CERES for Propositional Proof Schemata*

C. Dunchev[1], A. Leitsch[1], M. Rukhaia[1], and D. Weller[2]

1 Institute of Computer Languages, Vienna University of Technology.
2 Institute of Discrete Mathematics and Geometry, Vienna University of Technology.

Abstract

The cut-elimination method CERES (for first- and higher-order classical logic) is based on the notion of a characteristic clause set, which is extracted from an **LK**-proof and is always unsatisfiable. A resolution refutation of this clause set can be used as a skeleton for a proof with atomic cuts only (atomic cut normal form). This is achieved by replacing clauses from the resolution refutation by the corresponding projections of the original proof.

We present a generalization of this method to propositional proof schemata and define a schematic version of propositional sequent calculus called **LKS**, and a notion of proof schema based on primitive recursive definitions. A method is developed to extract schematic characteristic clause sets and schematic projections from these proof schemata. We also define a schematic resolution calculus for refutation of schemata of clause sets, which can be applied to refute the schematic characteristic clause sets. Finally the projection schemata and resolution schemata are plugged together and a schematic representation of the atomic cut normal forms is obtained. An algorithmic handling of the schematic cut-elimination method is supported by a recent extension of the CERES system.

1998 ACM Subject Classification F.4.1 Mathematical Logic

Keywords and phrases schemata, cut-elimination, resolution

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

1 Introduction

Cut-elimination was originally introduced by G. Gentzen in [12] as a theoretical tool from which results like decidability and consistency could be proven. Cut-free proofs are computationally explicit objects from which interesting information such as Herbrand disjunctions and interpolants can be easily extracted. When viewing formal proofs as a model for mathematical proofs, cut-elimination corresponds to the removal of lemmata, which leads to interesting applications (such as one described below).

For such applications to mathematical proofs, the cut-elimination method CERES (cut-elimination by resolution) was developed [7, 9, 13]. It essentially reduces cut-elimination for a proof π to a theorem proving problem: the refutation of the *characteristic clause set* $\mathrm{CL}(\pi)$. Given a resolution refutation γ of $\mathrm{CL}(\pi)$, an essentially cut-free proof can be constructed by a proof-theoretic transformation. In contrast to the usual Gentzen-style approach to cut-elimination, the CERES method performs a global instead of a local analysis of the proof.

The present work is motivated by an application of CERES to (a formalization of) a mathematical proof: Fürstenberg's proof of the infinity of primes [1, 6]. Since cut-elimination

* Supported by the FWF project I383.

in the presence of induction is problematic, the proof was formalized as a sequence of proofs $(\pi_k)_{k\in\mathbb{N}}$ showing that the assumption that there exist exactly k primes is contradictory. The application was performed in a semi-automated way: $\mathrm{CL}(\pi_k)$ was computed for some small values of k and from this, a general scheme $\mathrm{CL}(\pi_n)$, where n is a formal parameter, was constructed and subsequently analyzed by hand. The analysis finally showed that from Fürstenberg's proof, which makes use of topological concepts, Euclid's elementary proof could be obtained by cut-elimination. It was unsatisfactory that the parameterized object $\mathrm{CL}(\pi_n)$ was constructed ad-hoc and by hand, instead of being computed automatically from a parameterized representation of the sequence $(\pi_k)_{k\in\mathbb{N}}$. Further, for lack of a formalism, the analysis of $\mathrm{CL}(\pi_n)$ could not be performed in a computer-aided way.

Hence the present work is concerned with the investigation of a formal concept of *proof schema*. The aim is to define a CERES method for proof schemata that will yield a schematic representation of $\mathrm{CL}(\pi_k)$. This can be regarded as a generalization of CERES from proofs π to sequences of proofs $(\pi_k)_{k\in\mathbb{N}}$ that are given by such a proof schema. Not only will this close the gap in the application described above (and in future ones) by automatically computing the correct schema $\mathrm{CL}(\pi_n)$, but it will also allow (semi-)automated application of the method using theorem provers and proof assistants based on the formal definitions.

A central aspect of our notion of proof schema is its treatment of inductive reasoning. In other approaches [14, 10, 16], *induction rules* are considered. Usually, such a rule allows to infer, from proofs of $\varphi(0)$ and $\varphi(n) \supset \varphi(n+1)$, the statement $\varphi(n)$. In a sense, inductive reasoning is done on the level of formulas. In our approach, the treatment of induction is on the level of *proofs*: for example, we allow in a proof of $\varphi(n+1)$ a *reference* or *proof link* to a proof of $\varphi(n)$. This is similar to the *cyclic approach* taken in [15, 11].

We start our investigation into proof schemata with the case of propositional logic. A notion of *formula schema* was introduced and investigated in [2, 4]. The subclass of *regular schemata* was identified and shown decidable, and a tableau calculus STAB was defined and implemented [3]. In this paper, we will focus on this class of schemata, although it can in principle be applied to more expressive classes of formulas (this is clear in the propositional case, and the extension to first-order schemata will be investigated in the future).

This paper is structured in the following way: In Section 2, we introduce formula and proof schemata. Sections 3 and 4 are devoted to the data structures central to the CERES method: the characteristic clause term and the proof projection term respectively. In Section 5, a schematic resolution calculus is defined and the main theoretical result on the CERES method is stated. Finally, in Section 6, we apply the method semi-automatically to a proof that an n-bit adder is commutative.

2 Schematic Language

The aim of this section is to introduce a notion of propositional proof schema. Following [4] we first introduce the syntax of propositional formula schemata on which we then build our notion of schematic sequent calculus proof. To the best of our knowledge, there does not yet exist a sequent calculus for propositional formula schemata, although *cyclic proofs* which are similar to our proof schemata have been considered in the literature [15, 11].

We assume a countably infinite set of *index variables* (intended to be interpreted over $\mathbb{N}$), and a countably infinite set of *propositional symbols*. As we will see later in this section, index variables can be free or bound. Free index variables are called *parameters*. The set of *arithmetic expressions* is defined inductively from the constant 0 and the index variables over the signature $s, +$ (note that all our arithmetic expressions are linear). We will denote elements of $\mathbb{N}$ by $\alpha, \beta, \ldots$, bound index variables by $i, j, l, \ldots$, parameters by $k, m, n, \ldots$, and arithmetic expressions by $a, b, \ldots$. We will identify $\alpha \in \mathbb{N}$ with the numeral $s^{\alpha}(0)$. We say that an arithmetic expression a is *ground* if a does not contain variables. We make use of the standard rewrite system for arithmetic expressions, consisting of $a + 0 \to a, a + s(b) \to s(a+b)$ and so forth. This rewrite system is confluent and terminating for ground arithmetic expressions, and for simplicity we will identify such expressions with their normal forms $s^{\alpha}(0)$. A substitution is a function mapping every (free) index variable to an arithmetic expression.

▶ Definition 2.1 (Indexed proposition). An expression of the form p_a, where a is an arithmetic expression and p a propositional symbol, is called an *indexed proposition.*

▶ Definition 2.2 (Formula schemata). We define formula schemata inductively in the following way:

- An indexed proposition is an (atom) formula schema.
- If ϕ_1 and ϕ_2 are formula schemata, then so are $\phi_1 \vee \phi_2$, $\phi_1 \wedge \phi_2$ and $\neg\phi_1$.
- If ϕ is a formula schema, a, b are arithmetic expressions and i is an index variable not bound in ϕ, then $\bigwedge_{i=a}^{b} \phi$ and $\bigvee_{i=a}^{b} \phi$ are formula schemata such that i is bound in both formula schemata.

We denote formula schemata by $A, B, \ldots$. The notation $A(k)$ is used to indicate a parameter k in A, and $A(a)$ then denotes $A\{k \leftarrow a\}$.

2.1 Proof schemata

In this section we define a version of the classical propositional sequent calculus **LK**. It will differ from **LK** in two ways: The formulas it will operate on will be formula schemata, and special initial sequents called *proof links* will be allowed. Note that already in [2], a calculus STAB for formula schemata is introduced. Our calculus differs from it in some aspects: most importantly, we include the cut rule, which allows the formalization of proofs that use lemmata. Furthermore, instead of a looping rule (which is geared towards automated theorem proving), we use a different approach to the recursive specification of proofs which is more suited to the formalization of proofs found by humans.

