Integrating a Global Induction Mechanism into a Sequent Calculus

David M. Cerna[1*] and Michael Lettmann[2]

[1] Reseach Institute for Symbolic Computation
Johannes Kepler University, Linz, Austria
david.cerna@risc.jku.at
[2] Institute of Information Systems
Technische Universität Wien, Vienna, Austria
lettmann@logic.at

Abstract. Most interesting proofs in mathematics contain an inductive argument which requires an extension of the **LK**-calculus to formalize. The most commonly used calculi contain a separate rule or axiom which reduces the important proof theoretic properties of the calculus. In such cases cut-elimination does not result in analytic proofs, i.e. every formula occurring in the proof is a subformula of the end sequent. Proof schemata are a generalization of **LK**-proofs able to simulate induction by linking proofs, indexed by a natural number, together. Using a global cut-elimination method a normal form can be reached which allows a schema of *Herbrand Sequents* to be produced, an essential step for proof analysis in the presence of induction. However, proof schema have only been studied in a limited context and are currently defined for a very particular proof structure based on a slight extension of the **LK**-calculus. The result is an opaque and complex formalization. In this paper, we introduce a calculus integrating the proof schema formalization and in the process we elucidate properties of proof schemata which can be used to extend the formalism.

1 Introduction

The schematic construction of objects that forms the basis of proof schemata, as described in this paper, was introduced by V. Aravantinos et al. [2–7]. Initially, they considered formulas of an indexed propositional logic with a single *free numeric parameter* and with two new logical connectors, i.e. $\vee$-iteration and $\wedge$-iteration. They developed a tableau based decision procedure for the satisfiability of a monadic[3] fragment of this logic. An extension to a special case of multiple parameters was also investigated by D. Cerna [13]. In a more recent work, V. Aravantinos et al. [6] introduced a superposition resolution calculus for a clausal

*This research has been partially supported by the Austrian Science Fund (FWF) under the project P 28789-N32.

[3] In this fragment the use of schematic constructors is restricted to one free parameter per formula.

representation of indexed propositional logic. The calculus provided decidability results for an even larger fragment of the monadic fragment. The clausal form allows an easy extension to indexed predicate logic, though all decidability results are lost. In either case, the refutations producible by the calculus for unsatisfiable clause sets is quite restricted[4].

Nonetheless, these results inspired investigations into the use of schemata as an alternative formalization of induction for proof analysis and transformation. This is not the first alternative formalization of induction with respect to Peano arithmetic [25]. However, all existing examples [11, 12, 21], to the best of our knowledge, lack a proof normal form or *subformula-like property*[5], i.e. every formula occurring in the proof is a subformula of a formula occurring in the end sequent. What we mean by this is that performing cut-elimination in the presence of induction results in a non-analytic proof: some part of the argument is not directly connected to the theorem being proven. Two important constructions extractable from proofs with the subformula property, *Herbrand sequents* [19, 25] and *expansion trees* [22], are not extractable from proofs within these calculi. While Herbrand sequents allow the representation of the propositional content of first-order proofs, expansion trees generalize Herbrand's theorem.

Note that in [20] a finite representation of a sequence of Herbrand sequents is produced, a *Herbrand system*. Of course, such objects are not derivable from a finite set of ground instances, though instantiating the free parameter of a Herbrand systems results in a sequent derivable from a finite set of ground instances. Bounds on the size of these sets of ground instances, in terms of the free parameter, exists [16]. Representing a ground derivation for the Herbrand system as a proof schema itself is still an open problem and is a motivating factor of this paper. In some cases, the resulting Herbrand system is not formalizable as a proof schema in the sense of [16, 20] due to the restriction placed on proof schema construction, see [14]. Lifting these restrictions is non-trivial. Our goal is to design a calculus which easily allows one to relax the restrictions.

The first proof analysis carried out using a rudimentary schematic formalism was Baaz et al. [8], where proof analysis of Fürstenberg's proof of the infinitude of primes was successfully performed using **CERES** [9]. The formalism discussed in this paper is an extension of CERES introduced by C. Dunchev et al. [17]. It allows the extraction of a schema of Herbrand sequents from the resulting normal form produced by cut-elimination in the presence of induction. Problematically, the method of cut-elimination introduced in [17] is not known to be complete, in terms of cut-elimination, and is very difficult to use. For an example of the difficulties see Cerna and Leitsch's work [15]. A much improved version of this cut-elimination method has been introduced in [20]. using the superposition

[4] A formal analysis has not be performed but through conversations with the authors and their collaborators a polynomial bound on the size of the produced refutations is expected.

[5] A proof fulfilling the subformula property can be referred to as *analytic*. By subformula-like, we mean that the proof is non-analytic, but still allows the extraction of objects important for proof analysis which rely on analyticity.

resolution calculus of [6]. The method is complete and always produces a schema of Herbrand sequents, but as mentioned earlier, it is quite weak, also in comparison to the method of [17]. It relies on the superposition resolution calculus of [6]. The method of [17] can formalize proof normal forms with a non-elementary length with respect to the size of the end sequent[6].

In both cases, the concept of *proof schema* is designed to encapsulate a sequence of interacting proofs (the proofs are defined using the **LKS**-calculus) [17] which can be joined in a particular way allowing the construction of a valid **LK**-proof for any natural number. The **LKS**-calculus introduces concepts such as *links* but does not place restrictions on what a sound application of the rule is, rather an additional construction, *proof schemata*, provides the soundness conditions. However, as one might expect, most sequences of proofs will result in a invalid proof, multiple proofs when only one is desired, or proofs which are more complex than necessary, i.e. repetition or unnecessary constructions. These issues make extensions of proofs schemata, i.e. adding additional parameters and/or more complexity well-ordering conditions as well as compression and proof optimization, increasingly difficult. However, even more pressing is that the restrictions placed on *proof schemata* avoid sequences which will result in valid **LK**-proof when instantiated. As an example, the schema of Herbrand sequents extracted in [14] from the proof analysis of the infinitary Pigeonhole Principle can not be formulated as a proof schema in the current framework.

In this work we present a novel calculus for proof schemata which provides a better understanding of the restrictions placed on proof schema construction in previous work. The calculus implicitly enforces the sound application of inferences and in doing so it provides an easy mechanism for weakening the soundness conditions. Moreover, we show completeness with respect to the *k-induction* fragment of Peano arithmetic [20], thus showing that the current calculus is equivalent to previous formalisms. However, one of the most interesting results is that *component collections* (an abstraction of sequents used in our calculus) can be interpreted as a sequence of inductions (similar to the fusion method introduced by Gentzen [18]) rather than as a tree of inductions. This is unexpected given that proof schemata enforce a very specific tree structure which is partially what makes introduction of multiple parameters so difficult. This flattened structure allows us to easily consider a *component* as separate from the parameter of the proof schema and separately from the proof schema itself. Taking advantage of this property in order to weaken the current restrictions built into the framework and to formalize multiple parameter schemata is planned for future work.

The rest of this paper is as follows: In Section 2, we discuss the necessary background knowledge needed for the results. In Section 3 we discuss the evaluation and interpretation of proof schemata. In Section 4, we introduce the concept of the calculus. In Section 5, we show soundness and completeness of the calculus. In Section 6, we conclude the paper and mention possible applications, future work, and open problems.

