Primitive Recursive Proof Systems for Arithmetic

David M. Cerna[1][2] and Anela Lolic[3]*

[1] Research Institute for Symbolic Computation, Johannes Kepler University Linz, Austria
david.cerna@risc.jku.at
[2] Institute for Formal Models and Verification, Johannes Kepler University Linz, Austria
[3] Technical University of Vienna, Institute of Discrete Mathematics and Geometry, Vienna, Austria
anela@logic.at

Abstract

Peano arithmetic, as formalized by Gentzen, can be presented as an axiom extension of the **LK**-calculus with equality and an additional inference rule formalizing induction. While this formalism was enough (with the addition of some meta-theoretic argumentation) to show the consistency of arithmetic, alternative formulations of induction such as the infinitary ω-rule and cyclic reasoning provide insight into the structure of arithmetic proofs obfuscated by the inference rule formulation of induction. For example, questions concerning the elimination of cut, consistency, and proof shape are given more clarity. The same could be said for functional interpretations of arithmetic such as *system T* which enumerates the recursive functions provably total by arithmetic. A key feature of these variations on the formalization of arithmetic is that they get somewhat closer to formalizing the concept of induction directly using the inferences of the **LK**-calculus, albeit by adding extra machinery at the meta-level. In this work we present a recursive sequent calculus for arithmetic which can be syntactically translated into Gentzen formalism of arithmetic and allows proof *normalization* to the **LK**-calculus with equality.

1 Introduction

Proof schemata serve as an alternative formulation of induction based on primitive recursive proof specification. Essentially, the local soundness of an individual *proof component* is replaced by the global soundness of a collection of components. The relationship between local and global soundness is illustrated in [9] where a calculus integrating this global soundness is able to construct proof objects which are locally sound.

The seminal work concerning "proof as schema" was the analysis of Fürstenberg's proof of the infinitude of primes by Baaz et al. [1] using a rudimentary schematic formalism and **CERES** [2]. The schematic representation of induction aided the proof analysis by allowing one to consider a recursive representation of the cut structure as a set of clauses. From refutations of this recursive clause set and *projections* to the original schema one can extend Herbrand's theorem to fragment (k-induction) arithmetic [12, 17].

Note that k-induction is a fragment of arithmetic specifically defined for discussing the expressive power of the formalism introduced in [17] and is not a well studied fragment of arithmetic. It restricts fresh eigenvariable introduction within the main sequent of the induction inference rule. This corresponds to the "one free parameter" per proof schema restriction introduced in this work. The $\mathcal{S}i$**LK**-calculus introduced by Cerna & Lettmann [9], internalizes both global soundness conditions of [17] and the eigenvariable restrictions. Interestingly, enforcing the restrictions on eigenvariable introduction within the $\mathcal{S}i$**LK**-*components* (pairs of sequents) is quite artificial. Collections of $\mathcal{S}i$**LK**-components are to be interpreted as the main

*This research is supported by the FWF project I-2671-N35.

sequent of an induction inference after applying Gentzen's transformation for fusing multiple inductions [13]. It is not necessary to enforce the fresh eigenvariable condition on such induction inferences and thus implies the generalization we present in this work. Concerning proof construction within the $\mathcal{S}i$**LK**-calculus, proofs take a similar form as Curry's formalization of primitive recursive arithmetic [11], quantification of the eigenvariable resulting from induction inferences is not allowed. The constructions introduced in this work generalize this restriction as well by allowing so call *computational proof schema*.

Comparing the existing schematic formalism and our generalization, there are several key distinction when compared to other formalisms of induction and arithmetic. The calculus introduced in [18] is closely related to the sequent calculus of Peano arithmetic, and was developed as a way to eliminate cut in the presence of induction with out using the meta-theoretic method of Gentzen. This work can be contrasted with earlier work concerning proof schema in that it focuses on the elimination of cut without concern for maintaining analyticity of the proof resulting from transformation. Proof systems based on infinite descent [4, 5], otherwise known as cyclic proofs, maintain analyticity during the process of cut-elimination, though only for the infinitary representation of the proof. Such systems rely on regularity of the infinite tree in order to maintain finite representability which is loss during cut-elimination [23]. An interesting result of such formulations is that one can enumerate the finite traces of the regular infinite tree highlight semantic relationships between two inductive definitions [20]. When a given cyclic proof is cut free this is closely related to the resulting *Herbrand systems* of [17].

Formalisms such as system System T and the resulting functional interpretations of arithmetic [14], are closely related to our formalism but do not take the step of integrating the recursive constructions directly into the proof formalism. Though a restricted ω-rule has been discussed [21] where the sequence of proofs must be primitive recursively representable. Such ω-rule have been shown equivalent to Peano arithmetic, thus motivating the development of a proof system which integrates such a construction, that is rather than restricting an infinitary rule which simulates induction we explicitly make the primitive recursive definitions a part of the object language.

Part of the difficultly concerning the implementation of such a formalism is the precise formalization of primitive recursion one ought to use. We use an alternative construction of primitive recursive functions based on the separation of variables into *active* and *dormant* [22]. This has interesting implication concerning cut-elimination in the presence of infinitary derivation [6] given that this formalism of primitive recursion generalizes the simple recursive definitions of [17]. This entails that rather than reductive cut-elimination as is usually used in the presence of an ω-rule, one can use a global cut-elimination method, such as *CERES* [1, 2]. Note that extension of such methods to the introduced formal system is beyond the scope of this work, but it is planned for future investigation.

We provide a formalism which is provability-wise as expressive as **PA** without restricting the structure of the proof or the inductive argument. We do so by constructing proof schemata over a well-founded ordering and allow multiple *free parameters* (eigenvariables introduced by inductive inferences). Multi-parameter schemata provide quantifier introduction over numeric terms without the complete loss of the recursive structure and thus allow us to formalize strong totality statements[1]. Notice that a result of the syntactic translation we provide here is a non-trivial conservative reflection between **P**-schema and Gentzen's formalization of Peano arithmetic. For uses of reflection principles with respect to arithmetic see Parikh's results [19, 3] for the monadic version of **PA**, that is **PA***. We foresee application of similar methodology to other problems and see our approach as a method of formulating logical relationships between

[1] As noted in [15], a single inductively introduced quantifier suffices for formalizing **PA**.

a classical formulation of a theory and its schematic counterpart.

2 The Schematic Language and P-schema

We generalize previous work concerning *proof schemata* [9, 10, 12, 17]. Our formalism uses a multi-sorted first order language consisting of the ω sort, referred to as the *indexing sort*, and the ι sort, referred to as the *individual sort* consisting of the standard first-order term language together with a countable set of variable symbols. Given that this paper deals primarily with arithmetic the indexing sort ω will be assumed to be the sort of numerals unless otherwise stated, that is terms of ω are constructed from the signature $\{0, s(\cdot)\}$ together with a countable set of *parameter symbols.* Parameters are special constants which can be thought of as *eigenvariables* for recursive proof definitions. We will discuss this in more detail when defining P-schema.

In addition to the above term constructions one can also define so called *defined function symbols*, which can either be functions assigned a type constructed using the following grammar, $\tau := \omega|\iota|\tau \rightarrow \tau$, or primitive recursive definitions. For example, in [8] a proof schema was constructed whose term signature included a function f of type $\iota \rightarrow \omega$. Formulas are built from a countable set of predicate symbols $\mathcal{P}$ and the logical connectives $\neg, \wedge, \vee, \rightarrow, \forall, \exists$. The sort of proper first-order formula is denoted by o.

One can construct *defined predicate symbols* using the type grammar $\mu := \tau \rightarrow o$ where τ is the nonterminal introduced for typing defined function symbols. In general, defined symbols will be denoted by $\hat{\cdot}$, i.e. $\hat{P}$. We will refer to constructions of type o together with the defined predicate symbols as *formula schema.*

We assume a set of convergent rewrite rules $\mathcal{E}$ (equational theory) defining the syntactic behavior of defined symbols. The rules of $\mathcal{E}$ are of the form $\widehat{f}(\bar{t}) = E$, where $\bar{t}$ contains no defined symbols, and either $\widehat{f}$ is a function symbol and E is a term or $\widehat{f}$ is a predicate symbol and E is a formula schema. The $\mathcal{E}$-rule is reversible.

We generalize the notion of sequent to so called *schematic sequents* which are a pair of multisets of formula schemata Δ, Π denoted by $\Delta \vdash \Pi$. We will denote multisets of formula schemata by upper-case Greek letters. Let $S(\overline{x})$ be a sequent and $\overline{x}$ a vector of free variables, then $S(\bar{t})$ denotes $S(\overline{x})$ where $\overline{x}$ is replaced by $\bar{t}$ and $\bar{t}$ is a vector of terms of appropriate type. The rules of the standard **LK**-calculus as defined in [24] and will be extended to handle the defined constructions of schematic sequents. We refer to this extended calculus as the **LKE**-calculus which is the **LK**-calculus with the addition of the following rule:

Definition 1 (**LKE**)**.** *Let $\mathcal{E}$ be an equational theory. **LKE** is an extension of **LK** by the $\mathcal{E}$ inference rule* $\dfrac{S(t)}{S(t')}\,\mathcal{E}$ *where the term or formula schema t in the sequent S is replaced by a term or formula schema t' for $\mathcal{E} \models t = t'$.*

An **LKE**-derivation is a rooted tree s.t. every node is decorated by a sequent and an edge exists being two nodes of the tree iff there is a sound inference rule application with the sequent closer to the root playing the role of the main sequent and the other sequents playing the role of auxiliary sequents. If every branch of the tree ends at an initial sequent then we refer to the **LKE**-derivation as an **LKE**-proof. P-schema require an generalization of *initial sequents*, i.e. sequents of the form $A \vdash A$, which we cover in the following section.

