Herbrand Sequent Extraction [*]

Stefan Hetzl[1], Alexander Leitsch[1], Daniel Weller[1], and Bruno Woltzenlogel
Paleo[1]

{hetzl, leitsch, weller, bruno}@logic.at
Institute of Computer Languages (E185),
Vienna University of Technology,
Favoritenstraße 9, 1040 Vienna, Austria

Abstract. Computer generated proofs of interesting mathematical the-
orems are usually too large and full of trivial structural information,
and hence hard to understand for humans. Techniques to extract spe-
cific essential information from these proofs are needed. In this paper we
describe an algorithm to extract Herbrand sequents from proofs writ-
ten in Gentzen's sequent calculus **LK** for classical first-order logic. The
extracted Herbrand sequent summarizes the creative information of the
formal proof, which lies in the instantiations chosen for the quantifiers,
and can be used to facilitate its analysis by humans. Furthermore, we
also demonstrate the usage of the algorithm in the analysis of a proof of
the equivalence of two different definitions for the mathematical concept
of lattice, obtained with the proof transformation system `CERES`.

1 Introduction

Within mathematical knowledge management, the problem of analyzing and un-
derstanding computer generated proofs plays a fundamental role and its impor-
tance can be expected to grow, as automated and interactive deduction methods
and computer processing power improve. Such computer generated proofs are
formal, in the sense that they strictly follow *axioms* and *rules of inference* of
formal logical calculi, as Hilbert calculi, natural deduction calculi or sequent
calculi. The main advantages of formal proofs are:

- Formal proofs, when viewed and studied as a model and ideal for informal
  mathematical proofs, allow meta-mathematical investigations into the foun-
  dations of Mathematics.
- The correctness of formal proofs can be easily checked, by verifying whether
  the formal axioms and rules of the calculus were correctly employed.
- Formal proofs for formalized statements (formulas) can be constructed by
  computers executing automated or interactive theorem provers [15].
- Automated proof transformations can be employed to obtain new formal
  proofs from previously existing ones [2, 3]. Subsequently, the analysis and

interpretation of the new formal proofs might lead to the discovery of new informal proofs of the original theorems containing interesting mathematical arguments.

However, formal representations of real mathematical proofs or computer generated proofs of real mathematical problems usually have some drawbacks that make them difficult to be analyzed and understood by mathematicians. Firstly, the size of a formal proof is usually huge ([1]), which makes it hard to be visualized as a whole. Secondly, many of its individual inferences are only structural, necessary not to carry some essential idea about the proof, but only to satisfy the formalities of the calculus. Thirdly, inference rules of proof calculi not always correspond easily to natural inferences in informal proofs. Together these drawbacks imply that, given a formal proof, it is not easy for humans to understand its essential idea, because it is hidden in a large data structure of repetitive, bureaucratic and non-intuitive formalities. Therefore there is a need for summarization of formal proofs or for extraction of its hidden crucial information, whenever these proofs are intended to be analyzed and understood by humans. This need has become especially clear to us during the development and use of our automated proof transformation system CERES[1] for the cut-elimination of real mathematical proofs in classical first-order logic [2].

This paper describes one possible technique that helps to overcome these difficulties in the particular case of first-order logic. Our technique relies on the concept of Herbrand sequent, a generalization of Herbrand disjunction [10], which can be used to summarize the creative content of first-order formal proofs [13], which lies in the instantiations chosen for quantified variables. Although we use sequent calculi, the idea described here could be adapted to other calculi, since it relies on a general property of first-order logic, as stated by Herbrand's theorem, and not on specific features of particular calculi.

After describing the technique, we demonstrate its use with the analysis of a computer generated proof of the equivalence of two different lattice definitions.

2 The Sequent Calculus LKDe

Our formal proofs are written in an extension of Gentzen's sequent calculus **LK**, which is called **LKDe** and has the following additional features:

- Arbitrary but pre-defined atomic formulas are allowed in the axioms. This has the advantage that typical mathematical axioms (e.g. symmetry and reflexivity of equality, associativity of addition, . . .) do not need to be carried along all the formal proof in the antecedents of the sequents, but can instead appear simply as non-tautological axiom sequents in the leaf nodes of the proof. On the other hand, Gentzen's cut-elimination theorem [8] holds in this calculus only in a modified form, since atomic cuts are not necessarily eliminable.

---

[1] CERES Website: http://www.logic.at/ceres

– There are additional rules for equality and mathematical definitions, in order to make the calculus more comfortable to use in the formalization of real mathematical proofs, which use equality and definitions of concepts very often.

A partial description of the sequent calculus **LKDe** follows. A full description can be found in [3]. Additionally, by **LKe** we denote the **LKDe** calculus without definition rules.

Definition 1 (Sequent). *A* sequent *is a pair* $A_1, \ldots, A_n \vdash C_1, \ldots, C_m$ *of sequences of first-order logic formulas. The first sequence,* $A_1, \ldots, A_n$, *is the* antecedent *of the sequent and the second sequence,* $C_1, \ldots, C_m$, *is the* consequent *of the sequent. We use the symbols* Γ, Π, Λ *and* Δ, *possibly with subscripts, to denote sequences of formulas in the antecedent and consequent of sequents.*

1. **The Axioms:** We allow arbitrary atomic sequents as axioms. The logical axioms are of the form $A \vdash A$ for A atomic. For equality we use the reflexivity axiom scheme $\vdash t = t$ for all terms t.
2. **Propositional rules: LKDe** has rules for the propositional connectives: $\vee$, $\rightarrow$, $\neg$ and $\wedge$, as exemplified below:

$$\frac{\Gamma \vdash \Delta, A \quad \Pi \vdash \Lambda, B}{\Gamma, \Pi \vdash \Delta, \Lambda, A \wedge B} \wedge : r \qquad \frac{A, \Gamma \vdash \Delta}{A \wedge B, \Gamma \vdash \Delta} \wedge : l1 \qquad \frac{A, \Gamma \vdash \Delta}{B \wedge A, \Gamma \vdash \Delta} \wedge : l2$$

3. **First-order rules: LKDe** has rules for the existential ($\exists$) and universal ($\forall$) quantifiers.

$$\frac{A\{x \leftarrow t\}, \Gamma \vdash \Delta}{(\forall x)A, \Gamma \vdash \Delta} \forall : l \qquad \frac{\Gamma \vdash \Delta, A\{x \leftarrow \alpha\}}{\Gamma \vdash \Delta, (\forall x)A} \forall : r$$

$$\frac{A\{x \leftarrow \alpha\}, \Gamma \vdash \Delta}{(\exists x)A, \Gamma \vdash \Delta} \exists : l \qquad \frac{\Gamma \vdash \Delta, A\{x \leftarrow t\}}{\Gamma \vdash \Delta, (\exists x)A} \exists : r$$

The $\forall : r$ and $\exists : l$ rules must satisfy the eigenvariable condition: the variable α must not occur in Γ nor in Δ nor in A. Quantifiers introduced by them are called *strong quantifiers*. For the $\forall : l$ and the $\exists : r$ rules the term t must not contain a variable that is bound in A. Quantifiers introduced by them are called *weak quantifiers*.