[6] See Orevkov's proof [23] or Boolos' proof [10].

2 Preliminaries

In this section, we provide a formal construction of proof schemata.

2.1 Schematic Language

We work in a two-sorted version of classical first-order logic. The first sort we consider is ω, in which every term normalizes to a *numeral*, i.e a term inductively constructable by $N \Rightarrow s(N) \mid 0$, such that $s(N) \neq 0$ and $s(N) = s(N') \rightarrow N = N'$. We will denote numerals by lowercase greek letters, i.e. α, β, γ, etc. Furthermore, the omega sort includes a countable set of parameter symbols $\mathcal{N}$. For this work, we will only need a single parameter symbol which in most cases we denote by n. We use k, k' to represent numeric expressions containing the parameter. The parameter symbol n will be referred to as the *free parameter*.

The second sort ι (individuals) is a standard first-order term language extended by *defined function symbols* and *schematic variable symbols*. To distinguish defined and uninterpreted function symbols we partition the functions of ι into two categories, *uninterpreted function symbols* $\mathbf{F_u}$ and *defined function symbols* $\mathbf{F_d}$. Defined function symbols will be denoted with $\widehat{\cdot}$. Schematic variable symbols are variables of the type $\omega \rightarrow \iota$ used to construct sequences of variables, essentially a generalization of the standard concept of a variable. Given a schematic variable x instantiated by a numeral α we get a variable of the ι sort $x(\alpha)$.

Formula schemata, a generalization of formulas including defined predicate symbols, are defined inductively using the standard logical connectives from uninterpreted and defined predicate symbols. Analogously, we label symbols as defined predicate symbols with $\widehat{\cdot}$. A *schematic sequent* is a pair of two multisets of formula schemata Δ, Π denoted by $\Delta \vdash \Pi$. We will denote multisets of formula schemata by uppercase greek letters unless it causes confusion.

Note that we extend the **LK**-calculus [25] (see appendix **??**) to the **LKE**-calculus [17] by adding an inference rule for the construction of defined predicate and function symbols and a set of convergent rewrite rules $\mathcal{E}$ (equational theory) to our interpretation. The rules of $\mathcal{E}$ take the following form $\widehat{f}(\bar{t}) = E$, where $\bar{t}$ contains no defined symbols, and either $\widehat{f}$ is a function symbol of range ι and E is a term or $\widehat{f}$ is a predicate symbol and E is a formula schema.

Definition 1 (LKE). *Let $\mathcal{E}$ be an equational theory. **LKE** is an extension of **LK** by the $\mathcal{E}$ inference rule*

$$\frac{S(t)}{S(t')}\,\mathcal{E}$$

where the term or formula schema t in the sequent S is replaced by a term or formula schema t' for $\mathcal{E} \models t = t'$.

Example 1. Iterated version of $\vee$ and $\wedge$ (the defined predicates are abbreviated as $\bigvee$ and $\bigwedge$) can be defined using the following equational theory:

$$\bigvee_{i=0}^{0} P(i) = \bigwedge_{i=0}^{0} P(i) = P(0), \qquad \bigvee_{i=0}^{s(y)} P(i) = \bigvee_{i=0}^{y} P(i) \vee P(s(y)), \qquad \bigwedge_{i=0}^{s(y)} P(i) = \bigwedge_{i=0}^{y} P(i) \wedge P(s(y)).$$

2.2 The LKS-calculus and Proof Schemata

Schematic proofs are a finite ordered list of *proof schema components* which can interact with each other. This interaction is defined using so-called *links*, a 0-ary inference rule we add to **LKE**-calculus: Let $S(k, \bar{x})$ be a sequent where $\bar{x}$ is a vector of schematic variables. By $S(k, \bar{t})$ we denote $S(k, \bar{x})$ where $\bar{x}$ is replaced by $\bar{t}$, respectively, and $\bar{t}$ is a vector of terms of appropriate type. Furthermore, we assume a countably infinite set $\mathcal{B}$ of *proof symbols* denoted by $\varphi, \psi, \varphi_i, \psi_j$. The expression

$$\frac{(\varphi, k, \bar{t})}{S(k, \bar{t})}$$

is called a link with the intended meaning that there is a proof called φ with the end-sequent $S(k, \bar{x})$. Let k be a numeric expression, then $\mathcal{V}(k)$ is the set of parameters in k. We refer to a link as an *E-link* if $\mathcal{V}(k) \subseteq E$. Note that in this work $E = \{n\}$ or $E = \varnothing$.

Definition 2 (LKS). ***LKS*** *is an extension of* ***LKE****, where links may appear at the leaves of a proof.*

Definition 3 (Proof Schema Component). *Let $\psi \in \mathcal{B}$ and $n \in \mathcal{N}$. A* proof schema component $\mathbf{C}$ *is a triple $(\psi, \pi, \nu(k))$ where π is an* ***LKS****-proof only containing $\varnothing$-links and $\nu(k)$ is an* ***LKS****-proof containing $\{n\}$-links. The end-sequents of the proofs are $S(0, \bar{x})$ and $S(k, \bar{x})$, respectively. Given a proof schema component $\mathbf{C} = (\psi, \pi, \nu(k))$ we define $\mathbf{C}.1 = \psi$, $\mathbf{C}.2 = \pi$, and $\mathbf{C}.3 = \nu(k)$.*

If $\nu(k)$ of a proof schema component $(\psi, \pi, \nu(k))$ contains a link to ψ it will be referred to as *cyclic*, otherwise it is *acyclic*.

Definition 4. *Let $\mathbf{C}_1$ and $\mathbf{C}_2$ be proof schema components such that $\mathbf{C}_1.1$ is distinct from $\mathbf{C}_2.1$ and $n \in \mathcal{N}$. We say $\mathbf{C}_1 \succ^* \mathbf{C}_2$ if there are no links from $\mathbf{C}_2$ to $\mathbf{C}_1$ and all links that call $\mathbf{C}_1$ or $\mathbf{C}_2$ are $\{n\}$-links of the following form:*

$$\frac{(\mathbf{C}_1.1, k, \bar{a})}{S(k, \bar{a})} \qquad\qquad \frac{(\mathbf{C}_2.1, t, \bar{b})}{S'(t, \bar{b})}$$

where t is a numeric expression such that $\mathcal{V}(t) \subseteq \{n\}$, k' is a sub-term of k, and $\bar{a}$ and $\bar{b}$ are vectors of terms from the appropriate sort. $S(k, \bar{a})$ and $S'(t, \bar{b})$ are the end sequents of components $\mathbf{C}_1$ and $\mathbf{C}_2$ respectively.

Let Ψ be a set of proof schema components. We say $\mathbf{C}_1 \succ \mathbf{C}_2$ if $\mathbf{C}_1 \succ^* \mathbf{C}_2$ and $\mathbf{C}_1 \succ^* \mathbf{D}$ holds for all proof schema components $\mathbf{D}$ of Ψ with $\mathbf{C}_2 \succ^* \mathbf{D}$.

