Untyping Typed Algebraic Structures and Colouring Proof Nets of Cyclic Linear Logic

Damien Pous

Untyping Typed Algebraic Structures and Colouring Proof Nets of Cyclic Linear Logic

Damien Pous*

CNRS (LIG, UMR 5217)

Abstract. We prove "untyping" theorems: in some typed theories (semirings, Kleene algebras, residuated lattices, involutive residuated lattices), typed equations can be derived from the underlying untyped equations. As a consequence, the corresponding untyped decision procedures can be extended for free to the typed settings. Some of these theorems are obtained via a detour through fragments of cyclic linear logic, and give rise to a substantial optimisation of standard proof search algorithms.

1 Introduction

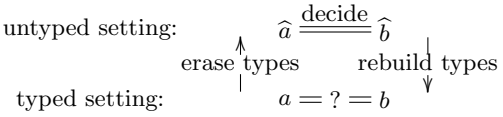
Motivations. The literature contains many decidability or complexity results for various algebraic structures. Some of these structures (rings, Kleene algebras [18], residuated lattices [26]) can be generalised to *typed* structures, where the elements come with a domain and a codomain, and where operations are defined only when these domains and codomains agree according to some simple rules. Although such typed structures are frequently encountered in practice (e.g., rectangular matrices, heterogeneous binary relations, or more generally, categories), there are apparently no proper tools to easily reason about these.

This is notably problematic in proof assistants, where powerful decision procedures are required to let the user focus on difficult reasoning steps by leaving administrative details to the computer. Indeed, although some important theories that can be decided automatically in Coq or HOL (e.g., Presburger arithmetic [24], elementary real algebra [13], rings [12]), there are no high-level tools to reason about heterogeneous relations or rectangular matrices.

In this paper, we show how to extend the standard decision procedures from the untyped structures to the corresponding typed structures. In particular, we make it possible to use standard tools to reason about rectangular matrices or heterogeneous relations, without bothering about types (i.e., matrix dimensions or domain/codomain information). The approach we propose is depicted below: we study "untyping" theorems that allow one to prove typed equations as follows: 1) erase type informations, 2) prove the equation using standard, untyped,

* Work partially funded by the French ANR projet blanc "Curry-Howard pour la Concurrence" CHOCO ANR-07-BLAN-0324

decision procedures, and 3) derive a typed proof from the untyped one.



Besides the theoretical aspects, an important motivation behind this work comes from a Coq library [4] in which we developed efficient tactics for partial axiomatisations of relations: the ideas presented here were used and integrated in this library to extend our tactics to typed structures, for free.

Overview. We shall mainly focus on the two algebraic structures we mentioned above, since they raise different problems and illustrate several aspects of these untyping theorems: Kleene algebras [17] and residuated lattices [16].

- The case of Kleene algebras is the simplest one. The main difficulty comes from the annihilating element (0): its polymorphic typing rule requires us to show that equational proofs can be factorised so as to use the annihilation laws at first, and then reason using the other axioms.
- The case of residuated structures is more involved: due to the particular form of axioms about residuals, we cannot rely on standard equational axiomatisations of these structures. Instead, we need to exploit an equivalent cut-free sequent proof system (first proposed by Ono and Komori [26]), and to notice that this proof system corresponds to the intuitionistic fragment of cyclic linear logic [32]. The latter logic is much more concise and the corresponding proof nets are easier to reason about, so that we obtain the untyping theorem in this setting. We finally port the result back to residuated lattices by standard means.

The above sequent proof systems have the sub-formula property, so that they yield decision procedures, using proof search algorithms. As an unexpected application, we show that the untyping theorem makes it possible to improve these algorithms by reducing the set of proofs that have to be explored.

Outline. We introduce our notations and make the notion of typed structure precise in §2. We study Kleene algebras and residuated lattices in §3 and §4, respectively. We conclude with applications, related work, and directions for future work in §5.

2 Notation, typed structures

Let $\mathcal{X}$ be an arbitrary set of *variables*, ranged over using letters x, y. Given a signature Σ, we let a, b, c range over the set $T(\Sigma + \mathcal{X})$ of *terms with variables*. Given a set $\mathcal{T}$ of *objects* (ranged over using letters n, m, p, q), a *type* is a pair (n, m) of objects (which we denote by $n \to m$, following categorical notation), a *type environment* $\Gamma : \mathcal{X} \to \mathcal{T}^2$ is a function from variables to types, and we will

define *type judgements* of the form $\Gamma \vdash a : n \to m$, to be read "in environment Γ, term a has type $n \to m$, or, equivalently, a is a morphism from n to m". By $\Gamma \vdash a, b : n \to m$, we mean that both a and b have type $n \to m$; type judgements will include the following rule for variables:

$$\frac{\Gamma(x) = (n, m)}{\Gamma \vdash x : n \to m}\ \mathrm{TV}$$

Similarly, we will define *typed equality* judgements of the form $\Gamma \vdash a = b : n \to m$: "in environment Γ, terms a and b are equal, at type $n \to m$". Equality judgements will generally include the following rules, so as to obtain an equivalence relation at each type:

$$\frac{\Gamma(x) = (n, m)}{\Gamma \vdash x = x : n \to m}\ \mathrm{V} \qquad \frac{\Gamma \vdash a = b : n \to m \quad \Gamma \vdash b = c : n \to m}{\Gamma \vdash a = c : n \to m}\ \mathrm{T} \qquad \frac{\Gamma \vdash a = b : n \to m}{\Gamma \vdash b = a : n \to m}\ \mathrm{S}$$

By taking the singleton set as set of objects ($\mathcal{T} = \{\emptyset\}$), we recover standard, untyped structures: the only typing environment is $\widehat{\emptyset} : x \mapsto (\emptyset, \emptyset)$, and types become uninformative (this corresponds to working in a one-object category; all operations are total functions). To alleviate notations, since the typing environment will always be either $\widehat{\emptyset}$ or an abstract constant value Γ, we shall leave it implicit in type and equality judgements, by relying on the absence or the presence of types to indicate which one to use. For example, we shall write $\vdash a = b : n \to m$ for $\Gamma \vdash a = b : n \to m$, while $\vdash a = b$ will denote the judgement $\widehat{\emptyset} \vdash a = b : \emptyset \to \emptyset$.

The question we study in this paper is the following one: given a signature and a set of inference rules defining a type judgement and an equality judgement, does the implication below hold, for all a, b, n, m such that $\vdash a, b : n \to m$?

$$\vdash a = b \quad \textit{entails} \quad \vdash a = b : n \to m\ .$$

In other words, in order to prove an equality in a typed structure, is it safe to remove all type annotations, so as to work in the untyped underlying structure?