4. **Equality rules:**

$$\frac{\Gamma \vdash \Delta, s = t \quad \Pi \vdash \Lambda, A[s]_\Xi}{\Gamma, \Pi \vdash \Delta, \Lambda, A[t]_\Xi} = (\Xi) : r1 \qquad \frac{\Gamma \vdash \Delta, s = t \quad A[s]_\Xi, \Pi \vdash \Lambda}{A[t]_\Xi, \Gamma, \Pi \vdash \Delta, \Lambda} = (\Xi) : l1$$

$$\frac{\Gamma \vdash \Delta, t = s \quad \Pi \vdash \Lambda, A[s]_\Xi}{\Gamma, \Pi \vdash \Delta, \Lambda, A[t]_\Xi} = (\Xi) : r2 \qquad \frac{\Gamma \vdash \Delta, t = s \quad A[s]_\Xi, \Pi \vdash \Lambda}{A[t]_\Xi, \Gamma, \Pi \vdash \Delta, \Lambda} = (\Xi) : l2$$

where Ξ is a set of positions in A and s and t do not contain variables that are bound in A.

5. **Structural rules:** weakening, contraction and permutation, as well as the following cut-rule:

$$\frac{\Gamma \vdash \Delta, A \quad A, \Pi \vdash \Lambda}{\Gamma, \Pi \vdash \Delta, \Lambda} cut$$

6. **Definition rules:** They correspond directly to the *extension principle* in predicate logic and introduce new predicate and function symbols as abbreviations for formulas and terms. Let A be a first-order formula with the free variables $x_1, \ldots, x_k$,denoted by $A(x_1, \ldots, x_k)$, and P be a *new k-ary* predicate symbol (corresponding to the formula A). Then the rules are:

$$\frac{A(t_1, \ldots, t_k), \Gamma \vdash \Delta}{P(t_1, \ldots, t_k), \Gamma \vdash \Delta} \ d:l \qquad \frac{\Gamma \vdash \Delta, A(t_1, \ldots, t_k)}{\Gamma \vdash \Delta, P(t_1, \ldots, t_k)} \ d:r$$

for arbitrary sequences of terms $t_1, \ldots, t_k$. Definition introduction is a simple and very powerful tool in mathematical practice, allowing the easy introduction of important concepts and notations (e.g. groups, lattices, ...) by the introduction of new symbols.

Definition 2 (Skolemization). *The skolemization of a sequent removes all its strong-quantifiers and substitutes the corresponding variables by* skolem-terms *in a validity preserving way (i.e. the skolemized sequent is valid iff the original sequent is valid).* **LKDe***-proofs can also be skolemized, as described in [4], essentially by skolemizing the end-sequent and recursively propagating the skolemization to the corresponding formulas in the premises above. Such skolemized proofs can still contain strong quantifiers that go into cuts.*

Remark 1. There are many algorithms for skolemization. They can be classified as either *prenex*, which firstly transform formulas and sequents into a prenex form (i.e. with all quantifiers occurring in the beginning of formulas), or *structural*, which leave weak quantifiers in their places. It has been shown that prenex skolemization can result in a non-elementary increase in the Herbrand Complexity of an **LK**-Proof [4]. Moreover, prenexification impairs the readability of formulas. Therefore we use structural skolemization algorithms [5], whenever skolemization is necessary or desirable for our proof transformations.

Example 1. The sequent

$$(\forall x)((\exists z)P(x, z) \wedge (\forall y)(P(x, y) \rightarrow P(x, f(y)))) \vdash (\forall x)(\exists y)P(x, f^2(y))$$

can be structurally skolemized to

$$(\forall x)(P(x, g(x)) \wedge (\forall y)(P(x, y) \rightarrow P(x, f(y)))) \vdash (\exists y)P(a, f^2(y))$$

where a is a skolem-constant and g is a skolem-function.

3   The *CERES* Method

Our motivation to devise and implement Herbrand sequent extraction algorithms was the need to analyze and understand the result of proof transformations performed automatically by the `CERES`-system, among which the main one is Cut-Elimination by Resolution: the *CERES* method [6].

The method transforms any **LKDe**-proof with cuts into an *atomic-cut normal form* (ACNF) containing no non-atomic cuts. The remaining atomic cuts

are, generally, non-eliminable, because **LKDe** admits non-tautological axiom sequents.

The ACNF is mathematically interesting, because cut-elimination in formal proofs corresponds to the elimination of lemmas in informal proofs. Hence the ACNF corresponds to an informal mathematical proof that is analytic in the sense that it does not use auxiliary notions that are not already explicit in the axioms or in the theorem itself.

The transformation to ACNF via Cut-Elimination by Resolution is done according to the following steps:

1. Construct the (always unsatisfiable [6]) *characteristic clause set* of the original proof by collecting, joining and merging sets of clauses defined by the ancestors of cut-formulas in the axioms of the proof.
2. Obtain from the characteristic clause set a grounded resolution refutation, which can be seen as an **LKe**-proof by exploiting the fact that the resolution rule is essentially a cut-rule restricted to atomic cut-formulas only.
3. For each clause of the characteristic clause set, construct a *projection* of the original proof with respect to the clause.
4. Construct the ACNF by plugging the projections into the leaves of the grounded resolution refutation tree (seen as an **LKe**-proof) and by adjusting the refutation accordingly. Since the projections do not contain cuts and the refutation contains atomic cuts only, the resulting **LKDe** proof will indeed be in atomic-cut normal form.

This method has been continuously improved and extended. The characteristic clause sets evolved to *proof profiles*, which are invariant under rule permutations and other simple transformations of proofs [12, 11]. The resolution and the sequent calculi are now being extended to restricted second-order logics.