Definition 5 (Proof Schema [17]). *Let $\mathbf{C}_1, \cdots, \mathbf{C}_m$ be proof schema components such that $\mathbf{C}_i.1$ is distinct for $1 \leqslant i \leqslant m$ and $n \in \mathcal{N}$. Let the end sequents of $\mathbf{C}_1$ be $S(0, \bar{x})$ and $S(k, \bar{x})$. We define $\Psi = \langle \mathbf{C}_1, \cdots, \mathbf{C}_m \rangle$ as a* proof schema *if $\mathbf{C}_1 \succ \ldots \succ \mathbf{C}_m$.*

We call $S(k, \bar{x})$ the end sequent of Ψ and assume an identification between the formula occurrences in the end sequents of the proof schema components so that we can speak of occurrences in the end sequent of Ψ. The class of all proof schemata will be denoted by $\boldsymbol{\Upsilon}$.

For any proof schema $\Phi \in \boldsymbol{\Upsilon}$, such that $\Phi = \langle \mathbf{C}_1, \cdots, \mathbf{C}_m \rangle$ we define $|\Phi| = m$ and $\Phi.i = \mathbf{C}_i$ for $1 \leqslant i \leqslant m$. Note that instead of using *proof schema pair* [17, 20] to define proof schemata we use proof schema components. The only difference is that proof schema components make the name explicit. All results concerning proof schemata built from proof schema pairs carry over for our above definition.

Example 2. Let us consider the proof schema $\Phi = \langle (\varphi, \pi, \nu(k)) \rangle$. The proof schema uses one defined function symbol $\widehat{S}(\cdot)$ to convert terms of the ω sort to the ι sort The equational theory $\mathcal{E}$ is as follows:

$$\mathcal{E} = \left\{ \widehat{S}(k+1) = f(\widehat{S}(k)) \ ; \ \widehat{S}(0) = 0 \ ; \ k + f(l) = f(k+l) \right\}.$$

We abbreviate the context as $\Delta = \{P(\alpha + 0), \forall x.P(x) \to P(f(x))\}$. The proofs π and $\nu(k)$ are as follows:

$$\pi = \dfrac{\dfrac{P(\alpha+0) \vdash P(\alpha+0)}{\Delta \vdash P(\alpha+0)}\, w\colon l}{\Delta \vdash P(\alpha + \hat{S}(0))}\, \mathcal{E}$$

$$\nu(k) = \dfrac{\dfrac{\dfrac{\dfrac{\overset{(\varphi,\, n,\, \alpha)}{\Delta \vdash P(\alpha + \hat{S}(n))} \qquad P(f(\alpha + \hat{S}(n))) \vdash P(f(\alpha + \hat{S}(n)))}{\Delta, P(\alpha + \hat{S}(n)) \to P(f(\alpha + \hat{S}(n))) \vdash P(f(\alpha + \hat{S}(n)))}\, \to\colon l}{\Delta, \forall x.P(x) \to P(f(x)) \vdash P(f(\alpha + \hat{S}(n)))}\, \forall\colon l}{\Delta, \forall x.P(x) \to P(f(x)) \vdash P(\alpha + f(\hat{S}(n)))}\, \mathcal{E}}{\Delta, \forall x.P(x) \to P(f(x)) \vdash P(\alpha + \hat{S}(n+1))}\, \mathcal{E}}{\Delta \vdash P(\alpha + \hat{S}(n+1))}\, c\colon l$$

Note that π contains no links, while $\nu(k)$ contains a single $\{n\}$-link.

3 Evaluation and Interpretation

Proof schemata are an alternative formulation of induction. In [20], it is shown that proof schemata are equivalent to a particular fragment of the induction arguments formalizable in Peano arithmetic, i.e. the so called *k-simple induction*. More specifically, k-simple induction limits the number of inductive eigenvariables[7] to one. In previous work [17, 20], **LKE** was extended by the following induction rule instead of links:

$$\dfrac{F(k), \Gamma \vdash \Delta, F(s(k))}{F(0), \Gamma \vdash \Delta, F(t)}\, \mathbf{IND}$$

where t is an arbitrary term of the numeric sort. The result is the calculus **LKIE**. To enforce k-simplicity we add the following constraint: let ψ be an **LKIE**-proof such that for any induction inference in ψ, $V(t) \subseteq \{k\}$ for some k. In [17, 20], the authors show that the following two proposition hold, and thus define the relationship between k-simple **LKIE**-proofs and proof schemata. Given that our calculus can be used to construct proof schemata, the relationship can be trivially extended to proofs resulting from our calculus.

[7] Inductive eigenvariables are eigenvariables occurring in the context of an induction inference rule.

Proposition 1. *Let Ψ be a proof schema with end-sequent $\mathcal{S}$. Then there exists a k-simple* **LKIE***-proof of $\mathcal{S}$.*

Proof. See Proposition 3.13 of [20].

Proposition 2. *Let π be a k-simple* **LKIE***-proof of $\mathcal{S}$. Then there exists a proof schema with end-sequent $\mathcal{S}$.*

Proof. See Proposition 3.15 of [20].

Unlike the induction proofs of the **LKIE**-calculus, proof schemata have a recursive structure and thus require an evaluation ("unrolling"), similar to primitive recursive functions. When we instantiate the free parameter, the following evaluation procedure suffices.

Definition 6 (Evaluation of proof schema [17]). *We define the rewrite rules for links*

$$\frac{(\varphi, 0, \bar{t})}{S(0, \bar{t})} \Rightarrow \pi \qquad\qquad \frac{(\varphi, k, \bar{t})}{S(k, \bar{t})} \Rightarrow \nu(k)$$

for all proof schema components $\mathbf{C} = (\varphi, \pi, \nu(k))$. Furthermore, for $\alpha \in \mathcal{N}$, we define $\mathbf{C}\,[k\backslash\alpha]\downarrow$ as a normal form of the link

$$\frac{(\varphi, \alpha, \bar{t})}{S(\alpha, \bar{t})}$$

under the above rewrite system extended by the rewrite rules for defined function and predicate symbols, i.e. the equational theory $\mathcal{E}$. Also, for a proof schema $\Phi = \langle \mathbf{C}_1, \ldots, \mathbf{C}_m \rangle$, we define $\Phi\,[n\backslash\alpha]\downarrow = \mathbf{C}_1\,[k\backslash\alpha]\downarrow$.

Example 3. Let Φ be the proof schema of example 2 and Δ defined equivalently. For $1 \in \mathbb{N}$ we can write down $\Phi\,[n\backslash 1]\downarrow$ as follows:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{P(\alpha+0) \vdash P(\alpha+0)}\,w\colon l}{\Delta \vdash P(\alpha+0)}\,\mathcal{E}}{\Delta \vdash P(\alpha+\hat{S}(0))}\,\mathcal{E} \qquad P(f(\alpha+\hat{S}(0))) \vdash P(f(\alpha+\hat{S}(0)))}{\Delta, P(\alpha+\hat{S}(0)) \to P(f(\alpha+\hat{S}(0))) \vdash P(f(\alpha+\hat{S}(0)))}\,{\to}\colon l}{\Delta, \forall x.P(x) \to P(f(x)) \vdash P(f(\alpha+\hat{S}(0)))}\,\forall\colon l}{\Delta \vdash P(f(\alpha+\hat{S}(0)))}\,c\colon l}{\Delta \vdash P(\alpha+f(\hat{S}(0)))}\,\mathcal{E}}{\Delta \vdash P(\alpha+\hat{S}(0+1))}\,\mathcal{E}}{\Delta \vdash P(\alpha+\hat{S}(1))}\,\mathcal{E}$$

The described evaluation procedure essentially defines a rewrite system for proof schemata with the following property.

