clause can be derived from the set Γ, Δ with initial set of support Δ and theory Γ.

3.2 Ordered Resolution with Selection

Ordered resolution with selection [3] (ORS($\succ, S$)) is another refinement of resolution parametrized by an Noetherian ordering $\succ$ on atoms which is stable under substitution and total on ground atoms, and by a selection function S that associates to each clause a subset of the negative literals of this clause. It consists in restricting the literals on which resolution can be applied: if $S(C)$ is not empty, then only the literals in $S(C)$ can be used; otherwise, only the maximal literals w.r.t. $\succ$ can be used. We will therefore say that a literal is selected in a clause C if it is in $S(C)$ or if $S(C)$ is empty and the literal is maximal in C. Ordered resolution with selection is refutationally complete whatever ordering or selection function are used.

3.3 ([Ordered] Polarized) Resolution Modulo

An extension of resolution based on deduction modulo, named Extended Narrowing and Resolution (ENAR), was defined in [18]. ENAR is a family of resolution calculi, each

parametrized by a rewriting system $\mathcal{R}$.[1] It consists in adding a new inference rule, called Extended Narrowing, which produces the clauses obtained by narrowing a clause by $\mathcal{R}$. Since narrowing a clause with a proposition rewriting rule can produce a formula which is not in clausal normal form, the latter has to be computed to find the generated clauses. The Extended Narrowing rule is therefore:

$$\text{Ext. Narr. } \frac{C}{D}\ C \underset{\mathcal{R}}{\rightsquigarrow} A,\ D \in \mathcal{C}\ell(A)$$

where $\mathcal{C}\ell(A)$ is the set of clauses of the clausal normal form of A.

We say that ENAR for $\mathcal{R}$ is complete if, whenever $\vdash A$ can be proved in the sequent calculus modulo $\mathcal{R}$, the empty clause can be derived from $\mathcal{C}\ell(\neg A)$ in ENAR for $\mathcal{R}$. Hermant [23] proved that the empty clause can be derived from $\mathcal{C}\ell(\neg A)$ in ENAR for $\mathcal{R}$ if and only if $\vdash A$ can be proved *without cut* in the sequent calculus modulo $\mathcal{R}$. This implies that ENAR for a rewriting system $\mathcal{R}$ is complete if and only if the sequent calculus modulo $\mathcal{R}$ admits cut.

In ENAR, formulas have to be put in clausal normal form dynamically, which may require fresh Skolem symbols each time. To avoid this, Dowek introduced the Polarized Resolution Modulo (PRM) [16]. As ENAR, this is a family of resolution calculi parametrized by a rewriting system, but this system is assumed to be polarized, and clausal, i.e., each negative rule is of the form $P \to^- C$, and each positive rule is of the form $P \to^+ \neg C$, where C is in clausal form. In that case, the Extended Narrowing rule becomes:

$$\text{Ext. Narr.}^- \ \frac{P \vee C}{\sigma(D \vee C)}\ \sigma = mgu(P, Q),\ Q \to^- D \in \mathcal{R}$$

$$\text{Ext. Narr.}^+ \ \frac{\neg Q \vee D}{\sigma(C \vee D)}\ \sigma = mgu(P, Q),\ P \to^+ \neg C \in \mathcal{R}$$

Gao proved that any rewriting system admitting cut can be transformed into an equivalent polarized and clausal one [22], so that PRM can be applied whenever ENAR can.

3.4 Polarized Rewriting Rules and One-Way Clauses

To each polarized clausal rewriting rule can be associated a clause in which one literal is selected. This clause is called a *one-way* clause [16]. For instance, to $P \to^- C$ is associated $\underline{\neg P} \vee C$, and to $P \to^+ \neg C$ is associated $\underline{P} \vee C$ (the selected literals are underlined). Conversely, to a clause and a literal occurrence in this clause can be associated a polarized clausal rewriting rule: to $\underline{P} \vee C$ is associated $P \to^+ \neg C$, and to $\underline{\neg P} \vee C$ is associated $P \to^- C$. It is worth remarking that applying Extended Narrowing on a clause C with a polarized clausal rule R leads to the same clause as applying Resolution on C and the one-way clause corresponding to R. Thus, polarized rewriting rules can be seen as special clauses with the following properties:

- only the selected literal can be used to resolve a one-way clause;
- two one-way clauses cannot be resolved together.

The results of this paper exploit this isomorphism between polarized clausal rewriting rules and one-way clauses.

---

[1] ENAR is originally parametrized by a rewriting system $\mathcal{R}$ and an equational theory $\mathcal{E}$, and the unification in the Resolution, Factoring and Extended Narrowing rules is performed modulo the equational theory $\mathcal{E}$. To keep it simple, we choose not to consider equational theories in this paper.