3 Kleene algebras

We study the case of residuated lattices in §4; here we focus on Kleene algebras. In order to illustrate our methodology, we actually give the proof in three steps, by considering two intermediate algebraic structures: monoids and semirings. The former admit a rather simple and direct proof, while the latter are sufficient to expose concisely the main difficulty in handling Kleene algebras.

3.1 Monoids

Definition 1. *Typed monoids* are defined by the signature $\{\cdot_2, 1_0\}$, together with the following inference rules, in addition to the rules from §2.

$$\frac{}{\vdash 1 : n \to n}\ \textsc{To} \qquad \frac{\vdash a : n \to m \qquad \vdash b : m \to p}{\vdash a \cdot b : n \to p}\ \textsc{Td} \qquad \frac{}{\vdash 1 = 1 : n \to n}\ \textsc{o}$$

$$\frac{\vdash a = a' : n \to m \qquad \vdash b = b' : m \to p}{\vdash a \cdot b = a' \cdot b' : n \to p}\ \textsc{d} \qquad \frac{\vdash a : n \to m}{\vdash 1 \cdot a = a : n \to m}\ \textsc{od}$$

$$\frac{\vdash a : n \to m \qquad \vdash b : m \to p \qquad \vdash c : p \to q}{\vdash (a \cdot b) \cdot c = a \cdot (b \cdot c) : n \to q}\ \textsc{da} \qquad \frac{\vdash a : n \to m}{\vdash a \cdot 1 = a : n \to m}\ \textsc{do}$$

In other words, typed monoids are just categories: 1 and $\cdot$ correspond to identities and composition. Rules (O) and (D) ensure that equality is reflexive at each type (point (i) below) and preserved by composition. As expected, equalities relate correctly typed terms only (ii):

Lemma 2. *(i) If* $\vdash a : n \to m$*, then* $\vdash a = a : n \to m$*.*
(ii) If $\vdash a = b : n \to m$*, then* $\vdash a, b : n \to m$*.*

Moreover, in this setting, type judgements enjoy some form of injectivity (types are not uniquely determined due to 1, which is typed in a polymorphic way):

Lemma 3. *If* $\vdash a : n \to m$ *and* $\vdash a : n' \to m'$*, then we have* $n = n'$ *iff* $m = m'$*.*

We need another lemma to obtain the untyping theorem: all terms related by the untyped equality admit the same type derivations.

Lemma 4. *If* $\vdash a = b$*; then for all* n, m*, we have* $\vdash a : n \to m$ *iff* $\vdash b : n \to m$*.*

Theorem 5. *If* $\vdash a = b$ *and* $\vdash a, b : n \to m$*, then* $\vdash a = b : n \to m$*.*

Proof. We reason by induction on the derivation $\vdash a = b$; the interesting cases are the following ones:

- the last rule used is the transitivity rule (T): we have $\vdash a = b$, $\vdash b = c$, $\vdash a, c : n \to m$, and we need to show that $\vdash a = c : n \to m$. By Lemma 4, we have $\vdash b : n \to m$, so that by the induction hypotheses, we get $\vdash a = b : n \to m$ and $\vdash b = c : n \to m$, and we can apply rule (T).
- the last rule used is the compatibility of $\cdot$ (D): we have $\vdash a = a'$, $\vdash b = b'$, $\vdash a \cdot b, a' \cdot b' : n \to m$, and we need to show that $\vdash a \cdot b = a' \cdot b' : n \to m$. By case analysis on the typing judgements, we deduce that $\vdash a : n \to p$, $\vdash b : p \to m$, $\vdash a' : n \to q$, $\vdash b' : q \to m$, for some p, q. Thanks to Lemmas 3 and 4, we have $p = q$, so that we can conclude using the induction hypotheses ($\vdash a = a' : n \to p$ and $\vdash b = b' : p \to m$), and rule (D). ∎

Note that the converse of Theorem 5 ($\vdash a = b : n \to m$ entails $\vdash a = b$) is straightforward, so that we actually have an equivalence.

3.2 Non-commutative semirings

Definition 6. *Typed semirings* are defined by the signature $\{\cdot_2, +_2, 1_0, 0_0\}$, together with the following rules, in addition to the rules from Def. 1 and §2.

$$\frac{}{\vdash 0 : n \to m}\ \textsc{Tz} \qquad \frac{\vdash a, b : n \to m}{\vdash a + b : n \to m}\ \textsc{Tp} \qquad \frac{\vdash a = a' : n \to m \qquad \vdash b = b' : n \to m}{\vdash a + b = a' + b' : n \to m}\ \textsc{p}$$

$$\frac{}{\vdash 0 = 0 : n \to m}\ \textsc{z} \qquad \frac{\vdash a : n \to m}{\vdash a + 0 = a : n \to m}\ \textsc{pz} \qquad \frac{\vdash a, b : n \to m}{\vdash a + b = b + a : n \to m}\ \textsc{pc}$$

$$\frac{\vdash a, b, c : n \to m}{\vdash (a + b) + c = a + (b + c) : n \to m}\ \textsc{pa} \qquad \frac{\vdash a : n \to m \qquad \vdash b, c : m \to p}{\vdash a \cdot (b + c) = a \cdot b + a \cdot c : n \to p}\ \textsc{dp}$$

$$\frac{\vdash a : n \to m}{\vdash a \cdot 0 = 0 : n \to p}\ \textsc{dz} \qquad \frac{\vdash a : n \to m}{\vdash 0 \cdot a = 0 : p \to m}\ \textsc{zd} \qquad \frac{\vdash a : n \to m \qquad \vdash b, c : p \to n}{\vdash (b + c) \cdot a = b \cdot a + c \cdot a : p \to m}\ \textsc{pd}$$

In other words, typed semiring are categories enriched over a commutative monoid: each homset is equipped with a commutative monoid structure, and composition distributes over these monoid structures.

Lemma 2 is also valid in this setting: equality is reflexive and relates correctly typed terms only. However, due to the presence of the annihilator element (0), Lemmas 3 and 4 no longer hold: 0 has any type, and we have $\vdash x \cdot 0 \cdot x = 0$ while $x \cdot 0 \cdot x$ only admits $\Gamma(x)$ as a valid type. Moreover, some valid proofs cannot be typed just by adding decorations: for example, $0 = 0 \cdot a \cdot a = 0$ is a valid untyped proof of $0 = 0$; however, this proof cannot be typed if a has a non-square type. Therefore, we have to adopt another strategy: we reduce the problem to the annihilator-free case, by showing that equality proofs can be factorised so as to use rules (PZ), (DZ), and (ZD) at first, as oriented rewriting rules.