Lemma 1. *The rewrite system for links is strongly normalizing, and for a proof schema Φ and $\alpha \in \mathcal{N}$, $\Phi\,[n\backslash\alpha]\downarrow$ is an* **LK***-proof.*

Proof. See Lemma 3.10 of [20].

Proposition 3 (Soundness [17]). *Let $\Phi = \langle \mathbf{C}_1, \ldots, \mathbf{C}_m \rangle$ be a proof schema with end-sequent $S(n, \bar{x})$ and let $\alpha \in \mathcal{N}$. Then $\Phi[n\backslash\alpha] \downarrow$ is an **LK**-proof of $S(n, \bar{x})$.*

Essentially, Proposition 3 states that $\mathbf{C}_1[k\backslash\alpha] \downarrow$ is an **LK**-proof of the end-sequent $S(n, \bar{x})[k\backslash\alpha] \downarrow$ where by $\downarrow$ we refer to normalization of the defined symbols in $S(n, \bar{x})$.

4 The $\mathcal{S}i$LK-calculus

The $\mathcal{S}i$**LK**-calculus ($\mathcal{S}$chematic $i$nduction **LK**-calculus, see Table 1 & 2) allows one to build a proof schema component-wise. We call the set of expressions in between two | a *component group*. Note that, unlike proof schemata we do not need proof symbols nor ordering because it is implied by the construction. Each component group consists of a multiset of *component pairs* which are pairs of **LKS**-sequents. A set of component groups is referred to as a *component collection*. Even though all auxiliary components (or component groups) are shifted to the left, we do not intend any ordering, i.e. writing, for instance, $(\top : A \vdash A)$ to the right of $\boldsymbol{\Pi}$ in $Ax_1 : r$ in Table 1 does not change the rule.

Table 1. The basic inference rules of the $\mathcal{S}i$**LK**-calculus.

$$\frac{\boldsymbol{\Pi}}{(\top : A \vdash A) \mid \boldsymbol{\Pi}}\, Ax_1 : r \qquad \frac{\Gamma \mid \boldsymbol{\Pi}}{(\top : A \vdash A), \Gamma \mid \boldsymbol{\Pi}}\, Ax_2 : r$$

$$\frac{(\top : \mathbf{[\,S\,]}), \Gamma \mid \boldsymbol{\Pi}}{(A \vdash^{f(n)} A : \mathbf{[\,S\,]}), \Gamma \mid \boldsymbol{\Pi}}\, Ax : l \qquad \frac{(\top : \mathbf{S}), (\top : \mathbf{S}), \Gamma \mid \boldsymbol{\Pi}}{(\top : \mathbf{S}), \Gamma \mid \boldsymbol{\Pi}}\, c_c : r$$

$$\frac{(\mathbf{Q} : \mathbf{[\,S\,]}), (\mathbf{Q} : \mathbf{[\,S\,]}), \Gamma \mid \boldsymbol{\Pi}}{(\mathbf{Q} : \mathbf{[\,S\,]}), \Gamma \mid \boldsymbol{\Pi}}\, c_c : l \qquad \frac{(\mathbf{Q} : \mathbf{[S]}), \Gamma \mid \boldsymbol{\Pi}}{(\top : \mathbf{[S]}), \Gamma' \mid \boldsymbol{\Pi}}\, br$$

$$\frac{(\top : \mathbf{Q}), \Gamma \mid \boldsymbol{\Pi}}{(\top : \mathbf{[\,Q\,]}), \Gamma \mid \boldsymbol{\Pi}}\, cl_{bc} \qquad \frac{(\top : \mathbf{[\,Q\,]}) \mid \boldsymbol{\Pi}}{(\mathbf{[\,]} : \mathbf{[\,Q\,]}) \mid \boldsymbol{\Pi}}\, cl_{LKE}$$

$$\frac{((\Pi \vdash^{\alpha} \Delta)[n\backslash\alpha] : \mathbf{[\,} (\Pi \vdash \Delta)[n\backslash 0] \mathbf{\,]}) \mid \boldsymbol{\Pi}}{(\mathbf{[\,} (\Pi \vdash \Delta)[n\backslash\alpha] \mathbf{\,]} : \mathbf{[\,} (\Pi \vdash \Delta)[n\backslash 0] \mathbf{\,]}) \mid \boldsymbol{\Pi}}\, cl_{sc}$$

$$\frac{(\mathbf{Q} : \mathbf{[\,S\,]}), (\mathbf{R} : \mathbf{[\,S\,]}), \Gamma \mid \boldsymbol{\Pi}}{(\mathbf{Q}' : \mathbf{[\,S\,]}), \Gamma \mid \boldsymbol{\Pi}}\, \rho_2^{sc} \qquad \frac{(\mathbf{Q} : \mathbf{[\,S\,]}), \Gamma \mid \boldsymbol{\Pi}}{(\mathbf{Q}' : \mathbf{[\,S\,]}), \Gamma \mid \boldsymbol{\Pi}}\, \rho_1^{sc}$$

$$\frac{(\top : \mathbf{S}), (\top : \mathbf{R}), \Gamma \mid \boldsymbol{\Pi}}{(\top : \mathbf{S}'), \Gamma \mid \boldsymbol{\Pi}}\, \rho_2^{bc} \qquad \frac{(\top : \mathbf{S}), \Gamma \mid \boldsymbol{\Pi}}{(\top : \mathbf{S}'), \Gamma \mid \boldsymbol{\Pi}}\, \rho_1^{bc}$$

To enforce correct construction of proof schema components we introduce a *closure* mechanism similar to *focusing* [1] (see the inferences cl_{bc}, cl_{LKE}, and cl_{sc} in Table 1). Let us consider a component pair $\mathbf{C} = (\mathbf{Q} : \mathbf{S})$ where $\mathbf{Q}$ is a sequent, a sequent in square brackets, or $\top$ and $\mathbf{S}$ is a sequent or a sequent in square brackets. The left side $\mathbf{Q}$ is the stepcase and the right side $\mathbf{S}$ is the basecase. We use pairs of sequents rather than individual sequents on different

branches to enforce the dependence between the stepcase and the basecase, that is, the both have the same end-sequent. The configuration $\mathbf{Q} = \top$ means that we are still allowed to apply rules to the basecase. If $\mathbf{S}$ is closed, i.e. $\mathbf{S}$ is of the form $[\ \Delta \vdash \Pi\ ]$ for an arbitrary sequent $\Delta \vdash \Pi$, we have closed the basecase (using inference rule cl_{bc}) and essentially fixed its end-sequent. Therefore, $\mathbf{Q}$ is always equal to $\top$ as long as the basecase is not closed. If $\mathbf{S}$ is closed we are allowed to apply rules to the stepcase. This fixing of the end-sequent essentially fixes the sequent we are allowed to introduce using the inference rule $\circlearrowright$.

