Peano Arithmetic and muMALL: an Extended Abstract

Matteo Manighetti and Dale Miller
1 Introduction

We propose to examine some of the proof theory of arithmetic by using three proof systems. A linearized version of arithmetic, named $\tilde{\mu}$MALL, is MALL plus logical connectives to treat first-order term structures: equality and inequality, first-order universal and existential quantification, and the least and greatest fixed point operators. The proof system $\tilde{\mu}$LKp is an extension of $\tilde{\mu}$MALL in which contraction and weakening are permitted and $\tilde{\mu}$LKp$^+$ is a further extension in which the cut rule is permitted. As their names implies, $\tilde{\mu}$LKp and $\tilde{\mu}$LKp$^+$ involves polarized classical formulas, as defined below.

It is known that $\tilde{\mu}$MALL has a cut-elimination result and is therefore consistent [2, 3]. We will show that $\tilde{\mu}$LKp is consistent by embedding it into second-order linear logic. We also show that $\tilde{\mu}$LKp$^+$ contains Peano arithmetic and that in a couple of different situations, $\tilde{\mu}$LKp is conservative over $\tilde{\mu}$MALL. Finally, we show that a proof that a relation represents a total function can be turned into a proof-search-based algorithm to compute that function.

Since we are interested in using $\tilde{\mu}$MALL to study arithmetic, we use first-order structures to encode natural numbers and fixed points to encode relations among numbers. This focus is in contrast to uses of the propositional subset of $\tilde{\mu}$MALL as a typing systems (see, for example, [7]). We shall limit ourselves to using invariants to reason about fixed points instead of employing other methods, such as infinitary proof systems (e.g., [4]) and cyclic proof systems (e.g., [6,17]).

1.1 Polarized and unpolarized formulas

Following Church’s Simple Theory of Types [5], we shall view the formulas and terms of arithmetic as simply typed $\lambda$-terms using the primitive types $o$ and $i$, respectively. Propositional connectives have the usual typing: $o \rightarrow o \rightarrow o$ for binary connectives and $o$ for the units. There are six connectives that have types involving $i$, namely, $=$ and $\neq$, both of type $i \rightarrow i \rightarrow o$; $\forall$ and $\exists$, both of type $(i \rightarrow o) \rightarrow o$; and $\mu$ and $\nu$, both of type $((i \rightarrow o) \rightarrow (i \rightarrow o)) \rightarrow (i \rightarrow o)$. These latter two connectives denote the least and greatest fixed point operators for one argument: additional such operators can easily be added to handle arities more than 1.

Formulas in our development of arithmetic are divided into two classes. Neither class will have atomic formulas, i.e., there are no (undefined) predicates. Unpolarized formulas are built using $\wedge$, $\bigvee$, $\forall$, $\exists$, $\mu$, and $\nu$. Polarized formulas are built using $\otimes$, 1, $\top$, $\perp$, $\&$, $\square$, 0, $\leftarrow$, $\not\in$, $\forall$, $\exists$, $\mu$, and $\nu$. Note that the six connectives that have $i$ in their typing are used in both of these classes: these are also the connectives that have an unambiguous polarity (we discuss polarity below).

As defined, both polarized and unpolarized formulas are in negation normal form in the sense that they contain no occurrences of negation. For convenience, we will occasionally allow implications in...
unpolarized formulas: in those cases, we treat $P \supset Q$ as $\overline{P} \lor Q$ where $\overline{P}$ is the result of replacing every occurrence of the logical connectives in $P$ with its De Morgan dual (following the usual conventions for classical and linear logics and where = and $\neq$ are duals, as are $\mu$ and $\nu$).

1.2 Polarity of formulas

The connectives used in polarized formulas are given a polarity. The connectives $\neg$, $\bot$, $\&$, $\top$, $\forall$, $\neq$, and $\forall$ are negative while their De Morgan duals are positive. A polarized formula is positive or negative depending only on the polarity of its top-most connective.

A polarized formula $\hat{Q}$ is a polarized version of the unpolarized formula $Q$ if every occurrence of $\&$ and $\otimes$ in $\hat{Q}$ is replaced by $\land$ in $Q$, every occurrence of $\neg$ and $\oplus$ in $\hat{Q}$ is replaced by $\lor$ in $Q$, every occurrence of $1$ and $\top$ in $\hat{Q}$ is replaced by $tt$ in $Q$, and every occurrence of $0$ and $\bot$ in $\hat{Q}$ is replaced by $ff$ in $Q$. Notice that if $Q$ has $n$ occurrences of propositional connectives, then there are $2^n$ formulas $\hat{Q}$ that are polarized versions of $Q$.

Fixed point expression, such as $((\mu \lambda x (B P x)) t)$, introduce variables of predicate type (here, $P$). In the case of the $\mu$ fixed point, any expression built using that predicate variable will be considered to be polarized positively. If the $\forall$ operator is used instead, any expressions built using the predicate variables it introduces is considered to be polarized negatively.

A formula is purely positive (resp., purely negative) if every logical connective it contains is positive (resp., negative). We generalize the familiar arithmetical hierarchy notation by using it to classify polarized formulas as follows. The $\Sigma_1$-formulas are exactly the purely positive formulas, and the $\Pi_1$-formulas are exactly the purely negative formulas. More generally, for $n \geq 1$, the $\Pi_{n+1}$-formulas are those negative formulas for which every positive subformula occurrence is a $\Sigma_n$-formula. Similarly, the $\Sigma_{n+1}$-formulas are those positive formulas for which every negative subformula occurrence is a $\Pi_n$-formula. A formula in $\Sigma_n$ or in $\Pi_n$ has at most $n - 1$ alternations of polarity. Clearly, the dual of a $\Sigma_n$-formula is a $\Pi_n$-formula, and vice versa. We shall also extend these classifications of formulas to abstractions over terms: thus, we say that the term $\lambda x. B$ of type $t \rightarrow o$ is in $\Sigma_n$ if $B$ is a $\Sigma_n$-formula.