Definition 7. Let a be a term; we denote by $a_\downarrow$ the *normal form of* a, obtained with the following convergent rewriting system:

$$a + 0 \to a \qquad 0 + a \to a \qquad 0 \cdot a \to 0 \qquad a \cdot 0 \to 0$$

We say that a is *strict* if $a_\downarrow \neq 0$. We let $\_ \vdash^+ \_ = \_ : \_ \to \_$ denote the *strict equality* judgement obtained by removing rules (DZ) and (ZD), and requiring a to be strict in rules (DP) and (PD).

On strict terms, we recover the injectivity property of types we had for monoids. Then, using the same methodology as previously, one easily obtain the untyping theorem for strict equality judgements.

Lemma 8. *For all strict terms a such that $\vdash a : n \to m$ and $\vdash a : n' \to m'$, we have $n = n'$ iff $m = m'$.*

Proposition 9. *If $\vdash^+ a = b$ and $\vdash a, b : n \to m$, then $\vdash^+ a = b : n \to m$.*

Note that the patched rules for distributivity, (DP$^+$) and (PD$^+$) are required in order to obtain the counterpart of Lemma 4: if a was not required to be strict, we would have $\vdash^+ 0 \cdot (x + y) = 0 \cdot x + 0 \cdot y$, and the right-hand side can be typed in environment $\Gamma = \{x \mapsto (3, 2),\ y \mapsto (4, 2)\}$ while the left-hand side cannot.

We now have to show that any equality proof can be factorised, so as to obtain a strict equality proof relating the corresponding normal forms:

Proposition 10. *If $\vdash a = b$, then we have $\vdash^+ a_\downarrow = b_\downarrow$.*

Proof. We first show by induction that whenever $\vdash a = b$, a is strict iff b is strict (†). Then we proceed by induction on the derivation $\vdash a = b$, we detail only some cases:

- (D) we have $\vdash^+ a_\downarrow = a'_\downarrow$ and $\vdash^+ b_\downarrow = b'_\downarrow$ by induction; we need to show that $\vdash^+ (a \cdot b)_\downarrow = (a' \cdot b')_\downarrow$. If one of a, a', b, b' is not strict, then $(a \cdot b)_\downarrow = (a' \cdot b')_\downarrow = 0$, thanks to (†), so that we are done; otherwise, $(a \cdot b)_\downarrow = a_\downarrow \cdot b_\downarrow$, and $(a' \cdot b')_\downarrow = a'_\downarrow \cdot b'_\downarrow$, so that we can apply rule (D).
- (DZ) trivial, since $(a \cdot 0)_\downarrow = 0$.
- (DP) we need to show that $\vdash^+ (a \cdot (b + c))_\downarrow = (a \cdot b + a \cdot c)_\downarrow$; if one of a, b, c is not strict, both sides reduce to the same term, so that we can apply Lemma 2(i) (which holds in this setting); otherwise we have $(a \cdot (b + c))_\downarrow = a_\downarrow \cdot (b_\downarrow + c_\downarrow)$ and $(a \cdot b + a \cdot c)_\downarrow = a_\downarrow \cdot b_\downarrow + a_\downarrow \cdot c_\downarrow$, so that we can apply rule (DP$^+$) ($a_\downarrow$ is obviously strict). ■

Since the normalisation procedure preserves types and respects equalities, we finally obtain the untyping theorem.

Lemma 11. *If $\vdash a : n \to m$, then $\vdash a_\downarrow : n \to m$ and $\vdash a = a_\downarrow : n \to m$.*

Theorem 12. *In semirings, for all a, b, n, m such that $\vdash a, b : n \to m$, we have $\vdash a = b$ iff $\vdash a = b : n \to m$.*

3.3 Kleene algebras

Kleene algebras are idempotent semirings equipped with a star operation [17]; they admit several important models, among which binary relations and *regular languages* (the latter is complete [21,18]; since equality of regular languages is decidable, so is the equational theory of Kleene algebras). Like previously, we type Kleene algebras in a natural way, where star operates on "square" types: types of the form $n \to n$, i.e., square matrices or homogeneous binary relations.

Definition 13. *Typed Kleene algebras* are defined by the signature $\{\cdot_2, +_2, \star_1, 1_0, 0_0\}$, together with the following rules, in addition that from Defs. 1 and 6, and §2), and where $\vdash a \leq b : n \to m$ is an abbreviation for $\vdash a + b = b : n \to m$.

$$\frac{\vdash a : n \to n}{\vdash a^\star : n \to n}\ \text{TS} \qquad \frac{\vdash a = b : n \to n}{\vdash a^\star = b^\star : n \to n}\ \text{S} \qquad \frac{\vdash a : n \to m}{\vdash a + a = a : n \to m}\ \text{PI}$$

$$\frac{\vdash a : n \to n}{\vdash 1 + a \cdot a^\star = a^\star : n \to n}\ \text{SP} \qquad \frac{\vdash a \cdot b \leq b : n \to m}{\vdash a^\star \cdot b \leq b : n \to m}\ \text{SL} \qquad \frac{\vdash b \cdot a \leq b : n \to m}{\vdash b \cdot a^\star \leq b : n \to m}\ \text{SR}$$

The untyped version of this axiomatisation is that from Kozen [18]: axiom (PI) corresponds to idempotence of $+$, the three other rules define the star operation (we omitted the mirror image of axiom (SP), which is derivable from the other ones [4]). One can extend the proofs from the previous section without unexpected difficulties: we add the rule $0^\star \to 1$ to the rewriting system used for normalising terms, and we require b to be strict in the strict versions of rules (SL) and (SR). We do not give more details here: complete proofs are available as Coq scripts [28].

Theorem 14. *In Kleene algebras, for all a, b, n, m such that $\vdash a, b : n \to m$, we have $\vdash a = b$ iff $\vdash a = b : n \to m$.*

4 Residuated lattices

We now move to our second main example, *residuated lattices*. These structures also admit binary relations as models; they are of special interest since they make it possible to reason algebraically about well-founded relations. For example, one can use residuation to prove Newman's Lemma in relation algebras [9]. We start with a simpler structure.

