a-logic with arrows

Murdoch J. Gabbay[1] Michael J. Gabbay[2]

Abstract

We present an extension of first-order predicate logic with a novel predicate '$\mathbf{at}\,t$' meaning intuitively "this term is a variable symbol". We give simple sequent proof-rules for it, we demonstrate cut-elimination for the resulting logic, and we give a semantics for which the logic is sound and complete.
Because we can now make assertions about what would normally be considered an intensional property of a term (being a variable symbol) we can now express inside the logic, properties of its terms and predicates which would normally be external to the logic. We give axiomatisations in a-logic, including of the lambda-calculus, and discuss what relevance this might have to logic programming.

1 Introduction

a-logic extends classical First-Order Logic (FOL) with a predicate $\mathbf{at}\,t$ such that if t is *not* a variable symbol then $\mathbf{at}\,t$ is contradictory. We read $\mathbf{at}\,t$ as 't is a variable symbol'.

Let a, b, c, and so on, be variable symbols. A simple inductive definition of substitution on *abstract syntax* without binding is:

$$a[a{:=}t] \equiv t \qquad a{\not\equiv}b \Rightarrow b[a{:=}t] \equiv b \qquad f(a_1,\ldots,a_n)[a{:=}t] \equiv f(a_1[a{:=}t],\ldots,a_n[a{:=}t])$$

Here f is a term-former and we write $s \equiv t$ for 's and t are identical terms', and $s \not\equiv t$ for 's and t are syntactically different terms'. For the first clause it is important that a is a variable symbol, and for the second clause it is important that *both* a and b are variable symbols. The third clause, as stated, ignores capture avoidance (if f is a binding term former) we assume that the reader is familiar with appropriate additional clauses/conventions to accommodate this.

We find that $\mathbf{at}$ gives just enough power to axiomatise the λ-calculus. We investigate the λ-calculus example in detail but by the end of the paper it should seem plausible that a-logic has broader applications.

So what is new? Plenty of other systems can express the syntax and reductions of the λ-terms. To our knowledge a-logic is unique in that it has this expressivity

[1] `http://www.gabbay.org.uk`

[2] `michael.gabbay(at)kcl.ac.uk`

while still being a first-order (that is, β-reduction is not primitive) logic such that: variables of the object theory are variables of the logic *and* λ and application are just binary term-formers taking pairs of terms and giving terms.

We would like to say that again, but we need some (informal) terminology. Say a map is a ***shallow embedding*** from one system (logic or calculus, say) to another, when variables map to variables (with a ***deep embedding*** variables map to constants) [2,29]; a map is ***compositional*** when the interpretation of an expression can be obtained directly by composing the interpretation of its parts in a syntax-directed manner. Say a system is ***first-order*** when its language of terms does not already contain β-reduction.

Our application of a-logic is unique in that it is to our knowledge the only compositional shallow embedding of the λ-calculus into a first-order logic.

The reader need not care about logic *or* about the λ-calculus. However, these are paradigmatic and they are the basis of logic and functional programming respectively. It has not previously been possible to directly (in the sense just given) map one to the other. We discuss potential applications and related work in the Conclusions.

Outline of the paper. a-logic is a straightforward classical first-order logic with a Boolean semantics. We define syntax in Subsection 2.2 and sequent rules in Subsection 2.3. We axiomatise substitution in Subsection 3.1, then the untyped λ-calculus in Subsection 3.2, and finally we briefly consider other examples. Section 4 proves cut-elimination. Section 5 constructs a class of models and proves soundness and completeness. In the Conclusions we recap on related work, summarise, and mention possible future work. In Section 6 we build a concrete semantics for the axioms of the λ-calculus, incidentally proving consistency of the main axiomatisations in this paper.

The 'a' in 'a-logic' refers to the fact that the logic is for making statements about its variable symbols a, b, c. Any resemblance to an indefinite article is coincidental.

2 a-logic

2.1 The aim of a-logic

Ultimately a-logic will be sensitive to its own syntax in addition to possessing a familiar denotational model theory. Truth in a model will therefore depend not only on the denotations of predicates, but also on their syntactic structure. For example the predicate 'x is a variable' will be true in a model not simply because of the denotation of 'x' but also because 'x' *is* a variable. Not every predicate is sensitive to term complexity in this way, some ignore it completely, others only partially. For example the predicate 't_1 is a simplification of t_2', if true, ought to remain true if t_1 is replaced only by *simpler* terms with the same denotation, and if t_2 is replaced only by *more complex* terms with the same denotation. To capture these intuitions formally, we assign to predicates an *arity* which indicates how it reacts to variation in the syntactic properties of its terms.

The syntactic property we are most interested in a-logic recognising is that of 'being a variable'. The model theory of a-logic shall contain special elements, call them *atoms*, which represent variables in the denotation. But the variables of our syntax should still range over the whole domain of the model. So we use the predicate $\mathbf{at}\,t$, which means that t is a variable symbol and is interpreted as and atom in the model. We can now extend our use of the term 'atom' to apply to variable symbols as well: a variable symbol a is an ***atom*** when we know that $\mathbf{at}\,a$.

2.2 Terms, directions, predicates

Assume a countably infinite set $a, b, c \in \mathbb{A}$ of variable symbols and some untyped language of abstract syntax trees. Write s, t, for arbitrary terms, built inductively from variable symbols and some countable set of ***term-formers*** of fixed arity applied to terms. Write $t[a{:=}t']$ for the usual substitution on terms. We may write $a \in t$ if a occurs in t and $a \notin t$ otherwise.[3]

We assume three ***directions*** $\uparrow$ ***up***, $\downarrow$ ***down***, and $\circlearrowleft$ ***up and down***. Where necessary, d will vary over directions. Assume some countable set of ***predicate constant symbols*** $p, q, r \ldots \in \mathbb{P}$, each with an arity $\delta = (d_1, \ldots, d_n)$ which is a sentence in $\{\uparrow, \downarrow, \circlearrowleft\}^*$ (thus, a possibly empty list of directions). We write $p : \delta$ for 'p ***has arity*** δ'. When δ has length n and some list of terms also has length n, we say that list has ***length appropriate to*** δ. We assume distinguished predicate constants:

(i) We call $\rightsquigarrow : (\uparrow, \downarrow)$ ***aequality*** (pronounced as 'ayquality').

(ii) We call $\mathbf{at} : (\downarrow)$ ***atom*** ('is an atom').

(iii) We call $\bot : ()$ ***false*** or ***contradiction***.

We may write ts as shorthand for a list of terms $t_1, \ldots, t_n$. Write $p(ts)$ for a predicate constant applied to a list of terms of length appropriate to its arity.

Consider $p : (d_1, \ldots, d_n)$ and a list of terms $(t_1, \ldots, t_n)$. When for all i such that $a \in t_i$, it is the case that $d_i \in \{\uparrow, \circlearrowleft\}$, we say a ***occurs up in*** $p(ts)$ and we write $a{\uparrow}p(ts)$. Similarly when for all i such that $a \in t_i$ we have $d_i \in \{\downarrow, \circlearrowleft\}$, write $a{\downarrow}p(ts)$. When for all i such that $a \in t_i$ we have $d_i = \circlearrowleft$, write $a{\circlearrowleft}p(ts)$.

The intuition of $\rightsquigarrow$ is a formalisation of a reduction relation (like β-reduction, for example). The intuition of $a{\downarrow}p(ts)$ is a 'subject reduction' property, that if $s \rightsquigarrow t$ then $p(ts)[a{:=}s]$ implies $p(ts)[a{:=}t]$. If $a{\uparrow}p(ts)$ and $s \rightsquigarrow t$ then $p(ts)[a{:=}t]$ implies $p(ts)[a{:=}s]$. Finally, if $a{\circlearrowleft}p(ts)$ and $s \rightsquigarrow t$ then $p(ts)[a{:=}s]$ if and only if $p(ts)[a{:=}t]$. More on this in Subsection 2.4.

Intuitively, $\mathbf{at}\,t$ is true when t is a variable symbol. The arity $(\uparrow, \downarrow)$ of $\rightsquigarrow$ is consistent with its intuition. $\mathbf{at}$ must have arity $(\downarrow)$ both to avoid inconsistencies (we see this later) and to remain true to the intuition.[4]

There is no interaction (yet) between directions and the terms, so $p(ts)$ is always well-formed so long as ts has the right length. ***Predicates*** are generated by the grammar

$$P ::= p(ts) \mid \bot \mid P \supset P \mid \forall a.P.$$

[3] If the language of terms admits binding then $a \in t$ when a occurs free in t.

[4] For example, if $\rightsquigarrow$ models a literal reduction relation then we can interpret $t \rightsquigarrow a$ as expressing that some complex term t reduces to an atom a. In that case we had better not allow the substitution of t for a in a (true) sentence such as $\mathbf{at}\,a$, for that would yield the (potentially false) sentence $\mathbf{at}\,t$.

$$\frac{}{\Gamma, P \vdash P, \Delta}(\mathbf{Ax}) \quad \frac{\Gamma, P \vdash Q, \Delta}{\Gamma \vdash P \supset Q, \Delta}(\supset \mathbf{R}) \quad \frac{\Gamma \vdash P, \Delta \quad \Gamma, Q \vdash \Delta}{\Gamma, P \supset Q \vdash \Delta}(\supset \mathbf{L})$$

$$\frac{\Gamma \vdash P, \Delta \quad [a \notin \Gamma, \Delta]}{\Gamma \vdash \forall a.P, \Delta}(\forall \mathbf{R}) \quad \frac{\Gamma, P[a{:=}t] \vdash \Delta}{\Gamma, \forall a.P \vdash \Delta}(\forall \mathbf{L})$$

$$\frac{\Gamma \vdash P, \Delta \quad \Gamma, P \vdash \Delta}{\Gamma \vdash \Delta}(\mathbf{Cut}) \quad \frac{}{\Gamma, \bot \vdash \Delta}(\bot \mathbf{L})$$

$$\frac{\Gamma, \mathbf{at}\, a \vdash \Delta \quad [a \notin \Gamma, \Delta]}{\Gamma \vdash \Delta}(\mathbf{at}\, \mathbf{L}) \quad \frac{\Gamma \vdash \mathbf{at}\, t, \Delta \quad [t \text{ not a variable}]}{\Gamma \vdash \Delta}(\mathbf{at}\, \mathbf{R})$$

$$\frac{}{\Gamma \vdash t \rightsquigarrow t, \Delta}(\rightsquigarrow \mathbf{R})$$

$$\frac{\Gamma \vdash p(ts)[a{:=}t'], \Delta \quad [a{\downarrow}p(ts)]}{\Gamma, t' \rightsquigarrow t \vdash p(ts)[a{:=}t], \Delta}(\rightsquigarrow \mathbf{L}{\downarrow}) \quad \frac{\Gamma \vdash p(ts)[a{:=}t], \Delta \quad [a{\uparrow}p(ts)]}{\Gamma, t' \rightsquigarrow t \vdash p(ts)[a{:=}t'], \Delta}(\rightsquigarrow \mathbf{L}{\uparrow})$$

Fig. 1. Derivation rules of a-logic with aequality

$\forall a.P$ is binding, nothing else is. As is usual we write $\forall a, b, \ldots .P$ as shorthand for $\forall a.\forall b. \ldots P$. We equate predicates up to α-equivalence; we will *not* talk about 'occurring free' or 'occurring bound'. Consistent with our convention for terms we write $P \equiv Q$ for 'P and Q are identical predicates up to α-equivalence'.