2 Linear and classical proof systems for polarized formulas

The μMALL proof system [2,3] for polarized formulas is the one-sided sequent calculus proof system given in Figure 1. The variable $y$ in the $\forall$-introduction rule is an eigenvariable: it is restricted to not be free in any formula in the conclusion of that rule. The application of a substitution $\theta$ to a signature $\Sigma$ (written $\Sigma \theta$ in the $\neq$ rule in Figure 1) is the signature that results from removing from $\Sigma$ the variables in the domain of $\theta$ and adding back any variable that is free in the range of $\theta$. In the $\neq$-introduction rule, if the terms $t$ and $t'$ are not unifiable, the premise is empty and immediately proves the conclusion.

The choice of using Church’s $\lambda$-notation provides an elegant treatment of higher-order substitutions (needed for handing induction invariants) and provides a simple treatment of fixed point expressions and the binding mechanisms used there. In particular, we shall assume that formulas in sequents are always treated modulo $\alpha \beta \eta$-conversion. We usually display formulas in $\beta \eta$-long normal form when presenting sequents. Note that formula expressions such as $B S \bar{t}$ (see Figure 1) are parsed as $(\cdots ((B S) t_1) \cdots t_n)$ if $\bar{t}$ is the list of terms $t_1, \ldots, t_n$.

If we were working in a two-sided sequent calculus, the $\forall$-rule in Figure 1 could be written in the
following two ways.

\[
\frac{\Gamma \vdash \mu \nu}{\Gamma \vdash \mu B\nu} \quad \frac{\Gamma \vdash \mu B(\mu B)\nu}{\Gamma \vdash \mu B\nu} \quad \frac{\Gamma \vdash \mu B\nu}{\Gamma \vdash \mu B, vB\nu}
\]

That is, the one rule for \( \nu \) yields both coinduction and induction. In general, we shall speak of the higher-order substitution term \( S \) used in both of these rules as the invariant of that rule (i.e., we will not use the term co-invariant even though that might be more appropriate in some settings).

We make the following observations about this proof system.

1. The \( \mu \nu \) rule is a limited form of the initial rule. The general form of the initial rule, namely, that the sequent \( \vdash Q, Q \) is provable, is admissible.

2. The \( \mu \) rule allows the \( \mu \) fixed point to be unfolded. This rule captures, in part, the identification of \( \mu B \) with \( B(\mu B) \); that is, \( \mu B \) is a fixed point of \( B \). This inference rule allows one occurrence of \( B \) in \( (\mu B) \) to be expanded to two occurrences of \( B \) in \( B(\mu B) \). In this way, unbounded behavior can appear in \( \bar{\mu} \text{MALL} \) where it does not occur in MALL.

3. The \textit{unfold} rule in Figure 2, which simply unfolds \( \nu \)-expression, is admissible in \( \bar{\mu} \text{MALL} \) by using the \( \nu \)-rule with the invariant \( S = B(\nu B) \).

4. The weakening and contraction rules are admissible in \( \bar{\mu} \text{MALL} \) for purely negative formulas.

5. The proof rules for equality guarantee that function symbols are all treated injectively: thus, function symbols will act only as term constructors.

We define \( \bar{\mu} \text{LKp} \) to be the proof system \( \bar{\mu} \text{MALL} \) but with the inference rules for contraction \( C \) and weakening \( W \) (see Figure 2) added to \( \bar{\mu} \text{MALL} \). In addition, we define \( \bar{\mu} \text{LKp}^+ \) to be \( \bar{\mu} \text{LKp} \) but with the cut rule added (also in Figure 2).

**Example 1.** The formula \( \forall x \forall y [x = y \lor x \neq y] \) can be polarized as either

\[
\forall x \forall y [x = y \lor x \neq y] \quad \text{or} \quad \forall x \forall y [x = y \lor x \neq y].
\]

Only the first of these is provable in \( \bar{\mu} \text{MALL} \), although both formulas are provable in \( \bar{\mu} \text{LKp} \).
3 \(\tilde{\mu}\text{LKp}^+\) and Peano Arithmetic

While the cut rule (Figure \ref{fig:cutrule}) is admissible in \(\tilde{\mu}\text{MALL}\) \cite{okada1991natural}, it is currently open as to whether or not the cut rule is admissible in \(\tilde{\mu}\text{LKP}\). We conjecture that \(\tilde{\mu}\text{LKp}\) and \(\tilde{\mu}\text{LKp}^+\) prove the same sequents. We can prove, however, that \(\tilde{\mu}\text{LKp}\) is consistent.

**Theorem 1.** \(\tilde{\mu}\text{LKp}\) is consistent.

The proof of this theorem can be found in Appendix \[A\]. It is worth noting that adding contraction to some logical systems with weak forms of fixed points can change that logic from being consistent to inconsistent. For example, both Girard \cite{girard1992linear} and Schroeder-Heister \cite{schroeder1981proof} describe a variant of linear logic with unfolding fixed points that is consistent, but when contraction is added, it becomes inconsistent. In their case, negations are allowed in the body of fixed point definitions. The theorem above proves that adding contraction to \(\tilde{\mu}\text{MALL}\) does not lead to inconsistency.

In order to show that Peano arithmetic is contained in \(\tilde{\mu}\text{LKp}^+\), we need to deal with the following three aspects of logic.