The `CERES`-system automates the method described above, using either Otter[2] or Prover9[3] as resolution-based first-order theorem provers to obtain the refutation of the characteristic clause set. However, current fully-automated resolution-based theorem provers have difficulties to refute some characteristic clause sets produced by `CERES` [1]. On the other hand, interactive theorem provers are typically not resolution-based. Therefore, we are currently developing our own flexible, interactive and resolution-based first-order theorem prover.

4 An Algorithm for Herbrand Sequent Extraction

Herbrand sequents are a generalization of Herbrand disjunctions [10] for the sequent calculus **LK**.

Definition 3 (Herbrand Sequents of a Sequent). *Let s be a closed sequent containing weak quantifiers only. We denote by s_0 the sequent s after removal of all its quantifiers. Any propositionally valid sequent in which the antecedent*

---

[2] Otter Website: http://www-unix.mcs.anl.gov/AR/otter/

[3] Prover9 Website: http://www.cs.unm.edu/ mccune/prover9/

(respectively, consequent) formulas are instances (i.e. their free variables are possibly instantiated by other terms) of the antecedent (respectively, consequent) formulas of s_0 is called a Herbrand sequent of s.

Let s be an arbitrary sequent and s' a skolemization of s. Any Herbrand sequent of s' is a Herbrand sequent of s.

Remark 2. In Gentzen's original sequent calculus **LK**, Herbrand sequents are tautologies. In a sequent calculus with arbitrary atomic axioms, as **LKDe**, a valid sequent is only valid with respect to the axioms used in the proof. Hence, the Herbrand sequent will not be a tautology, but only propositionally valid with respect to the axioms used in the proof.

Remark 3. It would be possible to define a Herbrand sequent of an arbitrary sequent s without using skolemization. This could be achieved by imposing eigenvariable conditions on the instantiations chosen for the originally strongly quantified variables. However, the use of skolemization is advantageous, because skolem symbols store information about how the originally strongly quantified variables depend on the weakly quantified variables. This information would be lost if, instead of skolem terms, we had eigenvariables. Hence, skolemization improves readability of the Herbrand sequent.

Example 2 (Herbrand Sequents). Consider the valid sequent

$$(\forall x)((\exists z)P(x,z) \land (\forall y)(P(x,y) \to P(x,f(y)))) \vdash (\forall x)(\exists y)P(x,f^2(y))$$

The following sequents are some of its Herbrand sequents, where g is a skolem-function and a is a skolem-constant produced by skolemization:

1. $P(a,g(a)) \land (P(a,g(a)) \to P(a,f(g(a)))),$
   $P(a,g(a)) \land (P(a,f(g(a))) \to P(a,f^2(g(a)))) \vdash P(a,f^2(g(a)))$
2. $P(g(a),g^2(a)) \land (P(g(a),g^2(a)) \to P(g(a),f(g^2(a)))),$
   $P(g(a),g^2(a)) \land (P(g(a),f(g^2(a))) \to P(g(a),f^2(g^2(a)))) \vdash P(g(a),f^2(g^2(a)))$
3. $P(b,g(b)) \land (P(b,c) \to P(a,f(c))), P(a,g(a)) \land (P(a,g(a)) \to P(a,f(g(a)))),$
   $P(a,g(a)) \land (P(a,f(g(a))) \to P(a,f^2(g(a)))) \vdash P(a,f^2(g(a))), P(a,f^2(d))$

The first two Herbrand sequents above are minimal in the number of formulas, while the third is not.

Apart from its usage as an analysis tool, as described in this paper, the concept of Herbrand disjunction (or Herbrand sequent) also plays an important role in the foundations of Logic and Mathematics, as expressed by Herbrand's Theorem. A concise historical and mathematical discussion of Herbrand's Theorem, as well as its relation to Gödel's Completeness Theorem, can be found in [7].

Theorem 1 (Herbrand's Theorem). *A sequent s is valid if and only if there exists a Herbrand sequent of s.*

Proof. Originally in [10], stated for Herbrand disjunctions. Also in [7] with more modern proof calculi.

Herbrand's theorem guarantees that we can always obtain a Herbrand sequent from a correct proof, a possibility that was realized and exploited by Gentzen in his Mid-Sequent Theorem for sequents consisting of prenex formulas only.

Theorem 2 (Mid-Sequent Theorem). *Let φ be a prenex **LK***-Proof without non-atomic cuts. Then there is an **LK***-Proof φ' of the same end-sequent such that no quantifier rule occurs above propositional and cut rules.*

Proof. The original proof, for Gentzen's original sequent calculus **LK** and cut-free **LK**-Proofs, defines rule permutations that shift quantifier rules downwards. The iterated application of the permutations eventually reduces the original proof to a normal form fulfilling the mid-sequent property [8]. The rule permutations can be easily extended to proofs containing atomic cuts.

Remark 4. φ' has a mid-sequent, located between its lower first-order part and its upper propositional part. This mid-sequent is a Herbrand sequent of the end-sequent of φ, after skolemization of φ'.

However, Gentzen's algorithm has one strong limitation: it is applicable only to proofs with end-sequents in prenex form. Although we could transform the end-sequents and the proofs to prenex form, this would compromise the readability of the formulas and require additional computational effort. Prenexification is therefore not desirable in our context, and hence, to overcome this and other limitations in Gentzen's algorithm, we developed three other algorithms. In this paper we describe one of them in detail [4], which was chosen to be implemented within `CERES`. The other two are described in [17, 16]. Another different approach, based on functional interpretation and aiming at proofs with cuts, can be found in [9].

4.1 Extraction via Transformation to Quantifier-rule-free LKe$_\mathbf{A}$

The algorithm requires a temporary transformation to an extension of the calculus **LKe**, called **LKe$_\mathbf{A}$** and obtained by the addition of the following two rules, which allow the formation of array formulas $\langle A_1, \ldots, A_n \rangle$ from arbitrary formulas A_j, $j \in \{1, 2, \ldots, n\}$. They will be used to replace some contraction rules in the original proof.

$$\frac{\Delta, A_1, \Gamma_1, \ldots, A_n, \Gamma_n, \Pi \vdash \Lambda}{\Delta, \langle A_1, \ldots, A_n \rangle, \Gamma_1, \ldots, \Gamma_n, \Pi \vdash \Lambda} \; \langle \rangle : l \qquad \frac{\Lambda \vdash \Delta, A_1, \Gamma_1, \ldots, A_n, \Gamma_n, \Pi}{\Lambda \vdash \Delta, \langle A_1, \ldots, A_n \rangle, \Gamma_1, \ldots, \Gamma_n, \Pi} \; \langle \rangle : r$$

To extract a Herbrand sequent of the end-sequent of a skolemized **LKDe**-proof φ such that cuts do not contain quantifiers (nor defined formulas that contain quantifiers), the algorithm executes two transformations:

1. Ψ (definition 4): produces a quantifier-rule-free **LKe$_\mathbf{A}$**-proof where quantified formulas are replaced by array-formulas containing their instances.