4 Cut-Admitting Presentations of Theories

4.1 Simulating set of support

We suppose that the theory is presented by means of a set of clauses. If not, it has to be transformed into clausal normal form using standard techniques.

▶ **Definition 3.** Given a set of clauses Γ, we define the polarized rewriting system $\mathcal{R}_\Gamma$ consisting of, for each clause C in Γ and each literal L in C,

- if $L = P$ is positive, a positive rewriting rule $P \to^+ \neg\forall x_1, \ldots, x_n.\ L_1 \vee \cdots \vee L_m$ where $x_1, \ldots, x_n$ are the free variables of C that are not free in P and $L_1, \ldots, L_m$ are the literals of C different from P;
- if $L = \neg P$ is negative, a negative rewriting rule $P \to^- \forall x_1, \ldots, x_n.\ L_1 \vee \cdots \vee L_m$ where $x_1, \ldots, x_n$ are the free variables of C that are not free in P and $L_1, \ldots, L_m$ are the literals of C different from $\neg P$.

▶ **Example 4.** Let Γ be the set of clauses corresponding to the definition of the inclusion:

$$\neg A \subseteq B \vee \neg(X \in A) \vee X \in B$$
$$A \subseteq B \vee \mathit{diff}(A, B) \in A$$
$$A \subseteq B \vee \neg(\mathit{diff}(A, B) \in B)$$

Then $\mathcal{R}_\Gamma$ is

$$A \subseteq B \to^- \forall x.\ \neg x \in A \vee x \in B$$
$$X \in A \to^- \forall b.\ \neg A \subseteq b \vee X \in b$$
$$X \in B \to^+ \neg\forall a.\ \neg a \subseteq B \vee X \in a$$
$$A \subseteq B \to^+ \neg \mathit{diff}(A, B) \in A$$
$$\mathit{diff}(A, B) \in A \to^+ \neg A \subseteq B$$
$$A \subseteq B \to^+ \neg\neg \mathit{diff}(A, B) \in B$$
$$\mathit{diff}(A, B) \in B \to^- A \subseteq B$$

▶ Remark. The number of rewriting rules in $\mathcal{R}_\Gamma$ is equal to the number of literal occurrences in Γ.

We then prove that the rewriting systems obtained as above enjoy cut admissibility. We follow three steps. The first one is the completeness of the set-of-support strategy:

▶ **Lemma 5.** *The consistency of a finite set of clauses Γ implies the completeness of the set-of-support strategy for Γ.*

Proof. This is the main theorem of [32]. ◀

The second step is to simulate the set-of-support strategy for Γ using PRM for $\mathcal{R}_\Gamma$:

▶ **Lemma 6.** *A derivation of the empty clause from a set of clauses Δ with the set-of-support strategy for Γ can be transformed into a derivation of the empty clause from a set of clauses Δ in PRM for $\mathcal{R}_\Gamma$.*

Proof. By induction on the length of the derivation. Note that Γ remains the same during the derivation, only the set of support Δ changes. Factoring steps are trivial, since they only involve clauses of Δ that can be factored also in PRM. If the first step resolves two clauses from the set of support (i.e. two clauses not in Γ), the same resolution step can be performed in PRM. If the first step is $\mathsf{Resolution}\ \dfrac{C \vee P \qquad D \vee \neg Q}{\sigma(C \vee D)}\ \sigma = mgu(P, Q)$ where

$D \vee \neg Q$ is in Γ, we know that there is a rule $Q \to^- \forall x_1, \ldots, x_n.\ D$ in $\mathcal{R}_\Gamma$. Therefore, we have the following derivation in PRM: $\mathsf{Ext.\ Narr.}^+ \dfrac{C \vee P}{\sigma(C \vee D)}\ \sigma = mgu(P, Q)$.

If the first step is $\mathsf{Resolution}\ \dfrac{C \vee \neg P \qquad D \vee Q}{\sigma(C \vee D)}\ \sigma = mgu(P, Q)$ where $D \vee Q$ is in Γ, we know that there is a rule $Q \to^+ \neg\forall x_1, \ldots, x_n.\ D$ in $\mathcal{R}_\Gamma$. Therefore, we have the following derivation in PRM: $\mathsf{Ext.\ Narr.}^- \dfrac{C \vee \neg P}{\sigma(C \vee D)}\ \sigma = mgu(P, Q)$. ◀

▶ **Corollary 7.** *The completeness of the set-of-support strategy for* Γ *implies the completeness of PRM for* $\mathcal{R}_\Gamma$*.*

The third step adapts an existing result on the equivalence of cut admissibility and completeness of standard resolution modulo to *polarized* resolution modulo.