▶ Definition 2.3 (Calculus **LKS**). An expression of the form $\Gamma \vdash \Delta$, where Γ and Δ are multisets of formula schemata, is called a *sequent schema* (or just sequent). If S is a sequent schema and a an arithmetic expression, the notation $S(a)$ is defined as for formula schemata. If $\Gamma \cup \Delta$ consists of atoms only, the sequent is called a *clause.*

We assume a countably infinite set of *proof symbols* denoted by $\varphi, \psi, \ldots$. If φ is a proof symbol, a is an arithmetic expression, and S a sequent schema, then the expression $\dfrac{(\varphi(a))}{S}$ is called a *proof link.*

An **LKS**-proof is a tree of sequents such that every leaf is either of the form $A \vdash A$ for atomic A, or is a proof link, and that is built according to the usual rules of classical propositional sequent calculus **LK** with the proviso that schematic formulas are considered modulo the equalities $A(0) = \bigwedge_{i=0}^{0} A(i)$ and $(\bigwedge_{i=0}^{n} A(i)) \wedge A(n+1) = \bigwedge_{i=0}^{n+1} A(i)$ (and analogously for $\bigvee$)[1].

The notion of *auxiliary formula* is defined as usual, and all occurrences of formulas in an **LKS**-proof are endowed with an *ancestor relation* in the usual way. If O is a set of formula

[1] These equalities are used so that the rules for $\wedge$ and $\vee$ can be applied to $\bigwedge$ and $\bigvee$.

occurrences and o is a formula occurrence, then o is an *O-ancestor* if o is an ancestor of some $o' \in O$. o is a *cut-ancestor* if it is an ancestor of an auxiliary formula of a cut.

The root sequent of an **LKS**-proof is called its *end-sequent.* An **LKS**-proof is called *ground* if it does not contain parameters and proof links.

Note that a ground **LKS**-proof is essentially an **LK**-proof (obtained by replacing ground $\bigwedge, \bigvee$ by a finite number of $\wedge, \vee$). We usually denote **LKS**-proofs by $\pi, \nu, \ldots$.

To be of practical use, we add a notion of recursion to the **LKS**-proofs defined above. This will yield our notion of *proof schemata*:

▶ Definition 2.4 (Proof schemata). Let ψ be a proof symbol and $S(n)$ be a sequent. Then a *proof schema pair for ψ* is a pair of **LKS**-proofs $(\pi, \nu(k+1))$ with end-sequents $S(0)$ and $S(k+1)$ respectively such that π may not contain proof links and $\nu(k+1)$ may contain only proof links of the form $\frac{(\psi(k))}{S(k)}$. For such a proof schema pair, we say that a proof link of the form $\frac{(\psi(a))}{S(a)}$ is a *proof link to ψ*. We say that $S(n)$ is the end-sequent of ψ, and we assume an identification between formula occurrences in the end-sequents of π and $\nu(k+1)$ so that we can speak of occurrences in the end-sequent of ψ.

Finally, a *proof schema* Ψ is a tuple of proof schema pairs $\langle p_1, \ldots, p_\alpha \rangle$ for $\psi_1, \ldots, \psi_\alpha$ respectively such that the **LKS**-proofs in p_β may also contain proof links to ψ_γ for $1 \leq \beta < \gamma \leq \alpha$. We also say that the end-sequent of ψ_1 is the end-sequent of Ψ.

We make a small detour to consider a possible objection to our definition of proof schema: that one could define a similar proof system in a simpler way by considering **LKS**-proofs together with an *induction rule.* This is indeed possible, and there are translations between such a system and our system of proof schemata. The reason that we are considering the formalism of proof schemata is that on one hand, it is very natural to write proofs in such a way: the proof π analyzed in [6] is an example. On the other hand, the translation from proof schemata to the system with an induction rule introduces additional cuts. Since we are interested in cut-elimination, and furthermore in the elimination of cuts from concrete proofs, the introduction of additional cuts should be avoided if at all possible. The authors have experienced this problem when trying to apply the CERES method to a formalization of π in higher-order logic, using the second-order induction axiom: the characteristic clause set of the proof was too large to be analyzed either manually of automatically.

Therefore we will continue with our investigation of proof schemata. In the following, we will consider a fixed proof schema $\Psi = \langle (\pi_1, \nu_1(k+1)), \ldots, (\pi_\alpha, \nu_\alpha(k+1)) \rangle$ for $\psi_1, \ldots, \psi_\alpha$. We now give a syntactic meaning to proof schemata: A proof schema Ψ with end-sequent $S(n)$ can be considered as a representation of a sequence $\lambda_0, \lambda_1, \ldots$ of ground **LKS**-proofs that prove the sequents $S(0), S(1), \ldots$. For this definition, we consider **LKS**-proofs as terms and define a rewrite system for them.

▶ Definition 2.5 (Evaluation of proof schemata). We define the rewrite rules for proof links

$$\frac{(\psi_\beta(0))}{S} \rightarrow \pi_\beta, \qquad \frac{(\psi_\beta(s(k)))}{S} \rightarrow \nu_\beta(k+1)$$

for all $1 \leq \beta \leq \alpha$. Now for $\gamma \in \mathbb{N}$ we define $\psi_\beta\downarrow_\gamma$ as a normal form of $\frac{(\psi_\beta(\gamma))}{S(\gamma)}$ under the rewrite system just given. Further, we define $\Psi\downarrow_\gamma = \psi_1\downarrow_\gamma$.

▶ **Example 2.6.** Let's consider the following proof schema $\Psi = \langle (\pi, \nu(k+1)) \rangle$ for ψ, where π is:

$$\dfrac{\dfrac{p_0 \vdash p_0}{\neg p_0, p_0 \vdash}\,\neg : l \qquad p_1 \vdash p_1}{p_0, \neg p_0 \vee p_1 \vdash p_1}\,\vee : l$$

and $\nu(k+1)$:

$$\dfrac{\dfrac{\dfrac{(\psi(k))}{p_0, \bigwedge_{i=0}^{k}(\neg p_i \vee p_{i+1}) \vdash p_{k+1}} \qquad \dfrac{\dfrac{p_{k+1} \vdash p_{k+1}}{\neg p_{k+1}, p_{k+1} \vdash}\,\neg : l \qquad p_{k+2} \vdash p_{k+2}}{p_{k+1}, \neg p_{k+1} \vee p_{k+2} \vdash p_{k+2}}\,\vee : l}{p_0, \bigwedge_{i=0}^{k}(\neg p_i \vee p_{i+1}), \neg p_{k+1} \vee p_{k+2} \vdash p_{k+2}}\,cut}{p_0, \bigwedge_{i=0}^{k+1}(\neg p_i \vee p_{i+1}) \vdash p_{k+2}}\,\wedge : l$$

Then $\Psi \downarrow_0$ is just π and $\Psi \downarrow_1$ is the following proof:

$$\dfrac{\dfrac{\dfrac{\dfrac{p_0 \vdash p_0}{\neg p_0, p_0 \vdash}\,\neg : l \qquad p_1 \vdash p_1}{p_0, \neg p_0 \vee p_1 \vdash p_1}\,\vee : l \qquad \dfrac{\dfrac{p_1 \vdash p_1}{\neg p_1, p_1 \vdash}\,\neg : l \qquad p_2 \vdash p_2}{p_1, \neg p_1 \vee p_2 \vdash p_2}\,\vee : l}{p_0, \neg p_0 \vee p_1, \neg p_1 \vee p_2 \vdash p_2}\,cut}{p_0, (\neg p_0 \vee p_1) \wedge (\neg p_1 \vee p_2) \vdash p_2}\,\wedge : l$$

▶ Proposition 2.7 (Soundness). For every $\gamma \in \mathbb{N}$ and $1 \leq \beta \leq \alpha$, $\psi_\beta \downarrow_\gamma$ is a ground **LKS**-proof with end-sequent $S_\beta(\gamma)$. Hence $\Psi \downarrow_\gamma$ is a ground **LKS**-proof with end-sequent $S(\gamma)$.

Proof. By induction on $\alpha - \beta$, we show that $\dfrac{(\psi_\beta(\gamma))}{S_\beta(\gamma)}$ rewrites to a ground **LKS**-proof with end-sequent $S_\beta(\gamma)$ for all $\beta \leq \alpha$ and all γ. If $\alpha = \beta$ we proceed by induction on γ: If $\gamma = 0$ then the proof link rewrites to π_α which is as desired by definition. If $\gamma > 0$ then the proof link rewrites to $\nu_\alpha(\gamma)$, to which we may apply the induction hypothesis since it contains only proof links to $\psi_\alpha(\gamma')$ with $\gamma' < \gamma$. This completes the induction on γ. Now let $1 \leq \beta < \alpha$. Again we proceed by induction on γ. If $\gamma = 0$ then the proof link rewrites to π_β, which by definition contains only proof links to $\psi_{\beta'}$ with $\beta' > \beta$, hence we may conclude by the (outer) induction hypothesis. If $\gamma > 0$ then the proof link rewrites to $\nu_\beta(\gamma)$ which only contains proof links to $\psi_{\beta'}(\gamma')$ with $\beta' > \beta$, and to $\psi_\beta(\gamma')$ with $\gamma' < \gamma$. The first case is treated with the outer induction hypothesis, the second case with the inner induction hypothesis. ◀

Next, we will consider the problem of cut-elimination for proof schemata. Note that trivially, for every $\gamma \in \mathbb{N}$ we can obtain a cut-free proof of $S(\gamma)$ by computing $\Psi \downarrow_\gamma$, which contains cuts, and then applying a usual cut-elimination algorithm. What we are interested in here is rather a *schematic* description of all the cut-free proofs for a parameter n. It is not possible to obtain such a description by naively applying Gentzen-style cut-elimination to the **LKS**-proofs in Ψ, since it is not clear how to handle the case

$$\dfrac{\dfrac{(\psi_1(a_1))}{\Gamma \vdash \Delta, C} \qquad \dfrac{(\psi_2(a_2))}{C, \Pi \vdash \Lambda}}{\Gamma, \Pi \vdash \Delta, \Lambda}\,cut$$

as this would require "moving the cut through a proof link". In this paper, we will go a different route: we will define a CERES method, which will be based on a *global* analysis of the proof schema. It will eventually yield the desired schematic description of the sequence of cut-free proofs, as expressed by Theorem 5.12.