2. Φ (definition 5): transforms the end-sequent of the resulting $\mathbf{LKe_A}$-proof into an ordinary sequent containing no array-formulas.

Remark 5. The restriction to proofs such that cuts do not contain quantifiers (nor defined formulas that contain quantifiers) is necessary because otherwise the cut-formulas in each branch of the cut would be substituted by different arrays and the corresponding cut inference in the $\mathbf{LKe_A}$-proof would be incorrect. The extracted sequent would not be propositionally valid, and therefore not a Herbrand sequent. This restriction is not a problem, because we analyze proofs in atomic-cut normal form produced by the *CERES* method.

Let φ be a skolemized $\mathbf{LKDe}$-Proof such that cuts do not contain quantifiers (nor defined formulas that contain quantifiers). A Herbrand sequent of the end-sequent of φ can be obtained by computing:

$$H(\varphi) \doteq \Phi(\text{end-sequent}(\Psi(\varphi)))$$

Definition 4 (Ψ: Transformation to Quantifier-rule-free $\mathbf{LKe_A}$).
The mapping Ψ transforms a skolemized $\mathbf{LKDe}$-Proof φ such that cuts do not contain quantifiers (nor defined formulas that contain quantifiers) into a quantifier-rule-free $\mathbf{LKe_A}$-Proof according to the inductive definition below:

Base Case, Initial Axiom Sequents:

$$\Psi(A_1, \ldots, A_n \vdash B_1, \ldots, B_m) \doteq A_1, \ldots, A_n \vdash B_1, \ldots, B_m$$

Proofs Ending with Quantifier-Rules:

$$\Psi \left(\begin{array}{c} [\varphi'] \\ \dfrac{A(t), \Gamma \vdash \Delta}{(\forall x)A(x), \Gamma \vdash \Delta} \ \forall : l \end{array} \right) \doteq \Psi(\varphi')$$

$$\Psi \left(\begin{array}{c} [\varphi'] \\ \dfrac{\Gamma \vdash \Delta, A(t)}{\Gamma \vdash \Delta, (\exists x)A(x)} \ \exists : r \end{array} \right) \doteq \Psi(\varphi')$$

Proofs Ending with Definition-Rules:

$$\Psi \left(\begin{array}{c} [\varphi'] \\ \dfrac{A(t_1, \ldots, t_n), \Gamma \vdash \Delta}{P(t_1, \ldots, t_n), \Gamma \vdash \Delta} \ d : l \end{array} \right) \doteq \Psi(\varphi')$$

$$\Psi \left(\begin{array}{c} [\varphi'] \\ \dfrac{\Gamma \vdash \Delta, A(t_1, \ldots, t_n)}{\Gamma \vdash \Delta, P(t_1, \ldots, t_n)} \ d : r \end{array} \right) \doteq \Psi(\varphi')$$

Proofs Ending with Contractions:

$$\Psi \left(\begin{array}{c} [\varphi'] \\ \dfrac{\Gamma, A, \Lambda_1 \ldots, A, \Lambda_n \vdash \Delta}{\Gamma, A, \Lambda_1, \ldots, \Lambda_n \vdash \Delta} \ c : l \end{array} \right) \doteq \begin{array}{c} [\Psi(\varphi')] \\ \dfrac{\Gamma^*, A_1, \Lambda_1^*, \ldots, A_n, \Lambda_n^* \vdash \Delta^*}{\Gamma^*, \langle A_1, \ldots, A_n \rangle, \Lambda_1^*, \ldots, \Lambda_n^* \vdash \Delta^*} \ \langle \rangle : l \end{array}$$

$$\Psi\left(\dfrac{\overset{[\varphi']}{\Gamma \vdash \Delta, A, \Lambda_1 \ldots, A, \Lambda_n}}{\Gamma \vdash \Delta, A, \Lambda_1, \ldots, \Lambda_n} \ c:r\right) \doteq \dfrac{\overset{[\Psi(\varphi')]}{\Gamma^* \vdash \Delta^*, A_1, \Lambda_1^*, \ldots, A_n, \Lambda_n^*}}{\Gamma^* \vdash \Delta^*, \langle A_1, \ldots, A_n\rangle, \Lambda_1^*, \ldots, \Lambda_n^*} \ \langle\rangle:r$$

Proofs Ending with other Unary Rules:

$$\Psi\left(\dfrac{\overset{[\varphi']}{A, \Gamma \vdash \Delta}}{A \wedge B, \Gamma \vdash \Delta} \ \wedge:l1\right) \doteq \dfrac{\overset{[\Psi(\varphi')]}{A^*, \Gamma^* \vdash \Delta^*}}{A^* \wedge B^*, \Gamma^* \vdash \Delta^*} \ \wedge:l1$$

For other unary rules, Ψ is defined analogously to the case for $\wedge:l1$ above.

Proofs Ending with other Binary Rules:

$$\Psi\left(\dfrac{\overset{[\varphi_1']}{A, \Gamma_1 \vdash \Delta_1} \quad \overset{[\varphi_2']}{B, \Gamma_2 \vdash \Delta_2}}{A \vee B, \Gamma_1, \Gamma_2 \vdash \Delta_1, \Delta_2} \ \vee:l\right) \doteq \dfrac{\overset{[\Psi(\varphi_1')]}{A^*, \Gamma_1^* \vdash \Delta_1^*} \quad \overset{[\Psi(\varphi_2')]}{B^*, \Gamma_2^* \vdash \Delta_2^*}}{A^* \vee B^*, \Gamma_1^*, \Gamma_2^* \vdash \Delta_1^*, \Delta_2^*} \ \vee:l$$

For other binary rules, Ψ is defined analogously to the case for $\vee:l$ above. Additionaly, the introduction of array-formulas requires an adaption of the positions in equality rules, so that replacements are executed in all formulas of the array.