2.1 The P-schema Formalism

To define **P**-schema we need to distinguish between three types of parameter symbols in the ω sort, namely, *active* parameters $\mathcal{N}_a$, *passive* parameters $\mathcal{N}_p$, and *internal* parameters $\mathcal{N}_i$[2]. These distinctions highlight three ways parameter symbols can be used within a proof schema:

- To guide proper construction of the recursive proof definition (active).
- To allow strong quantification, substitution, and proof normalization (passive).
- To allow value passing between different proofs within the **P**-schema (internal).

We will denote active parameters by lower-case Latin characters n, m, k, passive parameters by lower-case bold Greek characters $\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}$, and internal parameters by lower-case bold Latin characters $\mathbf{n}, \mathbf{m}, \mathbf{k}$. Furthermore we distinguish between certain types of numeric terms by considering four independent sets of terms dependent on the type of parameter symbols they contain $\mathcal{A}_{\mathbb{N}}$ (active), $\mathcal{P}_{\mathbb{N}}$ (passive), $\mathcal{I}_{\mathbb{N}}$ (internal), and $\mathcal{G}_{\mathbb{N}}$ (parameter free). Together these sets make the set of *schematic terms* denoted by $\mathcal{S}_{\mathbb{N}} = \mathcal{A}_{\mathbb{N}} \cup \mathcal{P}_{\mathbb{N}} \cup \mathcal{G}_{\mathbb{N}} \cup \mathcal{I}_{\mathbb{N}}$. Given that we are concerned with terms built from successor and zero they contain at most one parameter symbol.

A particular type of schematic sequent is needed for the construction of **P**-schema which accounts for intended roles of the various types of parameters. We refer to a sequent S as $(n, \mathcal{I})$-sequent if the only active parameter occurring in S is n and all internal parameters occurring in S are members of the set $\mathcal{I} \subset \mathcal{I}_{\mathbb{N}}$. If a sequent S is active parameter free and all internal parameters occurring in S are members of the set $\mathcal{I} \subset \mathcal{I}_{\mathbb{N}}$ it will be referred to as an $\mathcal{I}$-sequent.

The presence or absence of an active parameter within a given sequent S denotes whether the sequent is intended to be used as part of the step case or base case of an inductive definition over $\mathcal{G}_{\mathbb{N}}$. If the sequent in question denotes the end sequent of an inductively defined **LK**-proof then the step case and base case must be syntactic variations of each other modulo the occurrences of the active parameter. We refer to a pair of sequents (S, S'), where S is an $(n, \mathcal{I})$-sequent and S' is an $\mathcal{I}$-sequent, as an *inductive pair* if there exists $\alpha \in \mathcal{G}_{\mathbb{N}}$ s.t. $S\{n \leftarrow \alpha\} = S'$.

Based on the parameter type distinction we can refine the concept of an **LKE**-derivation. We will refer to an **LKE**-derivation which only contains $(n, \mathcal{I})$-sequents as a *multivariate* **LKE**-derivation (*mv***LKE**-derivation). Furthermore an *mv***LKE**-derivation whose branches end at initial sequents and whose root sequent does not contain active and internal parameters is an *mv***LKE**-proof. The end-sequent of an *mv***LKE**-proof (*mv***LKE**-derivation) φ will be denoted by $es(\varphi)$ and the set $\mathcal{V}_x(S)$ for $x \in \{a, p, i\}$ will denote the active, passive and internal parameters occurring in the sequent S, respectively. Notice that the calculus introduced so far cannot construct *mv***LKE**-proofs unless the entire derivation is active and internal parameter free; **P**-schema describe how a set of derivations join to produce *mv***LKE**-proofs.

Non-tautological leaves of *mv***LKE**-derivations can be seen as *links* to a yet to be described proof. If one is given a set of proofs Φ containing a proof ψ which has a non-tautological leaf $S\sigma$, if there exists a $\chi \in \Phi$ s.t. $es(\chi) = S$ then we can *link* the proof χ to ψ to get a proof. The non-tautological leaves can be substitution invariants of the end sequent of χ. Note that σ is defined over the internal and active parameters of χ. If after a finite number of linking steps the result is a *mv***LKE**-derivation π, where each branch ends at an initial sequent, π can be easily transformed into an *mv***LKE**-proof by substitution.

In order to properly describe the above procedure we need to provide *mv***LKE**-derivations with names, therefore we assume a countably infinite set $\mathcal{B}$ of *proof symbols*, i.e. $\varphi, \psi, \varphi_i, \psi_j$.

[2] A related terminology can be found in [22] which discusses a construction similar to ours.

Definition 2 (mv**LKS**)**.** *The mv**LKS**-calculus is an extension of mv**LKE**, where links may appear at the leaves of a derivation.*

Given that mv**LKS**-derivations contain links, an external object containing the derivations is necessary in order to guarantee a sound construction. This soundness condition is provided by the **P**-schema construction. To simplify some of the following definitions, we will refer to an mv**LKS**-derivation as *inactive* if it does not contain an active parameter and $\{n\}$-active if it contains only the active parameter n.

Example 1. *Consider the following $\mathcal{E}$ theory*

$$\mathcal{E} = \{\widehat{a}(s(n),\beta) = s(\widehat{a}(n,\beta));\ \widehat{a}(0,\beta) = \beta\},$$

where

$$\pi = \qquad \dfrac{\dfrac{\vdash 0 = 0}{\vdash a(0,0) = 0}\,\mathcal{E}}{\vdash a(0,0) = a(0,0)}\,\mathcal{E} \qquad\qquad \nu = \qquad \dfrac{\begin{array}{c}\chi(n)\\ \cdots\cdots\cdots\cdots\cdots\cdots\\ \vdash a(n,0) = a(0,n)\end{array} \qquad \begin{array}{c} S_1(\nu_1)\\ \hline \vdots \end{array}}{\vdash a(n+1,0) = a(0,n+1)}\,cut$$

$S_1(\nu_1) \equiv a(n,0) = a(0,n) \vdash s(a(n,0)) = s(a(0,n))$, *$\pi$ is an mv**LKS**-proof and ν is an mv**LKS**-derivation. Also, ν contains a link to the proof symbol χ and is $\{n\}$-active. Note that the end-sequent of ν, $es(\nu)$ is $es(\chi(n))\{n \leftarrow n+1\}$. Moreover, $es(\nu)$ is an $(n,\emptyset)$-sequent, i.e. internal parameter free. Its inductive pair also contains the sequent $es(\nu)\{n \leftarrow 0\} = \vdash a(0,0) = a(0,0)$, the end-sequent of π (an $\emptyset$-sequent). The triple (χ, π_1, ν_1) is referred to as an $(n,\emptyset)$-component.*

Definition 3 ($(n,\mathcal{I})$-component)**.** *Let $\psi \in \mathcal{B}$, $n \in \mathcal{N}_a$ and $\mathcal{I} \subset \mathcal{N}_i$. An* $(n,\mathcal{I})$-component **C** *is a triple (ψ, π, ν) where π is an inactive mv**LKS**-derivation ending with $S\{n \leftarrow \alpha\}$ and ν is an $\{n\}$-active mv**LKS**-derivation ending in an $(n,\mathcal{I})$-sequent S whose inductive pair is $S\{n \leftarrow \alpha\}$. Given a component $\mathbf{C} = (\psi, \pi, \nu)$ we define $\mathbf{C}.1 = \psi$, $\mathbf{C}.2 = \pi$, and $\mathbf{C}.3 = \nu$. We refer to $es(\mathbf{C}) = S$ as the* end sequent *of the component* **C**.

When possible, we will refer to an $(n,\mathcal{I})$-component as a component. A schematic proof is defined over finitely many components, which can be linked together. So far we have not restricted linking, not all links are sound. Whenever a component **C** links to another component **D** the passive parameters occurring in $es(\mathbf{D})$ must occur in the sequent associated with the link in **C**, what we refer to as *association.* In order to define associations between schematic sequents we introduce *schematic substitutions*, a function $\sigma : \mathcal{A}_{\mathbb{N}} \cup \mathcal{I}_{\mathbb{N}} \to \mathcal{S}_{\mathbb{N}}$ which replaces all occurrences of a parameter $x \in \mathcal{A}_{\mathbb{N}} \cup \mathcal{I}_{\mathbb{N}}$ with a term $t \in \mathcal{S}_{\mathbb{N}}$.

Definition 4 (association)**.** *Let $S(\bar{t})$ be an $(n,\mathcal{I})$-sequent and $S'(\bar{x})$ an $(m,\mathcal{I}')$-sequent where $\bar{t}$ is a sequence of terms from the ι sort and $\bar{x}$ is a sequence of free ι sort variables of the same length. We say* $S(\bar{t})$ associates with $S'(\bar{x})$ *if there exists a schematic substitution σ s.t. $S(\bar{t})\sigma = S'(\bar{t})$.*

Association defines the relationship between schematic sequents with roughly the same structure, however, for **P**-schema we need a slightly stronger relation which we refer to as *Linkability.* Essentially, linkability tells us when the end sequent of a given component can be attached to leaves of another component, i.e. the two sequents associate and certain restrictions on the passive parameters hold.

Definition 5 (Linkability)**.** *Two components* **C** *and* **D** *are said to be* $(\mathbf{C},\mathbf{D})$-linkable *if for each non-axiomatic leaf S in* **C** *of which $es(\mathbf{D})$ associates, $\mathcal{V}_p(es(\mathbf{D})) \subseteq \mathcal{V}_p(S)$. We say they are* strictly $(\mathbf{C},\mathbf{D})$-linkable *if it holds that $\mathcal{V}_p(es(\mathbf{D})) \subseteq \mathcal{V}_p(es(\mathbf{C}))$.*

Assuming we have a set of components Φ, linkability defines a partial order on the set.