3 Characteristic Clause Term

At the heart of the CERES method lies the *characteristic clause set*, which describes the cuts in a proof. The connection between cut-elimination and the characteristic clause set is that any resolution refutation of the characteristic clause set can be used as a skeleton of a proof containing only atomic cuts.

The characteristic clause set can either be defined directly as in [7], or it can be obtained via a transformation from a *characteristic clause term* as in [8]. We use the second approach here; the reason for this will be explained later.

Our main aim is to extend the usual inductive definition of the characteristic clause term to the case of proof links. This will give rise to a notion of *schematic characteristic clause term*. As usual, a *clause term* is a term built inductively from clauses and the binary symbols $\otimes, \oplus$. The usual definition of the characteristic clause term depends upon the cut-status of the formula occurrences in a proof (i.e. whether a given formula occurrence is a cut-ancestor, or not). But a formula occurrence in a proof schema gives rise to many formula occurrences in its evaluation, some of which will be cut-ancestors, and some will not. Therefore we need some machinery to track the cut-status of formula occurrences through proof links. Hence we call a set Ω of formula occurrences from the end-sequent of an **LKS**-proof π a *cut-configuration for* π.

We will represent the characteristic clause term of a proof link in our object language: For all proof symbols ψ and cut-configurations Ω we assume a unique indexed proposition symbol $\mathrm{cl}^{\Omega,\psi}$ called *clause term symbol*. The intended semantics of $\mathrm{cl}^{\Omega,\psi}_a$ is "the characteristic clause set of $\psi(a)$, with the cut-configuration Ω".

▶ Definition 3.1 (Characteristic clause term). Let π be an **LKS**-proof and Ω a cut-configuration. In the following, by $\Gamma_\Omega, \Delta_\Omega$ and Γ_C, Δ_C we will denote multisets of formulas of Ω- and cut-ancestors respectively. Let ρ be an inference in π. We define a clause term $\Theta_\rho(\pi, \Omega)$ inductively:

- if ρ is an axiom of the form $\Gamma_\Omega, \Gamma_C, \Gamma \vdash \Delta_\Omega, \Delta_C, \Delta$, then $\Theta_\rho(\pi,\Omega) = \Gamma_\Omega, \Gamma_C \vdash \Delta_\Omega, \Delta_C$
- if ρ is a proof link of the form $\dfrac{(\psi(a))}{\Gamma_\Omega, \Gamma_C, \Gamma \vdash \Delta_\Omega, \Delta_C, \Delta}$ then define Ω' as the set of formula occurrences from $\Gamma_\Omega, \Gamma_C \vdash \Delta_\Omega, \Delta_C$ and $\Theta_\rho(\pi,\Omega) = \vdash \mathrm{cl}^{\Omega',\psi}_a$
- if ρ is an unary rule with immediate predecessor ρ', then $\Theta_\rho(\pi,\Omega) = \Theta_{\rho'}(\pi,\Omega)$.
- if ρ is a binary rule with immediate predecessors ρ_1, ρ_2, then
  - if the auxiliary formulas of ρ are Ω- or cut-ancestors, then $\Theta_\rho(\pi,\Omega) = \Theta_{\rho_1}(\pi,\Omega) \oplus \Theta_{\rho_2}(\pi,\Omega)$,
  - otherwise $\Theta_\rho(\pi,\Omega) = \Theta_{\rho_1}(\pi,\Omega) \otimes \Theta_{\rho_2}(\pi,\Omega)$.

Finally, define $\Theta(\pi,\Omega) = \Theta_{\rho_0}(\pi,\Omega)$, where ρ_0 is the last inference of π, and $\Theta(\pi) = \Theta(\pi,\emptyset)$.

▶ **Example 3.2.** Let's consider the proof schema Ψ of the sequent $p_0, \bigwedge_{i=0}^{n}(\neg p_i \vee p_{i+1}) \vdash p_{n+1}$, defined in Example 2.6. We have two relevant cut-configurations: $\emptyset$ and $\{p_{n+1}\}$. The characteristic clause terms of Ψ for these cut-configurations are:

$$\begin{array}{lcl}
\Theta(\pi,\emptyset) & = & \vdash \otimes \vdash \\
\Theta(\pi,\{p_{n+1}\}) & = & \vdash \otimes \vdash p_1 \\
\Theta(\nu(k+1),\emptyset) & = & \vdash \mathrm{cl}^{\{p_{n+1}\},\psi}_k \oplus (p_{k+1} \vdash \otimes \vdash) \\
\Theta(\nu(k+1),\{p_{n+1}\}) & = & \vdash \mathrm{cl}^{\{p_{n+1}\},\psi}_k \oplus (p_{k+1} \vdash \otimes \vdash p_{k+2})
\end{array}$$

We say that a clause term is *ground* if it does not contain index variables and clause term symbols. Analogously to proof schemata, we define a notion of evaluation of characteristic clause terms:

▶ Definition 3.3 (Evaluation). We define the rewrite rules for clause term symbols for all proof symbols ψ_β and cut-configurations Ω:

$$\mathrm{cl}_0^{\Omega,\psi_\beta} \to \Theta(\pi_\beta, \Omega), \qquad \mathrm{cl}_{k+1}^{\Omega,\psi_\beta} \to \Theta(\nu_\beta(k+1), \Omega),$$

for all $1 \leq \beta \leq \alpha$. Next, let $\gamma \in \mathbb{N}$ and let $\mathrm{cl}^{\Omega,\psi_\beta} \downarrow_\gamma$ be a normal form of $\mathrm{cl}_\gamma^{\Omega,\psi_\beta}$ under the rewrite system just given. Then define $\Theta(\psi_\beta, \Omega) = \mathrm{cl}^{\Omega,\psi_\beta}$ and $\Theta(\Psi, \Omega) = \Theta(\psi_1, \Omega)$ and finally the *schematic characteristic clause term* $\Theta(\Psi) = \Theta(\Psi, \emptyset)$.

Now we can explain why we chose to define the characterstic clause set via the characteristic clause term: The clause term is closed under the rewrite rules we have given for the clause term symbols, while the notion of clause set is not (a clause will in general become a formula when subjected to the rewrite rules). Now, we prove that the notion of characteristic clause term is well-defined.

▶ Proposition 3.4. Let $\gamma \in \mathbb{N}$ and Ω be a cut-configuration, then $\Theta(\psi_\beta, \Omega) \downarrow_\gamma$ is a ground clause term for all $1 \leq \beta \leq \alpha$. Hence $\Theta(\Psi) \downarrow_\gamma$ is a ground clause term.

Proof. We proceed analogously to the proof of Proposition 2.7. ◀

Next, we show that evaluation and extraction of characteristic clause terms commute. We will later use this property to derive results on schematic characteristic clause sets from standard results on (non-schematic) CERES.

▶ Proposition 3.5. Let Ω be a cut-configuration and $\gamma \in \mathbb{N}$. Then $\Theta(\Psi \downarrow_\gamma, \Omega) = \Theta(\Psi, \Omega) \downarrow_\gamma$.

Proof. We proceed by induction on γ. If $\gamma = 0$, then $\Theta(\Psi \downarrow_0, \Omega) = \Theta(\pi_1, \Omega)$ and $\Theta(\Psi, \Omega) \downarrow_0 = \Theta(\pi_1, \Omega)$.

IH1: assume $\gamma > 0$ and for all $\beta < \gamma$, $\Theta(\Psi \downarrow_\beta, \Omega) = \Theta(\Psi, \Omega) \downarrow_\beta$. We proceed by induction on the number α of proof symbols in Ψ.