Note that the definition of Ψ is sound, as for every proof φ and for all formula occurrences F in the end-sequent of φ, we can associate a corresponding formula occurrence F^* in the end-sequent of $\Psi(\varphi)$.

The transformation Ψ essentially omits quantifier-rules and definition-rules. Then it replaces contractions by array formations, because the auxiliary formulas of contractions will not be generally equal to each other anymore after the omission of quantifier-rules.

Example 3. Let φ be the following **LKDe**-proof:

$$\dfrac{\dfrac{\dfrac{\overset{[\varphi']}{P(0), P(0) \rightarrow P(s(0)), P(s(0)) \rightarrow P(s^2(0)) \vdash P(s^2(0))}}{P(0), P(0) \rightarrow P(s(0)), (\forall x)(P(x) \rightarrow P(s(x)) \vdash P(s^2(0))} \ \forall:l}{\dfrac{P(0), \forall x(P(x) \rightarrow P(s(x)), (\forall x)(P(x) \rightarrow P(s(x)) \vdash P(s^2(0))}{P(0), \forall x(P(x) \rightarrow P(s(x)) \vdash P(s^2(0))} \ c:l}}{P(0) \wedge (\forall x)(P(x) \rightarrow P(s(x)) \vdash P(s^2(0))} \ \wedge:l$$

Let φ_c be the subproof of φ with the contraction as the last rule. Firstly we compute $\Psi(\varphi_c)$:

$$\dfrac{\overset{[\Psi(\varphi')]}{P(0), P(0) \rightarrow P(s(0)), P(s(0)) \rightarrow P(s^2(0)) \vdash P(s^2(0))}}{P(0), \langle P(0) \rightarrow P(s(0)), P(s(0)) \rightarrow P(s^2(0))\rangle \vdash P(s^2(0))} \ \langle\rangle:l$$

The instances of the auxiliary occurrences of the contraction, $P(0) \rightarrow P(s(0))$ and $P(s(0)) \rightarrow P(s^2(0))$, are not equal to each other anymore. A contraction is therefore generally not possible, and this is why Ψ replaces contractions by array formations. The array-formulas store all the instances that would have been contracted.

We proceed with the computation of $\Psi(\varphi)$:

$$[\Psi(\varphi')]$$

$$\cfrac{\cfrac{P(0), P(0) \to P(s(0)), P(s(0)) \to P(s^2(0)) \vdash P(s^2(0))}{P(0), \langle P(0) \to P(s(0)), P(s(0)) \to P(s^2(0)) \rangle \vdash P(s^2(0))} \ \langle\rangle : l}{P(0) \wedge \langle P(0) \to P(s(0)), P(s(0)) \to P(s^2(0)) \rangle \vdash P(s^2(0))} \ \wedge : l$$

Definition 5 (Φ: Expansion of Array Formulas). *The mapping Φ transforms array formulas and sequents into first-order logic formulas and sequents. In other words, Φ eliminates $\langle \ldots \rangle$ and can be defined inductively by:*

1. *If A is a first-order logic formula, then $\Phi(A) \doteq A$*
2. *$\Phi(\langle A_1, \ldots, A_n \rangle) \doteq \Phi(A_1), \ldots, \Phi(A_n)$*
3. *If $\Phi(A) = A_1, \ldots, A_n$, then $\Phi(\neg A) \doteq \neg A_1, \ldots, \neg A_n$*
4. *If $\Phi(A) = A_1, \ldots, A_n$ and $\Phi(B) = B_1, \ldots, B_m$, then $\Phi(A \circ B) \doteq A_1 \circ B_1, \ldots, A_1 \circ B_m, \ldots, A_n \circ B_1, \ldots, A_n \circ B_m$, for $\circ \in \{\wedge, \vee, \to\}$*
5. *$\Phi(A_1, \ldots, A_n \vdash B_1, \ldots, B_m) \doteq \Phi(A_1), \ldots, \Phi(A_n) \vdash \Phi(B_1), \ldots, \Phi(B_m)$*

*Φ has not been defined over formulas that contain array formulas in the scope of quantifiers. This is not necessary, because Ψ transforms **LKDe**-proofs into **LKe$_\mathbf{A}$**-proofs where this situation never occurs.*

Example 4. Let $\Psi(\varphi)$ be the **LKe$_\mathbf{A}$**-Proof in Example 3. Then, its end-sequent, after mapping array formulas to sequences of formulas, is:

$$\Phi(\textit{end-sequent}(\Psi(\varphi))) = \begin{pmatrix} P(0) \wedge (P(0) \to P(s(0))), \\ P(0) \wedge (P(s(0)) \to P(s^2(0))) \end{pmatrix} \vdash P(s^2(0))$$

Theorem 3 (Soundness). *Let φ be a skolemized **LKDe**-Proof in atomic cut normal form. Then the sequent $H(\varphi)$, extracted by the algorithm defined above, is a Herbrand sequent of the end-sequent of φ.*

Proof. A proof (fully available in [17]) can be sketched in the following way. We have to show that:

1. The formulas of $H(\varphi)$ are substitution instances of the formulas of the end-sequent of φ without their quantifiers.
2. $H(\varphi)$ is a valid sequent.

Item 1 follows clearly from the definitions of Φ and Ψ, because Ψ substitutes quantified sub-formulas of the end-sequent by array-formulas containing only substitution instances of the respective sub-formulas, and Φ expands the the array-formulas maintaining the structure of the formulas where they are located.

Item 2 can be proved by devising a transformation Φ_P that maps the intermediary **LKDe$_A$**-proof $\Psi(\varphi)$ to an **LKDe**-proof $\Phi_P(\Psi(\varphi))$ by substituting $\langle\rangle : l$ rules by sequences of $\wedge : l$ rules and $\langle\rangle : r$ rules by sequences of $\vee : r$ rules. $\Psi(\varphi)$ does not contain quantifier rules. Therefore $\Phi_P(\Psi(\varphi))$ is essentially a propositional **LKe**-proof, in which the arrays of the end-sequent were substituted by either nested $\wedge$ connectives or nested $\vee$ connectives. The end-sequent of $\Phi_P(\Psi(\varphi))$ can be shown, by structural induction, to be logically equivalent to the extracted sequent $\Phi(\textit{end-sequent}(\Psi(\varphi)))$. Therefore the extracted sequent is valid.