Write VP for ***the variables occurring in*** P, defined by: $Vp(t_1, \ldots, t_n) = \bigcup Vt_i$, $V(P \supset Q) = VP \cup VQ$, and $V(\forall b.P) = VP \backslash \{b\}$. We may write $a \in P$ for $a \in VP$, we read it 'a ***occurs in*** P'. Then $a \notin P$ means 'a does not occur in P'.

Write $P[a{:=}s]$ for P with every instance of the variable a replaced by the term s in the usual, capture-avoiding, manner.

2.3 Contexts and judgements

A ***(logical) context*** Γ is a finite set of predicates. We write $a \in \Gamma$ when $a \in P$ for some $P \in \Gamma$, and $a \notin \Gamma$ otherwise.

A ***judgement*** is a pair of contexts which we write $\Gamma \vdash \Delta$. When a context is on the right-hand side of a judgement we call it a ***cocontext***. The ***valid*** or ***derivable*** judgements are inductively defined by the rules in Figure 1.

In Figure 1 a and b are distinct variable symbols; bound by $\forall$, free otherwise (note that the a in $(\mathbf{at}\,\mathbf{L})$ must be a variable symbol). Formulae in square brackets are *side-conditions* whose satisfaction can be decided just by examining syntax. $(\mathbf{at}\,\mathbf{L})$ is not mis-typed. It *eliminates* a variable.

Call $\Gamma \vdash \Delta$ a ***theorem*** when a derivation exists concluding in $\Gamma \vdash \Delta$. If $\Gamma = \emptyset$ write $\Gamma \vdash \Delta$ as $\vdash \Delta$. If $\Delta = \emptyset$ write $\Gamma \vdash \Delta$ as $\Gamma \vdash$.

A subset of the logic is classical predicate logic, so we use standard sugar such as writing $\neg P$ for $P \supset \bot$, $P \vee Q$ for $(\neg P) \supset Q$, $\exists a.P$ for $\neg(\forall a.\neg P)$, $P \wedge Q$ for $\neg(P \supset (\neg Q))$, and in general we shall use other well-known shorthands for classical equivalences.

2.4 Comments on atoms

($\mathbf{at\,L}$) states that atoms exist, and ($\mathbf{at\,R}$) states that we cannot indirectly describe them (that is, by means of a complex term).

As can be seen from this derivation, assuming t is not a variable,

$$\dfrac{\dfrac{}{\Gamma, \mathbf{at}\,t \vdash \mathbf{at}\,t, \Delta}(\mathbf{Ax})}{\Gamma, \mathbf{at}\,t \vdash \Delta}(\mathbf{at\,R})$$

($\mathbf{at\,R}$) is like ($\bot\mathbf{L}$); 'if t is a term then it is not a variable'. $\mathbf{at}\,\langle a, a\rangle \vdash$ is a theorem and can be derived by ($\mathbf{Ax}$) followed by ($\mathbf{at\,R}$). (We assume a pair term-former $\langle\text{-},\text{-}\rangle$.) There is no rule $\Gamma \vdash \mathbf{at}\,a, \Delta$. It would break Lemma 4.1 (the substitution lemma) which underlies the essential case of $\forall$ and which formalises the intuition 'variable symbols represent unknown terms'.

We cannot conclude $\mathbf{at}\,a$ just because a *is* a variable symbol! $\mathbf{at}\,a$ in Γ represents a promise that a will never be instantiated to a (non-variable) term. This explains ($\mathbf{at\,R}$) since there the promise has manifestly been broken, and also ($\mathbf{at\,L}$) since there a is fresh from Γ and thus immune to any nasty things Γ may do to it (such as contain $a = 0$). We may call a variable symbol a of which we know $\mathbf{at}\,a$ an ***atom***.

Define $a{\uparrow}P$, $a{\downarrow}P$, and $a{\circlearrowleft}P$ by:

$$\frac{a{\circlearrowleft}P}{a{\uparrow}P} \quad \frac{a{\circlearrowleft}P}{a{\downarrow}P} \quad \frac{a{\uparrow}P \quad a{\downarrow}Q}{a{\downarrow}(P{\supset}Q)} \quad \frac{a{\downarrow}P \quad a{\uparrow}Q}{a{\uparrow}(P{\supset}Q)} \quad \frac{a{\circlearrowleft}P \quad a{\circlearrowleft}Q}{a{\circlearrowleft}(P{\supset}Q)} \quad \frac{a{\uparrow}P}{a{\uparrow}\forall a.P} \quad \frac{a{\downarrow}P}{a{\downarrow}\forall a.P} \quad \frac{a{\circlearrowleft}P}{a{\circlearrowleft}\forall a.P}$$

Theorem 2.1 *$\rightsquigarrow$ is transitive and reflexive; $s \rightsquigarrow t,\ t \rightsquigarrow u \vdash s \rightsquigarrow u$ and $\vdash s \rightsquigarrow s$ are derivable.*

Proof. The latter is simply an instance of ($\rightsquigarrow\mathbf{R}$), the former follows by this derivation (noting that $a{\downarrow}(s \rightsquigarrow a)$):

$$\dfrac{\dfrac{}{s \rightsquigarrow t \vdash s \rightsquigarrow a[a{:=}t]}(\mathbf{Ax})}{s \rightsquigarrow t, t \rightsquigarrow u \vdash s \rightsquigarrow a[a{:=}u]}(\rightsquigarrow\mathbf{L}{\downarrow})$$

□

$\rightsquigarrow$ behaves like the transitive reflexive closure of a reduction/rewrite relation. By Theorem 2.2 below if $a{\downarrow}P$ then P satisfies 'subject reduction'; if $P[a{:=}s]$ and $s \rightsquigarrow t$ then $P[a{:=}t]$. But then in $P \supset Q$ (where Q does not mention a) the direction of the 'subject reduction' is reversed, thus we need $a{\uparrow}(P \supset Q)$. Similar issues arise with $\rightsquigarrow$ itself. For example if $(a \rightsquigarrow t)[a{:=}s]$ and $s' \rightsquigarrow s$ then $(a \rightsquigarrow t)[a{:=}s']$. This is why $\rightsquigarrow$ has arity $(\uparrow, \downarrow)$.

Theorem 2.2 *The following two rules are derivable:*

$$\frac{\Gamma \vdash P[a{:=}t'], \Delta \quad (a{\downarrow}P)}{\Gamma, t' \rightsquigarrow t \vdash P[a{:=}t], \Delta}(\rightsquigarrow\mathbf{L}{\downarrow}) \qquad \frac{\Gamma \vdash \Delta, P[a{:=}t] \quad (a{\uparrow}P)}{\Gamma, t' \rightsquigarrow t \vdash \Delta, P[a{:=}t']}(\rightsquigarrow\mathbf{L}{\uparrow})$$

Proof. It is not hard to check, by induction on P, that directions propagate point-wise but get flipped if they are in negative position (the left-hand side of an implication) and that $a{\circlearrowleft}P$ precisely when $a{\uparrow}P$ and $a{\downarrow}P$.

We now work by induction on P. The atomic case follows immediately from $(\rightsquigarrow\mathbf{L}\uparrow)$, $(\rightsquigarrow\mathbf{L}\downarrow)$ and the definition of $a{\downarrow}P$ in the case where P is atomic (see page 3). The induction cases are routine. For example suppose P is $(Q \supset R)$. We must show that

(i) if $a{\downarrow}(Q \supset R)$ and $\Gamma \vdash (Q \supset R)[a{:=}t']\Delta$ then $\Gamma, t' \rightsquigarrow t \vdash (Q \supset R)[a{:=}t], \Delta$

(ii) if $a{\uparrow}(Q \supset R)$ and $\Gamma \vdash (Q \supset R)[a{:=}t], \Delta$ then $\Gamma, t' \rightsquigarrow t \vdash (Q \supset R)[a{:=}t'], \Delta$

The antecedent of case (i) can be only if we have $a{\uparrow}Q$ and $a{\downarrow}R$, and the antecedent for case (ii) can occur only if $a{\downarrow}Q$ and $a{\uparrow}R$.

For case (i) we sketch a derivation as follows:

$$\dfrac{\dfrac{\dfrac{}{\Gamma, Q[a{:=}t] \vdash Q[a{:=}t], \Delta}(\mathbf{Ax})}{\Gamma, t' \rightsquigarrow t, Q[a{:=}t] \vdash Q[a{:=}t'], \Delta}\,\mathit{Ind.Hyp.}, (\rightsquigarrow\mathbf{L}\uparrow) \qquad \dfrac{\dfrac{}{\Gamma, R[a{:=}t'] \vdash R[a{:=}t'], \Delta}(\mathbf{Ax})}{\Gamma, t' \rightsquigarrow t, R[a{:=}t'] \vdash R[a{:=}t], \Delta}\,\mathit{Ind.Hyp.}(\rightsquigarrow\mathbf{L}\downarrow)}{\dfrac{\Gamma, t' \rightsquigarrow t, (Q \supset R)[a{:=}t'], Q[a{:=}t] \vdash R[a{:=}t], \Delta}{\Gamma, t' \rightsquigarrow t, (Q \supset R)[a{:=}t'] \vdash (Q \supset R)[a{:=}t], \Delta}(\supset\mathbf{R})}(\supset\mathbf{L})$$

We can now derive $\Gamma, t' \rightsquigarrow t \vdash P[a{:=}t], \Delta$ from the assumption that $\Gamma \vdash (Q \supset R)[a{:=}t'], \Delta$ and $(\mathbf{Cut})$.

The relevant derivation for case (ii) is easily obtained from the derivation above by swapping t with t' and relabelling the applications of $(\rightsquigarrow\mathbf{L}\uparrow)$ and $(\rightsquigarrow\mathbf{L}\downarrow)$ accordingly. $\square$

The standard derivation rules for equality are: [5]

$$\dfrac{}{\Gamma \vdash t = t, \Delta}(=\mathbf{R}) \qquad \dfrac{\Gamma \vdash P[a{:=}t'], \Delta}{\Gamma, t' = t \vdash P[a{:=}t], \Delta}(=\mathbf{L})$$

If we make sure that $a{\circlearrowleft}P$, then $\rightsquigarrow$ behaves just like an equality — at least for that a. Also, a-logic predicates can express a notion of equality:

Definition 2.3 Write $t = u$ in a-logic as shorthand for $t \rightsquigarrow u \wedge u \rightsquigarrow t$.