Let $\alpha = 1$. By the definition of characteristic clause term, constructions of $\Theta(\Psi \downarrow_\gamma, \Omega)$ and $\Theta(\Psi, \Omega) \downarrow_\gamma$ differ only on proof links, i.e. if $(\psi_1(k))$ is a proof link in $\nu_1(k+1)$, then by the definition of evaluation of proof schemata, $\Theta(\psi_1 \downarrow_\gamma, \Omega)$ contains the term $\Theta(\psi_1 \downarrow_\beta, \Omega')$ and by the definition of evaluation of term schemata, $\Theta(\Psi, \Omega) \downarrow_\gamma$ contains the term $\Theta(\Psi, \Omega') \downarrow_\beta$. Then by the assumption $\Theta(\psi_1 \downarrow_\beta, \Omega') = \Theta(\Psi, \Omega') \downarrow_\beta$ and we conclude that $\Theta(\psi_1 \downarrow_\gamma, \Omega) = \Theta(\Psi, \Omega) \downarrow_\gamma$.

Now, assume $\alpha > 1$ and proposition holds for all proof schemata with proof symbols less than α (IH2). Again, for proof links in $\nu_1(k+1)$ of the form $(\psi_1(k))$ the argument is the same as in the previous case. Let $(\psi_\iota(a))$, $1 < \iota \leq \alpha$, be a proof link in $\nu_1(k+1)$. Then, again, by the definition of evaluation of proof schemata, $\Theta(\psi_1 \downarrow_\gamma, \Omega)$ contains the term $\Theta(\psi_\iota \downarrow_\lambda, \Omega')$ and by the definition of evaluation of term schemata, $\Theta(\Psi, \Omega) \downarrow_\gamma$ contains the term $\Theta(\Phi, \Omega') \downarrow_\lambda$, where $\Phi = \langle (\pi_\iota, \nu_\iota(k+1)), \ldots, (\pi_\alpha, \nu_\alpha(k+1)) \rangle$. Clearly, Φ contains less than α proof symbols, then by IH2, $\Theta(\psi_\iota \downarrow_\lambda, \Omega') = \Theta(\Phi, \Omega') \downarrow_\lambda$ and we conclude that $\Theta(\psi_1 \downarrow_\gamma, \Omega) = \Theta(\Psi, \Omega) \downarrow_\gamma$. ◀

From the characteristic clause term we finally define the notion of characteristic clause set. Towards this, we define some operations on (sets of) sequents.

▶ Definition 3.6. Let $\Gamma \vdash \Delta$ and $\Pi \vdash \Lambda$ be arbitrary sequents, then we define $\Gamma \vdash \Delta \times \Pi \vdash \Lambda = \Gamma, \Pi \vdash \Delta, \Lambda$. We extend this relation to sets of sequents P, Q in a natural way: $P \times Q = \{S_P \times S_Q \mid S_P \in P, S_Q \in Q\}$.

▶ Definition 3.7 (Characteristic clause sets). Let Θ be a clause term. Then we define a clause set $|\Theta|$ in the following way:

- $|\Gamma \vdash \Delta| = \{\Gamma \vdash \Delta\}$,
- $|\Theta_1 \otimes \Theta_2| = |\Theta_1| \times |\Theta_2|$,
- $|\Theta_1 \oplus \Theta_2| = |\Theta_1| \cup |\Theta_2|$.

For an **LKS**-proof π and cut-configuration Ω, $\mathrm{CL}(\pi, \Omega) = |\Theta(\pi, \Omega)|$. We define the *standard characteristic clause set* $\mathrm{CL}(\pi) = \mathrm{CL}(\pi, \emptyset)$ and the *schematic characteristic clause set* $\mathrm{CL}(\Psi, \Omega) = |\Theta(\Psi, \Omega)|$ and $\mathrm{CL}(\Psi) = \mathrm{CL}(\Psi, \emptyset)$.

▶ **Example 3.8.** Let's consider the characteristic clause terms defined in Example 3.2. Then the sequence of $\mathrm{CL}(\Psi)\downarrow_0, \mathrm{CL}(\Psi)\downarrow_1, \mathrm{CL}(\Psi)\downarrow_2, \ldots$ is: $\{\vdash\}$, $\{\vdash p_1 \ ; \ p_1 \vdash\}$, $\{\vdash p_1 \ ; \ p_1 \vdash p_2 \ ; \ p_2 \vdash\}, \ldots$

Now we prove the main result about the characteristic clause set and lift it to the schematic case.

▶ Proposition 3.9. Let π be a ground **LKS**-proof. Then $\mathrm{CL}(\pi)$ is unsatisfiable.

Proof. By the identification of ground **LKS**-proofs with propositional **LK**-proofs, the result follows from Proposition 3.2 in [7]. ◀

▶ Proposition 3.10. $\mathrm{CL}(\Psi)\downarrow_\gamma$ is unsatisfiable for all $\gamma \in \mathbb{N}$ (i.e. $\mathrm{CL}(\Psi)$ is unsatisfiable).

Proof. By Propositions 3.5 and 2.7 $\mathrm{CL}(\Psi)\downarrow_\alpha = \mathrm{CL}(\Psi, \emptyset)\downarrow_\alpha = \mathrm{CL}(\Psi\downarrow_\alpha, \emptyset) = \mathrm{CL}(\Psi\downarrow_\alpha)$ which is unsatisfiable by Proposition 3.9. ◀

The rewrite rules from Definition 3.3 can be used as logical definitions. Hence any theorem prover for propositional schemata can be used to refute $\mathrm{CL}(\Psi)$.

4 Projections

The next step in the schematization of the CERES method consists in the definition of schematic proof projections. The aim is, in analogy with the preceding section, to construct a *schematic projection term* that can be evaluated to a set of ground **LKS**-proofs. As before, we introduce formal symbols representing sets of proofs, and again the notion of **LKS**-proof is not closed under the rewrite rules for these symbols, which is the reason for introducing the notion of projection term.

For our term notation we assume for every rule ρ of **LKS** a corresponding *rule symbol* that, by abuse of notation, we also denote by ρ. Given a unary rule ρ and an **LKS**-proof π, there are different ways to apply ρ to the end-sequent of π: namely, the choice of auxiliary formulas is free. Formally, the projection terms we construct will include this information so that evaluation is always well-defined, but we will surpress it in the notation since the choice of auxiliary formulas will always be clear from the context.

For every proof symbol ψ and cut-configuration Ω, we assume a unique proof symbol $\mathrm{pr}^{\Omega,\psi}$. Now, a *projection term* is a term built inductively from sequents and terms $\mathrm{pr}^{\Omega,\psi}(a)$, for some arithmetic expression a, using unary rule symbols, unary symbols $w^{\Gamma \vdash \Delta}$ for all sequents $\Gamma \vdash \Delta$ and binary symbols $\oplus, \otimes_\sigma$ for all binary rules σ. The symbols $\mathrm{pr}^{\Omega,\psi}$ are called *projection symbols*. The intended interpretation of $\mathrm{pr}^{\Omega,\psi}(a)$ is "the set of characteristic projections of $\psi(a)$, with the cut-configuration Ω".

▶ Definition 4.1 (Characteristic projection term). Let π be an **LKS**-proof and Ω an arbitrary cut-configuration for π. Let $\Gamma_\Omega, \Delta_\Omega$ and Γ_C, Δ_C be multisets of formulas corresponding to Ω- and cut-ancestors respectively. We define a projection term $\Xi_\rho(\pi, \Omega)$ inductively:

- If ρ corresponds to an initial sequent S, then we define $\Xi_\rho(\pi, \Omega) = S$.

- If ρ is a proof link in π of the form: $\dfrac{(\psi(a))}{\Gamma_\Omega, \Gamma_C, \Gamma \vdash \Delta_\Omega, \Delta_C, \Delta}$ then, letting Ω' be the set of formula occurrences from $\Gamma_\Omega, \Gamma_C \vdash \Delta_\Omega, \Delta_C$, define $\Xi_\rho(\pi, \Omega) = \mathrm{pr}^{\Omega', \psi}(a)$.
- If ρ is a unary inference with immediate predecessor ρ', then:
  - if the auxiliary formula(s) of ρ are Ω- or cut-ancestors, then $\Xi_\rho(\pi, \Omega) = \Xi_{\rho'}(\pi, \Omega)$,
  - otherwise $\Xi_\rho(\pi, \Omega) = \rho(\Xi_{\rho'}(\pi, \Omega))$.
- If σ is a binary inference with immediate predecessors ρ_1 and ρ_2, then:
  - if the auxiliary formulas of σ are Ω- or cut-ancestors, let $\Gamma_i \vdash \Delta_i$ be the ancestors of the end-sequent in the conclusion of ρ_i, for $i = 1, 2$, and define: $\Xi_\sigma(\pi, \Omega) = w^{\Gamma_2 \vdash \Delta_2}(\Xi_{\rho_1}(\pi, \Omega)) \oplus w^{\Gamma_1 \vdash \Delta_1}(\Xi_{\rho_2}(\pi, \Omega))$,
  - otherwise $\Xi_\sigma(\pi, \Omega) = \Xi_{\rho_1}(\pi, \Omega) \otimes_\sigma \Xi_{\rho_2}(\pi, \Omega)$.