**Terms** We introduce the primitive type \(i\) and the term-level signature \(\{z: i, s: i \to i\}\), for zero and successor. We shall write numerals in bold, that is, \(\mathbf{0}, \mathbf{1}, \mathbf{2}\), etc are abbreviations for \(z, (s\, z), (s\,(s\, z))\), etc. We also introduce an abbreviation for the predicate that holds only for such numerals.

\[
\text{nat} = \mu \lambda N \lambda n (n = 0 \oplus \exists m (n = (s\, m) \otimes N\, m))
\]

**Formulas** We define the mapping \((\cdot)\) that translates formulas in Peano arithmetic into polarized formulas. The propositional connectives \(\land, \top, \lor\) are mapped to polarized versions, say, \(\oplus, 1, \ominus, 0\), respectively. The connectives \(=\) and \(\neq\) map to themselves. The first-order quantifiers are mapped so that they become explicitly typed, as follows. Recall that in Church’s STT representation of quantified formulas, the universally quantified formula \(\forall x.B\) is an abbreviation for \(\forall (\lambda x.B)\): here, \(\forall\) is a constant of type \(i \to o \to o\). Similarly, the existential quantifier is coded by the constant \(\exists\) of the same type. The function \((\cdot)\) replaces every occurrence of \(\forall\) with \(\lambda B. \forall x (\text{nat} x \ominus (B\, x))\) and every occurrence of \(\exists\) with \(\lambda B. \exists x (\text{nat} x \otimes (B\, x))\).

**Proofs** Peano Arithmetic is usually presented as a theory consisting of the following axioms and axiom scheme.

\[
\begin{align*}
\forall x. (sx) \neq z & \quad \forall x. \forall y(x + sx) = s(x + y) \\
\forall x\forall y. (sx = sy) \supset (x = y) & \quad \forall x. (x \cdot z = z) \\
\forall x. (x + z = x) & \quad \forall x. (x \cdot y = (x \cdot y + x)) \\
(Az \land \forall x. (Ax \supset A(s\, x))) & \supset \forall x. Ax
\end{align*}
\]

Since we wish to avoid introducing the extra constructors \(+\) and \(\cdot\), we encode addition and multiplications as relations. We can then extend the translation \((\cdot)\) to include

\[
(x + y = w)^o := \mu \lambda P. n\lambda m\lambda p ((n = z \otimes m = p) \oplus \exists n'\exists p' (n = (s\, n') \otimes p = (s\, p') \otimes P\, n'\, m\, p'))
\]

and

\[
(x \cdot y = w)^o := \mu \lambda M. n\lambda m\lambda p ((n = z \otimes p = z) \oplus \exists n'\exists p' (n = (s\, n') \otimes plus\, m\, p' \otimes M\, n'\, m\, p'))
\]

using the following fixed point definitions.

\[
\begin{align*}
\text{plus} & := \mu \lambda p. n\lambda m\lambda p ((n = z \otimes m = p) \oplus \exists n'\exists p' (n = (s\, n') \otimes p = (s\, p') \otimes P\, n'\, m\, p')) \\
\text{mult} & := \mu \lambda M. n\lambda m\lambda p ((n = z \otimes p = z) \oplus \exists n'\exists p' (n = (s\, n') \otimes plus\, m\, p' \otimes M\, n'\, m\, p'))
\end{align*}
\]
Our reusing of the familiar notation for the arithmetic hierarchy for classifying polarized formula is partially justified in the following sense: for all \( n \geq 1 \), if \( B \) is an unpolarized \( \Pi_n \)-formula then \( B^\circ \) is \( \Pi_n \), and if \( B \) is an unpolarized \( \Sigma_n \)-formula then \( B^\circ \) is \( \Sigma_n \).

**Theorem 2** (\( \tilde{\mu}LKP^+ \) contains Peano arithmetic). Let \( Q \) be any unpolarized formula and let \( \hat{Q} \) be a polarized version of \( Q \). If \( Q \) is provable in Peano arithmetic then \( (\hat{Q})^\circ \) is provable in \( \tilde{\mu}LKP^+ \).

**Proof.** It is easy to prove that \mult \text{ and } \plus \text{ describe precisely the multiplication and addition operations on natural numbers. Furthermore, the translations of the Peano Axioms can all be proved in } \bar{\mu}LKp. We illustrate just one of these axioms here: a polarization of the translation of the induction scheme is

\[
\left( Az \otimes \forall x. \left( n\text{at} \ x \ \forall A x A(s \ x)\right) \ \forall \forall x. \left( n\text{at} \ x \ \forall A x\right) \right)
\]

An application of the \( v \) rule to the second occurrence of \( n\text{at} \ x \) can provide an immediate proof of this axiom. Finally, the cut rule in \( \bar{\mu}LKp^+ \) allows us to encode the inference rule of modus ponens. \( \square \)

### 4 Conservativity results for linearized arithmetic

The following theorem is our first conservativity result.

**Theorem 3.** \( \tilde{\mu}LKp \) is conservative over \( \tilde{\mu}MALL \) for \( \Sigma_1 \)-formulas and \( \Pi_1 \)-formulas. In particular, let \( B \) be a either a \( \Sigma_1 \) or a \( \Pi_1 \)-formula. Then \( \vdash B \) has a \( \tilde{\mu}LKp \) proof if and only if \( \vdash B \) has a \( \tilde{\mu}MALL \) proof.

The case for \( \Sigma_1 \)-formulas is proved by a straightforward argument about the permutation of proof rules for \( \tilde{\mu}LKp \). The case for \( \Pi_1 \)-formulas has a simpler proof since weakening and contraction are admissible rules in \( \tilde{\mu}MALL \) for \( \Pi_1 \)-formulas.

Note that it is clear that if there exists a \( \tilde{\mu}MALL \) proof of a purely positive formula, then that proof does not contain the \( v \) rule, i.e., it does not contain the induction rule. Finally, given that first-order Horn clauses can interpret Turing machines \cite{18}, and given that Horn clauses can easily be encoded using purely positive formulas, it is undecidable whether or not a purely positive expression has a \( \tilde{\mu}MALL \) proof. Similarly, purely positive formulas can be used to specify any general recursive function.

Our next conservativity result requires restricting the complexity of invariants used in the induction rule \( v \). We say that a sequent has a \( \tilde{\mu}LKp(\Sigma_1) \) proof if it has a \( \tilde{\mu}LKp \) proof in which all invariants of the proof are purely positive. This fragment is similar to the fragment \( I\Sigma_1 \) of Peano Arithmetic. A well-known result in the study of arithmetic is the following.

Peano arithmetic is \( \Pi_2 \)-conservative over Heyting arithmetic: if Peano arithmetic proves a \( \Pi_2 \)-formula \( A \), then \( A \) is already provable in Heyting arithmetic \cite{8}.