▶ **Lemma 8.** *The completeness of PRM for* $\mathcal{R}_\Gamma$ *implies the admissibility of the cut rule in the polarized sequent calculus modulo* $\mathcal{R}_\Gamma$*.*

Proof. Either direct proof by adapting Hermant's one for unpolarized deduction modulo [23], or combination of the following lemmas. ◀

As in [11], Section 2.2, from the polarized rewriting system $\mathcal{R}_\Gamma$ we define the unpolarized rewriting system $\mathcal{R}_\Gamma^{\mp}$ consisting of:

- a rule $Q \to Q \vee \neg C$ for each positive rule $Q \to^+ \neg C$ in $\mathcal{R}_\Gamma$;
- a rule $Q \to Q \wedge C$ for each negative rule $Q \to^- C$ in $\mathcal{R}_\Gamma$.

The intuition behind the form of these unpolarized rules is the following: if the translation of a negative rule is applied at a positive position, then instead of Q we have to prove $Q \wedge C$. Since $Q \wedge C$ is stronger than Q, the application of the rewriting rule at a position of the "wrong" polarity is useless. On the contrary, at a negative position, instead of proving $\neg Q$ we have to prove $\neg(Q \wedge C)$ which is implied by $\neg C$. So it is as if we had rewritten $\neg Q$ to $\neg C$.

▶ **Lemma 9.** *A derivation of the empty clause from a set of clauses* Δ *in PRM for* $\mathcal{R}_\Gamma$ *can be transformed into a derivation of the empty clause from a set of clauses* Δ *in ENAR for* $\mathcal{R}_\Gamma^{\mp}$*.*

Proof. By induction on the derivation length, the only interesting case is Extended Narrowing.

Suppose that we have $\mathsf{Ext.\ Narr.}^- \dfrac{P \vee C}{\sigma(D \vee C)}\ \sigma = mgu(P, Q),\ Q \to^- D$. To $Q \to^- D$ corresponds the unpolarized rule $Q \to Q \wedge D$. Hence, $P \vee C$ can be narrowed to $\sigma((Q \wedge D) \vee C)$, whose clausal normal form is $(\sigma(Q \vee C)) \wedge (\sigma(D \vee C))$. Hence, the Extended Narrowing rule of ENAR can infer the clause $\sigma(D \vee C)$.

Suppose that we have $\mathsf{Ext.\ Narr.}^+ \dfrac{\neg P \vee C}{\sigma(D \vee C)}\ \sigma = mgu(P, Q),\ Q \to^+ \neg D$ To $Q \to^+ \neg D$ corresponds the unpolarized rule $Q \to Q \vee \neg D$. Hence, $\neg P \vee C$ can be narrowed to $\sigma(\neg(Q \vee \neg D) \vee C)$, whose clausal normal form is $(\sigma(\neg Q \vee C)) \wedge (\sigma(D \vee C))$. Hence, the Extended Narrowing rule of ENAR can infer the clause $\sigma(D \vee C)$. ◀

▶ **Corollary 10.** *The completeness of PRM for* $\mathcal{R}_\Gamma$ *implies the completeness of ENAR for* $\mathcal{R}_\Gamma^{\mp}$*.*

▶ **Lemma 11.** *The completeness of ENAR for* $\mathcal{R}_\Gamma^{\mp}$ *implies the admissibility of the cut rule in the asymmetric sequent calculus modulo* $\mathcal{R}_\Gamma^{\mp}$*.*

Proof. This is a corollary of Theorems 1 and 2 of [23]. ◀

▶ **Lemma 12.** *The admissibility of the cut rule in the asymmetric sequent calculus modulo* $\mathcal{R}_\Gamma^{\mp}$ *implies the admissibility of the cut rule in the polarized sequent calculus modulo* $\mathcal{R}_\Gamma$*.*

Proof. This is a direct consequence of the equivalence theorem between the polarized sequent calculus modulo $\mathcal{R}_\Gamma$ and the asymmetric sequent calculus modulo $\mathcal{R}_\Gamma^{\mp}$ (Corollary 10 of [11]): a sequent is provable (resp. provable without cut) in the polarized sequent calculus modulo a polarized proposition rewriting system $\mathcal{R}$ iff it is provable (resp. provable without cut) in the asymmetric sequent calculus modulo the rewriting system $\mathcal{R}^{\mp}$. ◀

By combining Lemma 5, Corollary 7, and Lemma 8, we obtain:

▶ **Theorem 13.** *The consistency of a finite set of clauses* Γ *implies the admissibility of the cut rule in the polarized sequent calculus modulo* $\mathcal{R}_\Gamma$*.*

4.2 Cut admissibility through saturation

To reduce the number of rules, it is possible to associate a polarized rewriting system to a set of clauses for ordered resolution with selection by considering as left-hand sides only the literals that are selected in a clause. Thus, we would not produce a rule for each literal but only for those that are in $S(C)$ or that are maximal if $S(C)$ is empty.