Define $\Xi(\pi, \Omega) = \Xi_{\rho_0}(\pi, \Omega)$, where ρ_0 is the last inference of π.

We say that a projection term is *ground* if it does not contain index variables and projection symbols.

▶ **Example 4.2.** Let's consider the proof schema Ψ of the sequent $p_0, \bigwedge_{i=0}^{n}(\neg p_i \vee p_{i+1}) \vdash p_{n+1}$ defined in Example 2.6 and cut-configurations defined in Example 3.2. Then the projection terms of Ψ for those cut-configurations are:

$$\begin{array}{lcl}
\Xi(\pi, \emptyset) & = & \neg_l(p_0 \vdash p_0) \otimes_{\vee_l} p_1 \vdash p_1 \\
\Xi(\pi, \{p_{n+1}\}) & = & \neg_l(p_0 \vdash p_0) \otimes_{\vee_l} p_1 \vdash p_1 \\
\Xi(\nu(k+1), \emptyset) & = & \wedge_l(w^{\neg p_{k+1} \vee p_{k+2} \vdash p_{k+2}}(\mathrm{pr}^{\{p_{n+1}\}, \psi}(k)) \oplus \\
 & & w^{p_0, \bigwedge_{i=0}^{k}(\neg p_i \vee p_{i+1}) \vdash}(\neg_l(p_{k+1} \vdash p_{k+1}) \otimes_{\vee_l} p_{k+2} \vdash p_{k+2})) \\
\Xi(\nu(k+1), \{p_{n+1}\}) & = & \wedge_l(w^{\neg p_{k+1} \vee p_{k+2} \vdash}(\mathrm{pr}^{\{p_{n+1}\}, \psi}(k)) \oplus \\
 & & w^{p_0, \bigwedge_{i=0}^{k}(\neg p_i \vee p_{i+1}) \vdash}(\neg_l(p_{k+1} \vdash p_{k+1}) \otimes_{\vee_l} p_{k+2} \vdash p_{k+2}))
\end{array}$$

We now define the evaluation of projection terms, which is compatible with the respective definition for clause terms.

▶ Definition 4.3 (Evaluation). We define the rewrite rules for projection term symbols for all proof symbols ψ_β and cut-configurations Ω:

$$\mathrm{pr}^{\Omega, \psi_\beta}(0) \to \Xi(\pi_\beta, \Omega), \qquad \mathrm{pr}^{\Omega, \psi_\beta}(k+1) \to \Xi(\nu_\beta(k+1), \Omega),$$

for all $1 \leq \beta \leq \alpha$. Next, let $\gamma \in \mathbb{N}$ and let $\mathrm{pr}^{\Omega, \psi_\beta} \downarrow_\gamma$ be a normal form of $\mathrm{pr}^{\Omega, \psi_\beta}(\gamma)$ under the rewrite system just given. Then define $\Xi(\psi_\beta, \Omega) = \mathrm{pr}^{\Omega, \psi_\beta}$ and $\Xi(\Psi, \Omega) = \Xi(\psi_1, \Omega)$ and finally the *schematic projection term* $\Xi(\Psi) = \Xi(\Psi, \emptyset)$.

▶ Proposition 4.4. Let Ω be a cut-configuration and $\gamma \in \mathbb{N}$. Then $\Xi(\Psi\downarrow_\gamma, \Omega) = \Xi(\Psi, \Omega)\downarrow_\gamma$.

Proof. We proceed as in the proof of Proposition 3.5. ◀

We will define a map from ground projection terms to sets of ground **LKS**-proofs. For this, we need some auxiliary notation. The discussion regarding the notation for the application of rules from the beginning of this section applies here.

▶ Definition 4.5. Let ρ be an unary and σ a binary rule. Let φ, π be **LKS**-proofs, then $\rho(\varphi)$ is the **LKS**-proof obtained from φ by applying ρ, and $\sigma(\varphi, \pi)$ is the proof obtained from the proofs φ and π by applying σ.

Let P, Q be sets of **LKS**-proofs. Then $\rho(P) = \{\rho(\pi) \mid \pi \in P\}$, $P^{\Gamma \vdash \Delta} = \{\pi^{\Gamma \vdash \Delta} \mid \pi \in P\}$, where $\pi^{\Gamma \vdash \Delta}$ is π followed by weakenings adding $\Gamma \vdash \Delta$, and $P \times_\sigma Q = \{\sigma(\varphi, \pi) \mid \varphi \in P, \pi \in Q\}$.

▶ Definition 4.6. Let Ξ be a ground projection term. Then we define a set of ground **LKS**-proofs $|\Xi|$ in the following way:

- $|A \vdash A| = \{A \vdash A\}$,
- $|\rho(\Xi)| = \rho(|\Xi|)$ for unary rule symbols ρ,
- $|w^{\Gamma\vdash\Delta}(\Xi)| = |\Xi|^{\Gamma\vdash\Delta}$,
- $|\Xi_1 \oplus \Xi_2| = |\Xi_1| \cup |\Xi_2|$,
- $|\Xi_1 \otimes_\sigma \Xi_2| = |\Xi_1| \times_\sigma |\Xi_2|$ for binary rule symbols σ.

For ground **LKS**-proofs π and cut-configurations Ω we define $\mathrm{PR}(\pi, \Omega) = |\Xi(\pi, \Omega)|$ and the *standard projection set* $\mathrm{PR}(\pi) = \mathrm{PR}(\pi, \emptyset)$. For $\gamma \in \mathbb{N}$ we define $\mathrm{PR}(\Psi) \downarrow_\gamma = |\Xi(\Psi) \downarrow_\gamma |$.

The following result describes the relation between the standard projection set and characteristic clause set in the ground case. It will allow us to construct, together with a resolution refutation of $\mathrm{CL}(\Psi)$, essentially cut-free proofs of $S(\gamma)$ for all $\gamma \in \mathbb{N}$. Finally, the result is lifted to the schematic case.

▶ Proposition 4.7. Let π be a ground **LKS**-proof with end-sequent S, then for all clauses $C \in \mathrm{CL}(\pi)$, there exists a ground **LKS**-proof $\pi \in \mathrm{PR}(\pi)$ with end-sequent $S \circ C$.

Proof. By the identification of ground **LKS**-proofs with propositional **LK**-proofs, the result follows from the Definition 4.6 and Lemma 3.1 in [7]. ◀

▶ Proposition 4.8. Let $\gamma \in \mathbb{N}$, then $\mathrm{PR}(\Psi\downarrow_\gamma) = \mathrm{PR}(\Psi) \downarrow_\gamma$.

Proof. This result follows directly from Proposition 4.4. ◀

▶ Proposition 4.9. Let $\gamma \in \mathbb{N}$, then for every clause $C \in \mathrm{CL}(\Psi)\downarrow_\gamma$ there exists a ground **LKS**-proof $\pi \in \mathrm{PR}(\Psi)\downarrow_\gamma$ with end-sequent $C \circ S(\gamma)$.

Proof. By Proposition 3.5, $\mathrm{CL}(\Psi) \downarrow_\gamma = \mathrm{CL}(\Psi \downarrow_\gamma)$, and by Proposition 4.8, $\mathrm{PR}(\Psi) \downarrow_\gamma = \mathrm{PR}(\Psi\downarrow_\gamma)$. Then the result follows from Proposition 4.7, since $\Psi\downarrow_\gamma$ has end-sequent $S(\gamma)$ by definition. ◀

5 Resolution Schemata

In this section we define a notion of schematic resolution. In fact, schematic resolution refutations of $\mathrm{CL}(\Psi)$, combined with the schematic projections $\mathrm{PR}(\Psi)$ allow the construction of schematic atomic cut normal forms of the original proof schema Ψ – what is precisely the aim of a schematic CERES-method.

▶ Definition 5.1 (s-clause). Clause variables are s-clauses, and clauses are s-clauses. If s_1, s_2 are s-clauses then $s_1 \circ s_2$ is an s-clause. An s-clause *over the clause variables* $X_1, \ldots, X_\alpha$ is an s-clause with clause variables in $\{X_1, \ldots, X_\alpha\}$

▶ Definition 5.2 (clause schema). A clause schema is a term $t(a, X_1, \ldots, X_\alpha)$ w.r.t. a rewrite system $\mathcal{R}$ where a is an integer term, $X_1, \ldots, X_\alpha$ are clause variables and $\mathcal{R}$ is of the form

$$\{t(0, X_1, \ldots, X_\alpha) \to s_0; \quad t(i+1, X_1, \ldots, X_\alpha) \to t(i, s_1, \ldots, s_\alpha)\}$$

where $s_0, \ldots, s_\alpha$ are s-clauses over the variables $X_1, \ldots, X_\alpha$.