This result inspires the following theorem.

**Theorem 4.** \( \tilde{\mu}LKp(\Sigma_1) \) is conservative over \( \tilde{\mu}MALL \) for \( \Pi_2 \)-formulas. That is, if \( B \) is a \( \Pi_2 \)-formula such that \( \vdash B \) has a \( \tilde{\mu}LKp(\Sigma_1) \) proof, then \( \vdash B \) has a \( \tilde{\mu}MALL \) proof.

A proof for this theorem can be found in Appendix B.

**Example 2.** Example 1 lists two polarized formulas. The formula \( \forall x\forall y[x = y \ \forall x \neq y] \) is \( \Pi_2 \) and is provable in both \( \tilde{\mu}MALL \) and \( \tilde{\mu}LKp \), while the formula \( \forall x\forall y[x = y \oplus x \neq y] \) is \( \Pi_3 \) and is provable in \( \tilde{\mu}LKp \) but not in \( \tilde{\mu}MALL \).
5 Using proof search to compute functions

One way to prove that a binary relation \( \phi \) encodes a function is to prove the totality and determinancy properties of \( \phi \): that is, prove
\[
[\forall x \exists y. \phi(x,y)] \land [\forall x \forall y_1 \forall y_2. \phi(x,y_1) \supset \phi(x,y_2) \supset y_1 = y_2].
\]
Clearly, these properties imply that for every natural number \( x \), the predicate \( \lambda y. \phi(x,y) \) denotes a singleton set. If our logic contains a choice operator, such as Church’s definite description operator \( \tau \) \cite{5}, then this function can be represented via the expression \( \lambda x. \tau y. \phi(x,y) \). A more computationally-oriented approach to encoding such functions follows the Curry-Howard approach of relating proof theory to computation \cite{12}: one extracts from a natural deduction proof of \( \forall x \exists y. \phi(x,y) \) a \( \lambda \)-term, which can be seen as an algorithm for computing the implied function. The algorithmic content of such a \( \lambda \)-term arises from a non-deterministic rewriting process that iteratively selects \( \beta \)-redexes for reduction. In most typed \( \lambda \)-calculus systems, all such sequences of rewrites will end in the same normal form, although some sequences of rewrites might be very long, and others can be very short. This section will describe an alternative mechanism for computing functions from their relational specification that relies on using proof search mechanisms instead of the Curry-Howard correspondence.

Note that if \( P \) and \( Q \) are predicates of arity one and if \( P \) denotes a singleton, then \( \exists x [P x \land Q x] \) and
\( \forall x [P x \supset Q x] \) are logically equivalent. We assume here that \( P x \) is a purely positive expression with \( x \) as its only free variable. Notice that the proof search semantics of these equivalent formulas are surprisingly different. In particular, if we attempt to prove \( \exists x [P x \land Q x] \), then we must guess a term \( t \) and then check that \( t \) denotes the element of the singleton (by proving \( P(t) \)). In contrast, if we attempt to prove \( \forall x [P x \supset Q x] \) then we allocate an eigenvariable \( y \) (which we will eventually instantiate with \( t \)) and then attempt to prove the sequent \( \vdash P y \supset Q y \). Such an attempt at building a proof might actually compute the value \( t \) (especially if we can restrict proofs of that implication to not involve the general form of induction).

**Example 3.** The following derivation verifying that 4 is a sum of 2 and 2.

\[
\vdash 2 = (s \ 1) \quad \vdash 4 = (s \ 3) \quad \vdash \text{plus} \ 1 \ 2 \ 3 \quad \otimes \times 2 \\
\vdash 2 = (s \ 1) \otimes 4 = (s \ 3) \otimes (\text{plus} \ 1 \ 2 \ 3) \quad \otimes \times 2 \\
\vdash \exists n' \ P'(2 = (s \ n') \otimes 4 = (s \ 3) \otimes (\text{plus} \ n' \ 2 \ p')) \\
\vdash (2 = 0 \otimes 2 \ 4) \oplus \exists n' \ P'(2 = (s \ n') \otimes 4 = (s \ 3) \otimes (\text{plus} \ n' \ 2 \ p')) \quad \mu \\
\vdash \text{plus} \ 2 \ 2 \ 4 \\
\vdash \exists p. \text{plus} \ 2 \ 2 \ p \quad \exists
\]

To complete this proof, we must construct a similar subproof verifying that \( 1 + 2 = 3 \). In particular, the witness used to instantiate the final \( \exists p \) is, in fact, that sum. Unfortunately, proof construction in this system does not help us construct this sum’s value. Instead, the first step in building such a proof bottom-up starts with guessing a value and checking that it is the correct sum.

**Example 4.** Given the definition of addition on natural numbers above, the following totality and determinancy formulas

\[
[\forall x_1 \forall x_2. \text{nat} \ x_1 \supset \text{nat} \ x_2 \supset \exists y. (\text{plus}(x_1, x_2, y) \land \text{nat} \ y)] \\
[\forall x_1 \forall x_2. \text{nat} \ x_1 \supset \text{nat} \ x_2 \supset \forall y_1 \forall y_2. \text{plus}(x_1, x_2, y_1) \supset \text{plus}(x_1, x_2, y_2) \supset y_1 = y_2]
\]

can be proved in \( \bar{\mu} \text{MALL} \) where these formulas are polarized using the multiplicative connectives. These proofs require both induction and the \( \mu \land \nu \) rule. Using the cut rule with (the obvious) proofs of \( \text{nat} \ 2 \) and
unfolding instead of the more general induction rule: just using unfoldings leads to an unbounded proof

There is also a

Example 5. Let \( P \) be \( \mu (\lambda R.\lambda x.x = 0 \oplus (R (s x))) \). Clearly, \( P \) denotes the singleton set containing zero. There is also a \( \mu \text{MALL} \) proof that \( \forall x [P x \supset \text{nat } x] \), but there is no (cut-free) proof of this theorem that uses unfolding instead of the more general induction rule: just using unfoldings leads to an unbounded proof search attempt which roughly follows the following outline.