Theorem 2.4 $(=\mathbf{R})$ *and* $(=\mathbf{L})$ *are derivable in* a*-logic.*

Proof. If neither $a{\uparrow}p(ts)$ nor $a{\downarrow}p(ts)$ we view $p(ts)$ in terms of two variables fresh atoms a' and a'', one occurring exclusively up in $p(ts)$, the other occurring exclusively down. That is, we view $p(ts)$ as $p'(ts)[a'{:=}a, a''{:=}a]$ where $a'{\downarrow}p'(ts)$ and $a''{\uparrow}p'(ts)$, and where $a \notin p'(ts)$. It is then easy to verify that with one use each of $(\rightsquigarrow \mathbf{L}\uparrow)$ and $(\rightsquigarrow \mathbf{L}\downarrow)$ we can obtain the effect of $(=\mathbf{L})$. $(=\mathbf{R})$ is easy. $\square$

[5] A familiar alternative to $(=\mathbf{L})$ is this:

$$\dfrac{\Gamma, P[a{:=}t'] \vdash \Delta}{\Gamma, t' = t, P[a{:=}t] \vdash \Delta}(=\mathbf{L}')$$

$(\mapsto\#)$	$\forall a,v.\big(a\#v \supset \forall b.(\mathbf{at}\,b \supset v\langle a{\mapsto}b\rangle \rightsquigarrow v)\big)$
$(\mapsto\mathbf{aa})$	$\forall a,v.(\mathbf{at}\,a \supset v\langle a{\mapsto}a\rangle \rightsquigarrow v)$
$(\mapsto\mathbf{at})$	$\forall a,t.(\mathbf{at}\,a \supset a\langle a{\mapsto}t\rangle \rightsquigarrow t)$
$(\mapsto\mathbf{ren})$	$\forall a,b,u,v.\big((\mathbf{at}\,a \wedge b\#v) \supset v\langle a{\mapsto}b\rangle\langle b{\mapsto}u\rangle \rightsquigarrow v\langle a{\mapsto}u\rangle\big)$
$(\mapsto\mathbf{comm})$	$\forall a,b,v,t,u.\big((\mathbf{at}\,b \wedge a\#b \wedge a\#u) \supset v\langle a{\mapsto}t\rangle\langle b{\mapsto}u\rangle \rightsquigarrow v\langle b{\mapsto}u\rangle\langle a{\mapsto}t\langle b{\mapsto}u\rangle\rangle\big)$

Fig. 2. *a*-logic axioms for substitution

$(\#\mathbf{aa})$	$\forall a.(\neg a\#a)$
$(\#\mathbf{at})$	$\forall a,t.(a\#t \supset \mathbf{at}\,a)$
$(\#\rightsquigarrow)$	$\forall a,b.\big((\mathbf{at}\,a \wedge \mathbf{at}\,b) \supset (a \rightsquigarrow b \vee a\#b)\big)$

Fig. 3. *a*-logic axioms for freshness

$(\lambda\mathbf{ref})$	$\forall a,b,v.(\mathbf{at}\,a \wedge b\#v \supset \lambda a.v = \lambda b.v\langle a{\mapsto}b\rangle)$
$(\lambda\beta)$	$\forall a,v,t.(\mathbf{at}\,a \supset (\lambda a.v)t \rightsquigarrow v\langle a{\mapsto}t\rangle)$

Fig. 4. *a*-logic axioms for α-equality and for β-reduction

So *a*-logic combines elements of computation, given by $\rightsquigarrow$, of logic, given by the evident logical apparatus, and of something quite unusual, given by **at** .

3 Expressivity

We are now ready to write down some axioms:

3.1 Substitution

Definition 3.1 Suppose a ternary term-former $\langle\rangle$. We usually write $\langle\rangle(s,u,t)$ as $s\langle u{\mapsto}t\rangle$. Assume a binary predicate $\# : (\downarrow,\downarrow)$ ***freshness***. Then an *a*-logic theory of substitution is given by the axioms in Figure 2.

We intend # to express a notion of 'not free in'. This notation and intuition is inherited from nominal techniques [18], though the properties of # are here expressed as axioms in *a*-logic, whereas they are built in as primitive to the nominal framework. We intend $\langle\rangle$ to express a notion of 'capture-avoiding substitution for atoms'; a related nominal treatment is also available [17].

Remark 3.2 Note, in Figure 3 that $a\#t$ implies $\mathbf{at}\,a$. We use this for example when we omit an assumption $\mathbf{at}\, a$ in a number of axioms, e.g. ($\mapsto$**comm**), but we still give a the name 'a'.

A remark on syntax is particularly important:

Remark 3.3 $v\langle s\mapsto t\rangle$ is valid syntax whether or not s *is* a variable symbol. $\langle\rangle$ is just a ternary term-former. $\langle\rangle$ is distinct from substitution on terms $v[a{:=}t]$. For example $c[a{:=}b] \equiv c$ but $c\langle a\mapsto b\rangle \not\equiv c$ and furthermore $\vdash c\langle a\mapsto b\rangle \rightsquigarrow c$ is not derivable.

$v[s{:=}t]$ is not well-defined unless $s \equiv a$ for some variable symbol a. $v\langle s\mapsto t\rangle$ is a valid term always, but only if we have assumed $\mathbf{at}\, s$ can we prove anything useful about $v\langle s\mapsto t\rangle$ from the axioms in Figure 2. However, assuming $\mathbf{at}\, s$ of s that is not a variable symbol leads to immediate contradiction by ($\mathbf{at}\,\mathbf{R}$). Therefore in practise we only have occasion to usefully write down terms such as $t\langle a\mapsto u\rangle$.

We can now give an answer to the question "why $\rightsquigarrow$ not $=$?". Assume the axioms of Figure 2, except replacing $\rightsquigarrow$ with $=$ from Definition 2.3, which we proved in Theorem 2.4 has the properties we expect of equality. For example axiom ($\mapsto$**at**) becomes $\forall a, t.(\mathbf{at}\, a \supset a\langle a\mapsto t\rangle = t)$. Call this set of axioms Σ'. Then:

Theorem 3.4 *$\Sigma' \vdash$ is derivable. In words: "Σ' is inconsistent in a-logic".*

Proof. Note that all the axioms of Σ' are closed and that $\Sigma', \mathbf{at}\, a \vdash a\langle a\mapsto a\rangle = a$ is derivable using the replaced version of ($\mapsto$**aa**). Using (**Cut**) and ($=$**L**) also $\Sigma', \mathbf{at}\, a \vdash \mathbf{at}\,(a\langle a\mapsto a\rangle)$ is derivable and using ($\mathbf{at}\,\mathbf{R}$) we obtain $\Sigma', \mathbf{at}\, a \vdash$. Finally by ($\mathbf{at}\,\mathbf{L}$) we derive $\Sigma' \vdash$. □

So **at** in a-logic interacts with equality, and seems to require it to be directional.

In Section 6 we build a nontrivial model for all the axioms of Figures 2, 3 and 4. By a soundness result which we prove later in Theorem 6.7, this demonstrates that any subset or weakening of these axioms is neither inconsistent nor trivial.

Remark 3.5 The axioms of Figure 3 interact to yield some further theorems of substitution and freshness. For example, using ($\rightsquigarrow$ **R**) it follows that

$$b\#a, a \rightsquigarrow b \vdash b \rightsquigarrow b$$

and then using (#**aa**) we can derive that $b\#a \vdash \neg(a \rightsquigarrow b)$. Then using ($\# \rightsquigarrow$) we can derive

$$\mathbf{at}\, a, \mathbf{at}\, b, b\#a \vdash a\#b$$

so freshness is a symmetric relation when between atoms.

Remark 3.6 A corollary of Theorem 6.7 is that it is consistent to strengthen ($\mapsto$**comm**) to

$$\forall a, b, v, t, u.\big((\mathbf{at}\, b \wedge a\#b \wedge a\#u) \supset v\langle a\mapsto t\rangle\langle b\mapsto u\rangle = v\langle b\mapsto u\rangle\langle a\mapsto t\langle b\mapsto u\rangle\rangle\big)$$

This may be appropriate in some examples, for example such in the λ-calculus of Subsection 3.2.

Axiomatisations of substitution exist though we are aware of no authoritative source. Two examples are Crabbé [10, p.2] and Salibra [23, p.6]. Crabbé assumes a function symbol which we can identify with #; Salibra gives a more algebraic axiomatisation but the domain *must* be a model of the λ-calculus so he expresses $a\#s$ (a is not free in s) by writing s as $(\lambda x.s)a$. Salibra's and Crabbé's axiomatisations are almost equivalent; the authors do not cite one another but the interested reader can note that $(\lambda x.s)t$ in Salibra corresponds to $s[a{:=}t]$ in Crabbé [10, Proposition 3.1]. We must remove Salibra's axiom β_6, which is β-reduction. The first author with Mathijssen has recently investigated an axiomatisation of substitution using nominal algebra [16,17]. Nominal algebra makes more structure, such as freshness, primitive to the logic; making a formal connection with a-logic is future work.

3.2 λ-calculus

Definition 3.7 Assume the term-former substitution $\langle\rangle$ and predicate constant freshness $\# : (\downarrow, \downarrow)$ from Subsection 3.1. Also assume binary term-formers ***application*** $\cdot$ and ***lambda*** λ. Write $\cdot(t', t)$ as $t' \cdot t$. Write $\lambda(s, t)$ as $\lambda s.t$. An a-logic theory of the λ-calculus is given by the axioms in Figures 2, 3 and 4.

Note (as observed of $\langle\rangle$ in Remark 3.3) that application and λ are merely term formers; $\lambda s.t$ is a valid term for any s, but the axioms permit nothing of interest to be derived about it, unless $\mathbf{at}\, s$ is also derivable.

We can add an axiom for extensionality to those of Figure 4:

$$\forall b.\forall s.\mathbf{at}\, b \supset (\lambda b.s)b \rightsquigarrow s.$$

There are other interesting properties we might consider investigating in future work. For example:

- A unary predicate symbol $\Box$ with axiom $\Box s \Leftrightarrow \forall a.\mathbf{at}\, a \supset a\#s$. The intuition is '$s$ is closed'.
- A predicate 'the free atoms of s are precisely b' axiomatised by $(\neg b\#s) \wedge \forall a.\mathbf{at}\, a \wedge a \neq b \supset a\#s$.
- A predicate 'the free atoms of s are at most b' axiomatised by $\forall a.\mathbf{at}\, a \wedge a \neq b \supset a\#s$.

These are in the spirit of suggestions already made for nominal rewriting in [11, Subsection 9.2] though the technical details are very different.

The first author was instrumental in developing so-called Nominal techniques, including Nominal Logic [22,14]. These have similar applications to a-logic and they have a freshness predicate #. However they are inconsistent with the axiom of choice; for example Nominal Logic is inconsistent with a unary term-former f such that $(f s)\#s$ always. It seems that we *can* axiomatise such an f in a-logic, as $\forall s.\exists b.f s \rightsquigarrow b \wedge b\#s$. The use of $\rightsquigarrow$ helps to avoid contradictions and this might be useful in some applications of nominal techniques.

4 Cut-elimination

In this section we prove that a-logic (without additional axioms and term formers for substitution) satisfies Cut-Elimination. If $\Gamma = \{G_1, \ldots, G_n\}$ write $\Gamma[a{:=}t]$ for

$\{G_1[a{:=}t], \dots, G_n[a{:=}t]\}$. Similarly for $\Delta[a{:=}t]$.

Call the total number of instances of derivation rules in a derivation, its ***depth***.

Lemma 4.1 (Substitution Lemma) *If* $\Gamma \vdash \Delta$ *has a derivation* Π*, then* $\Gamma[a{:=}t] \vdash \Delta[a{:=}t]$ *has a derivation* $\Pi[a{:=}t]$ *that is no deeper than* Π*.*

Proof. By induction on the depth of derivations.

We consider some cases:

- The case of $(\mathbf{at}\,\mathbf{R})$. If v is not a variable in $\Gamma \vdash \mathbf{at}\,v, \Delta$ then $v[a{:=}t]$ is not a variable in $\Gamma[a{:=}t] \vdash \mathbf{at}\,v[a{:=}t], \Delta[a{:=}t]$ so substitution instances of instances of $(\mathbf{at}\,\mathbf{R})$ are still instances of $(\mathbf{at}\,\mathbf{R})$.
- The case of $(\mathbf{at}\,\mathbf{L})$. Suppose the sequent $\Gamma \vdash \Delta$ is derived from the sequent $\Gamma, \mathbf{at}\,b \vdash \Delta$ by $(\mathbf{at}\,\mathbf{L})$. Then b is a variable symbol such that $b \not\in \Gamma, \Delta$. We need to show that $\Gamma[a{:=}t] \vdash \Delta[a{:=}t]$ is derivable.