A *residuated monoid* is a tuple $(X, \leq, \cdot, 1, \backslash, /)$, such that $(X, \leq)$ is a partial order, $(X, \cdot, 1)$ is a monoid whose product is monotonic ($a \leq a'$ and $b \leq b'$ entail $a \cdot b \leq a' \cdot b'$), and $\backslash, /$ are binary operations, respectively called *left* and *right divisions*, characterised by the following equivalences:

$$a \cdot b \leq c \quad \Leftrightarrow \quad b \leq a \backslash c \quad \Leftrightarrow \quad a \leq c / b$$

Accordingly, divisions can be typed in a natural way using following rules:

$$\frac{\vdash c : n \to m \qquad \vdash a : n \to p}{\vdash a \backslash c : p \to m}\ \textsc{Tl} \qquad \frac{\vdash c : n \to m \qquad \vdash b : p \to m}{\vdash c / b : n \to p}\ \textsc{Tr}$$

Although we can easily define a set of axioms to capture equalities provable in residuated monoids [16], the transitivity rule (T) becomes problematic in this setting (there is no counterpart to Lemma 4). Instead, we exploit a characterisation due to Ono and Komori [26], based on a Gentzen proof system for the full Lambek calculus [22]. Indeed, the "cut" rule corresponding to this system, which plays the role of the transitivity rule, can be eliminated. Therefore, this characterisation allows us to avoid the problems we encountered with standard equational proof systems. In some sense, moving to cut-free proofs corresponds to using a factorisation system, like we did in the previous section (Prop. 10).

4.1 Gentzen proof system for residuated monoids

Let l, k, h range over lists of terms, let $l; k$ denote the concatenation of l and k, and let ϵ be the empty list. The Gentzen proof system is presented below;

it relates lists of terms to terms. It is quite standard [16]: there is an axiom rule (V), and, for each operator, an introduction and an elimination rule.

$$\frac{}{x \vdash x}\ \textsc{v} \qquad \frac{}{\epsilon \vdash 1}\ \textsc{Io} \qquad \frac{l \vdash a \qquad l' \vdash a'}{l; l' \vdash a \cdot a'}\ \textsc{Id} \qquad \frac{l; b \vdash a}{l \vdash a/b}\ \textsc{Ir} \qquad \frac{b; l \vdash a}{l \vdash b\backslash a}\ \textsc{Il}$$

$$\frac{l; l' \vdash a}{l; 1; l' \vdash a}\ \textsc{Eo} \qquad \frac{l; b; c; l' \vdash a}{l; b \cdot c; l' \vdash a}\ \textsc{Ed} \qquad \frac{k \vdash b \qquad l; c; l' \vdash a}{l; c/b; k; l' \vdash a}\ \textsc{Er} \qquad \frac{k \vdash b \qquad l; c; l' \vdash a}{l; k; b\backslash c; l' \vdash a}\ \textsc{El}$$

The axiom rule can be generalised to terms (*i*), the cut rule is admissible (*ii*), and the proof system is correct and complete w.r.t. residuated monoids (*iii*).

Proposition 15. *(i) For all a, we have $a \vdash a$.*
(ii) For all l, k, k', a, b such that $l \vdash a$ and $k; a; k' \vdash b$, we have $k; l; k' \vdash b$.
(iii) For all a, b, we have $a \vdash b$ iff $a \leq b$ holds in all residuated monoids.

Proof. Point (*i*) is easy; see [26,25,16] for cut admissibility and completeness. ■

Type decorations can be added to the proof system rather easily (see Fig. 2 in the appendix). However, using this proof system, we were able to prove the untyping theorem only for the unit-free fragment: we needed to assume that terms have at most one type, which is not true in presence of 1. This proof was rather involved, so that we did not manage to circumvent this difficulty in a nice and direct way. Instead, as hinted in the introduction, we have to move to the following more symmetrical setting.

4.2 Cyclic MLL

The sequent system for residuated monoids actually corresponds to a non-commutative version of intuitionistic multiplicative linear logic (IMLL) [11]: the product ($\cdot$) is a non-commutative tensor ($\otimes$), and left and right divisions ($\backslash, /$) are the corresponding left and right linear implications ($\multimap, \circ\!\!-$). Moreover, it happens that this system is just the intuitionistic fragment of cyclic multiplicative linear logic (MLL) [32]. The untyping theorem turned out to be easier to prove in this setting, which we describe below.

We assume a copy $\mathcal{X}^\perp$ of the set of variables ($\mathcal{X}$), and we denote by $x^\perp$ the corresponding elements which we call *dual variables*. From now on, we shall consider terms with both kinds of variables: $T(\Sigma + X + X^\perp)$. We keep an algebraic terminology to remain consistent with the previous sections; notice that using terminology from logic, a term is a formula and a variable is an atomic formula.

Definition 16. *Typed MLL terms* are defined by the signature $\{\otimes_2, \parr_2, 1_0, \perp_0\}$, together with the following typing rules:

$$\frac{\Gamma(x) = (n, m)}{\vdash x : n \to m}\ \textsc{Tv} \qquad \frac{}{\vdash 1 : n \to n}\ \textsc{T}_1 \qquad \frac{\vdash a : n \to m \qquad \vdash b : m \to p}{\vdash a \otimes b : n \to p}\ \textsc{T}_\otimes$$

$$\frac{\Gamma(x) = (n, m)}{\vdash x^\perp : m \to n}\ \textsc{Tv}^\perp \qquad \frac{}{\vdash \perp : n \to n}\ \textsc{T}_\perp \qquad \frac{\vdash a : n \to m \qquad \vdash b : m \to p}{\vdash a \parr b : n \to p}\ \textsc{T}_\parr$$

$$\frac{\Gamma(x) = (n, m)}{\vdash x^\perp; x : m}\ \mathrm{A} \qquad \frac{}{\vdash 1 : n}\ 1 \qquad \frac{\vdash l : n}{\vdash \bot; l : n}\ \bot \qquad \frac{\vdash l; a : n \qquad \vdash b; k : n}{\vdash l; a \otimes b; k : n}\ \otimes$$

$$\frac{\vdash a; b; l : n}{\vdash a \parr b; l : n}\ \parr \qquad \frac{\vdash a : n \to m \qquad \vdash l; a : m}{\vdash a; l : n}\ \mathrm{E}$$

Fig. 1. Typed Sequents for Cyclic MLL.