Definition 6 (Linkability ordering)**.** *Let $\mathbf{C}_1$ and $\mathbf{C}_2$ be distinct components s.t. they are (strictly) $(\mathbf{C}_1, \mathbf{C}_2)$-linkable. Then we say that $\mathbf{C}_1 \prec \mathbf{C}_2$ ($\mathbf{C}_1 \prec_s \mathbf{C}_2$).*

Our restriction on the number of active parameters per sequent blocks mutual recursion and linkability relation can be used to define a well-ordering thus enforcing a primitive recursive construction. However, the linkability ordering is not well-founded, it is defined over the set of all components and allows the definition of mutually linkable components, i.e. $(\mathbf{C}_1, \mathbf{C}_2)$-linkable and $(\mathbf{C}_2, \mathbf{C}_1)$-linkable. Thus, we further restrict our usage of the ordering and only consider well founded sub-orderings of the linkability ordering when defining **P**-schemata.

Definition 7 (**P**-schema)**.** *Let $\mathbf{P} \subset \mathcal{N}_p$, $\mathbf{C}_1$ an $(n, \mathcal{I})$-component and $\mathbf{C}_2, \cdots, \mathbf{C}_\alpha$ components s.t. for all $1 \leq i \leq \alpha$, $\mathbf{C}_i.1$ are distinct and $\mathcal{V}_p(\mathbf{C}_i) \subseteq \mathbf{P}$. We define $\Psi = \langle \mathbf{C}_1, \cdots, \mathbf{C}_\alpha \rangle$ as a* **P**-schema *(*strict **P**-schema*) over a well founded suborder $\prec^* \subset \prec$ ($\prec_s^* \subset \prec_s$) of $\{\mathbf{C}_1, \ldots, \mathbf{C}_\alpha\}$ with $\mathbf{C}_1$ as least element. We define $|\Psi| = \alpha$, $\Psi.i = \mathbf{C}_i$ for $1 \leq i \leq \alpha$, and $es(\Psi) = es(\mathbf{C}_1)$.*

Notice that the passive parameters are, in some sense, declared before the construction of the **P**-schema, i.e. **P** is a set of passive parameters. These symbols play a special role during the normalization procedure, i.e. when we unfold the primitive recursive definitions defining a **P**-schema and construct an **LKS**-Proof. One can think of passive parameters as constants acting as place holders for numerals, i.e. ground terms $\mathcal{G}_\mathbb{N}$. However, not all passive parameters in a **P**-schema need to occur in the end sequent of a given schema. Passive parameters can also play the role of numeric eigenvariables and be strongly quantified. This property enforces a special treatment of **P**-schema which are constructed from other **P**-schema.

Definition 8 (sub **P**-schema)**.** *Let $\Psi = \langle \mathbf{C}_1, \cdots, \mathbf{C}_\alpha \rangle$ be a **P**-schema and $\Psi' = \left\langle \mathbf{C}'_1, \cdots, \mathbf{C}'_\beta \right\rangle$ be a **P**-schema s.t. $\left\{ \mathbf{C}'_1, \cdots, \mathbf{C}'_\beta \right\} \subseteq \{\mathbf{C}_1, \cdots, \mathbf{C}_\alpha\}$. We refer to Ψ' as a* sub **P**-schema *of Ψ. Furthermore, consider a **P**-schema $\Phi = \left\langle \mathbf{C}, \mathbf{C}'_1, \cdots, \mathbf{C}'_\beta \right\rangle$, a component $C' \in \Psi'$ s.t. for all $D \in \Psi'$, if $D \neq C'$, then $C' \not\prec D$ and an occurrence of a link to C' in Ψ of the form $\begin{array}{c} C'.1(t, \cdots) \\ \cdots\cdots\cdots\cdots\cdots \\ es(C'.1)\sigma \end{array}$ Where σ is a substitution with $dom(\sigma) = \mathcal{V}_i(C') \cup \mathcal{V}_a(C')$ s.t. $\mathcal{V}_a(C')\sigma = \{t\}$. We refer to a sub **P**-schema as* ground *if $\mathcal{V}_a(t) = \mathcal{V}_i(t) = \mathcal{V}_p(t) = \emptyset$,* essential *if $\mathcal{V}_a(t) \neq \emptyset$ and $\mathcal{V}_p(t) = \emptyset$, or as* computational *if $\mathcal{V}_a(t) = \emptyset$ and $\mathcal{V}_p(t) \neq \emptyset$, but $\mathcal{V}_p(t) \cap \mathcal{V}_p(es(\Psi)) = \emptyset$.*

Example 2. *We can formalize associativity of addition as an $\{\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}\}$-schema $\Phi = \langle(\varphi, \pi, \nu)\rangle$ over the following $\mathcal{E}$ theory $\mathcal{E} = \{\widehat{a}(s(n), \beta) = s(\widehat{a}(n, \beta));\ \widehat{a}(0, \beta) = \beta\}$ where*

$$\pi = \dfrac{\dfrac{\vdash a(\mathbf{k}, \boldsymbol{\gamma}) = a(\mathbf{k}, \boldsymbol{\gamma})}{\vdash a(0, a(\mathbf{k}, \boldsymbol{\gamma})) = a(\mathbf{k}, \boldsymbol{\gamma})}\mathcal{E}}{\vdash a(0, a(\mathbf{k}, \boldsymbol{\gamma})) = a(a(0, \mathbf{k}), \boldsymbol{\gamma})}\mathcal{E} \qquad \nu = \dfrac{\begin{array}{c} \varphi(n, \mathbf{k}, \boldsymbol{\gamma}) \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ \vdash a(n, a(\mathbf{k}, \boldsymbol{\gamma})) = a(a(n, \mathbf{k}), \boldsymbol{\gamma}) \end{array} \qquad \begin{array}{c} (\nu_1) \\ \\ S(\nu_1) \end{array}}{\vdash a(n', a(\mathbf{k}, \boldsymbol{\gamma})) = a(a(n', \mathbf{k}), \boldsymbol{\gamma})}cut$$

*and $S(\nu_1) \equiv a(n, a(\mathbf{k}, \boldsymbol{\gamma})) = a(a(n, \mathbf{k}), \boldsymbol{\gamma}) \vdash a(n', a(\mathbf{k}, \boldsymbol{\gamma})) = a(a(n', \mathbf{k}), \boldsymbol{\gamma})$. Notice that ν is an mv**LKS**-derivation not an mv**LKS**-proof being that the end sequent of ν is $\{n\}$-active. We can extend Φ to $\Phi^* = \langle(\chi, \lambda, \mu), (\varphi, \pi, \nu)\rangle$ where*

$$\nu_1 = \dfrac{\dfrac{\dfrac{\begin{array}{c} a(n, a(\mathbf{k}, \boldsymbol{\gamma})) = a(a(n, \mathbf{k}), \boldsymbol{\gamma}) \vdash \\ s(a(n, a(\mathbf{k}, \boldsymbol{\gamma}))) = s(a(a(n, \mathbf{k}), \boldsymbol{\gamma})) \end{array}}{\begin{array}{c} a(n, a(\mathbf{k}, \boldsymbol{\gamma})) = a(a(n, \mathbf{k}), \boldsymbol{\gamma}) \vdash \\ a(n', a(\mathbf{k}, \boldsymbol{\gamma})) = s(a(a(n, \mathbf{k}), \boldsymbol{\gamma})) \end{array}}\,\mathcal{E}}{\begin{array}{c} a(n, a(\mathbf{k}, \boldsymbol{\gamma})) = a(a(n, \mathbf{k}), \boldsymbol{\gamma}) \vdash \\ a(n', a(\mathbf{k}, \boldsymbol{\gamma})) = a(s(a(n, \mathbf{k})), \boldsymbol{\gamma}) \end{array}}\,\mathcal{E}}{\begin{array}{c} a(n, a(\mathbf{k}, \boldsymbol{\gamma})) = a(a(n, \mathbf{k}), \boldsymbol{\gamma}) \vdash \\ a(n', a(\mathbf{k}, \boldsymbol{\gamma})) = a(a(n', \mathbf{k}), \boldsymbol{\gamma}) \end{array}}\,\mathcal{E}$$

$$\lambda = \begin{array}{c} \varphi(0, \boldsymbol{\beta}, \boldsymbol{\gamma}) \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ \vdash a(0, a(\boldsymbol{\beta}, \boldsymbol{\gamma})) = a(a(0, \boldsymbol{\beta}), \boldsymbol{\gamma}) \end{array}$$

$$\mu = \begin{array}{c} \varphi(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}) \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ \vdash a(\boldsymbol{\alpha}, a(\boldsymbol{\beta}, \boldsymbol{\gamma})) = a(a(\boldsymbol{\alpha}, \boldsymbol{\beta}), \boldsymbol{\gamma}) \end{array}$$

*The schema Φ^* ends with an mv**LKS**-proof and represents a sequence of mv**LKS**-proofs.*

Note that this formalization is a generalization of the formalization described in [17]. If we were to restrict ourselves to $(n, \mathcal{I})$-components and construct strict $\{\boldsymbol{\alpha}\}$-schema, the resulting formalism would be equivalent to proof schemata *á la* [17]. For example $\{\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}\}$-schema Φ provided in Example 2 has an $\{n\}$-active end sequent with a free internal parameter, these are nothing more than the free parameter and a free variable of the ι sort as discussed in [17]. We can extend this example to a proof of commutativity which is beyond the expressive power of previous formalizations.