  By the inductive hypothesis, we have that $\Gamma[b{:=}b'], \mathbf{at}\,b[b{:=}b'] \vdash \Delta[b{:=}b']$ is derivable, for any other variable b'. Choosing b' so that $b' \not\in \Gamma, \Delta, t$ it follows that the sequent $\Gamma, \mathbf{at}\,b' \vdash \Delta$ is derivable, and the derivation is no deeper.

  By the inductive hypothesis again

  $$\Gamma[a{:=}t], (\mathbf{at}\,b')[a{:=}t] \vdash \Delta[a{:=}t]$$

  is derivable. Now, b' was chosen so that $(\mathbf{at}\,b')[a{:=}t] \equiv \mathbf{at}\,b'$. So we can use $(\mathbf{at}\,\mathbf{L})$ to derive $\Gamma[a{:=}t] \vdash \Delta[a{:=}t]$ as required.
- The rules for $\rightsquigarrow$ are non-standard but it is not hard to verify that substitution in a valid instance of a rule is still a valid instance, just as we would do for the usual equality rules.

  For example in the case of $(\rightsquigarrow\mathbf{L}{\downarrow})$ if we have

  $$\dfrac{\begin{array}{c}\Pi \\ \vdots \\ \Gamma \vdash p(ts)[b{:=}t'], \Delta \quad [b{\downarrow}p(ts)]\end{array}}{\Gamma, t' \rightsquigarrow t \vdash p(ts)[b{:=}t], \Delta}\,(\rightsquigarrow\mathbf{L}{\downarrow})$$

  ...then by induction hypothesis we have:

  $$\dfrac{\begin{array}{c}\Pi[a{:=}s] \\ \vdots \\ \Gamma[a{:=}s] \vdash p(ts)[b{:=}t'[a{:=}s]], \Delta[a{:=}s] \quad [b{\downarrow}p(ts)]\end{array}}{\Gamma[a{:=}s], (t' \rightsquigarrow t)[a{:=}s] \vdash p(ts)[b{:=}t[a{:=}s]], \Delta[a{:=}s]}\,(\rightsquigarrow\mathbf{L}{\downarrow}).$$

  Other cases are routine.

□

Lemma 4.2 (Weakening) *If* $\Gamma \vdash \Delta$ *then* $\Gamma, \Gamma' \vdash \Delta, \Delta'$*.*

Proof. By induction on the derivation of $\Gamma \vdash \Delta$. We may need to rename variables generated by $(\mathbf{at}\,\mathbf{L})$, or $(\forall\mathbf{R})$ to avoid clashes with variables in Γ' or Δ'. For example if the derivation ends with $(\mathbf{Ax})$ then the weakened conclusion is itself derivable by

(**Ax**), if the derivation ends with ($\forall$**R**):

$$\dfrac{\begin{array}{c}\Pi\\ \vdots\\ \Gamma \vdash P, \Delta\end{array}}{\Gamma \vdash \forall a.P, \Delta}\,(\forall\mathbf{R})^{(*)}$$

…then by the induction hypothesis we have…

$$\begin{array}{c}\Pi[a{:=}c]\\ \vdots\\ \Gamma, \Gamma' \vdash P[a{:=}c], \Delta, \Delta'\end{array}$$

where $\Pi[a{:=}c]$ is obtained using Lemma 4.1 by replacing a in Π by some atom c that does not occur in Γ, Γ', Δ or Δ'. We know that $\Pi[a{:=}c]$ really is a derivation because a is not free in Γ or Δ. From this we can derive $\Gamma, \Gamma' \vdash \forall a.P, \Delta, \Delta'$ by an application of ($\forall$**R**) (recall from Subsection 2.2 that we equate predicates up to α-equivalence). The remaining cases follow by familiar application of the induction hypothesis. $\square$

For any instance of (**Cut**): $\dfrac{\Gamma \vdash P, \Delta \quad \Gamma, P \vdash \Delta}{\Gamma \vdash \Delta}\,(\mathbf{Cut})$ say its ***degree*** is the total number of instances of the symbols $\bot, \supset, \forall$ in P; and say its ***rank*** is the depth of the derivation of its conclusion.

Theorem 4.3 (**Cut**) *is an admissible rule in the system without it.*

Proof. By induction, lexicographically, on (d, r) where d is the degree and r is the rank of the earliest cut, counting from the leaves of the derivation down to the conclusion. The proof is standard for first-order logic with equality and uses weakening, and the substitution lemma for the essential cases. (**at L**) and (**at R**) have no essential case. Cuts are commuted in the usual way with the rules, including (**at L**) and (**at R**), to lower their rank forming essential cases where they are reduced to cuts of lower degree.

$$\dfrac{\begin{array}{c}\Pi_1\\ \vdots\\ \Gamma \vdash P, \Delta\end{array} \quad \dfrac{\begin{array}{c}\Pi_2\\ \vdots\\ \mathbf{at}\,a, \Gamma, P \vdash \Delta\end{array}}{\Gamma, P \vdash \Delta}\,(\mathbf{at\,L})}{\Gamma \vdash \Delta}\,(\mathbf{Cut}) \quad \Longrightarrow \quad \dfrac{\dfrac{\begin{array}{c}\Pi_1'\\ \vdots\\ \mathbf{at}\,a, \Gamma \vdash P, \Delta\end{array} \quad \begin{array}{c}\Pi_2\\ \vdots\\ \mathbf{at}\,a, \Gamma, P \vdash \Delta\end{array}}{\mathbf{at}\,a, \Gamma \vdash \Delta}\,(\mathbf{Cut})}{\Gamma \vdash \Delta}\,(\mathbf{at\,L})$$

Here we use Weakening (proved above) to extend Π_1 to Π_1' by adding $\mathbf{at}\,a$ to the premise of every sequent in Π_1. Since $a \notin \Gamma, \Delta, P$ we can rename all terms in Π_1 so that a does not occur at all in it, and then we can add $\mathbf{at}\,a$ without invalidating any rule applications in Π. The essential (and commutation) cases for the remaining connectives are familiar, and with one exception the essential cases for $\rightsquigarrow$ are also easy.

$$\dfrac{\dfrac{}{\Gamma \vdash t \rightsquigarrow t, \Delta}\,(\rightsquigarrow\mathbf{R}) \quad \dfrac{\begin{array}{c}\Pi\\ \vdots\\ \Gamma \vdash \Delta\end{array}}{\Gamma, t \rightsquigarrow t \vdash \Delta}\,(\rightsquigarrow\mathbf{L}{\downarrow})}{\Gamma \vdash \Delta} \quad \Longrightarrow \quad \begin{array}{c}\Pi\\ \vdots\\ \Gamma \vdash \Delta\end{array}$$

It is worth stating explicitly the problematic commutation case for aequality. This is the case where an aequality rule applies to the cut formula itself. Suppose $a{\downarrow}p(ts)$

and we have

$$\dfrac{\dfrac{\begin{matrix}\Pi' \\ \vdots \\ \Gamma \vdash p(ts)[a{:=}t'], \Delta\end{matrix}}{\Gamma, t' \rightsquigarrow t \vdash p(ts)[a{:=}t], \Delta}\,(\rightsquigarrow\mathbf{L}{\downarrow}) \qquad \begin{matrix}\Pi \\ \vdots \\ \Gamma, p(ts)[a{:=}t] \vdash \Delta\end{matrix}}{\Gamma, t' \rightsquigarrow t \vdash \Delta}\,(\mathbf{Cut})$$

then (by the other commutation cases) we can assume that the right premise of such a (**Cut**) where the cut formula is atomic, is actually the conclusion of (**Ax**), ($\bot$**L**), ($\rightsquigarrow$ **R**) or (**at R**) (that is, we can assume Π is empty).

If the right premise of the (**Cut**) is derived by ($\bot$**L**) or ($\rightsquigarrow$ **R**), or if the predicate relevant to its derivation is in Γ or Δ (and is not $p(ts)[a{:=}t]$), then the conclusion of the (**Cut**) can be derived directly by the rule that derived $\Gamma, p(ts)[a{:=}t] \vdash \Delta$.

If the right premise of the (**Cut**) is derived from (**Ax**) so that Δ is $\Delta', p(ts)[a{:=}t]$ then we may permute the derivation to

$$\dfrac{\dfrac{\begin{matrix}\Pi' \\ \vdots \\ \Gamma \vdash p(ts)[a{:=}t'], \Delta\end{matrix} \qquad \dfrac{}{\Gamma, p(ts)[a{:=}t'] \vdash p(ts)[a{:=}t'], \Delta'}\,(\mathbf{Ax})}{\Gamma \vdash p(ts)[a{:=}t'], \Delta'}\,(\mathbf{Cut})}{\Gamma, t' \rightsquigarrow t \vdash \Delta}\,(\rightsquigarrow\mathbf{L}{\downarrow})$$

The twins to these, where $a\uparrow p(ts)$, is similar. □

Lemma 4.4 *If* $\Gamma \vdash \Delta$*, where* Γ *and* Δ *do not contain* **at** *or* $\rightsquigarrow$ *and where the final rule application is* (**at L**) *or* (**at R**) *to the formula* **at** a*, then a (shorter) derivation can be found without this application.*

Proof. By induction on the length of a cut-free derivation that $\Gamma \vdash \Delta$. Suppose (**at L**) or (**at R**) follows to a derivation of length 1. Since Γ and Δ do not contain $\rightsquigarrow$ the derivation must be a single application of (**Ax**) or ($\bot$**L**). In the first case the derivation must be of one of these forms:

$$\dfrac{\dfrac{}{\Gamma', \mathbf{at}\, a, P \vdash P, \Delta'}\,(\mathbf{Ax})}{\Gamma \vdash \Delta}\,(\mathbf{at\,L}) \qquad \text{or} \qquad \dfrac{\dfrac{}{\Gamma', P \vdash P, \mathbf{at}\, t, \Delta'}\,(\mathbf{Ax})}{\Gamma \vdash \Delta}\,(\mathbf{at\,R})$$

where $\Gamma = \Gamma' \cup \{P\}$ and $\Delta = \Delta' \cup \{P\}$. This can be replaced by a single application of (**Ax**). The case for ($\bot$**L**) is similar.

Now suppose the derivation is of length > 1. Then the rule preceding the final application of (**at L**) cannot be any $\rightsquigarrow$ rule (for then Γ would contain $\rightsquigarrow$). Therefore the preceeding rule must be a *FOL* rule. Then we permute the application of (**at L**) or (**at R**) with the FOL rule (so the **at** rule applies to the premises of the FOL rule).