5 Analysis of the Lattice Proof

In this section, the usefulness of a Herbrand sequent for understanding a formal proof will be demonstrated on a simple example from lattice theory . There are several different, but equivalent, definitions of *lattice*. Usually, the equivalence of several statements is shown by proving a cycle of implications. While reducing the size of the proof, this practice has the drawback of not providing *direct* proofs between the statements. Cut-elimination can be used to automatically generate a direct proof between any two of the equivalent statements. In this section, we will demonstrate how to apply cut-elimination with the `CERES`-system followed by Herbrand sequent extraction for this purpose.

5.1 The Lattice Proof

Definitions 7, 8 and 10 list different sets of properties that a 3-tuple $\langle L, \cap, \cup \rangle$ or a partially ordered set $\langle S, \leq \rangle$ must have in order to be considered a lattice.

Definition 6 (Semi-Lattice). *A semi-lattice is a set L together with an operation $\circ$ which is*

- *commutative: $(\forall x)(\forall y)\ x \circ y = y \circ x$,*
- *associative: $(\forall x)(\forall y)(\forall z)\ (x \circ y) \circ z = x \circ (y \circ z)$ and*
- *idempotent: $(\forall x)\ x \circ x = x$.*

Definition 7 (Lattice: definition 1). *A L1-lattice is a set L together with operations $\cap$ (meet) and $\cup$ (join) s.t. both $\langle L, \cap \rangle$ and $\langle L, \cup \rangle$ are semi-lattices and $\cap$ and $\cup$ are "inverse" in the sense that*

$$(\forall x)(\forall y)\ x \cap y = x \leftrightarrow x \cup y = y.$$

Definition 8 (Lattice: definition 2). *A L2-lattice is a set L together with operations $\cap$ and $\cup$ s.t. both $\langle L, \cap \rangle$ and $\langle L, \cup \rangle$ are semi-lattices and the absorption laws*

$$(\forall x)(\forall y)\ (x \cap y) \cup x = x \quad and \quad (\forall x)(\forall y)\ (x \cup y) \cap x = x$$

hold.

Definition 9 (Partial Order). *A binary relation $\leq$ on a set S is called* partial order *if it is*

- *reflexive (R): $(\forall x)\ x \leq x$,*
- *anti-symmetric (AS): $(\forall x)(\forall y)\ ((x \leq y \wedge y \leq x) \rightarrow x = y)$ and*
- *transitive (T): $(\forall x)(\forall y)(\forall z)\ ((x \leq y \wedge y \leq z) \rightarrow x \leq z)$.*

Definition 10 (Lattice: definition 3). *A L3-lattice is a partially ordered set $\langle S, \leq \rangle$ s.t. for each two elements of S there exist*

- *greatest lower bound (GLB) $\cap$, i.e.*

$$(\forall x)(\forall y)(x \cap y \leq x \wedge x \cap y \leq y \wedge (\forall z)((z \leq x \wedge z \leq y) \rightarrow z \leq x \cap y)),$$

– *least upper bound (LUB)* $\cup$, *i.e.*

$$(\forall x)(\forall y)(x \leq x \cup y \wedge y \leq x \cup y \wedge (\forall z)((x \leq z \wedge y \leq z) \rightarrow x \cup y \leq z)).$$

The above three definitions of lattice are equivalent. We will formalize the following proofs of $L1 \rightarrow L3$ and $L3 \rightarrow L2$ in order to extract a direct proof of $L1 \rightarrow L2$, i.e. one which does not use the notion of partial order.

Proposition 1. *L1-lattices are L3-lattices.*

Proof. Given $\langle L, \cap, \cup \rangle$, define $x \leq y$ as $x \cap y = x$. By idempotence of $\cap$, $\leq$ is reflexive. Anti-symmetry of $\leq$ follows from commutativity of $\cap$ as $(x \cap y = x \wedge y \cap x = y) \rightarrow x = y$. To see that $\leq$ is transitive, assume (a) $x \cap y = x$ and (b) $y \cap z = y$ to derive

$$x \cap z =^{\text{(a)}} (x \cap y) \cap z =^{\text{(assoc.)}} x \cap (y \cap z) =^{\text{(b)}} x \cap y =^{\text{(a)}} x$$

So $\leq$ is a partial order on L.

By associativity, commutativity and idempotence of $\cap$, we have $(x \cap y) \cap x = x \cap y$, i.e. $x \cap y \leq x$ and similarly $x \cap y \leq y$, so $\cap$ is a lower bound for $\leq$. To see that $\cap$ is also the greatest lower bound, assume there is a z with $z \leq x$ and $z \leq y$, i.e. $z \cap x = z$ and $z \cap y = z$. Then, by combining these two equations, $(z \cap y) \cap x = z$, and therefore, $z \leq x \cap y$.

To show that $\cup$ is an upper bound, derive from the axioms of semi-lattices that $x \cup (x \cup y) = x \cup y$ which, by the "inverse" condition of L1 gives $x \cap (x \cup y) = x$, i.e. $x \leq x \cup y$ and similarly for $y \leq x \cup y$. Now assume there is a z with $x \leq z$ and $y \leq z$, i.e. $x \cap z = x$ and $y \cap z = z$ and by the "inverse" condition of L1: $x \cup z = z$ and $y \cup z = z$. From these two equations and the axioms of semi-lattices, derive $(x \cup y) \cup z = z$ which, by the "inverse" condition of $L1$, gives $(x \cup y) \cap z = x \cup y$, i.e. $x \cup y \leq z$.

Proposition 2. *L3-lattices are L2-lattices.*

Proof. We want to show the absorption law $(x \cap y) \cup x = x$. That $x \leq (x \cap y) \cup x$ follows immediately from $\cup$ being an upper bound. But $x \cap y \leq x$ because $\cap$ is a lower bound. Furthermore also $x \leq x$, so x is an upper bound of $x \cap y$ and x. But as $\cup$ is the lowest upper bound, we have $(x \cap y) \cup x \leq x$ which by anti-symmetry of $\leq$ proves $(x \cap y) \cup x = x$. For proving the other absorption law $(x \cup y) \cap x = x$, proceed symmetrically.

By concatenation, the above two proofs show that all L1-lattices are L2-lattices. However, this proof is not a direct one, it uses the notion of partially ordered set which occurs neither in L1 nor in L2. By cut-elimination we will generate a direct formal proof automatically.