Example 3. *We use the same $\mathcal{E}$ theory as presented in Example 2 and extend the $\{\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}\}$-schema of Example 2 to the $\{\boldsymbol{\alpha}, \boldsymbol{\beta}\}$-schema $\Phi' = \langle (\chi, \pi_1, \nu_1), (\psi, \pi_2, \nu_2), (\xi, \pi_3, \nu_3)(\varphi, \pi, \nu) \rangle$ using the following equational axioms:*

$$\begin{aligned} E_1 \equiv & a(\boldsymbol{\alpha}, 1) = a(1, \boldsymbol{\alpha}) \vdash a(a(\boldsymbol{\alpha}, 1), n) = a(a(1, \boldsymbol{\alpha}), n) \\ E_2 \equiv & a(a(1, \boldsymbol{\alpha}), n) = a(n', \boldsymbol{\alpha}), a(a(\boldsymbol{\alpha}, 1), n) = a(a(1, \boldsymbol{\alpha}), n) \vdash a(a(\boldsymbol{\alpha}, 1), n) = a(n', \boldsymbol{\alpha}) \\ E_3 \equiv & a(\boldsymbol{\alpha}, n) = a(n, \boldsymbol{\alpha}) \vdash s(a(\boldsymbol{\alpha}, n)) = s(a(n, \boldsymbol{\alpha})) \\ E_4 \equiv & a(\boldsymbol{\alpha}, a(1, n)) = a(a(\boldsymbol{\alpha}, 1), n), a(a(\boldsymbol{\alpha}, 1), n) = a(n', \boldsymbol{\alpha}) \vdash a(\boldsymbol{\alpha}, a(1, n)) = a(n', \boldsymbol{\alpha}) \end{aligned}$$

π_1 and ν_1 are as in Example 1.

$$\pi_2 = \dfrac{\dfrac{\dfrac{\vdash s(0) = s(0)}{\vdash a(0, s(0)) = s(0)}}{\vdash a(0, s(0)) = s(a(0, 0))}\,\mathcal{E}}{\vdash a(0, s(0)) = a(s(0), 0)}\,\mathcal{E}$$

$$\nu_2 = \begin{array}{c} \dfrac{\begin{array}{c} \psi(n) \\ \cdots\cdots\cdots\cdots \\ \vdash a(n, 1) = a(1, n) \end{array} \qquad S_1(\nu_2)}{\vdots}\,cut \\ \hline \vdash a(n', 1) = a(1, n') \end{array}$$

where $S_1(\nu_2) \equiv a(n, 1) = a(1, n) \vdash s(a(n, 1)) = s(a(1, n))$, and

$$\pi_3 = \begin{array}{c} \chi(\boldsymbol{\alpha}) \\ \cdots\cdots\cdots\cdots \\ \vdash a(\boldsymbol{\alpha}, 0) = a(0, \boldsymbol{\alpha}) \end{array}$$

$$\nu_{3_1} = \begin{array}{c} \varphi(\boldsymbol{\alpha}, 1, n) \\ \cdots\cdots\cdots\cdots\cdots\cdots \\ \vdash a(\boldsymbol{\alpha}, a(1, n)) = a(a(\boldsymbol{\alpha}, 1), n) \end{array}$$

$$\nu_3 = \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\begin{array}{c} \xi(n, \boldsymbol{\alpha}) \\ \cdots\cdots\cdots\cdots \\ \vdash a(\boldsymbol{\alpha}, n) = a(n, \boldsymbol{\alpha}) \end{array}}{\vdots} \qquad \dfrac{\begin{array}{c} \psi(\boldsymbol{\alpha}) \\ \cdots\cdots\cdots\cdots \\ \vdash a(\boldsymbol{\alpha}, 1) = a(1, \boldsymbol{\alpha}) \end{array}}{\vdots}}{}\,cut}{\vdots} \qquad \begin{array}{c} (\nu_{3_1}) \\ S(\nu_{3_1}) \end{array}}{\vdash a(\boldsymbol{\alpha}, a(1, n)) = a(n', \boldsymbol{\alpha})}\,cut}{\vdash a(\boldsymbol{\alpha}, s(a(0, n))) = a(n', \boldsymbol{\alpha})}\,\mathcal{E}}{\vdash a(\boldsymbol{\alpha}, n') = a(n', \boldsymbol{\alpha})}\,\mathcal{E}$$

where $S(\nu_{3_1}) \equiv\vdash a(\boldsymbol{\alpha}, a(1, n)) = a(a(\boldsymbol{\alpha}, 1), n)$. Notice that ξ is the least element of the order $\prec$ and the following relations concerning $\prec$ are also defined: $\xi \prec \varphi$, $\xi \prec \psi$, $\psi \prec \chi$. Note

*Evaluation of Φ' into an **LKS**-proof is not yet possible given that $es(\Phi')$ contains an active parameter. We can perform a similar extension as before to construct a $\mathbf{P}$-schema which can be evaluated. We can now quantify the passive parameters of the schema and derive the precise statement of commutativity as one would derive in **PA**.*

Constructing an *mv***LKS**-proof from an *mv***LKS**-derivation is defined as follows:

Theorem 1. *Let $\Phi = \langle \mathbf{C}_1, \cdots, \mathbf{C}_\alpha \rangle$ be a $\mathbf{P}$-schema resulting in an mv**LKS**-derivation s.t. $0 < |\mathcal{V}_i(es(\Phi))| + |\mathcal{V}_a(es(\Phi))|$ and $\mathcal{V}_p(es(\Phi))| + |\mathcal{V}_i(es(\Phi))| + |\mathcal{V}_a(es(\Phi))| \leq |P|$. Then there exists $\Phi' = \langle \mathbf{C}^*, \mathbf{C}_1, \cdots, \mathbf{C}_\alpha \rangle$ s.t. $0 = |\mathcal{V}_i(es(\Phi))| + |\mathcal{V}_a(es(\Phi))|$.*

Proof. We add a new component to Φ resulting in the proof schema $\Phi' = \langle \mathbf{C}^*, \mathbf{C}_1, \cdots, \mathbf{C}_\alpha \rangle$ s.t. $\mathbf{C}_1 \prec \mathbf{C}^*$. Let $es(\Phi) = S(n, \boldsymbol{\alpha}_1, \cdots \boldsymbol{\alpha}_k, \mathbf{i}_1, \cdots, \mathbf{i}_m)$, $\mathbf{C}^* = (\chi, \lambda, \mu)$ and $\mathbf{C}_1 = (\varphi, \lambda', \mu')$. Furthermore, let $S = \{\gamma, \boldsymbol{\beta}_1, \cdots, \boldsymbol{\beta}_m\} \subset \mathbf{P}$ s.t. $\{\boldsymbol{\alpha}_1, \cdots \boldsymbol{\alpha}_k\} \cup S \equiv \emptyset$. Then we define λ and μ as follows

$$\lambda = \begin{array}{c} \varphi(0, \boldsymbol{\alpha}_1, \cdots \boldsymbol{\alpha}_k, \boldsymbol{\beta}_1, \cdots, \boldsymbol{\beta}_m) \\ \hdashline S(0, \boldsymbol{\alpha}_1, \cdots \boldsymbol{\alpha}_k, \boldsymbol{\beta}_1, \cdots, \boldsymbol{\beta}_m) \end{array} \qquad \mu = \begin{array}{c} \varphi(\gamma, \boldsymbol{\alpha}_1, \cdots \boldsymbol{\alpha}_k, \boldsymbol{\beta}_1, \cdots, \boldsymbol{\beta}_m) \\ \hdashline S(\gamma, \boldsymbol{\alpha}_1, \cdots \boldsymbol{\alpha}_k, \boldsymbol{\beta}_1, \cdots, \boldsymbol{\beta}_m) \end{array}$$

□

We will refer to such a $\mathbf{P}$-schema as the *completion* of Φ, or a *complete* $\mathbf{P}$-schema. In the next section we show that evaluation and soundness [17] can be extended to our formalism.

2.2 Evaluating P-Schemata

We now extend the soundness result of proof schema [17] and define evaluation of $\mathbf{P}$-schema.

Definition 9 (Evaluation of $\mathbf{P}$-schemata)**.** *Let $\Phi = \langle \mathbf{C}_1, \cdots, \mathbf{C}_\alpha \rangle$ be a complete strict $\mathbf{P}$-schema s.t. $C_i = (\psi_i, \pi_i, \nu_i)$. We define the rewrite rules for links as $\hat{\psi}_i(0, \mathcal{I}) \rightarrow \pi_i$ and $\hat{\psi}_i(s(n), \mathcal{I}) \rightarrow \nu_i$ where $\mathbf{C}_i$ is assumed to contain the active parameter n. The rewrite system for links is the union of these rules. Moreover, for a substitution $\sigma : \mathcal{N}_p \rightarrow \mathcal{G}_{\mathbb{N}}$ with domain P, we define $\Phi\sigma = \hat{\psi}_1\sigma$ as the normal form of Φ under these rewrite rules and the equational theory $\mathcal{E}$.*

Lemma 1. *Let Φ be a complete strict $\mathbf{P}$-schema and $\sigma : \mathcal{N}_p \rightarrow \mathcal{G}_{\mathbb{N}}$ with domain P. The rewrite system for the links of Φ is strongly normalizing and confluent, s.t. $\Phi\sigma$ is an **LK**-proof.*

Proof. By the restriction on occurrences of links, a proof schema can be seen as a set of primitive recursive definitions, and the rewrite rules for links are the standard rules for these definitions. It is well-known that such rewrite systems are strongly normalizing, see [16]. Finally, by the restriction to complete strict $\mathbf{P}$-schema links will not occur in the normal form and $\Phi\sigma$ is an *mv***LKE**-proof. Furthermore, since all $\mathcal{E}$-inferences in this proof are trivial and there are no parameters, we may consider it as an **LK**-proof.