Tensor ($\otimes$) and par ($\parr$) are typed like the previous dot operation; bottom ($\bot$) is typed like the unit (1); dual variables are typed by mirroring the types of the corresponding variables. We extend type judgements to lists of terms as follows:

$$\frac{}{\vdash \epsilon : n \to n}\ \textsc{Te} \qquad \frac{\vdash a : n \to m \qquad \vdash l : m \to p}{\vdash a; l : n \to p}\ \textsc{Tc}$$

(be careful not to confuse $\vdash a, b : n \to m$, which indicates that both a and b have type $n \to m$, with $\vdash a; b : n \to m$, which indicates that the list $a; b$ has type $n \to m$). *Linear negation* is defined over terms and lists of terms as follows:

$$(x)^\perp \triangleq x^\perp \qquad 1^\perp \triangleq \bot \qquad (a \otimes b)^\perp \triangleq b^\perp \parr a^\perp \qquad (a; l)^\perp \triangleq l^\perp; a^\perp$$
$$(x^\perp)^\perp \triangleq x \qquad \bot^\perp \triangleq 1 \qquad (a \parr b)^\perp \triangleq b^\perp \otimes a^\perp \qquad \epsilon^\perp \triangleq \epsilon$$

Note that since we are in a non-commutative setting, negation has to reverse the arguments of tensors and pars, as well as lists. Negation is involutive and mirrors type judgements:

Lemma 17. *For all l, $l^{\perp\perp} = l$; for all l, n, m, $\vdash l : n \to m$ iff $\vdash l^\perp : m \to n$.*

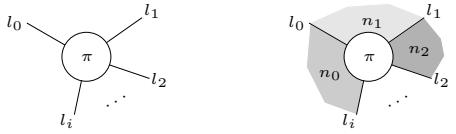
If we were using a two-sided presentation of MLL, judgements would be of the form $l \vdash k : m \to n$, intuitively meaning "$l \vdash k$ is derivable in cyclic MLL, and lists l and k have type $m \to n$". Instead, we work with one-sided sequents to benefit from the symmetrical nature of MLL. At the untyped level, this means that we replace $l \vdash k$ with $\vdash l^\perp; k$. According to the previous intuitions, the list $l^\perp; k$ has a square type $n \to n$: object m is hidden in the concatenation, so that it suffices to record the outer object (n). Judgements finally take the form $\vdash l : n$, meaning "the one-sided MLL sequent $\vdash l$ is derivable at type $n \to n$".

Definition 18. *Typed cyclic MLL* is defined by the sequent calculus from Fig. 1.

Except for type decorations, the system is standard: the five first rules are the logical rules of MLL [11]. Rule (E) is the only structural rule, this is a restricted form of the exchange rule, yielding cyclic permutations: sequents have to be thought of as rings [32]. As before, we added type decorations in a minimal way, so as to ensure that derivable sequents have square types, as explained above:

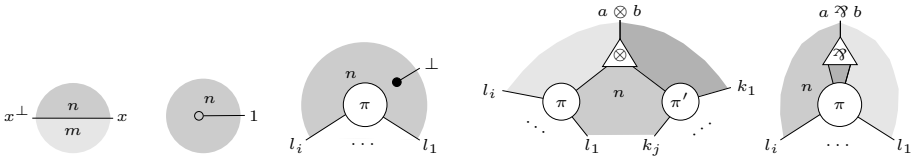
Lemma 19. *For all l, n, if $\vdash l : n$ then $\vdash l : n \to n$.*

We now give a graphical interpretation of the untyping theorem, using proof nets. Since provability is preserved by cyclic permutations, one can draw proof structures by putting the terms of a sequent on a circle [32]. For example, a proof π of a sequent $\vdash l_0, \ldots, l_i$ will be represented by a proof net whose interface is given by the left drawing below.



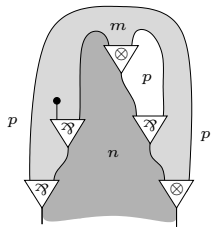
Suppose now that the corresponding list admits a square type: $\vdash l : n \to n$, i.e., $\forall j \leq i$, $\vdash l_j : n_j \to n_{j+1}$, for some $n_0, \ldots, n_{i+1}$ with $n = n_0 = n_{i+1}$. One can add these type decorations as background colours, in the areas delimited by terms, as we did on the right-hand side.

The logical rules of the proof system (Fig. 1) can then be represented by the proof net constructions below (thanks to this sequent representation, the exchange rule (E) is implicit). Since these constructions preserve planarity, all proof nets are planar [2], and the idea of background colours makes sense. Moreover, they can be coloured in a consistent way, so that typed derivations correspond to proof nets that can be entirely and consistently coloured.



Therefore, one way to prove the untyping theorem consists in showing that any proof net whose outer interface can be coloured can be coloured entirely. As an example, we give an untyped derivation below, together with the corresponding proof net. Assuming that $\Gamma(x) = n \to m$ and $\Gamma(y) = m \to p$, the conclusion has type $p \to p$, and the outer interface of the proof net can be coloured (here, with colours p and n). The untyping theorem will ensure that there exists a typed proof; indeed, the whole proof net can be coloured in a consistent way.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\vdash x^\perp; x}\,\mathrm{A} \quad \dfrac{}{\vdash y; y^\perp}\,\mathrm{E,A}}{\vdash x^\perp; (x \otimes y); y^\perp}\,\otimes}{\vdash x^\perp; (x \otimes y) \parr y^\perp}\,\parr \quad \dfrac{}{\vdash y; y^\perp}\,\mathrm{E,A}}{\vdash x^\perp; ((x \otimes y) \parr y^\perp) \otimes y; y^\perp}\,\otimes}{\vdash \perp; x^\perp; ((x \otimes y) \parr y^\perp) \otimes y; y^\perp}\,\perp}{\vdash y^\perp; \perp; x^\perp; ((x \otimes y) \parr y^\perp) \otimes y}\,\mathrm{E}}{\vdash y^\perp \parr \perp \parr x^\perp; ((x \otimes y) \parr y^\perp) \otimes y}\,\parr$$



We now embark in the proof of the untyping theorem for cyclic MLL; the key property is that the types of derivable sequents are all squares:

Proposition 20. *If $\vdash l$ and $\vdash l : n \to m$, then $n = m$.*

Proof. We proceed by induction on the untyped derivation $\vdash l$, but we prove a stronger property: "the potential types of all cyclic permutations of l are squares", i.e., for all h,k such that $l = h; k$, for all n, m such that $\vdash k; h : n \to m$, $n = m$. The most involved case is that of the tensor rule. Using symmetry arguments, we can assume that the cutting point belongs to the left premise: the conclusion of the tensor rule is $\vdash l; l'; a \otimes b; k$, we suppose that the induction hypothesis holds for $l; l'; a$ and $b; k$, and knowing that $\vdash l'; a \otimes b; k; l : n \to m$, we have to show $n = m$. Clearly, we have $\vdash l'; a : n \to p$, $\vdash b; k : p \to q$, and $\vdash l : q \to m$ for some p, q. By induction on the second premise, we have $p = q$, so that $\vdash l'; a; l : n \to m$. Since the latter list is a cyclic permutation of $l; l'; a$, we can conclude with the induction hypothesis on the first premise. ■