\[
\vdash P (s (s y)), \text{nat } y \\
\vdash \text{nat } 0 \\
\vdash P (s y), \text{nat } y \\
\vdash P y, \text{nat } y
\]

Although proof search can contain potentially unbounded branches, we can still use the proof search concepts of unification and non-deterministic search to compute the value within a singleton. We define a non-deterministic algorithm as follows. The state of this algorithm is a triple of the form

\[\langle x_1, \ldots, x_n; B_1, \ldots, B_m; t \rangle,\]

where \( t \) is a term, \( B_1, \ldots, B_m \) is a multiset of purely positive formulas, and all variables free in \( t \) and in the formulas \( B_1, \ldots, B_m \) are in the set of variables \( x_1, \ldots, x_n \). A success state is one of the form \( \langle \cdot; \cdot; t \rangle \) (that is, when \( n = m = 0 \)): such a state is said to have value \( t \).

Given the state \( S = \langle \Sigma; B_1, \ldots, B_m; t \rangle \) with \( m \geq 1 \), we can non-deterministically select one of the \( B_i \) formulas: for the sake of simplicity, assume that we have selected \( B_1 \). We define the transition \( S \Rightarrow S' \) of state \( S \) to state \( S' \) by a case analysis of the top-level structure of \( B_1 \).

- If \( B_1 \) is \( u = v \) and the terms \( u \) and \( v \) are unifiable with most general unifier \( \theta \), then we transition to \( \langle \Sigma \theta; B_2 \theta, \ldots, B_m \theta; t \theta \rangle \).
- If \( B_1 \) is \( B \otimes B' \) then we transition to \( \langle \Sigma; B, B', B_2, \ldots, B_m; t \rangle \).
- If \( B_1 \) is \( B \oplus B' \) then we transition to either \( \langle \Sigma; B, B_2, \ldots, B_m; t \rangle \) or \( \langle \Sigma; B', B_2, \ldots, B_m; t \rangle \).
- If \( B_1 \) is \( \mu B \bar{t} \) then we transition to \( \langle \Sigma; B(\mu B \bar{t}), B_2, \ldots, B_m; t \rangle \).
- If \( B_1 \) is \( \exists y. B \ y \) then we transition to \( \langle \Sigma, y; B y, B_2, \ldots, B_m; t \rangle \) assuming that \( y \) is not in \( \Sigma \).

This non-deterministic algorithm is essentially applying left-introduction rules in a bottom-up fashion and, if there are two premises, selecting (non-deterministically) just one premise to follow.
Lemma 1. Assume that \( P \) is a purely positive expression of type \( i \to o \) and that \( \exists y. Py \) has a \( \bar{\mu}Lkp \) proof. There is a sequence of transitions from the initial state \( \langle y; P y; y \rangle \) to a success state with value \( t \) such that \( P t \) has a \( \bar{\mu}MALL \) proof.

Proof. An augmented state is a structure of the form \( \langle \Sigma \mid \theta ; B_1 \mid \Xi_1, \ldots, B_m \mid \Xi_m ; t \rangle \), where

- \( \theta \) is a substitution with domain equal to \( \Sigma \) and which has no free variables in its range, and
- for all \( i \in \{1, \ldots, m\} \), \( \Xi_i \) is a \( \bar{\mu}MALL \) proof of \( \theta(B_i) \).

Clearly, if we strike out the augmented items (in red), we are left with a regular state. Given that we have a \( \bar{\mu}Lkp \) proof of \( \exists y. Py \), conservativity (Theorem 3) ensures us that we have a \( \bar{\mu}MALL \) proof of \( \exists y. Py \). Thus, we have a \( \bar{\mu}MALL \) proof \( \Xi_0 \) of \( P t \) for some term \( t \). Note that there is no occurrence of induction in \( \Xi_0 \). We now set the initial augmented state to \( \langle y \mid [y \mapsto t] ; P y \mid \Xi_0 ; y \rangle \). As we detail now,

the proof structures \( \Xi \) provide oracles that steer this non-deterministic algorithm to a success state with value \( t \). Given the augmented state \( \langle \Sigma \mid \theta ; B_1 \mid \Xi_1, \ldots, B_m \mid \Xi_m ; s \rangle \), we consider selecting the first pair \( B_1 \mid \Xi_1 \) and consider the structure of \( B_1 \).

- If \( B_1 \) is \( B' \otimes B'' \) then the last inference rule of \( \Xi_1 \) is \( \otimes \) with premises \( \Xi' \) and \( \Xi'' \), and we make a transition to \( \langle \Sigma \mid \theta ; B' \mid \Xi', B'' \mid \Xi'', \ldots, B_m \mid \Xi_m ; s \rangle \).

- If \( B_1 \) is \( B' \oplus B'' \) then the last inference rule of \( \Xi_1 \) is \( \oplus \) and that rule selects either the first or the second disjunct. In either case, let \( \Xi' \) be the proof of its premise. Depending on which of these disjuncts is selected, we make a transition to \( \langle \Sigma \mid \theta ; B' \mid \Xi', B_2 \mid \Xi_2, \ldots, B_m \mid \Xi_m ; s \rangle \) or \( \langle \Sigma \mid \theta ; B'' \mid \Xi'', B_2 \mid \Xi_2, \ldots, B_m \mid \Xi_m ; s \rangle \), respectively.

- If \( B_1 \) is \( \mu B_{\bar{\mu}} \) then the last inference rule of \( \Xi_1 \) is \( \mu \). Let \( \Xi' \) be the proof of the premise of that inference rule. We make a transition to \( \langle \Sigma \mid \theta ; B(\mu B)_{\bar{\mu}} \mid \Xi', B_2 \mid \Xi_2, \ldots, B_m \mid \Xi_m ; s \rangle \).