□

The conditions we place on $\mathbf{P}$-schema are sufficient for the development of a strongly normalizing and confluent rewrite system, but are not necessary. As has been pointed out we can relax these conditions by enforcing the components to obey a well-founded order. relaxing our conditions in this way would of course yield a more flexible formalism allowing the occurrence of arbitrary active parameters within a given schematic sequent, but at the same time the complexity of the formalism would have to increase. What is interesting about this work is that none of this complexity is necessary in order to achieve provability equivalence to expressive arithmetic theories.

Though there is still some room for us to relax the constraints. In particular we can allow proof schema to have computational sub-P-schema. starting from the sub-P-schema of a P-schema Φ which do not have any computational sub-P-schema we can apply Lemma 1 and construct an **LK**-proof. By structural induction on the **P**-schema construction we can consider the next largest computational sub-P-schema and glue the unfolded P-schema together and construct a larger proof. Though, we are most interested in complete strict P-schema and their relation to primitive recursive arithmetic. By theory extension, we mean any link to a sub **P**-schema can be treated as an axiom because it can be unrolled into an **LK**-proof and is therefore soundly constructed. Before discussing the soundness of **P**-schemata with respect to **LK**-proofs we need to introduce an additional concept concerning computational sub-**P**-schema.

Definition 10. *Let Ψ be a* **P***-schema. We refer to a* **P***-schema χ as a* direct *sub-***P***-schema of Ψ if there does not exists a sub-***P***-schema ξ other than Ψ itself s.t. χ is also a sub-***P***-schema of ξ and $\chi \not\equiv \Psi$.*

Definition 11. *Let Ψ be a* **P***-schema. We will refer to Ψ as 0-computational if Ψ is a complete* **P***-schema and does not contain any computational sub* **P***-schema. We refer to Ψ as k-computational if there exists a direct sub-***P***-schema of Ψ which is $(k-1)$-computational.*

Theorem 2 (Soundness of **P**-schemata)**.** *Let Ψ be a* **P***-schema and let $\sigma : \mathcal{N}_p \to \mathcal{G}_{\mathbb{N}}$ be a substitution whose domain is* **P***. Then $\Psi\sigma$ is an* ***LK****-proof over a theory T of $es(\Psi)\sigma$, where T is the set of end-sequents of the direct computational sub* **P***-schema Ψ.*

Proof. Let us assume that Ψ is a 0-computational **P**-schema implying that Ψ does not contain any computational sub-**P**-schema and that $es(\Psi)$ does not contain internal, nor active parameters. Thus by Lemma 1, $es(\Psi)\sigma$ can be transformed into an **LK**-proof.

For the induction hypothesis, we assume that for any k-computational **P**-schema Ψ, $\Psi\sigma$ can be transformed into a **LK**-proof over a theory T consisting of the end sequents of the direct sub **P**-schema of Ψ. We show that for a $(k+1)$-computational **P**-schema Ψ', $\Psi'\sigma'$ can be transformed into a **LK**-proof over a theory T' consisting of the end sequents of the direct sub **P**-schema of Ψ'. Let S be the set of all direct sub-**P**-schema of Ψ'.

Each $\chi \in S$ is an m-computational **P**-schema for $m \leq k$ and thus, we know that χ can be transformed into a complete **P**-schema and there exists a substitution σ_χ s.t. $\chi\sigma_\chi$ is an **LK**-proof over some theory T consisting of the end sequents of the direct sub-**P**-schema of χ. Let the end sequent of $\chi\sigma_\chi$ by $\Delta \vdash \Pi$. We can easily extend $\chi\sigma_\chi$ to a proof of $S_\chi = \vdash \bigwedge_{\phi\in\Delta} \phi \to \bigvee_{\phi\in\Pi} \phi$. Finally, this implies that we can construct a proof $\vdash \bigwedge_{\chi\in S'} S_\chi$. Thus, the conjunction of the end sequents of each sub-**P**-schema in $\chi \in S'$ is a sound construction and an **LK**-proof.

We can consider the Ψ^* which is the **P**-schema derived from Ψ by removing all components associated with sub-**P**-schema of S'. Occurrences of the end sequents of sub-**P**-schema of S' within Ψ^* can be treated as theory and thus $\Psi^*\sigma'$ is an **LK**-proof modulo the theory consisting of the end sequents of the computational direct sub **P**-schema of Ψ'. □

3 Local Induction and *mv*LKE

In [17], it was shown that proof schemata are equivalent to a particular fragment of arithmetic, i.e. the so called *k-simple induction*, which limits the introduction of fresh eigenvariables by induction. The induction rule on the lefthand side:

$$\dfrac{F(k), \Gamma \vdash \Delta, F(s(k))}{F(0), \Gamma \vdash \Delta, F(t)}\,\mathbf{IND} \qquad\qquad \dfrac{F(n), \Gamma \vdash \Delta, F(n+1)}{F(0)\sigma, \Gamma\sigma \vdash \Delta\sigma, F(t)\sigma}\,mv\mathbf{IND}$$

where t is a term of the numeric sort s.t. t either contains k (k is a free parameter in the sense of [17]) or is ground. Adding the above rule to **LKE** resulted in the **LKIE**-calculus. We will refer to this calculus as the *simple* **LKIE**-calculus. The *mv***LKE**-calculus and **P**-schema are related to a much more expressive induction rule. Essentially any term t can replace the active parameter n of the auxiliary sequent (including a term containing n) and the internal parameters can be instantiated with arbitrary terms. The instantiations must obey the restriction of at most one active parameter per schematic sequent. We refer to the calculus with the righthand side induction rule above as the *mv***LKIE**-calculus, where σ is a substitution whose domain is equivalent to the internal parameters occurring in the auxiliary sequent of the *mv***IND** induction inference. Note that internal parameters behave like free variables and thus can be instantiated by any term. In some sense they are internal with respect to a given component and thus, as we shall see later, are superfluous. We leave them as part of the formalism given that it is sensible for recursive definition to pass information. We do not restrict the terms σ can map internal parameters to other than the mapping must obey our restriction on active parameters, that is, only one active parameter per schematic sequent.

We consider an *mv***LKIE**-derivation ψ as an *mv***LKIE**-proof if the end-sequent of ψ does not contain internal or active parameters. We will first consider strict *mv***LKIE**-proofs which, like strict links, require preservation of the passive parameters, i.e. all passive parameters used in the proof must show up in the end sequent of the proof.

Example 4. *Here we present a strict mv**LKIE**-proof of the $\{\boldsymbol{\alpha}, \boldsymbol{\beta}\}$-schema of Example 3. Notice how the links are replaced by the induction rules in a similar fashion to the k-induction conversion introduced in [17]. We will further discuss this conversion with respect to* **P***-schema later in this paper.*

$$\dfrac{\begin{array}{c} a(\boldsymbol{\alpha}, n) = a(n, \boldsymbol{\alpha}) \vdash \\ a(\boldsymbol{\alpha}, n) = a(n, \boldsymbol{\alpha}) \end{array} \qquad \begin{array}{c} \begin{array}{c} a(n, 1) = a(1, n) \vdash \\ a(n', 1) = a(1, n') \\ \cdots\cdots\cdots\cdots\cdots\cdots \; mv\mathbf{IND} \\ a(0, 1) = a(1, 0) \vdash \\ a(\boldsymbol{\alpha}, 1) = a(1, \boldsymbol{\alpha}) \\ \hline \end{array} \\ \vdots \end{array}}{(\nu_1)}\,cut$$

$$\dfrac{\dfrac{\dfrac{\dfrac{(\nu_1) \qquad \begin{array}{c} \vdots \\ \dfrac{\begin{array}{c} a(n, a(\mathbf{m}, \mathbf{k})) = a(a(n, \mathbf{m}), \mathbf{k}) \vdash \\ a(n', a(\mathbf{m}, \mathbf{k})) = a(a(n', \mathbf{m}), \mathbf{k}) \end{array}}{\begin{array}{c} a(0, a(1, n)) = a(a(0, 1), n) \vdash \\ a(\boldsymbol{\alpha}, a(1, n)) = a(a(\boldsymbol{\alpha}, 1), n) \end{array}}\,mv\mathbf{IND} \\ \hline \\ \vdots \\ \vdash a(\boldsymbol{\alpha}, a(1, n)) = a(a(\boldsymbol{\alpha}, 1), n) \end{array}}{\begin{array}{c} a(\boldsymbol{\alpha}, n) = a(n, \boldsymbol{\alpha}) \vdash \\ a(\boldsymbol{\alpha}, a(1, n)) = a(n', \boldsymbol{\alpha}) \end{array}}\,cut}{\begin{array}{c} a(\boldsymbol{\alpha}, n) = a(n, \boldsymbol{\alpha}) \vdash \\ a(\boldsymbol{\alpha}, s(a(0, n))) = a(n', \boldsymbol{\alpha}) \end{array}}\,\mathcal{E}}{\begin{array}{c} a(\boldsymbol{\alpha}, n) = a(n, \boldsymbol{\alpha}) \vdash \\ a(\boldsymbol{\alpha}, n') = a(n', \boldsymbol{\alpha}) \end{array}}\,\mathcal{E}}{\begin{array}{c} a(\boldsymbol{\alpha}, 0) = a(0, \boldsymbol{\alpha}) \vdash \\ a(\boldsymbol{\alpha}, \boldsymbol{\beta}) = a(\boldsymbol{\beta}, \boldsymbol{\alpha}) \end{array}}\,mv\mathbf{IND}$$

Syntactically transforming **LKIE**-proofs into proof schemata, as done in [17], does not directly address the k-simplicity of the induction inferences. Rather, the transformation required a structural similarity between the usage of the indexing sort of the proof schema and the induction inferences in the **LKIE**-proof. The k-simple induction inference provides a perfect correspondence with proof schema as defined in [17]. Thus, if we are to follow a similar argumentation, we need to show that the *mv***IND** induction inference captures **P**-schema. This argument is easier to make when we restrict ourselves to strict proofs, i.e. transforming strict *mv***LKIE**-proofs into complete strict **P**-schema. However, using a similar argumentation as we used in Theorem 2, one can generalize this translation to a method for translating *mv***LKIE**-proofs into complete **P**-schema.