We can then apply the induction hypothesis. For example:

$$\dfrac{\dfrac{\begin{array}{c}\Pi\\ \vdots\\ \Gamma, \mathbf{at}\, a, P \vdash Q, \Delta\end{array}}{\Gamma, \mathbf{at}\, a \vdash P \supset Q, \Delta}(\supset\mathbf{R})}{\Gamma \vdash P \supset Q, \Delta}(\mathbf{at}\,\mathbf{L}) \quad \text{. . . gets permuted to. . .} \quad \dfrac{\dfrac{\begin{array}{c}\Pi\\ \vdots\\ \Gamma, \mathbf{at}\, a, P \vdash Q, \Delta\end{array}}{\Gamma, P \vdash Q, \Delta}(\mathbf{at}\,\mathbf{L})}{\Gamma \vdash P \supset Q, \Delta}(\supset\mathbf{R})$$

and the induction hypothesis applies to the derivation up to the application of $(\mathbf{at}\,\mathbf{L})$. □

Corollary 4.5 *a-logic is a conservative extension of FOL without identity.*

Proof. Suppose $\Gamma \vdash \Delta$, in the language of FOL without identity, is derivable in a-logic. We shall argue it is also derivable in FOL without identity. We know that $\Gamma \vdash \Delta$ can be derived in a cut-free derivation. If this derivation makes no use of the rules for **at** and $\leadsto$ then there is nothing to prove, otherwise our argument proceeds by induction on the length of the derivation. If the final rule is a FOL rule then the result follows by induction hypothesis on its premises. Given the conditions on Γ and Δ, the only other possibility for the final rule is that it is $(\mathbf{at}\,\mathbf{L})$ or $(\mathbf{at}\,\mathbf{R})$, in this case the result follows by Lemma 4.4. □

5 Models of a-logic

5.1 Semantic definitions and soundness

An ***a-domain*** $\mathbb{D}$ is a tuple $(|\mathbb{D}|, I_{\mathbb{D}}, \leq_{\mathbb{D}})$ — we may write $(|\mathbb{D}|, I, \leq)$ if $\mathbb{D}$ is understood — such that:

- $|\mathbb{D}|$ is a set.
  We call it the ***underlying set*** of $\mathbb{D}$. We may write $x \in \mathbb{D}$ for $x \in |\mathbb{D}|$.
- $(|\mathbb{D}|, \leq)$ is a poset on $|\mathbb{D}|$.
  That is, $\leq$ is a transitive reflexive asymmetric binary relation on $|\mathbb{D}|$.
- $I \subseteq |\mathbb{D}|$ is ***down-closed*** with respect to $\leq$. That is, for any $x, x' \in |\mathbb{D}|$, if $x' \in I$ and $x \leq x'$ then $x \in I$

A ***valuation*** ς to $\mathbb{D}$ is a function from the set of variable symbols $\mathbb{A}$ to $|\mathbb{D}|$ such that $\varsigma(a) \in I$ for at least one $a \in \mathbb{A}$. Write $\varsigma\{a \mapsto x\}$ for the valuation such that $\varsigma\{a \mapsto x\}(b) = \varsigma(b)$ for all b other than a, and $\varsigma\{a \mapsto x\}(a) = x$.

We need some notation. Write $|\mathbb{D}|^n$ for the set of n-tuples $(x_1, \ldots, x_n)$ of elements $x_i \in |\mathbb{D}|$ for $1 \leq i \leq n$. $|\mathbb{D}|^0$ contains just the 0-tuple $()$. Call a function f from $|\mathbb{D}|^n$ to $|\mathbb{D}|$ ***monotone*** when

$$\text{if } x_i \leq x_i' \text{ then } f(x_1, \ldots, x_i, \ldots, x_n) \leq f(x_1, \ldots, x_i', \ldots, x_n).$$

A ***model*** $[\![\text{-}]\!]$ extends an a-domain $\mathbb{D}$ with the following information:

- For each term-former f of arity n a monotone function $[\![f]\!]$ from $|\mathbb{D}|^n$ to $|\mathbb{D}|$ such that $[\![f]\!](x_1, \ldots, x_n) \notin I$ for all $x_1, \ldots, x_n$.

Intuitively, the result of applying the interpretation of a term-former to some arguments must never be the interpretation of a variable symbol. This reflects the intuition that $\mathbf{at}\, t$ is false if t is *not* a variable symbol.

- For each predicate constant symbol p of arity $(d_1, \dots, d_n)$ other than $\mathbf{at}$ and $\rightsquigarrow$, a set $[\![p]\!] \subseteq |\mathbb{D}|^n$ such that if $x_i \leq x'_i$ then:
  · If $d_i = \uparrow$ and $(x_1, \dots, x_i, \dots, x_n) \in [\![p]\!]$ then $(x_1, \dots, x'_i, \dots, x_n) \in [\![p]\!]$.
  · If $d_i = \downarrow$ and $(x_1, \dots, x'_i, \dots, x_n) \in [\![p]\!]$ then $(x_1, \dots, x_i, \dots, x_n) \in [\![p]\!]$.
  · If $d_i = \circlearrowleft$ then $(x_1, \dots, x_i, \dots, x_n) \in [\![p]\!]$ if and only if $(x_1, \dots, x'_i, \dots, x_n) \in [\![p]\!]$.
  Comparing these conditions with $(\rightsquigarrow\mathbf{L}\downarrow)$ and $(\rightsquigarrow\mathbf{L}\uparrow)$ from Figure 1 we see that $\leq$ models $\rightsquigarrow$, and we make this intuition precise below.

We extend the model $[\![\text{-}]\!]$ to an ***interpretation*** of terms inductively by

$$[\![a]\!]_\varsigma = \varsigma(a) \qquad [\![f(t_1, \dots, t_n)]\!]_\varsigma = [\![f]\!]([\![t_1]\!]_\varsigma, \dots, [\![t_n]\!]_\varsigma)$$

as is standard.

Given a valuation ς and a $[\![\text{-}]\!]$ we define a notion of ***validity*** in the model (for the valuation) as follows:

- $[\![\bot]\!]_\varsigma$ is invalid.
- $[\![P \supset Q]\!]_\varsigma$ is valid when $[\![P]\!]_\varsigma$ is invalid or $[\![Q]\!]_\varsigma$ is valid.
- $[\![\forall a.P]\!]_\varsigma$ is valid when $[\![P]\!]_{\varsigma\{a \mapsto x\}}$ is valid for all $x \in \mathbb{D}$.
- For $p \notin \{\rightsquigarrow, \mathbf{at}\}$, $[\![p(t_1, \dots, t_n)]\!]_\varsigma$ is valid when $([\![t_1]\!], \dots, [\![t_n]\!])_\varsigma \in [\![p]\!]$.
- $[\![t_1 \rightsquigarrow t_2]\!]_\varsigma$ is valid when $[\![t_2]\!]_\varsigma \leq [\![t_1]\!]_\varsigma$.
- $[\![\mathbf{at}\, t]\!]_\varsigma$ is valid when $[\![t]\!]_\varsigma \in I$.

We write that $[\![\Gamma]\!]_\varsigma$ is valid when $[\![G]\!]_\varsigma$ is valid for all $G \in \Gamma$, and we extend validity to judgements $\Gamma \vdash \Delta$ by: $[\![\Gamma \vdash \Delta]\!]_\varsigma$ is valid when either

- $[\![G]\!]_\varsigma$ is not valid for some $G \in \Gamma$, or
- $[\![D]\!]_\varsigma$ is valid for some $D \in \Delta$.

Finally we write

$$[\![\Gamma \vdash \Delta]\!] \text{ is valid when } [\![\Gamma \vdash \Delta]\!]_\varsigma \text{ for all valuations } \varsigma.$$

Lemmas 5.1 and 5.2 are technical results which are useful later:

Lemma 5.1 $[\![\mathbf{at}\, t]\!]_\varsigma$ *is valid if and only if t is a variable symbol and $\varsigma(t) \in I$.*

Proof. By definition, $[\![\mathbf{at}\, t]\!]_\varsigma$ is valid when $[\![t]\!]_\varsigma \in I$ and by the conditions on $[\![\text{-}]\!]$ and ς, $[\![t]\!]_\varsigma$ cannot be in I if t is a constant or a complex term (as no function symbol is interpreted to have values in I). $\square$

Lemma 5.2 (i) $[\![v[a{:=}t]]\!]_\varsigma = [\![v]\!]_{\varsigma\{a \mapsto [\![t]\!]_\varsigma\}}$.

(ii) $[\![P[a{:=}t]]\!]_\varsigma$ *is valid if and only if* $[\![P]\!]_{\varsigma\{a \mapsto [\![t]\!]_\varsigma\}}$ *is valid.*

Proof. The first part is by an easy induction on syntax. The second part is also by an easy induction, we consider only one case:

$[\![\mathbf{at}\,(v[a{:=}t])]\!]_\varsigma$ is valid if and only if $[\![v[a{:=}t]]\!]_\varsigma \in I$, and by part (i) this is the case if and only if $[\![v]\!]_{\varsigma\{a\mapsto[t]_\varsigma\}} \in I$.

Other cases are similar. □

Lemma 5.3 **((∀L))** (i) *If* $[\![\forall a.P]\!]_\varsigma$ *is valid then* $[\![P[a{:=}t]]\!]_\varsigma$.

(ii) *If* $[\![\Gamma, P[a{:=}t] \vdash \Delta]\!]_\varsigma$ *is valid then* $[\![\Gamma \wedge \forall a.P \vdash \Delta]\!]_\varsigma$.

Proof. The first part is direct from Lemma 5.2 and the definition of $[\![\forall a.P]\!]_\varsigma$. The second part is routine from the definition, using the first part. □

Lemma 5.4 *Suppose that* $a \notin t$, $a \notin G$ *and* $\varsigma(b) = \varsigma'(b)$ *for all other* $b \in \mathbb{A}$. *Then*

(i) $[\![t]\!]_\varsigma = [\![t]\!]_{\varsigma'}$

(ii) $[\![G]\!]_\varsigma$ *is valid iff* $[\![G]\!]_{\varsigma'}$ *is valid.*

As an easy corollary, if $a \notin \Gamma, \Delta$ *and* $\varsigma(b) = \varsigma'(b)$ *for all other* $b \in \mathbb{A}$ *then* $[\![\Gamma \vdash \Delta]\!]_\varsigma$ *if and only if* $[\![\Gamma \vdash \Delta]\!]_{\varsigma'}$.

Proof. We work by induction on t for (i) and then on G for (ii). We consider only two cases:

- The case $G \equiv \mathbf{at}\, t$. Suppose $[\![\mathbf{at}\, t]\!]_\varsigma$ is valid. By Lemma 5.1 t is a variable symbol and $\varsigma(t) \in I$. $t \equiv a$ is impossible because we assumed $a \notin t$. Therefore $\varsigma'(t)$ is valid. The result follows.

  The reverse implication is similar.
- The case $G \equiv \forall b.P$. We can assume b is distinct from a because we equate syntax up to α-conversion. $[\![\forall b.P]\!]_\varsigma$ is valid when $[\![P]\!]_{\varsigma\{b\mapsto x\}}$ for all $x \in \mathbb{D}$. The result follows by inductive hypothesis.

□

Lemma 5.5 **((∀R))** *If* $[\![\Gamma \vdash P, \Delta]\!]$ *and* $a \notin \Gamma, \Delta$ *then* $[\![\Gamma \vdash \forall a.P,\ \Delta]\!]$.

Proof. Fix ς. We now reason by cases:

- Fix some $G \in \Gamma$. By Lemma 5.4 $[\![G]\!]_\varsigma$ is not valid if and only if $[\![G]\!]_{\varsigma'}$ is not valid, for any other ς' such that $\varsigma(b) = \varsigma'(b)$ for all b other than a. Therefore $[\![\Gamma \vdash \forall a.P, \Delta]\!]_\varsigma$ is valid.
- Fix some $D \in \Delta$. If $[\![D]\!]_\varsigma$ is valid then by Lemma 5.4 also $[\![\Gamma \vdash \forall a.P, \Delta]\!]_\varsigma$ is valid.
- Suppose $[\![G]\!]_\varsigma$ is valid for all $G \in \Gamma$ and suppose $[\![D]\!]_\varsigma$ is not valid for all $D \in \Delta$. Then $[\![P]\!]_\varsigma$ is valid. By Lemma 5.4 $[\![P]\!]_{\varsigma\{a\mapsto x\}}$ must be valid for all $x \in \mathbb{D}$. Therefore $[\![\forall a.P]\!]_\varsigma$ is valid and so $[\![\Gamma \vdash \forall a.P, \Delta]\!]_\varsigma$ is valid as required.