5.2 Overview of the Analysis

The analysis of the lattice proof followed the steps below:

1. *Formalization of the lattice proof in the sequent calculus* **LKDe**: semi-auto-mated by `HLK`[4]. Firstly the proof was written in the language HandyLK, which can be seen as an intermediary language between informal mathematics and **LKDe**. Subsequently, `HLK` compiled it to **LKDe**.
2. *Cut-Elimination of the formalized lattice proof*: fully automated by `CERES`, employing the cut-elimination procedure based on resolution, sketched in Section 3, to obtain an **LKDe**-proof in Atomic-Cut Normal Form (ACNF), i.e. a proof in which cut-formulas are atoms.
3. *Extraction of the Herbrand sequent of the ACNF*: fully automated by `CERES`, employing the algorithm described in Section 4.
4. *Use of the Herbrand sequent* to interpret and understand the ACNF, in order to obtain a new direct informal proof.

5.3 Formalization of the Lattice Proof

The full formal proof has 260 rules (214 rules, if structural rules (except cut) are not counted). It is too large to be displayed here. Below we show only a part of it, which is close to the end-sequent and depicts the main structure of the proof, based on the cut-rule with $L3$ as the cut-formula. This cut divides the proof into two subproofs corresponding to propositions 1 and 2. The full proofs, conveniently viewable with `ProofTool`[5], are available in the website of `CERES`.

$$
\cfrac{
  \cfrac{
    \cfrac{
      \cfrac{[p_R]}{\vdash R} \quad
      \cfrac{
        \cfrac{[p_{AS}] \quad [p_T]}{\vdash AS \quad \vdash T}}{\vdash AS \wedge T} \wedge : r
    }{\vdash R \wedge (AS \wedge T)} \wedge : r
    }{\cfrac{\vdash POSET}{} }\ d : r \quad
  \cfrac{\cfrac{[p_{GLB}] \quad [p_{LUB}]}{L1 \vdash GLB \wedge LUB} \wedge : r}{}
}{
  \cfrac{L1 \vdash POSET \wedge (GLB \wedge LUB)}{L1 \vdash L3}\ d : r \quad \cfrac{[p_3^2]}{L3 \vdash L2}
}{L1 \vdash L2}\ cut
$$

- $L1 \equiv \forall x \forall y ((x \cap y) = x \supset (x \cup y) = y) \wedge ((x \cup y) = y \supset (x \cap y) = x)$
- $L2 \equiv \forall x \forall y (x \cap y) \cup x = x \wedge \forall x \forall y (x \cup y) \cap x = x$
- $L3 \equiv POSET \wedge (GLB \wedge LUB)$
- p_{AS}, p_T, p_R are proofs of, respectively, anti-symmetry (AS), transitivity (T) and reflexivity (R) of $\leq$ from the axioms of semi-lattices.
- p_3^2 is a proof that $L3$-lattices are $L2$-lattices, from the axioms of semi-lattices.

5.4 Cut-Elimination of the Lattice Proof

Prior to cut-elimination, the formalized proof is skolemized by `CERES`, resulting in a proof of the skolemized end-sequent $L1 \vdash (s_1 \cap s_2) \cup s_1 = s_1 \wedge (s_3 \cup s_4) \cap s_3 = s_3$, where s_1, s_2, s_3 and s_4 are skolem constants for the strongly quantified variables of $L2$. Then `CERES` eliminates cuts, producing a proof in atomic-cut normal (also available for visualization with `ProofTool` in the website of `CERES`).

---

[4] `HLK` Website: http://www.logic.at/hlk/
[5] `ProofTool` Website: http://www.logic.at/prooftool/

The ACNF is still quite large (214 rules; 72 rules not counting structural rules (except cut)). It is interesting to note, however, that the ACNF is smaller than the original proof in this case, even though in the worst case cut-elimination can produce a non-elementary increase in the size of proofs [14].

Although the ACNF of the Lattice proof is still large (214 rules), the extracted Herbrand sequent contains only 6 formulas, as shown in Subsection **??**. Therefore, the Herbrand sequent significantly reduces the amount of information that has to be analyzed in order to extract the direct mathematical argument contained in the ACNF.

5.5 Herbrand Sequent Extraction of the ACNF of the Lattice Proof

The Herbrand sequent of the ACNF, after set-normalization and removal of remaining sub-formulas introduced by weakening (or as the non-auxiliary formula of $\vee$ and $\wedge$ rules) in the ACNF, is:

$(A1)$ $s_1 \cup (s_1 \cup (s_1 \cap s_2)) = s_1 \cup (s_1 \cap s_2) \rightarrow s_1 \cap (s_1 \cup (s_1 \cap s_2)) = s_1,$

$(A2)$ $s_1 \cap s_1 = s_1 \rightarrow s_1 \cup s_1 = s_1,$

$(A3)$ $\underbrace{(s_1 \cap s_2) \cap s_1 = s_1 \cap s_2}_{(A3i)} \rightarrow (s_1 \cap s_2) \cup s_1 = s_1,$

$(A4)$ $(s_1 \cup (s_1 \cap s_2)) \cup s_1 = s_1 \rightarrow (s_1 \cup (s_1 \cap s_2)) \cap s_1 = s_1 \cup (s_1 \cap s_2),$

$(A5)$ $\underbrace{s_3 \cup (s_3 \cup s_4) = s_3 \cup s_4}_{(A5i)} \rightarrow s_3 \cap (s_3 \cup s_4) = s_3$

$(C1)$ $\vdash \underbrace{(s_1 \cap s_2) \cup s_1 = s_1}_{(C1i)} \wedge \underbrace{(s_3 \cup s_4) \cap s_3 = s_3}_{(C1ii)}$

5.6 Construction of the informal proof

After extracting a Herbrand sequent from the ACNF, the next step is to construct an informal, analytic proof of the theorem, based on the ACNF, but using only the information about the variable instantiations contained in its extracted Herbrand sequent. We want to stress that in the following, we are not performing syntactic manipulations of formulas of first-order logic, but instead we use the formulas from the Herbrand sequent of the ACNF as a *guide* to construct an analytical mathematical proof.