Theorem 21. *In cyclic MLL, if $\vdash l : n \to n$, then we have $\vdash l$ iff $\vdash l : n$.*

Proof. The right-to-left implication is straightforward; for the direct implication, we proceed by induction on the untyped derivation. The previous proposition is required in the case of the tensor rule: we know that $\vdash l; a$, $\vdash b; k$, and $\vdash l; a \otimes b; k : n \to n$, and we have to show that $\vdash l; a \otimes b; k : n$. Necessarily, there is some m such that $\vdash l; a : n \to m$ and $\vdash b; k : m \to n$; moreover, by Prop. 20, $n = m$. Therefore, we can apply the induction hypotheses (so that $\vdash l; a : n$ and $\vdash b; k : n$) and we conclude with the typed tensor rule. ■

4.3 Intuitionistic fragment

To deduce that the untyping theorem holds in residuated monoids, it suffices to show that the typed proof system from Fig. 2 corresponds to the intuitionistic fragment of that from Fig. 1. This is well-known for the untyped case, and type decorations do not add particular difficulties. Therefore, we just give a brief overview of the extended proof.

The idea is to define the following families of *input* and *output* terms (Danos-Regnier polarities [30,3]), and to work with sequents composed of exactly one output term and an arbitrary number of input terms.

$$\begin{array}{rl} i ::= & x^\perp \mid \bot \mid i \parr i \mid i \otimes o \mid o \otimes i \\ o ::= & x \mid 1 \mid o \otimes o \mid i \parr o \mid o \parr i \end{array}$$

Negation $(-^\perp)$ establishes a bijection between input and output terms. Terms of residuated monoids (IMLL formulae) are encoded into output terms as follows.

$$\begin{array}{lll} \lfloor a \cdot b \rfloor \triangleq \lfloor a \rfloor \otimes \lfloor b \rfloor & \lfloor a/b \rfloor \triangleq \lfloor a \rfloor \parr \lfloor b \rfloor^\perp & \lfloor x \rfloor \triangleq x \\ \lfloor 1 \rfloor \triangleq 1 & \lfloor a \backslash b \rfloor \triangleq \lfloor a \rfloor^\perp \parr \lfloor b \rfloor & \end{array}$$

This encoding is a bijection between IMLL terms and output MLL terms; it preserves typing judgements:

Lemma 22. *For all a, n, m, we have $\vdash a : n \to m$ iff $\vdash \lfloor a \rfloor : n \to m$.*

(Note that we heavily rely on overloading to keep notation simple.) The next proposition shows that we actually obtained a fragment of typed cyclic MLL; it requires the lemma below: input-only lists are not derivable. The untyping theorem for residuated monoids follows using Thm. 21.

Lemma 23. *If $\vdash l$, then l contains at least one output term.*

Proposition 24. *If $\vdash l, a : n \to m$, then $l \vdash a : n \to m$ iff $\vdash \lfloor l \rfloor^\perp; \lfloor a \rfloor : m$.*

Proof. The forward implication is proved by an induction on the sequent derivation. For the reverse direction, we actually prove the following stronger property, by induction on the untyped MLL derivation: "for all h, a, k such that $\vdash \lfloor h \rfloor^\perp; \lfloor a \rfloor; \lfloor k \rfloor^\perp$, for all n, m such that $\vdash h; k : n \to m$ and $\vdash a : n \to m$, we have $h; k \vdash a : n \to m$". Prop. 20 and Lemma 23 are required for the tensor rule; the generalisation with two lists h, k is required to handle the exchange rule. ■

Corollary 25. *In residuated monoids, if $\vdash l, a : n \to m$, then we have $l \vdash a$ iff $l \vdash a : n \to m$.*

4.4 Residuated lattices: additives.

The Gentzen proof system we presented for residuated monoids was actually designed for residuated *lattices* [26], obtained by further requiring the partial order $(X, \leq)$ to be a lattice $(X, \vee, \wedge)$. Binary relations fall into this family, by considering set-theoretic unions and intersections. The previous proofs scale without major difficulty: on the logical side, this amounts to considering the additive binary connectives $(\oplus, \&)$. By working in multiplicative additive linear logic (MALL) without additive constants, we get an untyping theorem for *involutive residuated lattices* [31]; we deduce the untyping theorem for residuated lattices by considering the corresponding intuitionistic fragment (see [28] for proofs).

On the contrary, and rather surprisingly, the theorem breaks if we include additive constants $(0, \top)$, or equivalently, if we consider *bounded* residuated lattices. The corresponding typing rules are given below, together with the logical rule for top (there is no rule for zero).

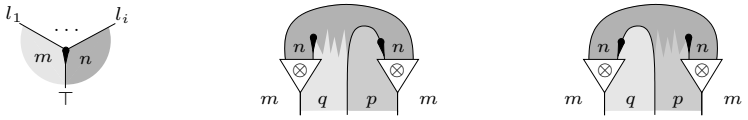
$$\frac{}{\vdash 0 : n \to m}\ \mathrm{T}_0 \qquad \frac{}{\vdash \top : n \to m}\ \mathrm{T}_\top \qquad \frac{\vdash l : m \to n}{\vdash \top; l : n}\ \top$$

The sequent $x^\perp \otimes \top; y^\perp; \top \otimes x$ gives a counter-example. This sequent basically admits the two following untyped proofs:

$$\dfrac{\dfrac{\dfrac{}{\vdash y^\perp; \top}\ \mathrm{E},\top \quad \dfrac{\dfrac{}{\vdash x; x^\perp}\ \mathrm{E},\mathrm{A} \quad \dfrac{}{\vdash \top}\ \top}{\vdash x; x^\perp \otimes \top}\ \otimes}{\vdash y^\perp; \top \otimes x; x^\perp \otimes \top}\ \otimes}{\vdash x^\perp \otimes \top; y^\perp; \top \otimes x}\ \mathrm{E} \qquad \dfrac{\dfrac{\dfrac{}{\vdash \top}\ \top \quad \dfrac{\dfrac{}{\vdash x; x^\perp}\ \mathrm{E},\mathrm{A} \quad \dfrac{}{\vdash \top; y^\perp}\ \top}{\vdash x; x^\perp \otimes \top; y^\perp}\ \otimes}{\vdash \top \otimes x; x^\perp \otimes \top; y^\perp}\ \otimes}{\vdash x^\perp \otimes \top; y^\perp; \top \otimes x}\ \mathrm{E},\mathrm{E}$$

However, this sequent admits the square type $m \to m$ whenever $\Gamma(x) = (n, m)$ and $\Gamma(y) = (p, q)$, while the above proofs cannot be typed unless $n = q$ or $n = p$,

respectively. Graphically, these proofs correspond to the proof nets below (where the proof net construction for rule $(\top)$ is depicted on the left-hand side); these proof nets cannot be coloured unless $n = q$ or $n = p$.