□

Lemma 5.6 **((at R),(at L))** (i) *If* $[\![\Gamma \vdash \mathbf{at}\, u, \Delta]\!]_\varsigma$ *is valid and* u *is not a variable symbol then* $[\![\Gamma \vdash \Delta]\!]_\varsigma$ *is valid.*

(ii) *If* $[\![\Gamma, \mathbf{at}\, a \vdash \Delta]\!]_\varsigma$ *is valid and* $[\![a]\!]_\varsigma \in I$ *then* $[\![\Gamma \vdash \Delta]\!]_\varsigma$ *is valid.*

Proof.

(i) By Lemma 5.1 $[\![\mathbf{at}\, u]\!]_\varsigma$ is not valid on any valuation ς. The result then follows from the definition of validity.

(ii) Since $[\![a]\!]_\varsigma \in I$ it follows that $[\![\mathbf{at}\,a]\!]_\varsigma$ is valid. By definition, either $[\![G]\!]_\varsigma$ is invalid for some $G \in \Gamma$ or $[\![D]\!]_\varsigma$ is valid for some $D \in \Delta$. Again by definition, $[\![\Gamma \vdash \Delta]\!]_\varsigma$.

□

Theorem 5.7 (Soundness) *Suppose $[\![\text{-}]\!]$ is a model. Then if $\Gamma \vdash \Delta$ is derivable then $[\![\Gamma \vdash \Delta]\!]$ is valid.*

Proof. We work by induction on the depth of derivations.

- Most cases are immediate from the definitions (we have sketched the reasoning in what we consider the more complex cases in lemmas above). Only rule $(\mathbf{at}\,\mathbf{L})$ is not straightforward.
- The case of $(\forall\mathbf{L})$. By part (ii) of Lemma 5.3.
- The case of $(\forall\mathbf{R})$. By Lemma 5.5.
- The case of $(\mathbf{at}\,\mathbf{R})$. By Lemma 5.1.
- The case of $(\mathbf{at}\,\mathbf{L})$. Suppose $\Gamma, \mathbf{at}\,a \vdash \Delta$ is derivable where $a \notin \Gamma, \Delta$. Let a' be such that $[\![a']\!]_\varsigma \in I$. By Lemma 4.1 for $[a{:=}a']$ it follows that $\Gamma, \mathbf{at}\,a' \vdash \Delta$ is derivable, and the derivation is no deeper. By inductive hypothesis $[\![\Gamma, \mathbf{at}\,a' \vdash \Delta]\!]_\varsigma$ is valid. The result now follows by part (ii) of Lemma 5.6.

□

5.2 Prime theories and completeness

Let $\mathcal{T}$ range over (possibly infinite) sets of formulae; call $\mathcal{T}$ a ***theory***. Call $\mathcal{T}$ ***inconsistent*** when there exists some (finite) $\Gamma \subseteq \mathcal{T}$ such that $\Gamma \vdash$ is derivable. Otherwise call $\mathcal{T}$ ***consistent***. Call $\mathcal{T}$ ***maximal*** when:

- $\mathcal{T}$ is consistent.
- If $\Gamma \subseteq \mathcal{T}$ and $\Gamma \vdash P$ then $P \in \mathcal{T}$. We call $\mathcal{T}$ ***deductively closed***.
- For all P, $P \in \mathcal{T}$ or $\neg P \in \mathcal{T}$.
- If $P[a{:=}t] \in \mathcal{T}$ for all t then $\forall a.P \in \mathcal{T}$.
- $\mathbf{at}\,a \in \mathcal{T}$ for at least one a.

Lemma 5.8 *If $\mathcal{T}$ is maximal then $\forall a.P \in \mathcal{T}$ if and only if $P[a{:=}t] \in \mathcal{T}$ for all t. Also $\neg\forall a.P \in \mathcal{T}$ if and only if $\neg P[a{:=}t] \in \mathcal{T}$ for some t.*

Proof. By easy calculations using the definition of maximal set. □

Lemma 5.9 *If Γ is a consistent and $a \notin \Gamma$ then $\Gamma \cup \{\mathbf{at}\,a\}$ is consistent.*

Proof. We suppose $\Gamma, \mathbf{at}\,a \vdash$ and we prove a contradiction. By assumption $a \notin \Gamma$; it is not hard to extend the derivation of $\Gamma, \mathbf{at}\,a \vdash$ to a derivation of $\Gamma \vdash \forall a.\neg\mathbf{at}\,a$.

We combine this using (**Cut**) with the following derivation

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma, \mathbf{at}\, a \vdash \mathbf{at}\, a}\,(\mathbf{Ax})}{\Gamma, \mathbf{at}\, a, \neg\mathbf{at}\, a \vdash}\,(\supset \mathbf{L})}{\Gamma, \mathbf{at}\, a, \forall a.\neg\mathbf{at}\, a \vdash}\,(\forall\mathbf{L})}{\Gamma, \forall a.\neg\mathbf{at}\, a \vdash}\,(\mathbf{at}\,\mathbf{L})$$

to produce a derivation of $\Gamma \vdash$. This contradicts our assumption that Γ is consistent. $\square$

Lemma 5.10 *If Γ is finite and consistent then there exists a maximal theory $\mathcal{T}$ such that $\Gamma \subseteq \mathcal{T}$.*

Proof. Since Γ is finite there are infinitely many variable symbols that do not appear in Γ. The language is countable so we enumerate its predicates $P_1, P_2 \ldots$ We now construct a sequence Γ_0, Γ_1, $\ldots$ as follows:

- $\Gamma_0 = \Gamma \cup \{\mathbf{at}\, b\}$ for some b such that $b \notin \Gamma$.
- Suppose Γ_n, P_{n+1} is inconsistent. Then $\Gamma_{n+1} = \Gamma_n \cup \{\neg P_{n+1}\}$.
- Suppose Γ_n, P_{n+1} is consistent and P_n is not of the form $\neg\forall a.P$ for any P. Then $\Gamma_{n+1} = \Gamma_n \cup \{P_{n+1}\}$.
- Suppose Γ_n, P_{n+1} is consistent and P_n is of the form $\neg\forall a.P$ for some P. Then $\Gamma_{n+1} = \Gamma_n \cup \{\neg P[a{:=}b]\}$ for some $b \notin \Gamma_n$.

Note that $\Gamma_m \subseteq \Gamma_n$ if $m \leq n$. Write $\mathcal{T} = \bigcup_n \Gamma_n$. By construction $\Gamma \subseteq \mathcal{T}$. If we can prove that $\mathcal{T}$ is maximal then we are done:

- $\mathcal{T}$ is consistent: Γ_0 is consistent by Lemma 5.9. By construction Γ_{n+1} is consistent if Γ_n is, so Γ_n is consistent for all $n > 0$.
  If $\mathcal{T}$ is inconsistent then it has a finite inconsistent subset. This must be contained in some Γ_n and this is impossible. The result follows.
- $\mathcal{T}$ is deductively closed: Suppose that $\mathcal{T} \vdash P$. $P \equiv P_n$ for some n and by construction $P \in \Gamma_{n+1}$.
- For every P, either $P \in \mathcal{T}$ or $\neg P \in \mathcal{T}$: $P \equiv P_n$ for some n. By construction $P_n \in \Gamma_{n+1}$ or $\neg P_n \in \Gamma_{n+1}$.
- If $P[a{:=}s] \in \mathcal{T}$ for all s then $\forall a.P \in \mathcal{T}$. Suppose that $\forall a.P \notin \mathcal{T}$, then $\neg\forall a.P \in \mathcal{T}$. Now $\neg\forall a.P \equiv P_n$ for some n. Therefore $\neg P[a{:=}b] \in \Gamma_{n+1}$ for some $b \notin \Gamma_n$ and so $P[a{:=}b] \notin \mathcal{T}$.
- $\mathbf{at}\, b \in \mathcal{T}$ for at least one b: By construction of Γ_0.

$\square$

For a given set of sentences $\mathcal{T}$, define $|t| = \{u \mid t \rightsquigarrow u \in \mathcal{T}\}$.

Lemma 5.11 *If $\mathcal{T}$ is maximal then $|u| \subseteq |t|$ if and only $t \rightsquigarrow u \in \mathcal{T}$.*

Proof. Suppose that $|u| \subseteq |t|$. Since $\mathcal{T} \vdash u \rightsquigarrow u$ it follows that $u \in |u|$ and so if $|u| \subseteq |t|$ we have that $u \in |t|$ and so $t \rightsquigarrow u$. Conversely suppose $t \rightsquigarrow u \in T$.

Now if $s \in |u|$ then $u \rightsquigarrow s \in \mathcal{T}$ and so $t \rightsquigarrow s \in \mathcal{T}$ by the maximality of $\mathcal{T}$ and the transitivity of $\rightsquigarrow$ (see Theorem 2.1), and so $s \in |t|$. Thus, $|u| \subseteq |t|$. $\square$

Lemma 5.12 *If* Γ *is consistent then there exists some model* $[\![\text{-}]\!]$ *and some valuation* ς *on* $[\![\text{-}]\!]$ *such that* $[\![\Gamma]\!]_\varsigma$ *is valid.*

Proof. Suppose Γ is consistent. Extend it using the construction in the proof of Lemma 5.10 to a maximal set $\mathcal{T}$ such that $\Gamma \subseteq \mathcal{T}$.

Define an a-domain $\mathbb{D}$ by:

- $|\mathbb{D}| = \{|t| \mid t \text{ is a term}\}$
- $I_{|\mathbb{D}|} = \{|t| \mid \mathbf{at}\, t \in \mathcal{T}\}$
- $|t| \leq_{|\mathbb{D}|} |u|$ when $|t| \subseteq |u|$.

The arity $\mathbf{at} : (\downarrow)$ and the rules for aequality entail that if $\mathbf{at}\, t$ and $t \rightsquigarrow u$ are derivable then so is $\mathbf{at}\, u$. It follows that I is down-closed. Therefore $\mathbb{D}$ is an a-domain.

We specify a model $[\![\text{-}]\!]$ and valuation ς by stipulating that:

(i) If f has arity n and $|t_1|, \dots, |t_n| \in |\mathbb{D}|$. Then $[\![f]\!](|t_1|, \dots, |t_n|) = |f(t_1, \dots, t_n)|$.

(ii) $[\![p]\!] = \{(|t_1|, \dots, |t_n|) \mid p(t_1, \dots, t_n) \in \mathcal{T}\}$.

(iii) $\varsigma(a) = |a| \in |\mathbb{D}|$.

We must verify the properties required of a model.

- If $[\![f]\!](|t_1| \dots |t_n|) \in I$, then $|f(t_1, \dots, t_n)| \in I$ and so $\mathbf{at}\, f(t_1, \dots, t_n) \in \mathcal{T}$. But this is impossible since $\mathbf{at}\, f(t_1, \dots, t_n) \vdash$ and $\mathcal{T}$ is consistent. Therefore $[\![f]\!](|t_1| \dots |t_n|) \notin I$.