- If \( B_1 \) is \( \exists y. B y \) then the last inference rule of \( \Xi_1 \) is \( \exists \). Let \( r \) be the substitution term used to introduce this \( \exists \) quantifier and let \( \Xi' \) be the proof of the premise of that inference rule. Then we make a transition to \( \langle \Sigma, w \mid \theta \circ \varphi ; B w \mid \Xi', B_2 \mid \Xi_2, \ldots, B_m \mid \Xi_m ; s \rangle \), where \( w \) is a variable not in \( \Sigma \) and \( \varphi \) is the substitution \([w \mapsto r] \). Here, we assume that the composition of substitutions satisfies the equation \( (\theta \circ \varphi)(x) = \varphi(\theta(x)) \).

- If \( B_1 \) is \( u = v \) and the terms \( u \) and \( v \) are unifiable with most general unifier \( \varphi \), then we make a transition to \( \langle \Sigma \varphi \mid \rho ; \varphi(B_2) \mid \Xi_2, \ldots, \varphi(B_m) \mid \Xi_m ; (\varphi t) \rangle \) where \( \rho \) is the substitution such that \( \theta = \varphi \circ \rho \).

In each of these cases, we must show that the transition is made to an augmented state. This is easy to show in all but the last two rules above. In the case of the transition due to \( \exists \), we know that \( \Xi' \) is a proof of \( \theta(B r) \), but that formula is simply \( \varphi(\theta(B w)) \) since \( w \) is new and \( r \) contains no variables free in \( \Sigma \). In the case of the transition due to equality, we know that \( \Xi_1 \) is a proof of the formula \( \theta(u = v) \) which means that \( \theta u \) and \( \theta v \) are the same terms and, hence, that \( u \) and \( v \) are unifiable and that \( \theta \) is a unifier. Let \( \varphi \) be the most general unifier of \( u \) and \( v \). Thus, there is a substitution \( \rho \) such that \( \theta = \varphi \circ \rho \) and, for \( i \in \{2, \ldots, m\} \), \( \Xi_i \) is a proof of \( \varphi \circ \rho(B_i) \). Finally, termination of this algorithm is ensured since the number of occurrences of inference rules in the included proofs decreases at every step of the transition. Since we have shown that there is an augmented path that terminates, we have that there exists a path of states to a success state with value \( t \).

This lemma ensures that our search algorithm can compute a member from a non-empty set, give a \( \bar{\mu}Lkp \) proof that that set is non-empty.

We can now prove the following theorem about singleton sets. We abbreviate \( (\exists x.P x) \land (\forall x_1 \forall x_2.P x_1 \supset P x_2 \supset x_1 = x_2) \) by \( \exists!x.P x \) in the following theorem.
Theorem 5. Assume that $P$ is a purely positive expression of type $i \to o$ and that $\exists y. P_y$ has a $\mu LKp$ proof. There is a sequence of transitions from the initial state $\langle y; P_y; y \rangle$ to a success state of value $t$ if and only if $P_t$ has a $\mu LKp$ proof.

Proof. Given a (cut-free) $\mu LKp$ proof of $\exists y. P_y$, that proof contains a $\mu MALL$ proof of $\exists y. P_y$. Since this formula is purely positive, there is a $\mu MALL$ proof for $\exists y. P_y$. The forward direction is immediate: given a sequence of transitions from the initial state $\langle y; P_y; y \rangle$ to the success state $\langle \cdot; \cdot; t \rangle$, it is easy to build a $\mu MALL$ proof of $P_t$. Conversely, assume that there is a $\mu MALL$ proof of $P_t$ for some term $t$. By conservativity, there is a $\mu MALL$ proof of $P_t$ and, hence, of $\exists y. P_y$. By Lemma 1 there is a sequence of transitions from initial state $\langle y; P_y; y \rangle$ to the success state $\langle \cdot; \cdot; s \rangle$, where $P_s$ has a $\mu MALL$ proof. Given that $P_t$ and $P_s$ and $\forall x_1 \forall x_2. P_{x_1} \supset P_{x_2} \supset x_1 = x_2$ all have $\mu LKp^+$ proofs, using the cut rule, we can conclude that $t = s$.

Thus, a (naive) proof-search algorithm involving both unification and non-deterministic search is sufficient for computing the functions encoded in relations.

While it is easy to encode the proof of totality for the Ackermann function in $\mu LKp$, it seems unlikely that a totality proof for that function can be done within $\mu MALL$. This separation between $\mu LKp$ and $\mu MALL$ was conjectured by Baelde [1, Section 3.5]. There are also several other linear logic style systems for which the totality of Ackermann’s function is known to be not provable. In particular, if we developed a Curry-Howard interpretation of $\mu MALL$, it would yield a system close to the linear $\lambda$-terms $H(\emptyset)$ of [14], which is known to capture exactly primitive recursive functions (see also similar results in [13]).

Acknowledgment: We thank the anonymous reviewers of an earlier draft of this paper for they valuable comments.

References

A Consistency of $\tilde{\mu}$LKP

By second-order linear logic, $LL_2$, we mean the logic of MALL with the addition of the following logical connectives: two exponentials $!$ and $?$, negation $(\cdot)^\bot$, equality and non-equality, and first-order and second-order quantification (no occurrences of fixed points are permitted). Cut-elimination of this version of $LL_2$ follows from Girard’s original cut-elimination proof [9] (see also [15]) and the cut-elimination proofs known for equality and non-equality [11][16].

We translate $\tilde{\mu}$LKP formulas into $LL_2$ formulas by translating fixed point expressions into second-order quantified formulas. The least fixed point expression $\mu B \vec{x}$ should be translated to a formula roughly of the form $\forall S \left( !\left( \forall \vec{y} \cdot B \vec{y} \rightarrow S \vec{y} \right) \rightarrow S \vec{x} \right)$. This translation must also insert $?$ into formulas in order to account for the fact that in $\tilde{\mu}$LKP, any formula can be contracted and weakened at any point in a proof. The translation is given as follows.