▶ **Example 14.** We consider the example of the inclusion again, with an ordering such that literals with $\subseteq$ are greater than literals with $\in$. The resulting rewriting system is reduced to

$$A \subseteq B \to^{-} \forall x.\ \neg x \in A \vee x \in B$$
$$A \subseteq B \to^{+} \neg \mathit{diff}(A, B) \in A$$
$$A \subseteq B \to^{+} \neg\neg \mathit{diff}(A, B) \in B$$

However, ordered resolution with selection is not compatible with the set-of-support strategy, in the sense that their combination jeopardizes completeness. Therefore, we cannot reuse the proof above and indeed, the rewriting system corresponding to the clauses may not admit cut. Nevertheless, a sufficient condition to ensure the completeness is the saturation of the set of clauses used as complement of the set of support (i.e. the theory): the clauses that can be inferred from it must either be in it or be redundant (i.e. they must be semantically implied by smaller clauses). We can redo the proof above, replacing consistency by saturation and using the set-of-support strategy for $\mathrm{ORS}(\succ, S)$ instead of resolution. Lemma 5 becomes:

▶ **Lemma 15.** *The saturation of a finite set of clauses* Γ *w.r.t. ORS(*$\succ, S$*) implies the completeness of the set-of-support strategy for* Γ *in ORS(*$\succ, S'$*), where* S' *is* S *on clauses of* Γ *and maps the empty set to other clauses.*

Proof. Since Γ is saturated, no relevant inference can be done among its clauses. Since $\mathrm{ORS}(\succ, S')$ is complete, so is the set-of-support strategy for it, since it only prevents inferences among clauses of Γ. ◀

Then, Corollary 7 is replaced by:

▶ **Lemma 16.** *The completeness of the set-of-support strategy for* Γ *in ORS(*$\succ, S'$*) implies the completeness of PRM for* $\mathcal{R}'_\Gamma$ *where* $\mathcal{R}'_\Gamma$ *is* $\mathcal{R}_\Gamma$ *restricted to the rules whose left-hand side is selected in the corresponding clause for ORS(*$\succ, S'$*).*

Proof. As in proof of Lemma 6, a derivation of the empty clause in $\mathrm{ORS}(\succ, S')$ with the set-of-support strategy for Γ can be simulated by a derivation in PRM for $\mathcal{R}'_\Gamma$. ◀

Combining Lemmas 15, 16 and 8, we obtain:

▶ **Theorem 17.** *The saturation of a finite set of clauses Γ w.r.t. ORS($\succ, S$) implies the admissibility of the cut rule in the polarized sequent calculus modulo the rewrite system $\mathcal{R}_\Gamma$ restricted to rules whose left-hand side is selected (or maximal if none are selected) w.r.t. S and $\succ$.*

Saturating a set of clauses is undecidable, but it can be semi-automated. First-order automated theorem provers like SPASS [31] actually work by trying to saturate the input set of clauses, unless the empty clause is derived. Running SPASS on the example above (with precedence $\subseteq > \in > \mathit{diff}$ and $\subseteq$ and $\in$ dominant predicates), the saturation generates two new clauses

$$\underline{\neg X \in A} \vee \mathit{diff}(A, B) \in A \vee X \in B$$
$$\underline{\neg \mathit{diff}(A, B) \in B} \vee \neg X \in A \vee X \in B$$

The following rewriting system therefore admits cuts:

$$\begin{aligned} A \subseteq B &\to^- \forall x.\ \neg x \in A \vee x \in B \\ A \subseteq B &\to^+ \neg \mathit{diff}(A, B) \in A \\ A \subseteq B &\to^+ \neg\neg \mathit{diff}(A, B) \in B \\ X \in A &\to^- \forall x.\ \mathit{diff}(A, B) \in A \vee x \in B \\ \mathit{diff}(A, B) \in B &\to^- \forall x.\ \neg X \in A \vee X \in B \end{aligned}$$

5 Experimental Comparison of Orientation Techniques

We have compared several techniques that transform a set of axioms into a rewriting system. Two of them, being based on Theorems 13 and 17, are therefore proved to be complete, in the sense that the resulting rewriting system admits cut. The other ones are merely heuristics.

5.1 Description of the different techniques

We compared six different ways of transforming a set of axioms into a rewriting system.

ClausalAll: The set of axioms is put in clausal normal form (using E [29]), and it is transformed into a rewriting system as described in Definition 3.

Sat: The set of axioms is saturated using E, and it is then transformed into a rewriting system restricted to the selected literals, as in Theorem 17. Of course, the saturation may not terminate, so this technique does not always succeed.