This counter-example for MALL also gives a counter-example for IMALL: the above proofs translate to intuitionistic proofs of $y \cdot (\top \backslash x) \vdash \top \cdot x$, which is also not derivable in the typed setting, unless $n = q$ or $n = p$. The problem is actually even stronger: while $S \cdot (\top \backslash R) \subseteq \top \cdot R$ holds for all homogeneous binary relations R, S (by the above proofs, for example), this law does not hold for arbitrary heterogeneous relations (take, e.g., the empty relation from the empty set to $\{\emptyset\}$ for R, and an arbitrary non-empty relation for S). This shows that we cannot always reduce the analysis of typed structures to that of the underlying untyped structures. Here, the equational theory of heterogeneous binary relations does not reduce to the equational theory of homogeneous binary relations.

5 Conclusions and directions for future work

We proved untyping theorems for several standard structures, allowing us to extend decidability results to the typed settings. All results have been formally checked [28] with the Coq proof assistant. We conclude by discussing applications, related work, and directions for future work.

5.1 Applications

Improving proof search for residuated structures. The sequent proof systems we mentioned in this paper have the sub-formula property, so that provability is decidable in each case, using a simple proof search algorithm [25]. Surprisingly, the concept of types can be used to cut off useless branches. Indeed, recall Prop. 20: "the types of any derivable sequent are squares". By contrapositive, given an untyped sequent l, one can easily compute an abstract 'most general type and environment' $(n \to m, \Gamma)$, such that $\Gamma \vdash l : n \to m$ holds (taking $\mathbb{N}$ as the set of objects, for example); if $n \neq m$, then the sequent is not derivable, and proof search can fail immediately on this sequent.

We did some experiments with a simple prototype [28]: we implemented focused [1] proof search for cyclic MALL, i.e., a recursive algorithm composed of two phases: an *asynchronous* phase which is deterministic, and a *synchronous* phase, where branching occurs (e.g., when applying rule $(\otimes)$). The optimisation consists in checking that the most general type of the sequent is square before entering the synchronous phase. The overall complexity remains exponential—provability is NP-complete [27]—but we get an exponential speed-up: the optimisation allows one to abort proof search immediately on approximately two

sequents over three. Even on small examples (sequents with twenty leaves over ten variables), we gain an order of magnitude; see Fig. 3 in the appendix.

Decision of typed Kleene algebras in Coq. The untyping theorem for typed Kleene algebras is quite important in the ATBR Coq library [4]: it allows one to use our tactic for Kleene algebras [5] in typed settings, and, in particular, with heterogeneous binary relations. The underlying decision procedure being quite involved, we can hardly imagine proving its soundness w.r.t. typed settings. Even writing a type-preserving version of the algorithm seems challenging.

At another level, we used the untyping theorem for semirings in order to formalise Kozen's completeness proof [18] for Kleene algebras (which we had to formalise to reach all models). Indeed, this proof heavily relies on matrix constructions, so that having adequate lemmas and tactics for working with possibly rectangular matrices was a big plus: this allowed us to avoid the ad-hoc constructions Kozen used to inject rectangular matrices into square ones.

5.2 References and related work

The relationship between residuated lattices and substructural logics is due to Ono and Komori [26]; see [16] for a detailed survey about these structures. Cyclic linear logic was suggested by Girard and studied by Yetter [32]. To the best of our knowledge, the idea of adding types to the above structures is new. The axiomatisation of Kleene algebras is due to Kozen [18].

Our typed structures can be seen as very special cases of *partial* algebras [6], where the domain of partial operations is defined by typing judgements. Similarly, one could use *many-sorted* algebras [14] to mimic types using sorts. Several encodings from partial algebras to total ones were proposed in the literature [23,7]. Although they are quite general, these results do not apply here: these encodings do not preserve the considered theory since they need to introduce new symbols and equations; as a consequence, ordinary untyped decision procedures can no longer be used after the translation. Dojer has shown that under some conditions, convergent term rewriting systems for total algebras can be used to prove existence equations in partial algebras [8]. While it seems applicable to semirings, this approach does not scale to Kleene algebras or residuated lattices, for which decidability does not arise from a term rewriting system.

Closer to our work is that from Kozen, who first proposed the idea of untyping typed Kleene algebras, in order to avoid the aforementioned matrix constructions [20]. He provided a different answer, however: using model-theoretic arguments, he proved an untyping theorem for the Horn theory of "1-free Kleene algebras". The restriction to 1-free expressions is required, as shown by the following counter-example: $\vdash 0 = 1 \Rightarrow a = b$ is a theorem of semirings, although there are non trivial typed semirings where $0 = 1$ holds at some types (e.g., empty matrices), while $a = b$ is not universally true at other types.

5.3 Handling other structures

Action algebras [29,15] are a natural extension of the structures we studied in this paper: they are also called *residuated Kleene algebras*; they combine the ingredients from residuated lattices and Kleene algebras. Although we do not know whether the untyping theorem holds in this case, we can think of two strategies to tackle this problem: 1) find a cut-free extension of the Gentzen proof system for residuated lattices and adapt our current proof—such an extension is left as an open question in [15], it would entail decidability of the equational theory of action algebras; 2) find a "direct" proof of the untyping theorem for residuated monoids, without using a Gentzen proof system, so that the methodology we used for Kleene algebras can be extended.

Kleene algebras with tests [19] are another extension of Kleene algebras, which is useful in program verification. Their equational theory is decidable, and although the details have to be checked, the untyping theorem is likely to hold: a possible difficulty could appear with the complement operation from the Boolean algebras of tests, but tests are inherently homogeneous.

Our proofs about semirings can be adapted to handle the cases of *allegories* and *distributive allegories* [10] (see [28] for proofs); however, the case of *division allegories*, where left and right divisions are added, remains open.

5.4 Towards a generic theory

The typed structures we focused on can be described in terms of enriched categories, and the untyping theorems can be rephrased as asserting the existence of faithful functors to one-object categories. It would therefore be interesting to find out whether category theory may help to define a reasonable class of structures for which the untyping theorem holds. In particular, how could we exclude the counter-example with additive constants in MALL?