- $[t = s] = ?(t = s)$ and $[t \neq s] = ?(t \neq s)$
- $[\forall x. Px] = ?\forall x. [Px]$ and $[\exists x. Px] = ?\exists x. [Px]$.
- $[B \otimes C] = ?([B] \otimes [C])$, $[B \oslash C] = ?([B] \oslash [C])$, $[B \& C] = ?([B] \& [C])$, $[B \oplus C] = ?([B] \oplus [C])$
- $[1] = ?1$, $[\bot] = ?\bot$, $[0] = ?0$, $[\top] = ?\top$
- $[\mu B \vec{x}] = ?\forall S[?(?) y. [B] S \vec{y} \oslash (S \vec{y})^\bot \oslash S \vec{x}]$
- $[\nu B \vec{x}] = ?\exists S[?(?) y. [B] S \vec{y} \oslash (S \vec{y})^\bot \oslash S \vec{x}]$
- $[A] = A$ where $A$ is an atomic formula.
• The $\cdot$ operator commutes with $\lambda$-abstraction: $[\lambda x.B] = \lambda x.[B]$. This feature of $\cdot$ permits translating invariants and the body of fixed point expressions.

• The $\cdot$ operator can be applied to a multiset of formulas: $[\Gamma] = \{[P] \mid P \in \Gamma\}$.

Note that when $B$ is the $\lambda$-abstraction $\lambda p \lambda \bar{x}.C$, where $C$ is a $\mu$MALL formula, $p$ is a first-order predicate variable, and $\bar{x}$ is a list of first-order variables, then $[B][S]\bar{t}$ is equal to $[BS]\bar{t}$ up to $\lambda$-conversion. We shall also need the following inference rule in $LL_2$, which is a kind of generalization of the cut rule.

$$\frac{\Gamma, BQ\bar{t}}{\Gamma, \neg(Q\bar{t}).P\bar{t}}$$

Here, of course, the first-order variables $\bar{x}$ are new. Also, the expression $B$ has the type that takes a first-order predicate to a first-order predicate and also monotonic, meaning that there are no occurrences of negated predicate variables in $B$. It is proved in [3, Proposition 2] that this rule is admissible in $LL_2$.

This rule essentially allows us to move from the fact that $Q \subseteq P$ and to the fact that $BQ \subseteq BP$.

**Lemma 2.** If $\vdash \Gamma$ is derivable in $\mu$LK$P$ then $\vdash [\Gamma]$ is derivable in $LL_2$.

**Proof.** We proceed by induction on the structure of cut-free $\mu$LK$P$ proofs. In particular, assume that $\vdash \Gamma$ has a cut-free $\mu$LK$P$ proof $\Xi$.

**Case:** The last inference rule of $\Xi$ comes from Figure[1] i.e., it is an introduction rules for a propositional connective, a unit, or a quantifier. For example, assume that this last inference rule is the following $\otimes$ introduction rule.

$$\frac{\Gamma, P}{\Gamma, \Delta, Q \otimes Q}$$

By the inductive assumption, $\vdash [\Gamma], [P]$ and $\vdash [\Delta], [Q]$ have $LL_2$ proofs. Hence, $\vdash [\Gamma], [\Delta], [P] \otimes [Q]$ has an $LL_2$ proof. By using the dereliction rule for $\otimes$ and the definition of $[\cdot]$, we know that $\vdash [\Gamma, \Delta], [P \otimes Q]$ has an $LL_2$ proof.

**Case:** The last inference rule is either weakening $W$ or contraction $C$. Since the image of $[\cdot]$ always has a $\otimes$ exponential as its top-level connective, the corresponding $LL_2$ inference rule is built with the same structural rule.

**Case:** The last inference rule of $\Xi$ is one of the fixed point rules from Figure[1] Assume, for example, that the last rule is

$$\vdash \mu B\bar{t}, \nu B\bar{t}$$

The desired translation of this inference rule into $LL_2$ is

$$\frac{\vdash B[S\bar{y}], [B]((\lambda \bar{w}(S\bar{w})^+)\bar{y})}{\vdash (S\bar{y})^+, ((S\bar{x})^+)}$$

An induction on the structure of the formula $B$ provides a proof that there is an $LL_2$ proof of remaining open premise.
Assume instead that the last rule of $\Xi$ is the introduction for $\nu$, namely,
\[
\frac{\vdash \Gamma, S\bar{t}}{\vdash \Gamma, \nu B\bar{t}} \quad \nu.
\]
The higher-order quantifier that appears in the $LL2$ encoding is instantiated with $[S]$. Thus, the desired $LL2$ proof is
\[
\vdash [BS\bar{x}], [S\bar{x}] \vdash \neg((S\bar{x})), \neg([S]\bar{x}) \quad \text{cut}
\]
\[
\vdash (\Gamma), [S\bar{t}] \quad \text{init}
\]
\[
\vdash (\Gamma), !((\forall y' . [S]y' \neg\neg([S]y')) \otimes [S]\bar{t}) \quad \text{cut}
\]
By the inductive hypothesis, the leftmost and rightmost premises have $LL2$ proof. Induction on first-order abstractions such as $S$ shows that the middle premise also has an $LL2$ proof.

Assume instead that the last rule of $\Xi$ is the introduction for $\mu$, namely,
\[
\frac{\vdash \Gamma, B(\mu B)\bar{t}}{\vdash \Gamma, \mu B\bar{t}} \quad \mu.
\]
We first show that $\vdash [B] [\mu B\bar{t}] \rightarrow [\mu B\bar{t}]$ has an $LL2$ proof for all $B$ and $\bar{t}$.

Here, $\Xi$ is a straightforward $LL2$ proof. Finally, using this proof of $\vdash [B] [\mu B\bar{t}]$, $\mu B\bar{t}$ and the cut rule for $LL2$, we have shown the soundness of the $\mu$ rule in Figure 1.

Proof of Theorem 1 Assume that $\vdash B$ and $\vdash B$ have $\mu LKp$ proofs. By Lemma 2, we know that $\vdash [B]$ and $\vdash [B]$ have $LL2$ proofs. While it is not the case that $[B] = ([B])^{+}$, a simple induction on the structure of $B$ shows that $[B]$ is provable in $LL2$. Since $LL2$ has a cut rule, we know that there is an $LL2$ proof of $\vdash -$ (the empty sequent). By the cut-elimination theorem of $LL2$, this sequent also has a cut-free $LL2$ proof, which is impossible.