Apart from schematic proofs, simple **LKE**-proofs can be constructed by keeping the stepcase equal to $\top$. If we instead intend to construct a schema of proofs we have to build a stepcase. The end sequent $(\Pi \vdash \Delta)[n\backslash 0]$ characterizes the stepcase sequent modulo the parameter value, i.e. $(\Pi \vdash^{\alpha} \Delta)[n\backslash\alpha]$ where α depends on the applications of $\circlearrowright$ - or $\rightsquigarrow$ -inferences (see Table 2). The point of this labelling α of the sequents is to indicate what value of the free parameter must be produced in order to close the component. Normally $\alpha = n'$, or the successor of n. Note that $f(n)$, $g(n)$, and $h(n)$ are intended to be arbitrary primitive recursive functions and may be introduced as an extension of the equational theory.

When a component group contains a single component pair and the end sequents of the basecase and stepcase are the same modulo the substitution of the free parameter we can close the component group using cl_{sc} (see Table 1). Alternatively, we can close a group by applying cl_{LKE} if the stepcase is equal to $\top$. We refer to such a group as a *closed group* and any group which is not closed is referred to as an *open group*. As a convention, inference rules can only be applied to open groups. Concerning $\rightsquigarrow$ (see Table 2), it may be the case that the closed group whose end sequent we use to introduce a link has free variables other than the free parameter. We assume correspondence between the free variables of the closed group and the main component, meaning that in a call

$$\Big(\ (\Lambda \vdash^{f(n)} \Gamma)\,[n\backslash g(n)]\,[\bar{x}\backslash\bar{t}]\ :[\ \mathbf{S}\ ]\ \Big)$$

of a component with free variables $\bar{x}$ all occurrences of $\bar{x}$ in the proof of

$$(\ [\ (\Lambda \vdash \Gamma)\,[n\backslash h(n)]\ ]:[\ \mathbf{R}\ ]\ )$$

are replaced with $\bar{t}$. Essentially, this rule is inductive lemma introduction. $h(n)$ is used to represent a non-standard instantiation of the free parameter, i.e. other than n'. Though, non-essentially for this work, in future work when we consider more complex inductive definitions and orders, thus such concepts will be necessary.

An $\mathcal{S}i\mathbf{LK}$-derivation is a sequence of $\mathcal{S}i\mathbf{LK}$ inferences rules ending in a component collection with at least one open component group. A $\mathcal{S}i\mathbf{LK}$-proof ends in a component collection where all components are closed. As we shall show, not every derivation can be extended into a proof.

We also consider a special case of $\mathcal{S}i\mathbf{LK}$-derivations (proofs) which we refer to as *pre-proof schema normal form*. A $\mathcal{S}i\mathbf{LK}$-derivation (proof) is in pre-proof schema normal form if for every application of $Ax_1 : r$ the context $\boldsymbol{\Pi}$ is a $\mathcal{S}i\mathbf{LK}$-proof. This enforces a stricter order on the construction of components than is

Table 2. The linking rules of the $\mathcal{S}i\mathbf{LK}$-calculus.

$$\frac{\left(\top : [\ (\Pi \vdash \Delta)\,[n\backslash 0]\ ]\right), \mathbb{\Gamma} \mid \boldsymbol{\Pi}}{\left((\Pi \vdash^{(n+1)} \Delta)\,[\bar{x}\backslash\bar{t}] : [\ (\Pi \vdash \Delta)\,[n\backslash 0]\ ]\right), \mathbb{\Gamma} \mid \boldsymbol{\Pi}}\ \circlearrowright$$

where $\bar{x}$ is the vector of all free variables of $(\Pi \vdash \Delta)$ and $\bar{t}$ is an arbitrary vector of terms which has the same length as $\bar{x}$.

$$\frac{\left(\top : [\ \mathbf{S}\ ]\right), \mathbb{\Gamma} \mid \boldsymbol{\Delta} \mid \left([\ (\Lambda \vdash \Gamma)\,[n\backslash h(n)]\ ] : [\ \mathbf{R}\ ]\right) \mid \boldsymbol{\Pi}}{\left((\Lambda \vdash^{f(n)} \Gamma)\,[n\backslash g(n)]\,[\bar{y}\backslash\bar{t}] : [\ \mathbf{S}\ ]\right), \mathbb{\Gamma} \mid \boldsymbol{\Pi'}}\ \rightsquigarrow$$

where $\bar{y}$ is the vector of all free variables of $(\Lambda \vdash \Gamma)$ and $\bar{t}$ is an arbitrary vector of terms which has the same length as $\bar{y}$. Also, g and h are arbitrary primitive recursive functions.

already enforced by the use of the $\rightsquigarrow$-inference which can be used to construct proof schemata.

Let $\mathcal{I}$ be the customary evaluation function of sequents, i.e. $\mathcal{I}(\Delta \vdash \Gamma) \equiv \bigwedge_{F\in\Delta} F \rightarrow \bigvee_{F\in\Gamma} F$ for an **LKE**-sequent $\Delta \vdash \Gamma$ and assume an $\mathcal{S}i\mathbf{LK}$-proof ending in the component collection

$$\mathbf{C} \equiv (\ [\ \mathbf{Q}_0\ ] : [\ \mathbf{S}_0\ ]\ ) \ \Big|\ \cdots\ \Big|\ (\ [\ \mathbf{Q}_m\ ] : [\ \mathbf{S}_m\ ]\ )$$

such that $(\ [\ \mathbf{Q}_0\ ] : [\ \mathbf{S}_0]\ )$ is the last component group closed in the proof of C (In the following we will refer to this component as the *leading component*). We extend the evaluation function to the schematic case and define the *evaluation function* of a closed component collection similar to [18] by

$$\mathcal{I}_{\mathcal{S}i\mathbf{LK}}(\mathbf{C}) \equiv \mathcal{I}(\mathbf{S}_0),$$

if $\nu_0 \equiv \top$ and $\mathcal{I}_{\mathcal{S}i\mathbf{LK}}(\mathbf{C}) \equiv$

$$\bigwedge_{i=0}^{m} \mathcal{I}(\mathbf{S}_i) \wedge \forall.x\Big(\bigwedge_{i=0}^{m} \big(\mathcal{I}(\mathbf{Q}_i\,[n\backslash x]) \rightarrow \mathcal{I}(\mathbf{Q}_i\,[n\backslash(x+1)])\big)\Big) \rightarrow \forall x.(\mathcal{I}(\mathbf{Q}_0\,[n\backslash x]),$$

otherwise. Implicitly, the closure rules imply an order. In general, all closed component groups are considered lower in the implied ordering than open component groups. Essentially, the ordering comes from the $\rightsquigarrow$-rule which can only be applied if the auxiliary component is closed. For example, a component may be forced to be closed last, and thus, would be consider the top of the implied ordering.