Lemma 2. *For strict*[3] $\mathbf{P}$*-schema* Ψ *there exists a strict mv****LKIE****-derivation of* $es(\Psi)$*.*

Proof. Let us consider a strict $\{\boldsymbol{\alpha}\}$-schema Ψ. As we discussed earlier such $\mathbf{P}$-schema are equivalent to the proof schemata defined in [17]. We repeat the argument provided in the earlier work translating strict $\{\boldsymbol{\alpha}\}$-schema into strict *mv*$\mathbf{LKIE}$-derivation of $es(\Psi)$.

Let us consider all $C \in \Psi$ s.t. for any $D \in \Psi$ if $C \neq D$, then $C \not\prec D$. Furthermore, let us consider the sub-$\{\boldsymbol{\alpha}\}$-schema of Ψ, Φ. We show how one can construct a *mv*$\mathbf{LKIE}$-derivation for such a component C. Being that C can only link to itself and we are restricted to a single passive parameter, it is only necessary to consider proof links in $C.3$. Now we produce a proof ν from $C.3$ by replacing all links of the form

$$\frac{C.1(n, t_1, \cdots, t_n)}{es(C.1)(n, t_1, \cdots, t_n)}$$

by the schematic sequent $\mathcal{F}(es(C.1))\sigma \vdash \mathcal{F}(es(C.1))\sigma$ where σ is a substitution implementing the substitution of the link, i.e. $\sigma : \mathcal{V}_i(es(C.1)) \rightarrow \mathcal{S}_{\mathbb{N}}$. There may be more then one occurrence of a link to $C.1$ within $C.3$ and thus multiple substitutions may occur. After applying this transformation the end sequent of ν becomes $\mathcal{F}(es(C.1)) \vdash \mathcal{F}(es(C.1))\{n \leftarrow n+1\}$. The resulting end sequent can be used as an auxiliary sequent for an *mv*$\mathbf{IND}$ inference. So far we only considered self referential links, thus we do not need to consider substitutions for the internal parameters, thus the *mv*$\mathbf{IND}$ inference behaves like a $\mathbf{IND}$ inference. In the step case there will be a slight complication arising when a link occurs pointing to another proof symbol. In such cases internal parameters can be instantiated as part of the *mv*$\mathbf{IND}$ inferences. Thus, when using this proof as a subproof of a larger proof one must use the appropriate substitution. To complete the construction we apply a cut inference to the main sequent of the *mv*$\mathbf{IND}$ inference and the end sequent of $C.2$, i.e. the basecase.

For the step case let $C \in \Psi$ s.t. for all $D_1, \cdots, D_m \in \Psi$ of which $D_m \prec \cdots \prec D_1 \prec C$ it must be the case that $m \leq n$. We assume as our induction hypothesis that the translation is sound for such components and show that it also holds when $m \leq n+1$. Note that this assumption implies that all direct sub-$\mathbf{P}$-schema of Ψ to which C links are translatable. We will refer to these direct sub-$\mathbf{P}$-schema of Ψ, ignoring self referential links (these are handle as in the basecase), as $\Phi_1, \cdots, \Phi_k$, and each of them has a sequence of substitutions for each occurance of the links in C, i.e. $\sigma_i^1, \cdots, \sigma_i^{r_i}$. We may now replace each of these links by an initial sequent $\mathcal{F}(es(\Phi_i))\sigma_i^j \vdash \mathcal{F}(es(\Phi_i))\sigma_i^j$. The result is the following end sequent of $C.3$:

$$\mathcal{F}(es(\Phi_1))\sigma_1^1, \cdots, \mathcal{F}(es(\Phi_1))\sigma_1^{r_1}, \cdots, \mathcal{F}(es(\Phi_1))\sigma_k^{r_k}, \mathcal{F}(es(C.3))\sigma_C^1, \cdots, \mathcal{F}(es(C.3))\sigma_C^{r_c}$$

$$\vdash \mathcal{F}(es(C.3))\{n \leftarrow n+1\}$$

We may now cut $\mathcal{F}(es(\Phi_i))\sigma_i^j$ using the translation of Φ_i and treat $\mathcal{F}(es(C.3))\sigma_C^{r_i}$ in a similar manor as in the basecase.

Now assuming that the theorem holds for $|\mathbf{P}| < n+1$ we can show that the theorem holds when $|\mathbf{P}| \leq n+1$. This result is quite a simple extension of the case when $|P| = 1$. Let us consider a component C s.t. all of its direct sub-$\mathbf{P}$-schema have at most n passive parameters but C contain $n+1$. because Ψ is strict, there are no passive parameters below C which are not contained in C. Thus, the same construction can be applied again. □

[3] $\mathcal{V}_p(es(\Psi))) \equiv P$

Note that the translation defined in Lemma 2, along with Lemma 3, provides equivalence between strict **P**-schema and primitive recursive arithmetic (**PRA**) being that strict **P**-schema do not allow the expression of totality of any of the constructions.

Lemma 3. *Let* Π *be a strict mv**LKIE**-derivation of* $\mathcal{S}$ *containing* α *inductions of the form*

$$\dfrac{F_\beta(n,\mathbf{m}_1,\cdots,\mathbf{m}_\gamma),\Gamma_\beta \vdash \Delta_\beta, F_\beta(n',\mathbf{m}_1,\cdots,\mathbf{m}_\gamma)}{F_\beta(0,\mathbf{a}_1,\cdots,\mathbf{a}_\gamma),\Gamma_\beta \vdash \Delta_\beta, F_\beta(t,\mathbf{a}_1,\cdots,\mathbf{a}_\gamma)}\; mv\mathbf{IND}$$

where $1 \leq \beta \leq \alpha$, *and if* $\eta < \beta$ *then the induction inference with conclusion* $F_\beta(0,\mathbf{a}_1,\cdots,\mathbf{a}_\gamma),\Gamma_\beta \vdash \Delta_\beta, F_\beta(t,\mathbf{a}_1,\cdots,\mathbf{a}_\gamma)$ *is above the induction inference with conclusion* $F_\eta(0,\mathbf{a}'_1,\cdots,\mathbf{a}'_{\gamma*}),\Gamma_\eta \vdash \Delta_\eta, F_\eta(t,\mathbf{a}'_1,\cdots,\mathbf{a}'_{\gamma*})$ *in* Π*. Then there exists a strict* **P***-schema with end-sequent* $\mathcal{S}$*.*

Proof. Let T be the transformation taking an *mv***LKIE**-derivation φ to an *mv***LKS**-derivation by replacing the induction inferences $F_\eta(0,\mathbf{a}_1,\ \cdots,\ \mathbf{a}_\gamma),\ \Gamma_\eta \vdash \Delta_\eta,\ F_\eta(t,\mathbf{a}_1,\ \cdots,\ \mathbf{a}'_\gamma)$, $\eta < \beta$, with a proof link $\psi_\eta(t,\mathbf{a}'_1,\cdots,\mathbf{a}'_{\gamma'})$. If the transformation reaches the induction inference β it replaces the $F_\beta(0,\mathbf{a}_1,\ \cdots,\ \mathbf{a}_\gamma),\ \Gamma_\beta \vdash \Delta_\beta,\ F_\beta(t,\ \mathbf{a}_1,\ \cdots,\ \mathbf{a}_\gamma)$ with a proof link $\psi_\beta(t,\mathbf{m}_1,\cdots,\mathbf{m}_\gamma)$ and sequent of the proof link $F_\beta(0,\mathbf{m}_1,\ \cdots,\ \mathbf{m}_\gamma),\ \Gamma_\beta \vdash \Delta_\beta, F_\beta(n,\mathbf{m}_1,\ \cdots,\ \mathbf{m}_\gamma)$. The instantiation is placed in the construction of the component for the predecessor of β. Such a transformation obviously constructs a strict **P**-schema from a strict *mv***LKIE**-derivation.