Theorem 4. *All L1-lattices $\langle L, \cap, \cup \rangle$ are L2-lattices.*

Proof. As both lattice definitions have associativity, commutativity and idempotence in common, it remains to show that the absorption laws hold for $\langle L, \cap, \cup \rangle$. We notice that, as expected, these properties coincide with the conjunction $(C1)$ for arbitrary $s_1, \ldots, s_4$ on the right hand side of the Herbrand sequent and so we proceed by proving each conjunct for arbitrary $s_1, \ldots, s_4 \in L$:

1. *We notice that $(A3i) + (A3)$ imply $(C1i)$. So we prove these properties:*

(a) First we prove (A3i):

$$s_1 \cap s_2 =^{(\text{idem.})} (s_1 \cap s_1) \cap s_2 =^{(\text{assoc.})} s_1 \cap (s_1 \cap s_2) =^{(\text{comm.})}$$
$$s_1 \cap (s_2 \cap s_1) =^{(\text{assoc.})} (s_1 \cap s_2) \cap s_1$$

(b) Assume $(s_1 \cap s_2) \cap s_1 = s_1 \cap s_2$. By definition of L1-lattices, $(s_1 \cap s_2) \cup s_1 = s_1$. Thus, we have proved (A3).

2. Again, we notice that $(A5i) + (A5) + commutativity$ imply $(C1ii)$ and use this fact:

(a) $s_3 \cup s_4 =^{(\text{idem.})} (s_3 \cup s_3) \cup s_4 =^{(\text{assoc.})} s_3 \cup (s_3 \cup s_4)$. We have proved (A5i).

(b) Assume $s_3 \cup (s_3 \cup s_4) = s_3 \cup s_4$. By definition of L1-lattices, $s_3 \cap (s_3 \cup s_4) = s_3$. This proves (A5).

So we have shown that for arbitrary $s_1, \ldots, s_4 \in L$, we have $(s_1 \cap s_2) \cup s_1 = s_1$ and $(s_3 \cup s_4) \cap s_3 = s_3$, which completes the proof.

Contrary to the proof in Section 5.1, we can now directly see the algebraic construction used to prove the theorem. This information was hidden in the synthetic argument that used the notion of partially ordered sets and was revealed by cut-elimination.

This example shows that the Herbrand sequent indeed contains the essential information of the ACNF, since an informal direct proof corresponding to the ACNF could be constructed by analyzing the extracted Herbrand sequent only.

6   Conclusion

We have described a new algorithm for Herbrand sequent extraction, which is better than Gentzen's mid-sequent reduction because it can be applied to proofs that are not in prenex form. Its use as a tool for the analysis of computer generated proofs was successfully demonstrated with a simple proof about lattices, which was automatically transformed to atomic-cut normal form by the `CERES` system. The Herbrand sequent significantly reduced the amount of information that had to be analyzed in order to understand the atomic-cut normal form produced by `CERES`:

Our algorithm still lacks support for definition rules, because they are removed by the transformation to **LKe$_\mathbf{A}$**. We are planning to improve on this and to be able to reinsert defined formulas in the extracted Herbrand sequent, in order to further improve its readability.

The technique described here is not limited to sequent calculi, since it relies on Herbrand's theorem, which is applicable to first-order logic in general. Many computer-generated proofs are obtained, for example, by automated theorem provers that do not work with sequent calculi, but with resolution calculi. Such resolution proofs are usually even harder for humans to understand, and therefore we are planning to extend the general idea behind Herbrand sequent extraction to an algorithm applicable to resolution proofs as well. This could be done by firstly translating a resolution refutation into a corresponding proof in sequent calculus.

References

1. Matthias Baaz, Stefan Hetzl, Alexander Leitsch, Clemens Richter, and Hendrik Spohr. Ceres: An Analysis of Fürstenberg's Proof of the Infinity of Primes. to appear in Theoretical Computer Science.

2. Matthias Baaz, Stefan Hetzl, Alexander Leitsch, Clemens Richter, and Hendrik Spohr. Cut-Elimination: Experiments with CERES. In Franz Baader and Andrei Voronkov, editors, *Logic for Programming, Artificial Intelligence, and Reasoning (LPAR) 2004*, volume 3452 of *Lecture Notes in Computer Science*, pages 481–495. Springer, 2005.

3. Matthias Baaz, Stefan Hetzl, Alexander Leitsch, Clemens Richter, and Hendrik Spohr. Proof Transformation by CERES. In Jonathan M. Borwein and William M. Farmer, editors, *Mathematical Knowledge Management (MKM) 2006*, volume 4108 of *Lecture Notes in Artificial Intelligence*, pages 82–93. Springer, 2006.

4. Matthias Baaz and Alexander Leitsch. On skolemization and proof complexity. *Fundamenta Matematicae*, 20:353–379, 1994.

5. Matthias Baaz and Alexander Leitsch. Cut normal forms and proof complexity. *Annals of Pure and Applied Logic*, 97:127–177, 1999.

6. Matthias Baaz and Alexander Leitsch. Cut-elimination and Redundancy-elimination by Resolution. *Journal of Symbolic Computation*, 29(2):149–176, 2000.

7. S. R. Buss. On Herbrand's theorem. *Lecture Notes in Computer Science*, 960:195, 1995.

8. G. Gentzen. Untersuchungen über das logische Schließen. In M.E.Szabo, editor, *The Collected Papers of Gerhard Gentzen*, pages 68–131. North-Holland Publishing Company, Amsterdam - London, 1969.

9. Philipp Gerhardy and Ulrich Kohlenbach. Extracting herbrand disjunctions by functional interpretation. *Archive for Mathematical Logic*, 44(5), 2005.

10. J. Herbrand. Recherches sur la théorie de la démonstration. *Travaux de la Societé des Sciences et des Lettres de Varsovie, Class III, Sciences Mathématiques et Physiques*, 33, 1930.

11. Stefan Hetzl. *Characteristic Clause Sets and Proof Transformations*. PhD thesis, Vienna University of Technology, 2007.

12. Stefan Hetzl and Alexander Leitsch. Proof transformations and structural invariance. *Algebraic and Proof-theoretic aspects*, LNAI 4460:201–230, 2007.

13. H. Luckhardt. Herbrand-Analysen zweier Beweise des Satzes von Roth: polynomiale Anzahlschranken. *Journal of Symbolic Logic*, 54:234–263, 1989.

14. Richard Statman. Lower bounds on Herbrand's theorem. *Proceedings of the American Mathematical Society*, 75:104–107, 1979.

15. G. Sutcliffe and C. Suttner. The State of CASC. *AI Communications*, 19(1):35–48, 2006.

16. Bruno Woltzenlogel Paleo. Herbrand sequent extraction. Master's thesis, Technische Universitaet Dresden; Technische Universitaet Wien, Dresden, Germany; Wien, Austria, 2007.

17. Bruno Woltzenlogel Paleo. *Herbrand Sequent Extraction*. VDM-Verlag, Saarbruecken, Germany, 2008.