Equiv(ClausalAll): Depending on their shape, axioms are transformed into rewriting rules:

Axioms of the form	are translated into	Axioms of the form	are translated into
$\forall x_1, \ldots, x_n.\ (P \Leftrightarrow A)$	$P \to^\pm A$	$\forall x_1, \ldots, x_n.\ (\neg P \Leftrightarrow A)$	$P \to^\pm \neg A$
$\forall x_1, \ldots, x_n.\ (A \Leftrightarrow P)$	$P \to^\pm A$	$\forall x_1, \ldots, x_n.\ (A \Leftrightarrow \neg P)$	$P \to^\pm \neg A$
$\forall x_1, \ldots, x_n.\ (P \Rightarrow A)$	$P \to^- A$	$\forall x_1, \ldots, x_n.\ (\neg P \Rightarrow A)$	$P \to^+ \neg A$
$\forall x_1, \ldots, x_n.\ (A \Rightarrow P)$	$P \to^+ \neg\neg A$	$\forall x_1, \ldots, x_n.\ (A \Rightarrow \neg P)$	$P \to^- \neg A$
$\forall x_1, \ldots, x_n.\ P$	$P \to^+ \neg\bot$	$\forall x_1, \ldots, x_n.\ \neg P$	$P \to^- \bot$

where P is atomic. Then, these potentially non-clausal rewriting rules are transformed into a clausal rewriting system using Gao's approach [22]. Intuitively, their right-hand sides are put in CNF (one rule can therefore leads to several rules), using E. Axioms not of these shapes are transformed using the ClausalAll technique above.

Presat(ClausalAll): Since saturation may not terminate, we can decide to stop it after a certain amount of clauses have been generated. In this technique, we saturate the set of axioms using E until 1000 clauses have been processed. These processed clauses are transformed into a rewriting system restricted to the selected literals as in Sat. The unprocessed clauses generated during the saturation are transformed using the ClausalAll technique.

Presat(Id): This is the same technique as Presat(ClausallAll), except that unprocessed clauses are not transformed into rewrite rules and are used as normal clauses.

Presat(Nil): This is the same technique as Presat(ClausallAll), except that unprocessed clauses are thrown away.

We have implemented a tool that automatically transforms the set of axioms of a problem into a rewriting system using one of these techniques. It is available on `http://www.ensiie.fr/~guillaume.burel/empty_autotheo.html.en`.

5.2 Experiment

We compared these techniques on the set of FOF problems of last-year CASC (`http://www.cs.miami.edu/~tptp/CASC/J6/`), consisting of 450 problems in first-order logic. For each problem, we first transformed the set of axioms of the problem (the formulas whose TPTP role is indicated as axiom) into a rewriting system thanks to each of the techniques above, with a timeout of 30s. We then used iProver modulo, our integration of PRM into iProver [8], to search for proofs, with a timeout of 300s. Since completeness is not ensured for some of the techniques, in some cases iProver modulo terminates without finding a proof. Because rewriting rules only affect the resolution calculus of iProver Modulo, and not the Inst-Gen calculus of iProver [26], we turned the latter one off to better quantify the effect of the rewriting systems. We therefore compared our results to the original iProver with the Inst-Gen calculus turned off, but also to iProver (v0.7, the same as in iProver modulo) with default options (therefore with Inst-Gen turned on) and to E v1.4, to have state-of-the-art results. Experiments were performed on an Intel Core i3 CPU @ 2.13GHz with 3GB of RAM.

The results are summarized in Table 1. "Transformed theories" indicates the number of problems whose axioms could be transformed into a rewriting system within the given timeout. "Theorem" indicates the number of problems for which a proof was found; "No proof" to the problems where the prover terminated without finding a proof; "Unknown" to the problems for which the prover could not find a proof with the given resources (timeout or memory limit outreached). Note that even for complete techniques, iProver modulo may terminate without finding a proof, because of the way it handles equalities. Indeed, on a problem with equalities, iProver Modulo adds a set of rewriting rules that define the equality predicate in the current signature, but that may not be compatible with the other rewriting rules. "Average time" is the average CPU time over all solved problems. "SOTAC" (state-of-the-art contribution) is the average SOTAC over all solved problems, where the SOTAC of a problem is the inverse of the number of systems that solved the problem in last CASC plus one. Of course, since the settings are different, this

Table 1 Comparison of transformation techniques

	Transformed theories	Result			Average time	SOTAC
		Theorem	No proof	Unknown		
ClausalAll	419	142	11	266	17.28	0.0917
Sat	71	37	6	28	18.24	0.0877
Equiv(ClausalAll)	419	127	38	254	17.26	0.1053
Presat(ClausalAll)	374	109	12	253	19.12	0.0921
Presat(Id)	417	113	7	297	28.41	0.0912
Presat(Nil)	417	39	241	137	15.69	0.0924
iProver w/o IG	n.r.	129	0	321	9.41	0.0895
iProver	n.r.	228	0	222	20.15	0.1043
E	n.r.	339	0	111	23.00	0.1156

SOTAC cannot be compared with the SOTAC of last CASC, but it gives an indication to compare the techniques among themselves. Detailed results for all problems can be found on `http://www.ensiie.fr/~guillaume.burel/empty_autotheo.html.en`.