We will inductively construct a strict **P**-schema $\langle \mathbf{C}_1,\cdots,\ \mathbf{C}_\alpha\rangle$ where $\mathbf{C}_\beta = (\psi_\beta,\pi,\nu)$ has the end sequent $F_\beta(0,\mathbf{m}_1,\cdots\ ,\mathbf{m}_\gamma),\Gamma_\beta \vdash \Delta_\beta, F_\beta(n,\mathbf{m}_1,\cdots,\mathbf{m}_\gamma)$ for some active parameter n. Assume that we have already constructed such proofs for $\mathbf{C}_{\beta+1},\cdots,\mathbf{C}_\alpha$ and consider the induction inference with the following main sequent $F_\beta(0,\mathbf{a}_1,\cdots,\mathbf{a}_\gamma),\Gamma_\beta \vdash \Delta_\beta, F_\beta(t,\mathbf{a}_1,\cdots,\mathbf{a}_\gamma)$. Let ξ be the derivation above the induction. We set π to $F_\beta(0,\mathbf{m}_1,\ \cdots,\ \mathbf{m}_\gamma),\ \Gamma_\beta \vdash \Delta_\beta,\ F_\beta(0,\ \mathbf{m}_1,\ \cdots,\ \mathbf{m}_\gamma)$ which by definition fulfills the requirements of links. Furthermore, let ν be the proof

$$\dfrac{\dfrac{\begin{matrix}\psi_\beta(n,\bar{\mathbf{m}}_\gamma)\\ S(\psi_\beta(n,\bar{\mathbf{m}}_\gamma))\end{matrix} \qquad\qquad \begin{matrix}T(\xi)\\ S(T(\xi))\end{matrix}}{F_\beta(0,\bar{\mathbf{m}}_\gamma),\Gamma_\beta,\Gamma_\beta \vdash \Delta_\beta,\Delta_\beta,F_\beta(n',\bar{\mathbf{m}}_\gamma)}\; mv\mathbf{IND}}{F_\beta(0,\bar{\mathbf{m}}_\gamma),\Gamma_\beta \vdash \Delta_\beta,F_\beta(n',\bar{\mathbf{m}}_\gamma)}\; c*$$

where $S(\psi_\beta(n,\bar{\mathbf{m}}_\gamma)) \equiv F_\beta(0,\bar{\mathbf{m}}_\gamma),\Gamma_\beta \vdash \Delta_\beta,F_\beta(n,\bar{\mathbf{m}}_\gamma)$, $S(T(\xi\ \ )) \equiv F_\beta(n,\bar{\mathbf{m}}_\gamma),\Gamma_\beta \vdash \Delta_\beta,F_\beta(n',\bar{\mathbf{m}}_\gamma)$, which also clearly satisfies the requirement on links. Summarizing, $\mathbf{C}_\beta$ is a component with end-sequent $F_\beta(0,\bar{\mathbf{m}}_\gamma),\Gamma_\beta \vdash \Delta_\beta,F_\beta(n,\bar{\mathbf{m}}_\gamma)$. Linkability and the partial ordering come for free from the construction of strict *mv***LKIE**-derivations. □

Notice that Lemma 3 does not put a restriction on the number of passive parameters in the end sequent, but limits the partial ordering of components to a total linear ordering. A simple corollary of Lemma 3 removes the restriction on the ordering of components, one can join sub-**P**-schema whose components are totally ordered into one whose components are not totally ordered. Proving the corollary requires the same induction argument over a the more complex order structure (the linkability ordering). We would have to join chains of components together using cuts.

Corollary 1. *Let* Π *be a strict mv**LKIE**-derivation of* $\mathcal{S}$*. Then there exists a strict* **P***-schema with end-sequent* $\mathcal{S}$*.*

Concerning strict *mv***LKIE**-derivations, notice that the need for passive, internal, and active parameters is no longer there. The three parameters aided the formalization of **P**-schema by removing mutual recursion and parameter instantiation, which are difficult to handle. Essentially, a reasonable class of strict **P**-schema could not be constructed without the three types of parameters. But for strict *mv***LKIE**-derivations, the construction is obvious and enforced by the proof structure, thus, we can replace internal and active parameters by the corresponding constants and passive parameters. The resulting rule is essentially the induction rule of arithmetic. However, given the inclusion of the $\mathcal{E}$ rule, which allows arbitrarily complex equational theories, the language is at least a conservative extension of **PRA**if $\mathcal{E}$ is limited to functions provably total in **PRA**. We show that for a particular choice of equational theory and using the standard equational axioms, the $\mathcal{E}$ rule is admissible and thus strict *mv***LKIE**-derivations are precisely as expressive as **PRA** and by transitivity so is the strict **P**-schema formulation.

Furthermore, by dropping the strictness requirement, that is allowing computational sub **P**-schema, It follows that the **P**-schema formulation is provability equivalent to **PA**.

4 P-schema, PRA, and PA

We will consider the **P**-schema formulation over the following equational theory

$$\mathcal{E}_{\text{PA}} = \left\{\ \widehat{a}(s(n),\beta) = s(\widehat{a}(n,\beta));\ \widehat{a}(0,\beta) = \beta\widehat{m}(s(n),\beta) = \widehat{a}(\widehat{m}(n,\beta),\beta);\ \widehat{m}(0,\beta) = 0\ \right\}.$$

Furthermore we we extend the **LK**-calculus by the following initial sequents of arithmetic and equational reasoning over the ω sort:

$$\begin{array}{lll} \vdash t = t & s_1 = t_1, \cdots s_n = t_n \vdash f(s_1, \cdots s_n) = f(t_1, \cdots t_n) & s(0) = 0 \vdash \\ s(x) = s(y) \vdash x = y & s_1 = t_1, \cdots s_n = t_n, P(s_1, \cdots s_n) \vdash P(t_1, \cdots t_n) & \end{array}$$

More information concerning equational reasoning and the axioms of arithmetic may be found in [24]. We will refer to these axioms as Ax_{PA}. The **LK**-calculus extended by these initial sequents, the definition of addition and multiplication as well as the induction inference (as discused in [24]) will be referred to as the **PA**-calculus (or **PRA**-calculus when we are referring to primitive recursive arithmetic alone).

Lemma 4. *Let* Π *be a strict mv**LKIE**-proof using* Ax_{PA} *and* $\mathcal{E}_{\text{PA}}$*. Then there exists a strict mv**LKIE**-proof* Π' *without the* $\mathcal{E}$ *inference rule (*Π' *is* $\mathcal{E}$*-free) ending with* $es(\Pi)$*.*

Proof. The rewrite rules of $\mathcal{E}_{\text{PA}}$ are precisely the axioms of Ax_{PA} for addition and multiplication. Thus, from those axioms and the initial sequents introduced above anything provable by the $\mathcal{E}$ inference rule can be proven using the above mentioned axioms and atomic cuts. □

Now that we have $\mathcal{E}$-free strict *mv***LKIE**-proofs we can consider translation to the **PA**-calculus without quantification. As the end sequent of a strict *mv***LKIE**-proof only has passive parameters. We can push the passive parameters up the proof tree and replace each active parameter by a fresh passive parameter. Thus, the resulting proof only contains passive parameters and constants and is a proof in the **PRA**-calculus.

Theorem 3. *There exists a* $\mathcal{E}$*-free strict mv**LKIE**-proof of a sequent* S *iff there exists a **PRA**-calculus proof of* S*.*

Proof. Let us refer to the $\mathcal{E}$-free strict *mv***LKIE**-proof of the sequent S as Φ. Note that on any branch starting at the root, the sequents of the branch are internal parameter free until we reach a *mv***LKIE** induction rule. Thus if Φ does not contain any *mv***LKIE**induction rules it is a **PRA**-calculus proof of S. Let us assume if for any proof Φ s.t. every branch starting from the root contains at most n mv**LKIE** induction inferences than we can transform the proof in a **PRA**-calculus proof of S. We now show that the same is possible when there is a single branch which contain $n+1$ mv**LKIE** induction inferences. Let us consider the lowest mv**LKIE** induction inference occurring on the branch with $n+1$ inferences.

$$\cfrac{\begin{array}{c}(\varphi)\\ F(n), \Gamma \vdash \Delta, F(n+1)\end{array}}{F(0)\sigma, \Gamma\sigma \vdash \Delta\sigma, F(t)\sigma}\ mv\mathbf{IND}$$

We can apply the substitution to the auxiliary sequent and all sequents of φ which are not auxiliary sequents of an mv**LKIE** induction inference. This application of substitution transforms the mv**IND** inference in the standard induction inference and by the induction hypothesis the result is a **PRA**-calculus proof of S. □

Example 5. *The following proof of commutativity is the result of applying the translation from $\mathcal{E}$-free strict mv**LKIE**-proofs to **PA**-calculus proofs. Note that the proof below was first introduced in Example 3.*

$$\cfrac{\begin{array}{c}a(\boldsymbol{\alpha}, \boldsymbol{\gamma}) = a(\boldsymbol{\gamma}, \boldsymbol{\alpha}) \vdash\\ a(\boldsymbol{\alpha}, \boldsymbol{\gamma}) = a(\boldsymbol{\gamma}, \boldsymbol{\alpha})\end{array} \qquad \begin{array}{c} a(\boldsymbol{\mu}, 1) = a(1, \boldsymbol{\mu}) \vdash \\ a(s(\boldsymbol{\mu}), 1) = a(1, s(\boldsymbol{\mu})) \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \ \mathbf{IND} \\ a(0, 1) = a(1, 0) \vdash \\ a(\boldsymbol{\alpha}, 1) = a(1, \boldsymbol{\alpha}) \\ \hline \vdots \end{array}}{(\nu_1)}\ cut$$

$$\cfrac{(\nu_1) \qquad \begin{array}{c}\vdots\\ a(\boldsymbol{\nu}, a(1,\boldsymbol{\gamma})) = a(a(\boldsymbol{\nu},1),\boldsymbol{\gamma}) \vdash \\ a(s(\boldsymbol{\nu}), a(1,\boldsymbol{\gamma})) = a(a(s(\boldsymbol{\nu}),1),\boldsymbol{\gamma}) \ \mathbf{IND} \\ \hline a(0, a(1,\boldsymbol{\gamma})) = a(a(0,1),\boldsymbol{\gamma}) \vdash \\ a(\boldsymbol{\alpha}, a(1,\boldsymbol{\gamma})) = a(a(\boldsymbol{\alpha},1),\boldsymbol{\gamma}) \\ \hline \vdots \\ \vdash a(\boldsymbol{\alpha}, a(1,\boldsymbol{\gamma})) = a(a(\boldsymbol{\alpha},1),\boldsymbol{\gamma}) \end{array}}{\begin{array}{c} a(\boldsymbol{\alpha}, \boldsymbol{\gamma}) = a(\boldsymbol{\gamma}, \boldsymbol{\alpha}) \vdash \\ a(\boldsymbol{\alpha}, a(1,\boldsymbol{\gamma})) = a(s(\boldsymbol{\gamma}),\boldsymbol{\alpha}) \\ (1)\end{array}}\ cut$$