Note that Definition 5.2 admits the representation of clauses of variable length, in contrast to the sequents of our input language. We will see in Section 6 that clause variables will be needed for schematic refutations where the number of atoms in clauses increases with the parameter n.

▶ **Example 5.3.** Let X, Y be clause variables; then $Y \circ (\vdash p_0) \circ X$ and $(\vdash p_{i+1}) \circ X$ are s-clauses. The term $t(n, X, Y)$ w.r.t

$$\{t(0, X, Y) \rightarrow Y \circ (\vdash p_0) \circ X; \quad t(i+1, X, Y) \rightarrow t(i, (\vdash p_{i+1}) \circ X, Y)\}$$

is a clause schema. The normal forms of the terms $t(\alpha, \ \vdash, \ \vdash q_0)$ are just the clauses $\vdash q_0, p_0, \ldots, p_\alpha$ for $\alpha \geq 0$.

Below we generalize the concept of resolution deductions to so-called resolution terms, which define some kind of skeleton for resolution deductions.

▶ Definition 5.4 (resolution term). We define resolution terms inductively:

- s-clauses are resolution terms.
- clause schemata are resolution terms.
- Let t_1 and t_2 be resolution terms w.r.t. $\mathcal{R}_1$ and $\mathcal{R}_2$ and p an indexed atom. Then $r(t_1; t_2; p)$ is a resolution term w.r.t. $\mathcal{R}_1 \cup \mathcal{R}_2$.

Let $\mathcal{D}$ be a set of clause schemata. A resolution term t *based on* $\mathcal{D}$ is a resolution term s.t. all s-clauses and clause schemata in t are also in $\mathcal{D}$.

▶ **Example 5.5.** Let $t(n, X, Y)$ be the clause schema w.r.t. rewrite system defined in Definition 5.2. Then

$$r(r(t(n, X, Y); \ p_n \vdash; \ p_n); \ q_0, q_1 \vdash; \ q_0)$$

is a resolution term. The normal form of this term for $n = 1$, $X = \vdash$, $Y = \vdash q_0$ is $r(r(\vdash q_0, p_0, p_1; \ p_1 \vdash; \ p_1); \ q_0, q_1 \vdash; \ q_0)$. This term even represents a resolution deduction.

r-expressions without clause variables can be evaluated to resolution deductions in the usual sense:

▶ Definition 5.6 (resolvent). Let $C\colon C_1 \vdash C_2$, $D\colon D_1 \vdash D_2$ be clauses and P an atom. Then $|r(C, D, P)| = C_1, D_1 \setminus P \vdash C_2 \setminus P, D_2$, where $C_2 \setminus P$ denotes the multi-set of atoms in C_2 after removal of all occurrences of P. The clause $|r(C, D, P)|$ is called a *resolvent* of C and D on P.

Note that, in case P does not occur in D_1 and/or C_2 the clause $|r(C, D, P)|$ is not a resolvent in the usual sense, but a clause which is subsumed by C or D; thus, also in this case, $|r(C, D, P)|$ is a logical consequence of C and D.

▶ Definition 5.7 (resolution deduction). If C is a clause then C is a resolution deduction and $ES(C) = C$. If ϱ_1 and ϱ_2 are resolution deductions and $ES(\varrho_1) = D_1$, $ES(\varrho_2) = D_2$ and $|r(D_1, D_2, P)| = D$ then $r(\varrho_1, \varrho_2, P)$ is a resolution deduction and $ES(r(\varrho_1, \varrho_2, P)) = D$.

Let t be a resolution deduction and $\mathcal{C}$ be the set of all clauses occurring in t; then t is called a *resolution refutation* of $\mathcal{C}$ if $ES(t) = \vdash$.

Any resolution deduction ϱ in Definition 5.7 can easily be transformed into a resolution tree $T(\varrho)$ in an obvious way.

▶ **Example 5.8.** $T(r(r(\vdash q_0, p_0, p_1; \ p_1 \vdash; \ p_1); \ q_0, q_1 \vdash; \ q_0))$ is the resolution tree

$$\dfrac{\dfrac{\vdash q_0, p_0, p_1 \quad p_1 \vdash}{\vdash q_0, p_0} \qquad q_0, q_1 \vdash}{q_1 \vdash p_0}$$

We define a notion of resolution proof schema in the spirit of Definition 2.4:

▶ Definition 5.9 (resolution proof schema). A resolution proof schema with clause variables $X_1, \ldots, X_\beta$ is a structure $((\varrho_1, \ldots, \varrho_\alpha), \mathcal{R})$ with $\mathcal{R}$: $\mathcal{R}_1 \cup \ldots \cup \mathcal{R}_\alpha$, where the $\mathcal{R}_i$ (for $0 \leq i \leq \alpha$) are defined as follows:

$$\mathcal{R}_i = \{\varrho_i(0, X_1, \ldots, X_\beta) \to s_i, \\ \varrho_i(k+1, X_1, \ldots, X_\beta) \to t_i[\varrho_i(k, \bar{s}^i_0), \varrho_{l_1}(a^i_1, \bar{s}^i_1), \ldots, \varrho_{l_{j(i)}}(a^i_{j(i)}, \bar{s}^i_{j(i)})]\}$$

where

- s_i is a parameter-free resolution term,
- $a^i_1, \ldots, a^i_{j(i)}$ are arithmetic terms,
- $\bar{s}^i_0, \ldots, \bar{s}^i_{j(i)}$ are vectors of clause schemata,
- the $t_i[\varrho_i(k, \bar{s}^i_0), \varrho_{l_1}(a^i_1, \bar{s}^i_1), \ldots, \varrho_{l_{j(i)}}(a^i_{j(i)}, \bar{s}^i_{j(i)})]$ are resolution terms after replacement of some clause schemata by the terms $\varrho_i(k, \bar{s}^i_0), \varrho_{l_1}(a^i_1, \bar{s}^i_1), \ldots, \varrho_{l_r}(a^i_{j(i)}, \bar{s}^i_{j(i)})$ where $i < \min\{l_1, \ldots, l_{j(i)}\}$ and $\max\{l_1, \ldots, l_{j(i)}\} \leq \alpha$.

▶ Definition 5.10 (resolution refutation schema). A resolution proof schema is called a *resolution refutation schema of a clause schema* $\mathcal{C}(n)$ if there exist clauses $C_1, \ldots, C_\alpha$ s.t., for every assignment β for n, $\varrho_1(n, C_1, \ldots, C_\alpha)\downarrow_\beta$ is a resolution refutation of $\mathcal{C}(n)\downarrow_\beta$.

▶ **Example 5.11.** Let CL(Ψ) be the schema of characteristic clauses from Example 2.6. Note that $\mathrm{CL}(\Psi)\downarrow_\alpha = \{\vdash p_1;\ p_1 \vdash p_2;\ \ldots;\ p_{\alpha-1} \vdash p_\alpha;\ p_\alpha \vdash\}$ for $\alpha > 0$. The resolution proof schema $((\varrho, \delta), \mathcal{R})$ is a resolution refutation schema of CL(Ψ), where

$$\begin{aligned}\mathcal{R} \quad = \quad & \{\varrho(0) \to \vdash,\ \varrho(k+1) \to r(\delta(k);\ p_{k+1} \vdash;\ p_{k+1}),\\ & \delta(0) \to \vdash p_1,\ \delta(k+1) \to r(\delta(k);\ p_{k+1} \vdash p_{k+2};\ p_{k+1})\}.\end{aligned}$$

A refutation of the clause set $\mathrm{CL}(\Psi)\downarrow_\alpha$ is then defined by the term $\varrho(n)\downarrow_\alpha$.

Note that in this refutation schema we did not make use of clause variables.

Finally, we can summarize the CERES method of cut-elimination for proof schemata by defining the whole CERES-procedure CERES-s on schemata (where Ψ is a proof schema):

Phase 1 of CERES-s: (schematic construction)

- compute CL(Ψ);
- compute PR(Ψ);
- construct a resolution refutation schema ϱ of CL(Ψ).

Phase 2 of CERES-s: (evaluation, given a number α)

- compute $\mathrm{CL}(\Psi)\downarrow_\alpha$;
- compute $\mathrm{PR}(\Psi)\downarrow_\alpha$;
- compute $\varrho_1(n, C_1, \ldots, C_\beta)\downarrow_\alpha$ and $T_\alpha\colon T(\varrho_1(n, C_1, \ldots, C_\beta)\downarrow_\alpha)$;
- append the corresponding projections in $\mathrm{PR}(\Psi)\downarrow_\alpha$ to T_α and propagate the contexts down in the proof.