We use the following denotations for construction of our inference rules. Context variables within schematic sequents will be denoted by uppercase greek letters Δ, Π, etc. Context variables within component groups will be denoted by blackboard bold uppercase greek letters $\mathbb{\Delta}, \mathbb{\Pi}$, etc. Context variables within the component collection will be denoted by fat bold uppercase greek letters,

$\boldsymbol{\Delta}, \boldsymbol{\Pi}$, etc. We use bold uppercase latin letters to denote schematic sequents, $\mathbf{R}, \mathbf{S}$, etc. The inference rules ρ_1^{sc}, ρ_2^{sc}, ρ_1^{bc}, ρ_2^{bc} apply an **LKE** inference rule ρ to the auxiliary sequents to get the main sequent. By the subscript we denote the arity of the inference rule. For example, $(\forall : l)_1^{sc}$ applies the universal quantifier rule to the left side of the stepcase. And finally, we use the following abbreviations:

$$\mathbb{\Gamma}' \equiv (\mathbb{\Gamma}, (\ \mathbf{Q} : \mathbf{[S]}\ )), \text{ and}$$
$$\boldsymbol{\Pi'} \equiv \boldsymbol{\Delta},\ (\ \mathbf{[}\ (\Lambda \vdash \Gamma)\{n \leftarrow \alpha\}\ \mathbf{]} : \mathbf{[\ R\ ]}\ )\ \Big|\ \boldsymbol{\Pi}.$$

The following example illustrates the construction of a simple $\mathcal{S}i\mathbf{LK}$-proof.

Example 4. For the construction of the following $\mathcal{S}i\mathbf{LK}$-proof we use the equational theory $\mathcal{E} \equiv \{\widehat{f^0}(x) = x; \widehat{f^{s(n)}}(x) = f\widehat{f^n}(x)\}$ and the abbreviations

$$\Delta \equiv P(0), \forall x.P(x) \to P(f(x)) \text{ and } \mathbf{S} \equiv \Delta \vdash P(\widehat{f^0(0)}).$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\left(\top \vdots P(0) \vdash P(0)\right)\ |}\,Ax_1 : r}{\left(\top \vdots P(0) \vdash P(\widehat{f^0}(0))\right)\ |}\,\mathcal{E}_1^{bc}}{\left(\top \vdots P(0), \forall x.P(x) \to P(f(x)) \vdash P(\widehat{f^0}(0))\right)\ |}\,(w : l)_1^{bc}}{\left(\top \vdots \mathbf{[}\Delta \vdash P(\widehat{f^0}(0))\ \mathbf{]}\right)\ |}\,cl_{bc}}{\left(P(f\widehat{f^n}(0)) \vdash^{s(n)} P(f\widehat{f^n}(0)) \vdots \mathbf{[\ S\ ]}\right)\ |}\,Ax : l}{\left(\top \vdots \mathbf{[\ S\ ]}\right), \left(P(f\widehat{f^n}(0)) \vdash^{s(n)} P(f\widehat{f^n}(0)) \vdots \mathbf{[\ S\ ]}\right)\ |}\,br}{\left(\Delta \vdash^{s(n)} P(\widehat{f^n}(0)) \vdots \mathbf{[\ S\ ]}\right), \left(P(f\widehat{f^n}(0)) \vdash^{s(n)} P(f\widehat{f^n}(0)) \vdots \mathbf{[\ S\ ]}\right)\ |}\,\circlearrowright}{\left(\Delta, P(\widehat{f^n}(0)) \to P(f\widehat{f^n}(0)) \vdash^{s(n)} P(f\widehat{f^n}(0)) \vdots \mathbf{[\ S\ ]}\right)\ |}\,(\to : l)_2^{sc}}{\left(\Delta, \forall x.P(x) \to P(f(x)) \vdash^{s(n)} P(\widehat{f^{s(n)}}(0)) \vdots \mathbf{[\ S\ ]}\right)\ |}\,(\forall : l)_1^{sc}}{\left(\Delta \vdash^{s(n)} P(\widehat{f^{s(n)}}(0)) \vdots \mathbf{[\ S\ ]}\right)\ |}\,(c : l)_1^{sc}}{\left(\mathbf{[}\ \Delta \vdash P(\widehat{f^{s(n)}}(0))\ \mathbf{]} \vdots \mathbf{[\ S\ ]}\right)\ |}\,cl_{sc}$$

By applying the evaluation function $\mathcal{I}_{\mathcal{S}i\mathbf{LK}}$ we get

$$\mathcal{I}_{\mathcal{S}i\mathbf{LK}}\left(\left(\ \mathbf{[}\ \Delta \vdash P(\widehat{f^{s(n)}}(0))\ \mathbf{]} : \mathbf{[\ S\ ]}\ \right)\right) \equiv$$
$$\Big(\big(\Delta' \to P(\widehat{f^0}(0))\big) \wedge \forall x.\Big(\big(\Delta' \to P(\widehat{f^x}(0))\big) \to \big(\Delta' \to P(\widehat{f^{x+1}}(0))\big)\Big)\Big) \to$$
$$\big(\Delta' \to \forall n.P(\widehat{f^n}(0))\big)$$

where $\Delta' \equiv P(0) \wedge \forall x.P(x) \to P(f(x))$. Essentially, what we have proven with this $\mathcal{S}i\mathbf{LK}$-proof is the sequent

$$P(0), \forall x.P(x) \to P(f(x)) \vdash \forall n.P(\widehat{f^n}(0)).$$

By extending the equational theory and by applying the $\circlearrowright$-inference we can easily strengthen the provable sequent of Example 4.

Example 5. Let $\mathcal{E}$ be the equational theory of Example 4, $\boldsymbol{\Pi}$ the proof of Example 4 and

$$\begin{aligned}
\Delta &\equiv P(0), \forall x.P(x) \to P(f(x)),\\
\mathbb{\Gamma} &\equiv \left(\left[\ \Delta \vdash P(\widehat{f^{s(n)}}(0))\ \right] : \left[\ \Delta \vdash P(\widehat{f^0}(\widehat{0}))\ \right] \right),\\
2 &\equiv s(s(0)),\\
\mathbf{S}' &\equiv \Delta \vdash P(\widehat{f^{2^0}}(0)),\\
\mathcal{E}' &\equiv \mathcal{E} \cup \{\widehat{f^{2^0}}(x) = f(x), \widehat{f^{2^{s(n)}}}(x) = \widehat{f^{2^n}}\widehat{f^2}(x)\}.
\end{aligned}$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\boldsymbol{\Pi}}{\left(\top : P(f(0)) \vdash P(f(0))\right) \mid \mathbb{\Gamma} \mid}\,Ax_1 : r}{\left(\top : P(f(0)) \vdash P(\widehat{f^{2^0}}(0))\right) \mid \mathbb{\Gamma} \mid}\,\mathcal{E}_1^{bc}}{\left(\top : P(0) \vdash P(0)\right), \left(\top : P(f(0)) \vdash P(\widehat{f^{2^0}}(0))\right) \mid \mathbb{\Gamma} \mid}\,Ax_2 : r}{\left(\top : P(0), P(0) \to P(f(0)) \vdash P(\widehat{f^{2^0}}(0))\right) \mid \mathbb{\Gamma} \mid}\,(\to : l)_2^{bc}}{\left(\top : P(0), \forall x(P(x) \to P(f(x))) \vdash P(\widehat{f^{2^0}}(0))\right) \mid \mathbb{\Gamma} \mid}\,(\forall : l)_1^{bc}}{\left(\top : \left[\ \Delta \vdash P(\widehat{f^{2^0}}(0))\ \right]\right) \mid \mathbb{\Gamma} \mid}\,cl_{bc}}{\left(\Delta \vdash^{2^{s(n)}} P(\widehat{f^{2^{s(n)}}}(0)) : \left[\ \mathbf{S}'\ \right]\right) \mid \mathbb{\Gamma} \mid}\,\widehat{\ }}{\left(\left[\Delta \vdash P(\widehat{f^{2^{s(n)}}}(0))\right] : \left[\ \mathbf{S}'\ \right]\right) \mid \mathbb{\Gamma} \mid}\,cl_{sc}$$

Notice that we were able to get a much stronger theorem without significantly extending the proof. Though, the instantiation of the second proof will be exponentially larger than an instantiation of the first proof for the same value of n, the second proof is only double the number of inferences. This is precisely the method one can use to formalize either Orevkov's proof [23] or Boolos' proof [10].