$$\cfrac{\begin{array}{c}(1)\\ a(\boldsymbol{\alpha}, \boldsymbol{\gamma}) = a(\boldsymbol{\gamma}, \boldsymbol{\alpha}) \vdash \\ a(\boldsymbol{\alpha}, a(1,\boldsymbol{\gamma})) = a(s(\boldsymbol{\gamma}),\boldsymbol{\alpha})\end{array} \qquad \begin{array}{c} a(\boldsymbol{\alpha}, a(1,\boldsymbol{\gamma})) = a(s(\boldsymbol{\gamma}),\boldsymbol{\alpha}), \\ \boldsymbol{\alpha} = \boldsymbol{\alpha}, s(\boldsymbol{\gamma}) = s(\boldsymbol{\gamma}), \\ a(1,\boldsymbol{\gamma}) = s(a(0,\boldsymbol{\gamma})) \vdash \\ a(\boldsymbol{\alpha}, s(a(0,\boldsymbol{\gamma}))) = a(s(\boldsymbol{\gamma}),\boldsymbol{\alpha}) \end{array}}{\begin{array}{c}\vdots \\ \hline a(\boldsymbol{\alpha}, \boldsymbol{\gamma}) = a(\boldsymbol{\gamma}, \boldsymbol{\alpha}) \vdash \\ a(\boldsymbol{\alpha}, s(a(0,\boldsymbol{\gamma}))) = a(s(\boldsymbol{\gamma}),\boldsymbol{\alpha}) \\ \hline (\nu_2)\end{array}}\ cut$$

$$\cfrac{(\nu_2) \qquad \begin{array}{c}\vdots\\ a(\boldsymbol{\alpha}, s(a(0,\boldsymbol{\gamma}))) = a(s(\boldsymbol{\gamma}),\boldsymbol{\alpha}), \\ \boldsymbol{\alpha} = \boldsymbol{\alpha}, s(\boldsymbol{\gamma}) = s(\boldsymbol{\gamma}), \\ s(a(0,\boldsymbol{\gamma})) = s(\boldsymbol{\gamma}) \vdash \\ a(\boldsymbol{\alpha}, s(\boldsymbol{\gamma})) = a(s(\boldsymbol{\gamma}),\boldsymbol{\alpha}) \end{array}}{\begin{array}{c}\vdots \\ \hline a(\boldsymbol{\alpha}, \boldsymbol{\gamma}) = a(\boldsymbol{\gamma}, \boldsymbol{\alpha}) \vdash \\ a(\boldsymbol{\alpha}, s(\boldsymbol{\gamma})) = a(s(\boldsymbol{\gamma}),\boldsymbol{\alpha}) \ \mathbf{IND} \\ \hline a(\boldsymbol{\alpha}, 0) = a(0, \boldsymbol{\alpha}) \vdash \\ a(\boldsymbol{\alpha}, \boldsymbol{\beta}) = a(\boldsymbol{\beta}, \boldsymbol{\alpha})\end{array}}\ cut$$

To get from **PRA** to **PA** we need to remove the requirement of only considering strict mv**LKIE**-proofs. In terms of P-schema, this would mean allowing computational sub-P-schema. We can build a mv**LKIE**-proof χ containing a sub-derivation ψ which is a strict mv**LKIE**-proof by allowing strong quantification on the passive parameters of ψ in χ. The problem is that doing so can possibly destroy the translation of Section 3. To show this is not possible we just have to consider translation of χ in parts, first we translate ψ and then we translate χ without ψ, that is replacing ψ with a theory axiom during translation. Once we finish the translation of both parts we glue them back together to get a translation of the original proof χ. This results in the following theorems:

Theorem 4. *A* **P***-schema* Ψ *with* $es(\Psi) = \mathcal{S}$ *exists iff a mv***LKIE***-derivation of* $\mathcal{S}$ *exists.*

Proof. We can convert Ψ into an mv**LKIE**-derivation by structural induction over the number of passive parameters not associated with computational sub-**P**-schema. First we consider strict **P**-schema (Theorem 3). As the IH we assume that the theorem holds for the first n computational sub-**P**-schema of Ψ, then we show it for $n+1$. □

Finally, we can extend the results of this section to **PA**.

Theorem 5. *There is an $\mathcal{E}$-free mv**LKIE**-derivation of a sequent S iff there is a **PA**-calculus proof of S.*

Proof. Note that the **PA** induction rule is a special case of the *mv***LKIE** induction rule and thus making backwards translation trivial. □

5 Conclusion

In this paper we generalized the proof schemata formalism of [17] to a much larger fragment of arithmetic. We refer to this generalization as **P**-schema and prove that a subset of **P**-schema, which we refer to as strict **P**-schema, is equivalent to **PRA** by providing a translation between strict **P**-schema and a standard formulation of primitive recursive arithmetic using the **LK**-calculus. Furthermore, we show that removing this restriction results in a formalism equivalent **PA** formalized as a theory extension of the **LK**-calculus. Also, as was addressed in [9], we would like to develop a calculus for construction of **P**-schema directly. One topic concerning proof schemata which has not been investigated is using inductive definitions other than the natural numbers to index the proof. We plan to investigate generalizations of the indexing sort in future work.

Concerning reflection principles for theories equivalent to **PA** [19], this work provides an interesting and non-trivial example of such a reflection providing an alternative and advantageous perspective of the theory of **PA**. So far most research into schematic formalisms has been focused on proof transformation, the area which gave to birth the concept. By providing an equivalence result with a strong arithmetic theory, we hope others will find interest in this formalism.

Acknowledgements: We would like to thank Michael Lettmann, Alexander Leitsch and Matthias Baaz for contributing to the polishing of this work.

References

[1] Matthias Baaz, Stefan Hetzl, Alexander Leitsch, Clemens Richter, and Hendrik Spohr. Ceres: An analysis of Fürstenberg's proof of the infinity of primes. *Theoretical Computer Science*, 403(2-3):160–175, August 2008.

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

[3] Matthias Baaz and Piotr Wojtylak. Generalizing proofs in monadic languages. *Annals of Pure and Applied Logic*, 154(2):71 – 138, 2008.

[4] 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. 2005.

[5] James Brotherston and Alex Simpson. Sequent calculi for induction and infinite descent. *Journal of Logic and Computation*, 21(6):1177–1216, 2010.

[6] Wilfried Buchholz. Notation systems for infinitary derivations. *Archive for Mathematical Logic*, 30(5):277–296, Sep 1991.

[7] David M. Cerna. A General Recursive Construction for Schematic Resolution Derivations. Risc report series, Research Institute for Symbolic Computation (RISC), Johannes Kepler University Linz, Schloss Hagenberg, 4232 Hagenberg, Austria, 2018. submitted for review.

[8] 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.

[9] David M. Cerna and Michael Lettmann. Integrating a global induction mechanism into a sequent calculus. In *TABLEAUX'17*, Lecture Notes in Comp. Sci., Sept. 2017.

[10] David M. Cerna and Michael Lettmann. Towards a clausal analysis of proof schemata. In *SYNASC'17*, IEEE Xplorer. IEEE, Sept. 2017.

[11] Haskell Curry. A formalization of recursive arithmetic. *American Journal of Mathematics*, 1941.

[12] Cvetan Dunchev, Alexander Leitsch, Mikheil Rukhaia, and Daniel Weller. Cut-elimination and proof schemata. In *Logic, Language, and Computation*, pages 117–136, 2013.

[13] 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.

[14] Kurt Gödel. Über eine bisher noch nicht bentzte erweiterung des finiten standpunktes. *Dialectica*, 12(3–4):280–287, 1958.

[15] Stefan Hetzl and Tin Lok Wong. Restricted notions of provability by induction. abs/1704.01930, 2017.

[16] Jeroen Ketema, Jan Willem Klop, and Vincent van Oostrom. Vicious circles in orthogonal term rewriting systems. *Electr. Notes Theor. Comput. Sci.*, 124(2):65–77, 2005.

[17] Alexander Leitsch, Nicolas Peltier, and Daniel Weller. CERES for first-order schemata. *J. Log. Comput.*, 27(7):1897–1954, 2017.

[18] Raymond Mcdowell and Dale Miller. Cut-elimination for a logic with definitions and induction. *Theoretical Computer Science*, 232, 1997.

[19] R. J. Parikh. Some results on the length of proofs. *Transactions of the American Mathematical Society*, 177:pp. 29–36, 1973.

[20] Reuben N. S. Rowe and James Brotherston. Realizability in cyclic proof: Extracting ordering information for infinite descent. In *Automated Reasoning with Analytic Tableaux and Related Methods - 26th International Conference, TABLEAUX 2017, Brasília, Brazil, September 25-28, 2017, Proceedings*, pages 295–310, 2017.

[21] J. R. Shoenfield. On a restricted ω-rule. *Bulletin de l'Acadimie Polonaise des Sciences*, 7:405–407, 1959.

[22] Harold Simmons. The realm of primitive recursion. *Archive for Mathematical Logic*, 27(2):177–188, Sep 1988.

[23] Alex Simpson. Cyclic arithmetic is equivalent to peano arithmetic. In *Foundations of Software Science and Computation Structures - 20th International Conference, FOSSACS 2017, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2017, Uppsala, Sweden, April 22-29, 2017, Proceedings*, pages 283–300, 2017.

[24] Gaisi Takeuti. *Proof Theory*, volume 81 of *Studies in logic and the foundations of mathematics*. American Elsevier Pub., 1975.