As expected, the orientation of the axioms using the Sat technique only succeeded in a few cases (16%). However, proofs for half of the problems were found thereafter. Presat(ClausalAll) also failed more than the other techniques, because processing 1000 clauses can generate a lot of unprocessed clauses, and ClausalAll produces a clause for each *literal* of these clauses, so that the time limit was reached before each was achieved. Problems that could not be oriented by any of the techniques were problems with too many axioms to be processed within the time limit, typically problems including file `CSR003+2.ax` which contains 55592 axioms.

The technique giving the best results is ClausalAll, mostly because the other techniques terminate more often without finding a proof because of their incompleteness; this is the case in particular for Presat(Nil), which was expected because a lot of information can be lost when the unprocessed clauses are dropped. Compared to iProver without Inst-Gen restricted on problems whose orientation succeeded, mainly ClausalAll and Equiv(ClausalAll) really perform better. In particular, the results for ClausalAll implies that it is better to do no selection at all on the clauses of the theory and to use the set-of-support strategy, than to select literals and to have to abandon this strategy to guarantee completeness.

An interesting result is the SOTAC of Equiv(ClausalAll), which is higher than the one of iProver (which is biased against iProver, since it participated in CASC, and is therefore counted twice). Equiv(ClausalAll) permits in particular to solve some problems that where not solved by E with the same conditions. Notably, this technique allows iProver modulo to prove many problems among `SCT???+?.p` and `SWW???+?.p` which are, according to their header, most likely generated by Sledgehammer. This can be explained by the fact that Sledgehammer adds a lot of related lemmas as axioms of the goal to be solved. These lemmas, often of the form $\forall x_1, \ldots, x_n.\ \mathit{Atomic\_hypothesis} \Rightarrow A$, can in most cases be oriented. There are two advantages: First, lemmas are not resolved with one another, what probably avoids to generate clauses that are not relevant for the given goal. Second, a lemma is only triggered when the *Atomic_hypothesis* above can be unified with a literal of a given clause, what possibly prevents a lot of pointless lemmas to be used. This suggests that orienting axioms into rewriting systems can be of real help in the context of tools such as Sledgehammer that links a goal to be proved with (maybe) relevant lemmas: not only is the choice of lemmas

critical, but also the way they are handled thereafter to search for the proof.

6 Conclusion and Further Work

In this paper, we have given two methods to present *any* first-order theory as a rewriting system admitting cut. We have presented the results of experiments comparing several techniques derived from these methods. They show that even if the first technique can lead to big rewriting systems, it is nonetheless better to employ it than to use axioms and to rely on ordered selection to reduce the search space. All the more, it allows to prove problems not solved by state-of-the-art provers. This work therefore constitutes a strong basis to the automatic transformation of theories into rewriting systems that can be used in tools based on deduction modulo, such as the integration of PRM into iProver (`http://www.ensiie.fr/~guillaume.burel/empty_iProverModulo.html.en`) or Dedukti, a proof checker for deduction modulo (`https://www.rocq.inria.fr/deducteam/Dedukti/`). It therefore constitutes an important step towards the automatic production of provers adapted to a specific theory. The experiments show that it is notably the case for theories represented by a set of lemmas chosen to search for a proof in a larger theory, as done by Sledgehammer. This work also strengthens the use of deduction modulo as a universal framework to handle proofs. Indeed, since simple type theory and pure type systems can be presented in deduction modulo, Dedukti can be used to check proofs produced by different proof assistant such as HOL and Coq. This paper shows that it can also be used for proofs built within a theory. All this opens many questions that we are now considering.

Logical Strength. In this paper, we show that the consistency of a theory implies the cut admissibility of (a presentation of) it. Since cut admissibility also implies consistency, Gödel's second incompleteness theorem implies that cut admissibility cannot be proved in the theory itself. But we may wonder whether it can be proved in the theory plus the assumption of its consistency. To this purpose, we have to investigate the proof of Section 4.1. We conjecture that "consistency implies cut admissibility" can be proved in first-order arithmetic, that is, that cut admissibility is not logically stronger than consistency.