  Also if $|t_i| \leq |t_i'|$ then $|t_i| \subseteq |t_i'|$ and then $t_i' \rightsquigarrow t_i \in \mathcal{T}$. But since $\rightsquigarrow$ has arity $(\uparrow, \downarrow)$ we have that

$$\dfrac{\dfrac{}{\vdash f(t_1, \dots, t_i', \dots, t_n) \rightsquigarrow f(t_1, \dots, t_i', \dots, t_n)}(\rightsquigarrow \mathbf{R})}{t_i' \rightsquigarrow t_i \vdash f(t_1, \dots, t_i', \dots, t_n) \rightsquigarrow f(t_1, \dots, t_i, \dots, t_n)}(\rightsquigarrow \mathbf{L}{\downarrow})$$

  and so $f(t_1, \dots, t_i', \dots, t_n) \rightsquigarrow f(t_1, \dots, t_i, \dots, t_n) \in \mathcal{T}$. Therefore

$$|f(t_1, \dots, t_i, \dots, t_n)| \subseteq |f(t_1, \dots, t_i', \dots, t_n)|$$

  which entails that $|f(t_1, \dots, t_i, \dots, t_n)| \leq |f(t_1, \dots, t_i', \dots, t_n)|$. Thus if $|t_i| \leq |t_i'|$ then $[\![f]\!](|t_1|, \dots, |t_i|, \dots, |t_n|) \leq [\![f]\!](|t_1|, \dots, |t_i'|, \dots, |t_n|)$ and we have shown that $[\![f]\!]$ is monotone.

- By construction ς maps atoms to elements of $\mathbb{D}$. By construction there exists an a such that $\mathbf{at}\, a \in \mathcal{T}$. Also by construction $|\mathbf{at}\, a| \in I$. Therefore there exists an a such that $\varsigma(a) \in I$.

- Suppose the arity of p is $(d_1 \dots d_n)$ and suppose $|t_i| \leq |t_i'|$. By definition $|t_i| \subseteq |t_i'|$ and by Lemma 5.11 $t_i' \rightsquigarrow t_i \in \mathcal{T}$. There are now three cases:

  · The case that $d_i = \uparrow$:
  If $(|t_1|, \dots, |t_i|, \dots, |t_n|) \in [\![p]\!]$, then $p(t_1 \dots, t_i, \dots, t_n) \in \mathcal{T}$. By the following

derivation and the maximality of $\mathcal{T}$, we have that $p(t_1\dots,t'_i,\dots,t_n)\in\mathcal{T}$:

$$\dfrac{\dfrac{}{p(ts)[a{:=}t_i]\vdash p(ts)[a{:=}t_i]}\,(\mathbf{Ax}) \qquad [a{\uparrow}p(ts)]}{t'_i\rightsquigarrow t_i,\,p(ts)[a{:=}t_i]\vdash p(ts)[a{:=}t'_i]}\,(\rightsquigarrow\mathbf{L}{\uparrow})$$

Therefore $(|t_1|,\dots,|t'_i|,\dots,|t_n|)\in[\![p]\!]$.

· The cases $d_i=\downarrow$ and $d_i=\circlearrowleft$ are similar.

We prove by induction that $[\![t]\!]_\varsigma=|t|$ for any t:

- The case $t\equiv a$. $[\![a]\!]_\varsigma=\varsigma(a)=|a|$ by construction.
- The case $t\equiv f(t_1,\dots,t_n)$. Then

$$[\![f]\!]([\![t_1]\!]_\varsigma,\dots,[\![t_n]\!]_\varsigma)=[\![f]\!](|t_1|,\dots,|t_n|)=|f(t_1,\dots,t_n)|.$$

We prove by induction that $[\![P]\!]_\varsigma$ is valid if and only if $P\in\mathcal{T}$:

- The case that $P\equiv\mathbf{at}\,t$. $[\![\mathbf{at}\,t]\!]_\varsigma$ is valid when $[\![t]\!]_\varsigma\in I$, which happens when $|t|\in I$, and this happens when $\mathbf{at}\,t\in\mathcal{T}$.
- The case that $P\equiv t_1\rightsquigarrow t_2$. $[\![t_2\rightsquigarrow t_1]\!]_\varsigma$ is valid when $[\![t_1]\!]_\varsigma\leq[\![t_2]\!]_\varsigma$, which happens when $|t_1|\leq|t_2|$, which happens when $|t_1|\subseteq|t_2|$, and by Lemma 5.11 this happens when $(t_2\rightsquigarrow t_1)\in\mathcal{T}$.
- The case that $P\equiv p(t_1,\dots,t_n)$ for some $p\not\in\{\mathbf{at}\,,\rightsquigarrow\}$. $[\![p(t_1,\dots,t_n)]\!]_\varsigma$ is valid when $([\![t_1]\!]_\varsigma,\dots,[\![t_n]\!]_\varsigma)\in[\![p]\!]$. Now $[\![t_i]\!]_\varsigma=|t_i|$ for $1\leq i\leq n$, so this happens when $(|t_1|,\dots,|t_n|)\in[\![p]\!]$. By construction this happens when $p(t_1,\dots,t_n)\in\mathcal{T}$.
- Other cases are straightforward. For example:

$$\begin{array}{lll}
[\![\forall a.P]\!]_\varsigma \text{ is valid} & \text{iff} & [\![P]\!]_{\varsigma\{a\mapsto|t|\}} \text{ is valid for all } |t| \\
& \text{iff} & [\![P[a{:=}t]]\!]_\varsigma \text{ is valid for all } t \text{ (Lemma 5.2)} \\
& \text{iff} & P[a{:=}t]\in\mathcal{T} \text{ is valid for all } t \text{ (induction hypothesis)} \\
& \text{iff} & \forall a.P\in\mathcal{T}
\end{array}$$

We conclude that $[\![P]\!]_\varsigma$ is valid for every $P\in\mathcal{T}$, and it follows that $[\![\Gamma]\!]_\varsigma$ is valid.□

Theorem 5.13 (Completeness) *If $[\![\Gamma\vdash\Delta]\!]$ is valid in all models then $\Gamma\vdash\Delta$.*

Proof. a-logic is classical. Therefore $\Gamma\vdash\Delta$ when $\Gamma\cup\{\neg D : D\in\Delta\}\vdash$. It suffices to show that if $\Gamma\cup\{\neg D : D\in\Delta\}$ is consistent then there is a model $[\![\text{-}]\!]$ and valuation ς such that $[\![P]\!]_\varsigma$ is valid for all $P\in\Gamma\cup\{\neg D : D\in\Delta\}$. This follows by Lemma 5.12. □

6 Consistency and nontriviality of the axiomatisation of the λ-calculus

We build a model, in the sense of Subsection 5.1, of the theory for the untyped λ-calculus from Subsection 3.2. This illustrates an easy but non-trivial example of an a-logic model and it suffices to prove consistency of the axioms in Figures 2, 3 and 4.

Let Λ be the set of untyped λ-terms, generated by the grammar

$$q, r ::= a \mid \lambda a.q \mid qq.$$

Here a ranges over variables of a-logic; it is convenient to identify these with variables of the λ-calculus. We define α-conversion and β-reduction on λ-terms set as usual [4]. Write $fv(q)$ for the free variables of q. For example $a \not\in fv(\lambda a.a)$.

Write $q[r/a]$ for the capture-avoiding substitution of r for a in q, making some fixed but arbitrary choice about how to rename variable symbols to avoid capture. For example $(\lambda a.b)[a/b] = \lambda a'.a$ for some fixed but arbitrary choice of a'.

Remark 6.1 Note that our use of $q[r/a]$ for capture-avoiding substitution in λ-terms is distinct from the (syntactic) substitution action $[a{:=}t]$ of Subsection 2.2: $[q/a]$ is an operation on the λ-terms which we are about to use to construct a concrete model for the theory of Figures 2 and 4; $[a{:=}t]$ is an operation on a-logic terms.

Write $q \to_{\alpha\beta} q'$ for the relation generated by an α-conversion followed by a β-reduction followed by α-conversion, and call this $\alpha\beta$***-reduction***.[6]

Write $\to^*_{\alpha\beta}$ for the transitive reflexive closure of $\to_{\alpha\beta}$. That is, $q' \to^*_{\alpha\beta} q$ when either $q' = q$ or there exists some chain $q_2, \dots, q_{n-1}$ such that

$$q' \to_{\alpha\beta} q_2 \to_{\alpha\beta} \dots \to_{\alpha\beta} q_{n-1} \to_{\alpha\beta} q.$$

Write $q'{\downarrow_{\alpha\beta}}$ for the relational image of $\{q'\}$ under $\to^*_{\alpha\beta}$. That is,

$$q'{\downarrow_{\alpha\beta}} = \{q \mid q' \to^*_{\alpha\beta} q\}.$$

All of these definitions are standard [4].

Definition 6.2 Let ¿ be some symbol not in Λ and read it as ***error***. We define an a-domain $\mathbb{D}$ by:

- $|\mathbb{D}| = \{q{\downarrow_{\alpha\beta}} \mid q \in \Lambda\} \cup \{¿\}$. x, y, z will range over elements of $|\mathbb{D}|$.
- $x \leq y$ when $x \subseteq y$ or $x = y = ¿$.
- $I = \{\{a\} \mid a \in \mathbb{A}\} \cup \{\emptyset\}$.

We define a valuation ς by $\varsigma(a) = \{a\}$. We define a model $[\![\text{-}]\!]$ by:

- $[\![\lambda]\!](x, y) = (\lambda a.r){\downarrow_{\alpha\beta}}$ if $x \cap \mathbb{A} = \{a\}$ and $y = r{\downarrow_{\alpha\beta}}$.
  $[\![\lambda]\!](x, y) = \{¿\}$ otherwise.
- $x[\![\,\cdot\,]\!]y = (qr){\downarrow_{\alpha\beta}}$ if $x = q{\downarrow_{\alpha\beta}}$ and $y = r{\downarrow_{\alpha\beta}}$.
  $x[\![\,\cdot\,]\!]y = \{¿\}$ otherwise.
- $[\![\langle\rangle]\!](z, x, y) = (q[r/a]){\downarrow_{\alpha\beta}}$ if $z = q{\downarrow_{\alpha\beta}}$, $x \cap \mathbb{A} = \{a\}$, and $y = r{\downarrow_{\alpha\beta}}$.
  $[\![\langle\rangle]\!](z, x, y) = \{¿\}$ otherwise.
- $(x, y) \in [\![\#]\!]$ when $x \cap \mathbb{A} = \{a\}$ and $y = r{\downarrow_{\alpha\beta}}$ and $a \not\in fv(r)$.

[6] The first author has studied different ways these definitions can be expressed. This is not the issue for constructing the model; for our purposes it suffices to know that the relation exists.

$(x, y) \notin [\![\#]\!]$ otherwise.

λ, $\cdot$, and $\langle\rangle$ in the language of $\mathcal{L}$ are term-formers with arities 2, 2, and 3 respectively. a-logic permits 'silly terms' such as $\lambda(\lambda a.\lambda b.ab).c$. Standard methods are available to exclude them; for instance a sorting system for a-logic could be developed. On the other hand, we could even 'embrace the silliness', allow such terms, and investigate their properties. All of this is future work; for our purposes we need only include ¿ an error value, and in Definition 6.2 we let the denotation of a silly term such as $\lambda(\lambda a.\lambda b.ab).c$ be $\{¿\}$. For more future work, it may be possible to identify ¿ with the $\perp$ of Scott domain models of the λ-calculus [27]. However, this requires a more sophisticated construction, perhaps involving nominal domain theory [26].

We now sketch a proof that we have indeed constructed a model of $\mathcal{L}$. Lemmas 6.3 and 6.4 are technical lemmas needed for Corollary 6.5.

Lemma 6.3 *Suppose $x, x' \in |\mathbb{D}|$ and suppose $x \leq x'$. Then:*

- *If $x' = r'{\downarrow_{\alpha\beta}}$ for some r' then $x = r{\downarrow_{\alpha\beta}}$ for some $r \in x'$ (that is, $x = r{\downarrow_{\alpha\beta}}$ for some r such that $r' \to^*_{\alpha\beta} r$).*
- *If $x' \cap \mathbb{A} = \{a\}$ then $x \cap \mathbb{A} = \{a\}$.*

Proof.