4.1 From $\mathcal{S}i$LK-Proof to Proof Schema

It is possible to construct a proof schema from any $\mathcal{S}i$**LK**-Proof, though it is much easier to perform the translation from $\mathcal{S}i$**LK**-Proof in pre-proof schema normal form. We now show that every $\mathcal{S}i$**LK**-Proof has a pre-proof schema normal form.

Lemma 2. *Let Φ be a $\mathcal{S}i$**LK**-Proof of a component collection* $\mathbf{C}$*. Then there exists a $\mathcal{S}i$**LK**-Proof Φ' of* $\mathbf{C}$ *in pre-proof schema normal form.*

Proof. We prove the statement by rearranging the application of the $\mathcal{S}i$**LK** rules. Let

$$\mathbf{C} \equiv (\ [\ \mathbf{Q}_0\ ]\ :[\ \mathbf{S}_0\ ]\ ) \ \Big|\ \cdots\ \Big|\ (\ [\ \mathbf{Q}_m\ ]\ :[\ \mathbf{S}_m\ ]\ )$$

be the ending component collection of Φ. We identify each component group $\mathcal{CG}_i = (\ [\ \mathbf{Q}_i\ ]\ :[\ \mathbf{S}_i\ ]\ )$ with its ancestors in Φ, i.e. all component groups that

are connected via a $\mathcal{S}i\mathbf{LK}$ rule, exempting the closed component groups of the $\rightsquigarrow$ rule, to $\mathcal{CG}_i$. Afterwards, we find the component group $\mathcal{CG}_i$ which is closed first, i.e. reading top to bottom the component group to which cl_{st} or cl_{LKE} is applied first. Since there is no other component closed earlier there are no $\rightsquigarrow$ rule identified with $\mathcal{CG}_i$, thus we can consider all rules identified with $\mathcal{CG}_i$ to be independent. This implies that we can rearrange Φ such that all rules identified with $\mathcal{CG}_i$ are at the top[8]. This part of the proof will not change any more. Now, we look again for the topmost cl_{st} or cl_{LKE} rule apart from the one we already considered. The corresponding component group $\mathcal{CG}_j$ and its identified rules contain only $\rightsquigarrow$ rules that link to components that are already rearranged and, hence, we can shift all rules identified with $\mathcal{CG}_j$ directly after the already rearranged ones. If we repeat this procedure, we end up with a proof in pre-proof schema normal form.

The important property of pre-proof schema normal form is that the construction of components is organized such that between any two closure rules is an **LKS**-proof.

Theorem 1. *Let Φ be an $\mathcal{S}i\mathbf{LK}$-Proof of the component collection*

$$(\mathbf{[}\ \mathbf{Q}_0\ \mathbf{]}\ \mathbf{:}\ \mathbf{[}\ \mathbf{S}_0\ \mathbf{]}\ ) \ \Big|\ \cdots\ \Big|\ (\mathbf{[}\ \mathbf{Q}_m\ \mathbf{]}\ \mathbf{:}\ \mathbf{[}\ \mathbf{S}_m\ \mathbf{]}\ )$$

such that $\nu_0 \neq \top$ is the leading component, then there exists a proof schema $\langle \mathbf{C}_0, \mathbf{C}_1, \cdots, \mathbf{C}_k \rangle$, for $k \leqslant m$, where for every $0 \leqslant i \leqslant k$ there exists $0 \leqslant i \leqslant j \leqslant m$, where the end sequents of $C_i.2$ and π_j match as well as the end sequents of $C_i.3$ and ν_j.

Proof. By Lemma 2 we know that Φ has a pre-proof schema normal form Φ'. Note that, in a pre-proof schema normal form the leading component, i.e. ν_0, is the leftmost component. In Φ', we delete all component groups whose stepcase is equal to $\mathbf{[}\ \mathbf{]}$ and get Ψ which contains k component groups. This is allowed because $\rightsquigarrow$ cannot link to component groups with stepcase $\mathbf{[}\ \mathbf{]}$. We construct the proof schema directly from Ψ where each proof schema component corresponds to a component group of Ψ. A proof schema component $\mathbf{C}_i$ is constructed from a component proof $\mathcal{CG}_j = (\mathbf{[}\ \mathbf{Q}_j\ \mathbf{]}\ \mathbf{:}\ \mathbf{[}\ \mathbf{S}_j\ \mathbf{]}\ )$ as follows: $\mathbf{C}_i.2$ is the proof containing all rules that are identified with $\mathcal{CG}_j$ and that are applied at the top of cl_{bc}. $\mathbf{C}_i.3$ is the proof containing all rules that are identified with $\mathcal{CG}_j$ and that are between cl_{bc} and cl_{sc}. We translate each component group according to the order of the pre-proof schema normal form, i.e. from right to left, to a proof schema component and construct thereby the proof schema of the theorem.

5 Properties of the Calculus

In this section we discuss the decision problem for validity, soundness of the calculus, and completeness with respect to k-simple proof schemata.

[8] In general, the context is not empty. Since the rules, exempting the $\rightsquigarrow$, are independent from the context, we can always adjust the context.

5.1 Decidability

Following the formalization of our calculus, we can state a semi-decidability theorem. This follows from our choice to distinguish between component collections where the leading component is equal but there is a variation in the other components.

Theorem 2. *Let $\boldsymbol{\Pi}$ be a collection of closed components that has a $\mathcal{S}i$**LK**-proof then we find the proof in a finite number of inferences.*

Proof. By Lemma 2 we can construct proofs from right to left. In general, the basecase is an **LKE**-sequent that is itself semi-decidable. The rightmost component cannot contain any $\rightsquigarrow$-inferences in the stepcase such that it behaves as an **LKE**-proof plus an additional theory axiom for the $\circlearrowright$-rule and is, therefore, semi-decidable. In the next component's stepcase, we consider all $\rightsquigarrow$-rules again as theory axioms, such that we end up in a semi-decidable fragment again. By the finite number of components the semi-decidability of $\boldsymbol{\Pi}$ follows.

The more interesting decidability property is of course whether we are able to extend the number of components on the right of a given component such that the new collection of components has a $\mathcal{S}i$**LK**-proof. To see that this is not even semi-decidable we will formalize Robinson arithmetic [24] in our system.

Theorem 3. *Let C be a closed component group. Then deciding if there exists a closed component collection $\boldsymbol{\Pi}$ such that $C \mid \boldsymbol{\Pi}$ is $\mathcal{S}i$**LK**-provable is undecidable.*

Proof. The ω sort obeys the axioms of Robinson arithmetic concerning successor and zero. We can add the addition and multiplication axioms to the equational theory. The most important axiom of Robinson arithmetic $\forall x(x = 0 \vee \exists y(s(y) = x))$ is intrinsically part of the link mechanism. Because Robinson arithmetic is *essentially undecidable* then showing that there is an extension of a given component collection to a $\mathcal{S}i$**LK**-provable collection must be as well.