For structures that are varieties, another approach would consist in using term rewriting theory to obtain generic factorisation theorems (Lemma 10, which we used to handle the annihilating element in semirings, would become a particular case). This seems rather difficult, however, since these kind of properties are quite sensitive to the whole set of operations and axioms that are considered.

Acknowledgements

We warmly thank Olivier Laurent and Tom Hirschowitz for the highly stimulating discussions we had about this work. More generally, we are grateful to all members of the Choco band.

References

1. J.-M. Andreoli. Logic programming with focusing proofs in linear logic. *J. of Logic and Comput.*, 2(3):297–347, 1992.
2. G. Bellin and A. Fleury. Planar and braided proof-nets for MLL with mix. *Archive for Math. Log.*, 37:309–325, 1998.
3. G. Bellin and P. J. Scott. On the pi-calculus and linear logic. *TCS*, 135(1):11–65, 1994.
4. T. Braibant and D. Pous. Coq library: ATBR, algebraic tools for binary relations. `http://sardes.inrialpes.fr/~braibant/atbr/`, May 2009.
5. T. Braibant and D. Pous. An efficient coq tactic for deciding Kleene algebras. In *Proc. ITP*, LNCS. Springer, 2010. (to appear).
6. P. Burmeister. *Algebras and Orders*, chapter Partial Algebra. Kluwer Ac. Pub., 1993.
7. R. Diaconescu. An encoding of partial algebras as total algebras. *Inf. Process. Lett.*, 109(23-24):1245–1251, 2009.
8. N. Dojer. Applying term rewriting to partial algebra theory. *Fund. Inf.*, 63(4):375–384, 2004.
9. H. Doornbos, R. Backhouse, and J. van der Woude. A calculational approach to mathematical induction. *TCS*, 179(1-2):103–135, 1997.
10. P. Freyd and A. Scedrov. *Categories, Allegories*. North Holland, 1990.
11. J.-Y. Girard. Linear logic. *TCS*, 50:1–102, 1987.
12. B. Grégoire and A. Mahboubi. Proving equalities in a commutative ring done right in Coq. In *Proc. TPHOL '05*, volume 3603 of *LNCS*, pages 98–113. Springer, 2005.
13. J. Harrison. A HOL decision procedure for elementary real algebra. In *HUG*, volume 780 of *LNCS*, pages 426–435. Springer, 1993.
14. P. J. Higgins. Algebras with a scheme of operators. *Math. Nachrichten*, 27:115–132, 1963.
15. P. Jipsen. From semirings to residuated Kleene lattices. *Studia Logica*, 76(2):291–303, 2004.
16. P. Jipsen and C. Tsinakis. A survey of residuated lattices. *Ordered Algebraic Structures*, 2002.
17. S. C. Kleene. Representation of events in nerve nets and finite automata. In *Automata Studies*, pages 3–41. Princeton University Press, 1956.
18. D. Kozen. A completeness theorem for Kleene algebras and the algebra of regular events. *Inf. and Comput.*, 110(2):366–390, 1994.
19. D. Kozen. Kleene algebra with tests. *Trans. Prog. Lang. and Sys.*, 19(3):427–443, 1997.
20. D. Kozen. Typed Kleene algebra. Technical Report TR98-1669, Cornell University., 1998.
21. D. Krob. Complete systems of B-rational identities. *TCS*, 89(2):207–343, 1991.
22. J. Lambek. The mathematics of sentence structure. *Amer. Math. Monthly*, 65:154–170, 1958.
23. T. Mossakowski. Relating CASL with other specification languages: the institution level. *TCS*, 286(2):367–475, 2002.
24. M. Norrish. Complete integer decision procedures as derived rules in hol. In *TPHOLs*, volume 2758 of *LNCS*, pages 71–86. Springer, 2003.
25. M. Okada and K. Terui. The finite model property for various fragments of intuitionistic linear logic. *J. Sym. Log.*, 64(2):790–802, 1999.
26. H. Ono and Y. Komori. Logics without the contraction rule. *J. Sym. Log.*, 50(1):169–201, 1985.
27. M. Pentus. Lambek calculus is NP-complete. *TCS*, 357(1-3):186–201, 2006.
28. D. Pous. Web appendix to this paper, 2010. `http://sardes.inrialpes.fr/~pous/utas/`.
29. V. R. Pratt. Action logic and pure induction. In *JELIA*, volume 478 of *LNCS*, pages 97–120. Springer, 1990.
30. L. Regnier. *Lambda-calcul et réseaux*. Thèse de doctorat, Université Paris VII, 1992.
31. A. M. Wille. A Gentzen system for involutive residuated lattices. *Alg. Univ.*, 54:449–463, 2005.
32. D. N. Yetter. Quantales and (noncommutative) linear logic. *J. Sym. Log.*, 55(1):41–64, 1990.

A Additional material

$$\frac{\Gamma(x) = (n, m)}{x \vdash x : n \to m}\ \textsc{v} \qquad \frac{}{\epsilon \vdash 1 : n \to n}\ \textsc{Io} \qquad \frac{\vdash b : m \to p \qquad l; b \vdash a : n \to p}{l \vdash a/b : n \to m}\ \textsc{Ir}$$

$$\frac{l \vdash a : n \to m \qquad l' \vdash a' : m \to p}{l; l' \vdash a \cdot a' : n \to p}\ \textsc{Id} \qquad \frac{\vdash b : p \to m \qquad b; l \vdash a : p \to n}{l \vdash b\backslash a : m \to n}\ \textsc{Il}$$

$$\frac{l; l' \vdash a : n \to m}{l; 1; l' \vdash a : n \to m}\ \textsc{Eo} \qquad \frac{\vdash l' : m \to q \qquad k \vdash b : n \to m \qquad l; c; l' \vdash a : p \to q}{l; c/b; k; l' \vdash a : p \to q}\ \textsc{Er}$$

$$\frac{l; b; c; l' \vdash a : n \to m}{l; b \cdot c; l' \vdash a : n \to m}\ \textsc{Ed} \qquad \frac{\vdash l : p \to m \qquad k \vdash b : m \to n \qquad l; c; l' \vdash a : p \to q}{l; k; b\backslash c; l' \vdash a : p \to q}\ \textsc{El}$$

Fig. 2. Typed Sequents for Residuated Monoids.

	synchronous phase	asynchronous phase
without optimisation	12 400 kc	3 390 kc
with optimisation	500 kc	133 kc
square sequents	164 k	-

Fig. 3. Number of recursive calls to the focused proof search algorithm, with and without the optimisation from §5.1 (tests performed in MALL without additive constants, by running the algorithm on 10 000 random sequents with 20 leaves and 10 distinct variables; 1kc=1000 calls).