Equality. In this paper, we only considered theories of first-order logic without equality. However, theories are often presented in first-order logic with equality. Adding the axioms for equality (reflexivity, symmetry, transitivity and congruence w.r.t. the function symbols and the predicates) and transforming them as presented in this paper is a theoretical way to obtain presentations of such theories. However, it does not take into account the specificity of equality, and the way it can be integrated into a deduction system thanks to deduction modulo. A first improvement is to put the equational axioms into an equational theory modulo which rewriting and unification is performed (see footnote page 6). Nevertheless, existing provers perform unification and rewriting modulo only for specific equational theories, such as commutativity of a function symbol. Only such axioms should therefore be presented this way. The other equational axioms should be transformed into *term* rewriting rules. It remains to be proved that using term rewriting rules for equational axioms and proposition rewriting rules as obtained as in this paper for the other axioms is complete. We conjecture that it is the case as long as the term rewriting system is confluent and commutes with the proposition rewriting system. The confluence of the term rewriting system can be ensured by the standard completion of Knuth and Bendix [25].

The next step is to design proof-search procedures based on deduction modulo for first-order logic with equality. A good candidate would be an extension of the superposition calculus [2] with an Extended Narrowing rule, but we currently do not know if cut admissibility is enough to prove its completeness. In particular, we do not know how to take this assumption into account in a completeness proof based on saturation, the kind of proof usually used for the completeness of superposition.

Axiom Schemata. This paper only considers finite theories, but usual theories, such as for instance arithmetic, use axiom schemata. A way to handle such theories is to consider the work of Kirchner [24] who transforms an axiom schema into a finite number of axioms, most of them being directly orientable into rewriting rules.

Combination with other kinds of presentations. In this paper, we have shown how to present any first-order theories as rewriting systems. However, for some specific theories, rewriting is probably not the best way to present them. For instance, to search for proofs in linear arithmetic, it is probably more efficient to use a combination with the simplex method than to use a rewriting system for linear arithmetic. Therefore, we would like to investigate how it could be possible to combine deduction modulo with other ways to present theories. A first lead would be to study canonized rewriting [13], where (ground) rewriting is combined with Shostak theories to get SMT solvers modulo AC. Then, we would need a way to recognize theories during proof search to trigger the most appropriate method [9].

Termination. Cut admissibility is not the only property of interest for a rewriting system. Termination is another good requirement, since it implies for instance the decidability of proof checking in the case of deduction modulo. However, note that even if rewriting terminates, narrowing may not, so that it seems less important for proof search. The systems produced by this paper's method may not terminate in general. We have to investigate if we can restrict the number of rules to ensure the termination of the rewriting system as well as its cut admissibility. In the same line of work, we should investigate whether we can obtain rewriting systems that provide decision procedures for some theories.

Intuitionistic Logic. Since it is based on resolution, the method described in this paper only works for classical logic. In intuitionistic logic, it is known that some theories cannot be transformed into a rewriting system with cut admissibility. In [5], we have proposed a procedure inspired from our work in [11] that is able to transform a large class of intuitionistic theories into a rewriting system admitting cuts. Since it is undecidable if such a transformation is possible, the procedure is of course non-terminating. We need to investigate whether the method proposed here can improve the transformation of intuitionistic theories, but it does not seem plausible.

References

1 Franz Baader and Tobias Nipkow. *Term* Re*writing and* all That. Cambridge University Press, 1998.

2 L. Bachmair and H. Ganzinger. Rewrite-based equational theorem proving with selection and simplification. *Journal of Logic and Computation*, 4(3):1–31, 1994.

3 Leo Bachmair and Harald Ganzinger. Resolution theorem proving. In *Handbook of Automated Reasoning*, pages 19–99. Elsevier and MIT Press, 2001.

4 Richard Bonichon and Olivier Hermant. A semantic completeness proof for TaMed. In Miki Hermann and Andrei Voronkov, editors, *LPAR*, volume 4246 of *LNCS*, pages 167–181. Springer, 2006.

5 Guillaume Burel. Automating theories in intuitionistic logic. In Silvio Ghilardi and Roberto Sebastiani, editors, *FroCoS*, volume 5749 of *LNAI*, pages 181–197. Springer, 2009.

6 Guillaume Burel. Embedding deduction modulo into a prover. In Anuj Dawar and Helmut Veith, editors, *CSL*, volume 6247 of *LNCS*, pages 155–169. Springer, 2010.

7 Guillaume Burel. Efficiently simulating higher-order arithmetic by a first-order theory modulo. *Logical Methods in Computer Science*, 7(1:3):1–31, 2011.

8 Guillaume Burel. Experimenting with deduction modulo. In Viorica Sofronie-Stokkermans and Nikolaj Bjørner, editors, *CADE 2011*, volume 6803 of *LNAI*, pages 162–176. Springer, 2011.