5.2 Soundness & Completeness

We provide a proof of soundness using our translation procedure of Section 4.1.

Theorem 4 (Soundness of the $\mathcal{S}i$LK-calculus). *If a closed component collection C is $\mathcal{S}i$**LK**-provable then it is valid.*

Proof. Let Φ be an $\mathcal{S}i$**LK**-proof of C. By Section 4.1 we can transform Φ to a pre-proof schema normal form Φ' and then construct a proof schema Ψ from it. By Proposition 3, we show the validity of the leading component and, therefore, of the evaluation itself, i.e. the $\mathcal{S}i$**LK**-calculus is sound.

To show completeness we technically need a conversion from proof schemata to $\mathcal{S}i$**LK**-proofs which can be easily derived given the procedure defined in Section 4.1. One needs to construct a closed component group for each component of the proof schema whose proofs can be read off from the proofs of the component's stepcase and basecase. The links are replaced by applications of the $\rightsquigarrow$-rule or the $\circlearrowright$-rule. Due to space constraints we avoid formally defining the procedure.

Theorem 5 (Completeness). *If a close component collection C represents a valid n-induction statement then it is $\mathcal{S}i$**LK**-provable.*

Proof. By the theorems and definitions of Section 3, we know that if C represents a valid n-induction statement then a proof can be found in the **LKIE**-calculus. Any **LKIE**-proof can be transformed into a proof schema Φ (Section 3). We have not shown that Φ can be transformed into $\mathcal{S}i$**LK**-proofs, but it is quite obvious that the procedure defined in Section 4.1 is reversible. Thus, there is a $\mathcal{S}i$**LK**-proof for C.

6 Conclusion

In this paper we introduce a calculus for the construction of proof schemata the $\mathcal{S}i$**LK**-calculus which elucidates the restriction found in the formalism of [17] and provides a mechanism to weaken them. Initially, proof schemata were formalized by first defining an extension of the calculus **LK**, the calculus **LKS**, which adds so called links and an equational theory rule. Using this extended calculus a formal definition for proof schemata was developed over sequences of proofs. While interesting results followed [20], the results of proof analysis using the method of [17] could not be formalized within the same framework as the original proof [14]. Also, restrictions on the ordering used, the type of induction, and number of parameters are not easy to relax in the existing framework. By flattening the tree structure of proof schema, separating the instantiation of a proof from its definition, and removing the implicit ordering of components which complicates construction of schema with multiple parameters or mutual recursion, we can easily consider extensions of proof schemata and plan to do so in future work.

References

1. Jean-Marc Andreoli. Logic programming with focusing proofs in linear logic. *J. Log. Comput.*, 2(3):297–347, 1992.
2. Vincent Aravantinos, Ricardo Caferra, and Nicolas Peltier. A schemata calculus for propositional logic. In *TABLEAUX'09*, volume 5607 of *Lecture Notes in Comp. Sci.*, pages 32–46. Springer, 2009.
3. Vincent Aravantinos, Ricardo Caferra, and Nicolas Peltier. A decidable class of nested iterated schemata. In *Proceedings of IJCAR'10*, pages 293–308. Springer, 2010.
4. Vincent Aravantinos, Ricardo Caferra, and Nicolas Peltier. Decidability and undecidability results for propositional schemata. *Journal of Artificial Intelligence Research*, 40(1):599–656, January 2011.
5. Vincent Aravantinos, Ricardo Caferra, and Nicolas Peltier. Linear temporal logic and propositional schemata, back and forth. In *Proceedings of TIME '11*, pages 80–87. IEEE, 2011.
6. Vincent Aravantinos, Mnacho Echenim, and Nicolas Peltier. A resolution calculus for first-order schemata. *Fundamenta Informaticae*, 2013.

7. Vincent Aravantinos and Nicolas Peltier. Schemata of smt-problems. In *Proceedings of TABLEAUX'11*, pages 27–42. Springer-Verlag, 2011.
8. 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(2-3):160–175, August 2008.
9. Matthias Baaz and Alexander Leitsch. Cut-elimination and redundancy-elimination by resolution. *Journal of Symbolic Computation*, 29:149–176, 2000.
10. George Boolos. Don't eliminate cut. *Journal of Philosophical Logic*, 13(4):373–378, 1984.
11. James Brotherston. Cyclic proofs for first-order logic with inductive definitions. In *Tableaux'05*, volume 3702 of *Lecture Notes in Comp. Sci.*, pages 78–92. Springer, 2005.
12. James Brotherston and Alex Simpson. Sequent calculi for induction and infinite descent. *Journal of Logic and Computation*, 2010.
13. David Cerna. A tableaux-based decision procedure for multi-parameter propositional schemata. In *Intelligent Computer Mathematics*, volume 8543 of *Lecture Notes in Comp. Sci.*, pages 61–75. Springer, 2014.
14. David M. Cerna. *Advances in schematic cut elimination*. PhD thesis, Technical University of Vienna, 2015. http://media.obvsg.at/p-AC12246421-2001.
15. David M. Cerna and Alexander Leitsch. Schematic cut elimination and the ordered pigeonhole principle. In *Automated Reasoning - 8th International Joint Conference, IJCAR 2016, Coimbra, Portugal, June 27 - July 2, 2016, Proceedings*, pages 241–256, 2016.
16. Cvetan Dunchev. *Automation of cut-elimination in proof schemata*. PhD thesis, Technical University of Vienna, 2012.
17. Cvetan Dunchev, Alexander Leitsch, Mikheil Rukhaia, and Daniel Weller. Cut-elimination and proof schemata. *Journal of Language, Logic, and Computation*, 2014.
18. Gerhard Gentzen. Fusion of several complete inductions. In M.E. Szabo, editor, *The Collected Papers of Gerhard Gentzen*, volume 55 of *Studies in Logic and the Foundations of Mathematics*, pages 309 – 311. Elsevier, 1969.
19. Stefan Hetzl, Alexander Leitsch, Daniel Weller, and Bruno Woltzenlogel Paleo. Herbrand sequent extraction. In *Intelligent Computer Mathematics*, pages 462–477. Springer, 2008.
20. Alexander Leitsch, Nicolas Peltier, and Daniel Weller. Ceres for First-Order Schemata. *Journal of Logic and Computation*, 2017. To appear.
21. Raymond Mcdowell and Dale Miller. Cut-elimination for a logic with definitions and induction. *Theoretical Computer Science*, 232, 1997.
22. Dale A. Miller. A compact representation of proofs. *Studia Logica*, 46(4):347–370, 1987.
23. V.P. Orevkov. Proof schemata in Hilbert-type axiomatic theories. *Journal of Soviet Mathematics*, 55(2):1610–1620, 1991.
24. R. M. Robinson. An essentially undecidable axiom system. In *Proceedings of the International Congress of Mathematics*, pages 729–730, 1950.
25. Gaisi Takeuti. *Proof Theory*, volume 81 of *Studies in logic and the foundations of mathematics*. American Elsevier Pub., 1975.