▶ **Theorem 5.12.** *Let Ψ be a proof schema with end-sequent $S(n)$. Then the evaluation of CERES-s produces for all $\alpha \in \mathbb{N}$ a ground* **LKS**-*proof π of $S(\alpha)$ with at most atomic cuts such that its size $|\pi|$ polynomial in $|\varrho_1(n, C_1, \ldots, C_\beta)\downarrow_\alpha| \cdot |\mathrm{PR}(\Psi)\downarrow_\alpha|$.*

Proof. Let $\alpha \in \mathbb{N}$. By Proposition 4.8 we obtain for any clause in $\mathcal{C}_\alpha\colon \mathrm{CL}(\Psi)\downarrow_\alpha$ a corresponding projection of the ground proof ψ_α in $\mathrm{PR}(\Psi)\downarrow_\alpha$. Let $R = ((\varrho_1, \ldots, \varrho_\beta), \mathcal{R})$ be a resolution refutation schema for CL(Ψ) constructed in phase 1 of CERES-s and T_α the corresponding tree. Clearly the length of any projection is at most $|\mathrm{PR}(\Psi)\downarrow_\alpha|$ and $|T(\alpha)|$ is polynomial in $\varrho_1(n, C_1, \ldots, C_{\beta'})\downarrow_\alpha$. Moreover, the resulting proof π_α of $S(\alpha)$ obtained in the last step of phase 2 contains at most atomic cuts. ◀

While the schematic characteristic clause set and the schematic projections could be obtained fully automatically, this is not the case for the resolution refutation schemata. For regular schemata, however, it is known that schematic tableaux-proofs can always be computed (see [5]). In [5] a transformation of the schematic tableaux-proofs to specifications of resolution proof schemata is given. However, the format of this specification is complex, hard to read and problematic to human interpretation (which is always the last step in an application of CERES); hence, a translation to our format of resolution refutation schemata appears desirable and will be part of future investigations.

6 The Adder Proof

In this section we give a more complex example of schematic cut-elimination. We formalize a proof of the theorem that *a circuit bit adder is commutative* and use the lemma that *the carry bits are equal*. First we introduce some "shortcuts" for formulas ($\hat{\mathbf{S}}$ denotes the sum and $\hat{\mathbf{C}}$ the carry bit computation):

$$\begin{array}{rcl}
A \oplus B & =_{def} & (A \wedge \neg B) \vee (\neg A \wedge B) \\
A \Leftrightarrow B & =_{def} & (\neg A \vee B) \wedge (\neg B \vee A) \\
\hat{\mathbf{S}}_i & =_{def} & S_i \Leftrightarrow (A_i \oplus B_i) \oplus C_i \\
\hat{\mathbf{S}}'_i & =_{def} & S'_i \Leftrightarrow (B_i \oplus A_i) \oplus C'_i \\
\hat{\mathbf{C}}_i & =_{def} & C_{i+1} \Leftrightarrow (A_i \wedge B_i) \vee (C_i \wedge A_i) \vee (C_i \wedge B_i) \\
\hat{\mathbf{C}}'_i & =_{def} & C'_{i+1} \Leftrightarrow (B_i \wedge A_i) \vee (C'_i \wedge B_i) \vee (C'_i \wedge A_i) \\
Adder_n & =_{def} & \bigwedge_{i=0}^{n} \hat{\mathbf{S}}_i \wedge \bigwedge_{i=0}^{n} \hat{\mathbf{C}}_i \wedge \neg C_0 \\
Adder'_n & =_{def} & \bigwedge_{i=0}^{n} \hat{\mathbf{S}}'_i \wedge \bigwedge_{i=0}^{n} \hat{\mathbf{C}}'_i \wedge \neg C'_0 \\
EqC_n & =_{def} & \bigwedge_{i=0}^{n} (C_i \Leftrightarrow C'_i) \\
EqS_n & =_{def} & \bigwedge_{i=0}^{n} (S_i \Leftrightarrow S'_i)
\end{array}$$

The schematic proof Ψ is defined via the proof symbols $\psi \prec \varphi \prec \phi \prec \chi$ (by abuse of notation, we use the same meta symbols for proofs and proof symbols) and the following definitions:

$$\cfrac{\cfrac{\cfrac{(\varphi(k))}{\neg C_0, \neg C'_0, \bigwedge_{i=0}^{k} \hat{\mathbf{C}}_i, \bigwedge_{i=0}^{k} \hat{\mathbf{C}}'_i \vdash EqC_k} \qquad \cfrac{(\chi(k))}{EqC_k, \bigwedge_{i=0}^{k} \hat{\mathbf{S}}_i, \bigwedge_{i=0}^{k} \hat{\mathbf{S}}'_i \vdash EqS_k}}{\neg C_0, \neg C'_0, \bigwedge_{i=0}^{k} \hat{\mathbf{C}}_i, \bigwedge_{i=0}^{k} \hat{\mathbf{C}}'_i, \bigwedge_{i=0}^{k} \hat{\mathbf{S}}_i, \bigwedge_{i=0}^{k} \hat{\mathbf{S}}'_i \vdash EqS_k}\ cut}{Adder_k, Adder'_k \vdash EqS_k}\ \wedge\colon l*$$

Because of lack of space we omit all basic cases and the purely propositional parts of the proofs.[2] We define $\varphi(k+1)$ as:

$$\cfrac{\cfrac{\cfrac{(\varphi(k))}{\neg C_0, \neg C'_0, \bigwedge_{i=0}^{k} \hat{\mathbf{C}}_i, \bigwedge_{i=0}^{k} \hat{\mathbf{C}}'_i \vdash EqC_k} \qquad \cfrac{(\phi(k))}{\neg C_0, \neg C'_0, \bigwedge_{i=0}^{k} \hat{\mathbf{C}}_i, \bigwedge_{i=0}^{k} \hat{\mathbf{C}}'_i \vdash C_{k+1} \Leftrightarrow C'_{k+1}}}{\neg C_0, \neg C'_0, \bigwedge_{i=0}^{k} \hat{\mathbf{C}}_i, \bigwedge_{i=0}^{k} \hat{\mathbf{C}}'_i \vdash EqC_{k+1}}\ \wedge\colon r, c\colon l*}{\neg C_0, \neg C'_0, \bigwedge_{i=0}^{k+1} \hat{\mathbf{C}}_i, \bigwedge_{i=0}^{k+1} \hat{\mathbf{C}}'_i \vdash EqC_{k+1}}\ \wedge\colon l*$$

where $\phi(k+1)$ is:

[2] A fully formal proof can be found here: http://www.logic.at/asap/

$$\dfrac{\dfrac{\begin{array}{c}(\phi(k))\\ \hdashline \neg C_0, \neg C'_0, \bigwedge_{i=0}^{k} \hat{\mathbf{C}}_i, \bigwedge_{i=0}^{k} \hat{\mathbf{C}}'_i \vdash C_{k+1} \Leftrightarrow C'_{k+1}\end{array} \qquad \begin{array}{c}\vdots\\ C_{k+1} \Leftrightarrow C'_{k+1}, \hat{\mathbf{C}}_{k+1}, \hat{\mathbf{C}}'_{k+1} \vdash C_{k+2} \Leftrightarrow C'_{k+2}\end{array}}{\neg C_0, \neg C'_0, \bigwedge_{i=0}^{k} \hat{\mathbf{C}}_i, \bigwedge_{i=0}^{k} \hat{\mathbf{C}}'_i, \hat{\mathbf{C}}_{k+1}, \hat{\mathbf{C}}'_{k+1} \vdash C_{k+2} \Leftrightarrow C'_{k+2}}\, cut}{\neg C_0, \neg C'_0, \bigwedge_{i=0}^{k+1} \hat{\mathbf{C}}_i, \bigwedge_{i=0}^{k+1} \hat{\mathbf{C}}'_i \vdash C_{k+2} \Leftrightarrow C'_{k+2}}\, \wedge\colon l*$$

and finally, $\chi(k+1)$ is:

$$\dfrac{\dfrac{\begin{array}{c}(\chi(k))\\ \hdashline EqC_k, \bigwedge_{i=0}^{k} \hat{\mathbf{S}}_i, \bigwedge_{i=0}^{k} \hat{\mathbf{S}}'_i \vdash EqS_k\end{array} \qquad \begin{array}{c}\vdots\\ C_{k+1} \Leftrightarrow C'_{k+1}, \hat{\mathbf{S}}_{k+1}, \hat{\mathbf{S}}'_{k+1} \vdash S_{k+1} \Leftrightarrow S'_{k+1}\end{array}}{EqC_k, \bigwedge_{i=0}^{k} \hat{\mathbf{S}}_i, \bigwedge_{i=0}^{k} \hat{\mathbf{S}}'_i, C_{k+1} \Leftrightarrow C'_{k+1}, \hat{\mathbf{S}}_{k+1}, \hat{\mathbf{S}}'_{k+1} \vdash EqS_{k+1}}\, \wedge\colon r}{EqC_{k+1}, \bigwedge_{i=0}^{k+1} \hat{\mathbf{S}}_i, \bigwedge_{i=0}^{k+1} \hat{\mathbf{S}}'_i \vdash EqS_{k+1}}\, \wedge\colon l*$$