9 Guillaume Burel and Simon Cruanes. Detection of first-order axiomatic theories. Submitted, available on the author's webpage, 2013.

10 Guillaume Burel and Gilles Dowek. How can we prove that a proof search method is not an instance of another? In *LFMTP'09*, ACM International Conference Proceeding Series, pages 84–87. ACM, 2009.

11 Guillaume Burel and Claude Kirchner. Regaining cut admissibility in deduction modulo using abstract completion. *Information and Computation*, 208(2):140–164, 2010.

12 Agata Ciabattoni, Nikolaos Galatos, and Kazushige Terui. From axioms to analytic rules in nonclassical logics. In Frank Pfenning, editor, *LICS*. IEEE Computer Society, 2008.

13 Sylvain Conchon, Évelyne Contejean, and Mohamed Iguernelala. Canonized rewriting and ground AC completion modulo Shostak theories. In Parosh A. Abdulla and K. Rustan M. Leino, editors, *TACAS 2011*, LNCS. Springer, 2011.

14 Gilles Dowek. What is a theory? In Helmut Alt and Afonso Ferreira, editors, *STACS*, volume 2285 of *LNCS*, pages 50–64. Springer, 2002.

15 Gilles Dowek. Confluence as a cut elimination property. In Robert Nieuwenhuis, editor, *RTA*, volume 2706 of *LNCS*, pages 2–13. Springer, 2003.

16 Gilles Dowek. Polarized resolution modulo. In Cristian S. Calude and Vladimiro Sassone, editors, *IFIP TCS*, volume 323 of *IFIP AICT*, pages 182–196. Springer, 2010.

17 Gilles Dowek, Thérèse Hardin, and Claude Kirchner. HOL-$\lambda\sigma$ an intentional first-order expression of higher-order logic. *Mathematical Structures in Computer Science*, 11(1):1–25, 2001.

18 Gilles Dowek, Thérèse Hardin, and Claude Kirchner. Theorem proving modulo. *Journal of Automated Reasoning*, 31(1):33–72, 2003.

19 Gilles Dowek and Alexandre Miquel. Cut elimination for Zermelo's set theory. Available on authors' web page, 2006.

20 Gilles Dowek and Benjamin Werner. Arithmetic as a theory modulo. In Jürgen Giesl, editor, *RTA*, volume 3467 of *LNCS*, pages 423–437. Springer, 2005.

21 Jean H. Gallier. *Logic for Computer Science: Foundations of Automatic Theorem Proving*, volume 5 of *Computer Science and Technology Series*. Harper & Row, New York, 1986.

22 Jianhua Gao. Clausal presentation of theories in deduction modulo. In *International Workshop on Proof-Search in Axiomatic Theories and Type Theories 2011*, 2011. `http://hal.inria.fr/inria-00614251/en/`.

23 Olivier Hermant. Resolution is cut-free. *Journal of Automated Reasoning*, 44(3):245–276, 2009.

24 Florent Kirchner. A finite first-order theory of classes. In Thorsten Altenkirch and Conor McBride, editors, *TYPES*, volume 4502 of *LNCS*, pages 188–202. Springer, 2006.

25 Donald E. Knuth and P. B. Bendix. Simple word problems in universal algebras. In J. Leech, editor, *Computational Problems in Abstract Algebra*, pages 263–297. Pergamon Press, Oxford, 1970.

26 Konstantin Korovin. iProver – an instantiation-based theorem prover for first-order logic (system description). In Alessandro Armando and Peter Baumgartner, editors, *IJCAR*, volume 5195 of *LNAI*, pages 292–298. Springer, 2008.

27 Sara Negri and Jan von Plato. Cut elimination in the presence of axioms. *The Bulletin of Symbolic Logic*, 4(4):418–435, 1998.

28 J. A. Robinson. A machine-oriented logic based on the resolution principle. *Journal of the ACM*, 12:23–41, 1965.

29 Stephen Schultz. System description: E 0.81. In David A. Basin and Michaël Rusinowitch, editors, *IJCAR*, volume 3097 of *LNCS*, pages 223–228. Springer, 2004.

30 Geoff Sutcliffe. The TPTP Problem Library and Associated Infrastructure: The FOF and CNF Parts, v3.5.0. *Journal of Automated Reasoning*, 43(4):337–362, 2009.

31 C. Weidenbach, D. Dimova, A. Fietzke, R. Kumar, M. Suda, and P. Wischnewski. SPASS version 3.5. In Renate A. Schmidt, editor, *CADE*, volume 5663 of *LNCS*, pages 140–145. Springer, 2009.

32 Larry Wos, George A. Robinson, and Daniel F. Carson. Efficiency and completeness of the set of support strategy in theorem proving. *J. ACM*, 12(4):536–541, 1965.