B $\mu LKp(\Sigma 1)$ is conservative over $\mu MALL$ for $\Pi 2$-formulas

In this section we prove that any $\Pi 2$ formula provable in $\mu LKp(\Sigma 1)$ is provable in $\mu MALL$. This conservativity result can be applied to the formulas stating the totality and determinacy properties (see Section 5) of relations defined by $\Sigma 1$-formulas, since they are all $\Pi 2$ formulas. The proof of this result would be aided greatly if we had a focusing theorem for $\mu LKp$. If we take the focused proof system for $\mu MALL$ given in [2,3] and add contraction and weakening in the usual fashion, we have a natural candidate for a focused proof system for $\mu LKp$. However, the completeness of that proof system is currently open. As Girard points out in [10], the completeness of such a focused (cut-free) proof system would
allow the extraction of the constructive content of classical $\Pi^0_2$ theorems, and we should not expect such a result to follow from the usual ways that we prove cut-elimination and the completeness of focusing. As a result of not possessing such a focused proof system for $\tilde{\mu}LKp$, we must reproduce aspects of focusing in order to prove our conservation result.

**Definition 1.** A reduced sequent is a sequent that contains only purely negative, purely positive, and $\Pi_2$ formulas. If $\Gamma_1$ and $\Gamma_2$ are reduced sequents, we say that $\Gamma_1$ contains $\Gamma_2$ if $\Gamma_2$ is a sub-multiset of $\Gamma_1$. Finally, we say that a reduced sequent is a pointed sequent if it contains exactly one formula that is either purely positive or $\Pi_2$.

**Definition 2.** A positive region is a cut-free $\tilde{\mu}LKp(\Sigma_1)$ proof that contains only the inference rules $\mu\nu$, contractions, weakening, and introduction rules for the positive connectives.

**Definition 3.** The $C\nu\nu$ rule is the following derived rule of inference.

\[
\begin{array}{c}
\vdash \Gamma, S\vec{t}, U\vec{t} \\
\vdash BU_{\vec{x}}, U\vec{x} \\
\vdash BS_{\vec{x}}, S\vec{x}
\end{array}
\]

\[
\Gamma, vB\vec{t} \vdash C
\]

The $C\nu\nu$ rule is justified as the following combination of $\nu$ and contraction rules.

\[
\begin{array}{c}
\vdash \Gamma, S\vec{t}, U\vec{t} \\
\vdash BU_{\vec{x}}, U\vec{x} \\
\vdash \Gamma, vB\vec{t}, S\vec{t} \\
\vdash BS_{\vec{x}}, S\vec{x}
\end{array}
\]

\[
\Gamma, vB\vec{t} \vdash C
\]

Since we are working within $\tilde{\mu}LKp(\Sigma_1)$, the invariants $S$ and $U$ are purely positive.

**Definition 4.** A negative region is a cut-free $\tilde{\mu}LKp(\Sigma_1)$ partial proof in which the open premises are all reduced sequent and where the only inference rules are introductions for negative connectives plus the $C\nu\nu$ rule.

**Lemma 3.** If a reduced sequent $\Gamma$ has a positive region proof then $\Gamma$ contains a pointed sequent that has a $\tilde{\mu}$MALL proof.

**Proof.** This proof is a simple generalization of the proof of Theorem 3. \qed

**Lemma 4.** If every premise of a negative region contains a pointed sequent with a $\tilde{\mu}$MALL proof, then the conclusion of the negative region contains a pointed sequent with a $\tilde{\mu}$MALL proof.

**Proof.** This proof is by induction on the height of the negative region. The most interesting case to examine is the one where the last inference rule of the negative region is the $C\nu\nu$ rule. Referring to the inference rule displayed above, the inductive hypothesis ensures that the reduced sequent $\vdash \Gamma, S\vec{t}, U\vec{t}$ contains a pointed sequent $\Delta, C$ where $\Delta$ is a multiset of purely negative formula in $\Gamma$ and where the formula $C$ (that is either purely positive or is $\Pi_2$) is either a member of $\Gamma$ or is equal to either $S\vec{t}$ or $U\vec{t}$. In the first case, $\Delta, C$ is also contained in the endsequent $\Gamma, vB\vec{t}$. In the second case, we have one of the following proofs:

\[
\begin{array}{c}
\vdash \Delta, S\vec{t} \\
\vdash BS_{\vec{x}}, S\vec{x}
\end{array}
\]

\[
\Gamma, vB\vec{t} \vdash C
\]

\[
\begin{array}{c}
\vdash \Delta, U\vec{t} \\
\vdash BU_{\vec{x}}, U\vec{x}
\end{array}
\]

\[
\Gamma, vB\vec{t} \vdash C
\]

depending on whether or not $C$ is $S\vec{t}$ or $U\vec{t}$. \qed
Lemma 5. If the reduced sequent $\Gamma$ has a cut-free $\bar{\mu}L\bar{K}(\Sigma_1)$ proof then $\Gamma$ has a proof that can be divided into a negative region that proves $\Gamma$ in which all its premises have positive region proofs.

Proof. This lemma is proved by appealing to the permutation of inference rules. As shown in [2], the introduction rules for negative connectives permute down over all inference rules in $\bar{\mu}$MALL. Not considered in that paper is how such negative introduction rules permute down over contractions. It is easy to check that such permutations do, in fact, happen except in the case of the $\nu$ rule. In general, contractions below a $\nu$ rule will not permute upwards, and, as a result, the negative region is designed to include the $C\nu\nu$ rule (where contraction is stuck with the $\nu$ rule). As a result, negative rules (including $C\nu\nu$) permute down while contraction and introductions of positive connectives permute upward. This gives rise to the two-region proof structure.

By combining the results of this section we have a proof of Theorem [4]