The schematic clause term and the projection term of this proof schema can be found at http://www.logic.at/asap/. Below we give first the sequence of characteristic clause sets CL_α (for $\mathrm{CL}_\alpha = \mathrm{CL}(\Psi){\downarrow_\alpha}$) and then a resolution refutation schema R of $\mathrm{CL}(\Psi)$.

$$\begin{array}{l}
\mathrm{CL}_\alpha = \{C_0 \vdash;\ C'_0 \vdash\} \cup \mathcal{D}_\alpha \cup \mathcal{C}_\alpha.\ \ \mathcal{D}_0 = \emptyset,\ \mathcal{D}_1 = \{C_1 \vdash C'_1;\ C'_1 \vdash C_1\}.\\
\mathcal{D}_{\beta+1} = \mathcal{D}_\beta \cup \{C_{\beta+1} \vdash C'_{\beta+1}, C_\beta;\ C'_{\beta+1} \vdash C_{\beta+1}, C'_\beta;\\
\quad C'_\beta, C_{\beta+1} \vdash C'_{\beta+1};\ C_\beta, C'_{\beta+1} \vdash C_{\beta+1}\}.\\
\mathcal{C}_0 = \{\vdash C_0, C'_0;\ C_0, C'_0 \vdash\},\ \mathcal{C}_{\beta+1} = \mathcal{C}_\beta \circ \{\vdash C_{\beta+1}, C'_{\beta+1}\} \cup \mathcal{C}_\beta \circ \{C_{\beta+1}, C'_{\beta+1} \vdash\}.
\end{array}$$

A resolution refutation schema of $\mathrm{CL}(\Psi)$ is $R = ((\varrho, \delta, \eta), \mathcal{R})$ where $\mathcal{R}$ is the following system:

$$\begin{array}{rcl}
\{\varrho(0, X) & \to & r(r((\vdash C_0, C'_0) \circ X;\ C_0 \vdash;\ C_0);\ C'_0 \vdash;\ C'_0),\\
\varrho(k+1, X) & \to & r(r(\varrho(k, (\vdash C_{k+1}, C'_{k+1}) \circ X);\ \eta(k);\ C'_{k+1});\\
& & \quad r(\delta(k);\ \varrho(k, (C_{k+1}, C'_{k+1} \vdash) \circ X);\ C'_{k+1});\\
& & \quad C_{k+1}),\\
\delta(0) & \to & C_1 \vdash C'_1,\\
\delta(k+1) & \to & r(C_{k+2} \vdash C'_{k+2}, C_{k+1};\ r(\delta(k);\ C'_{k+1}, C_{k+2} \vdash C'_{k+2};\ C'_{k+1});\ C_{k+1}),\\
\eta(0) & \to & C'_1 \vdash C_1,\\
\eta(k+1) & \to & r(C'_{k+2} \vdash C_{k+2}, C'_{k+1};\ r(\eta(k);\ C_{k+1}, C'_{k+2} \vdash C_{k+2};\ C_{k+1});\ C'_{k+1})\}
\end{array}$$

Note that, for the specification of this resolution refutation schema, the use of the clause variable X is vital. For the derivations $\delta(n)$ and $\eta(n)$ we get $ES(\delta(n){\downarrow_\alpha}) = C_{\alpha+1} \vdash C'_{\alpha+1}$, $ES(\eta(n){\downarrow_\alpha}) = C'_{\alpha+1} \vdash C_{\alpha+1}$. The resolution refutation of CL_α is defined by $\varrho(n, \vdash){\downarrow_\alpha}$.

It is easy to verify that, for all α, $\varrho_\alpha\colon \varrho(n, \vdash){\downarrow_\alpha}$ is a resolution refutation of CL_α. For ϱ_0 this is trivially realized. Assume that ϱ_α is a resolution refutation of CL_α. Then, using this induction hypothesis, $\varrho_{\alpha+1}$ is a resolution refutation if

$$\begin{array}{l}
r(r(\vdash C_{\alpha+1}, C'_{\alpha+1};\ C'_{\alpha+1} \vdash C_{\alpha+1};\ C'_{\alpha+1});\\
\quad r(C_{\alpha+1} \vdash C'_{\alpha+1};\ C_{\alpha+1}, C'_{\alpha+1} \vdash;\ C'_{\alpha+1});\\
\quad C_{\alpha+1})
\end{array}$$

is a resolution refutation, which it obviously is. That $\varrho_{\alpha+1}$ is also a refutation of $\mathrm{CL}_{\alpha+1}$ follows from the recursive definition of CL_α.

Although the above proof is not of mathematical importance, it has some interesting properties. First, we observe that for all $\alpha \in \mathbb{N}$, CL_α contains $2^{\alpha+2}$ clauses. Second, while the original proof schema (with cuts) used the lemma that all carry bits are equal, from the resolution refutation (which is a skeleton of the cut-free proof) we can see that the cut-free proof now derives the equality of the carry bits one-by-one.

Acknowledgment

The authors would like to thank Vincent Aravantinos for many discussions and insightful remarks on the topic of the present paper.

References

1 Martin Aigner and Günter Ziegler. *Proofs from THE BOOK*. Springer, 1999.

2 Vincent Aravantinos, Ricardo Caferra, and Nicolas Peltier. A schemata calculus for propositional logic. In *Automated Reasoning with Analytic Tableaux and Related Methods*, volume 5607 of *Lecture Notes in Computer Science*, pages 32–46, 2009.

3 Vincent Aravantinos, Ricardo Caferra, and Nicolas Peltier. RegSTAB: A SAT-Solver for Propositional Iterated Schemata. In *International Joint Conference on Automated Reasoning*, pages 309–315, 2010.

4 Vincent Aravantinos, Ricardo Caferra, and Nicolas Peltier. Decidability and undecidability results for propositional schemata. *Journal of Artificial Intelligence Research*, 40:599–656, 2011.

5 Vincent Aravantinos and Nicolas Peltier. Generating schemata of resolution proofs. In Martin Giese and Roman Kuznets, editors, *TABLEAUX 2011 Workshops, Tutorials, and Short Papers*, pages 16–30, 2011.

6 Matthias Baaz, Stefan Hetzl, Alexander Leitsch, Clemens Richter, and Hendrik Spohr. CERES: An analysis of Fürstenberg's proof of the infinity of primes. *Theoretical Computer Science*, 403:160–175, 2008.

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

8 Matthias Baaz and Alexander Leitsch. Towards a clausal analysis of cut-elimination. *Journal of Symbolic Computation*, 41(3-4):381–410, 2006.

9 Matthias Baaz and Alexander Leitsch. *Methods of Cut-Elimination*, volume 34 of *Trends in Logic*. Springer, 2011.

10 David Baelde and Dale Miller. Least and greatest fixed points in linear logic. In *LPAR 2007*, volume 4790 of *LNCS*, pages 92–106, 2007.

11 James Brotherston. Cyclic proofs for first-order logic with inductive definitions. In B. Beckert, editor, *Automated Reasoning with Analytic Tableaux and Related Methods*, volume 3702 of *Lecture Notes in Computer Science*, pages 78–92, 2005.

12 Gerhard Gentzen. Untersuchungen über das logische Schließen I. *Mathematische Zeitschrift*, 39(1):176–210, dec 1935.

13 Stefan Hetzl, Alexander Leitsch, and Daniel Weller. CERES in higher-order logic. *Annals of Pure and Applied Logic*, 162(12):1001–1034, 2011.

14 Raymond McDowell and Dale Miller. Cut-elimination for a logic with definitions and induction. *Theoretical Computer Science*, 232(1–2):91–119, 2000.

15 Christoph Sprenger and Mads Dam. On the structure of inductive reasoning: Circular and tree-shaped proofs in the μ-calculus. In *FOSSACS 2003*, volume 2620 of *LNCS*, pages 425–440, 2003.

16 William W. Tait. Normal derivability in classical logic. In *The Syntax and Semantics of Infinitary Languages*, volume 72 of *Lecture Notes in Mathematics*, pages 204–236. Springer Berlin, 1968.