- Suppose $x' = r'{\downarrow_{\alpha\beta}}$ and suppose $x \leq x'$. Note that $¿ \notin x'$. By definition of $\leq$ we know $x \subseteq x'$. It follows that $¿ \notin x$ and so $x = r{\downarrow_{\alpha\beta}}$ for some r. It follows that $r \in x'$ and therefore that $r' \to^*_{\alpha\beta} r$.
- Suppose $x' \cap \mathbb{A} = \{a\}$ and $x \leq x'$. By construction of $|\mathbb{D}|$ it follows that $x' = r'{\downarrow_{\alpha\beta}}$ and $r' \to^*_{\alpha\beta} a$. By the first part of this result $x = r{\downarrow_{\alpha\beta}}$ and $r' \to^*_{\alpha\beta} r$. It is a fact that a does not β-reduce. By confluence of λ-calculus reduction it follows that $r \to^*_{\alpha\beta} a$. The result follows.

□

Lemma 6.4 *If $x \in |\mathbb{D}|$ then precisely one of the following holds:*

- *$x \cap \mathbb{A} = \{a\}$ for some $a \in \mathbb{A}$.*
- *$x \cap \mathbb{A} = \emptyset$.*

Proof. We work by cases on possible forms for $x \in |\mathbb{D}|$:

- The case $x = q{\downarrow_{\alpha\beta}}$. It is a fact of λ-calculus reduction that if $q \to^*_{\alpha\beta} a$ for some $a \in \mathbb{A}$ then $q \not\to^*_{\alpha\beta} b$ for any other b.
- The case $x = \{¿\}$. Immediate.

□

Corollary 6.5 $[\![\lambda]\!]$, $[\![\cdot]\!]$, *and* $[\![\langle\rangle]\!]$ *are monotone.*

Proof. It suffices to verify that:

- If $x \leq x'$ and $y \leq y'$ then $[\![\lambda]\!](x, y) \leq [\![\lambda]\!](x', y')$.
- If $x \leq x'$ and $y \leq y'$ then $[\![\cdot]\!](x, y) \leq [\![\cdot]\!](x', y')$.
- If $x \leq x'$, $y \leq y'$, and $z \leq z'$ then $[\![\langle\rangle]\!](x, y, z) \leq [\![\langle\rangle]\!](x', y', z')$.

We will consider only the first of these calculations, for λ; the calculations for $\cdot$ and $\langle\rangle$ are similar. Suppose that $x \leq x'$ and $y \leq y'$.

By Lemma 6.4 either $x' \cap \mathbb{A} = \emptyset$ or $x \cap \mathbb{A} = \{a\}$ for some $a \in \mathbb{A}$. It is helpful, though, to distinguish three cases:

- The case that $y' = \{¿\}$. Then $[\![\lambda]\!](x', y') = \{¿\}$. We assumed $y \leq y'$. By construction of $|\mathbb{D}|$ it follows that $y = \{¿\}$. Therefore $[\![\lambda]\!](x, y) = \{¿\}$ and the result follows.
- The case that $x' \cap \mathbb{A} = \emptyset$. Then $[\![\lambda]\!](x', y') = \{¿\}$. We assumed $x \leq x'$ and so (since the definition of $\leq$ unpacks to $x \subseteq x'$) $x \cap \mathbb{A} = \emptyset$. Therefore $[\![\lambda]\!](x, y) = \{¿\}$ and the result follows.
- The case that $x' \cap \mathbb{A} = \{a\}$ and $y' = r'{\downarrow_{\alpha\beta}}$. Then $[\![\lambda]\!](x', y') = (\lambda a.r'){\downarrow_{\alpha\beta}}$. By part 1 of Lemma 6.3 since $x \leq x'$ it must be that $x \cap \mathbb{A} = \{a\}$. By part 2 of Lemma 6.3 since $y \leq y'$ it must be that $y = r{\downarrow_{\alpha\beta}}$ for some r such that $r' \to^*_{\alpha\beta} r$. By construction $[\![\lambda]\!](x, y) = (\lambda a.r){\downarrow_{\alpha\beta}}$. By properties of $\alpha\beta$-reduction $\lambda a.r' \to^*_{\alpha\beta} \lambda a.r$ and so ($[\![\lambda]\!](x, y) \subseteq [\![\lambda]\!](x', y')$ and so by definition) $[\![\lambda]\!](x, y) \leq [\![\lambda]\!](x', y')$ as required.

$\square$

Theorem 6.6 *$\mathbb{D}$ is an a-domain. ς is a valuation. $[\![\text{-}]\!]$ is a model.*

Proof. We verify all the required properties in turn. $\mathbb{D}$ is an a-domain:

- $|\mathbb{D}|$ is a set: This is a fact.
- $\subseteq$ is a partial order on $|\mathbb{D}|$: This is also a fact.
- I is down-closed: Suppose $x' \in I$ and $x \leq x'$. By definition of I we know that $x' = \{a\}$ for some $a \in \mathbb{A}$. By part 1 of Lemma 6.3 we know $x = x'$. It follows that $x \in I$.

ς is a valuation: By construction ς maps a to $\{a\} \in |\mathbb{D}|$. Also, $\varsigma(a) \in I$ for all $a \in \mathbb{A}$, and therefore $\varsigma(a) \in I$ for at least one $a \in \mathbb{A}$.

$[\![\text{-}]\!]$ is a model:

- λ is a term-former with arity 2. By construction $[\![\lambda]\!]$ is a function from $|\mathbb{D}|^2$ to $|\mathbb{D}|$. It is a fact of λ-calculus reduction that if $\lambda a.q \to^*_{\alpha\beta} q'$ then q' is not a variable symbol. It follows that $[\![\lambda]\!](x, y) \notin I$ always. We proved in Corollary 6.5 that $[\![\lambda]\!]$ is monotone.
- The cases of application and substitution are similar.
- Recall that $\# : (\downarrow, \downarrow)$. Suppose that $x \leq x'$, $y \leq y'$, and $(x', y') \in [\![\#]\!]$. We must show that $(x, y) \in [\![\#]\!]$.

  By construction $x' = \{a\}$ for some $a \in \mathbb{A}$, $y' = r'{\downarrow_{\alpha\beta}}$, and $a \notin \mathit{fv}(r')$. By part 1 of Lemma 6.3 since $x \leq x'$ we know $x = \{a\}$. By part 2 of Lemma 6.3 since $y \leq y'$ it must be that $y = r{\downarrow_{\alpha\beta}}$ for some r such that $r' \to^*_{\alpha\beta} r$. It is a fact of the λ-calculus that $a \notin \mathit{fv}(r)$. The result follows.

$\square$

Theorem 6.7 *The construction above yields a model of $\mathcal{L}$.*

Proof. We verify that the axioms given in in turn. We consider only two cases.

($\mapsto$#)

We must show that if $(x, y) \in [\![\#]\!]$ then

- $x = \{a\} \in I$,
- for all $\{b\} \in I$ such that $b \neq a$ it is the case that $y \subseteq [\![\langle\rangle]\!](y, \{a\}, \{b\})$.

Suppose that $(x, y) \in [\![\#]\!]$. By definition $x = \{a\}$, so $x \in I$. Also by definition $y = r{\downarrow_{\alpha\beta}}$ and $a \notin fv(r)$. By definition $[\![\langle\rangle]\!](r{\downarrow_{\alpha\beta}}, \{a\}, q{\downarrow_{\alpha\beta}}) = (r[q/a]){\downarrow_{\alpha\beta}}$. But, since $a \notin fv(r)$, $r[q/a] \equiv r$. And so $y = r{\downarrow_{\alpha\beta}} \subseteq r[q/a]){\downarrow_{\alpha\beta}}$. The result follows.

($\mapsto$**comm**)

Suppose that $x_2 = \{a_2\} \in I$, $(x_1, x_2) \in [\![\#]\!]$ and $(x_1, y_2) \in [\![\#]\!]$. We must show that

- $[\![\langle\rangle]\!]\Big([\![\langle\rangle]\!](z, x_2, y_2), x_1, [\![\langle\rangle]\!](y_1, x_2, y_2)\Big) \subseteq [\![\langle\rangle]\!]\Big([\![\langle\rangle]\!](z, x_1, y_1), x_2, y_2\Big)$

We may suppose that $y_1 = r_1{\downarrow_{\alpha\beta}}$, $y_2 = r_2{\downarrow_{\alpha\beta}}$ and $z = q{\downarrow_{\alpha\beta}}$. It follows that $x_1 = \{a_1\} \in I$ and that $a \notin fv(r_2)$. It also follows that $a_1 \notin fv(a_2)$ and, since a_2 is just a variable, this entails that $a_1 \neq a_2$.

Now $[\![\langle\rangle]\!](z, x_2, y_2) = q[r_2/a_2]{\downarrow_{\alpha\beta}}$ and $[\![\langle\rangle]\!](y_1, x_2, y_2) = r_1[r_2/a_2]{\downarrow_{\alpha\beta}}$ by definition. So $[\![\langle\rangle]\!](q[r_2/a_2]{\downarrow_{\alpha\beta}}, x_1, r_1[r_2/a_2]{\downarrow_{\alpha\beta}}) = q\big[r_2/a_2\big]\big[r_1[r_2/a_2]/a_1\big]{\downarrow_{\alpha\beta}}$.

Similarly it follows that $[\![\langle\rangle]\!]\Big([\![\langle\rangle]\!](z, x_1, y_1), x_2, y_2\Big) = q[r_1/a_1][r_2/a_2]{\downarrow_{\alpha\beta}}$

But it is a fact of the λ-calculus that if $a_1 \neq a_2$ and $a \notin fv(r_2)$ that $q[r_1/a_1][r_2/a_2] \equiv q\big[r_2/a_2\big]\big[r_1[r_2/a_2]/a_1\big]$.

The other axioms in Figures 2, 3 and 4 can be verified similarly. $\square$

7 Conclusions

a-logic is a classical first-order logic with a two-valued model theory, one of many [5,13]. The twist to the story is that **at** can identify variables, and accordingly truth-values are given also to open predicates.

In the semantics, predicates are sets of (tuples of) elements of the domain. **at** is a unary predicate. Atoms are simply a subset of the underlying domain; those in $[\![$**at**$]\!]$ which is the interpretation of **at** in the domain. Elements of $[\![$**at**$]\!]$ are special; they cannot be in the image of the interpretation of any term-formers.

We believe that a-logic could provide a logical semantics for *isvar* in PROLOG, as used for example in practical programming in CIAO PROLOG [9]. *isvar* is a *non-logical predicate* which is true when its argument has not yet been unified with a concrete value — that is, when it is a variable. This is used exactly as we have used it in this paper, to master shallow embeddings into CIAO of languages with variables. **at** and *isvar* appear to be compatible but problems arise interpreting PROLOG conjunction, which ceases to be commutative in the presence of *isvar*. This is a known issue which we are attempting to solve by using a modified translation of CIAO conjunction which is will be the topic of a future paper.

A previous paper considered a variant of a-logic with equality [15]. There, in order to axiomatise substitution we had to introduce a (in the authors' opinion) *ad hoc* notion which we called 'essentially a term'. In this paper we take a different path and introduce a *directed* equality $\rightsquigarrow$ — the *arrow* in the title of this paper. This solves the issues we encountered in [15], arguably in more elegant way which seems to benefit the resulting theory and axioms. In particular, our principal axiomatisations (of the λ-calculus and of substitution) are improved.

It is relevant how conveniently a-logic can be implemented in a theorem-prover such as Isabelle [21]. We do not know.

$\rightsquigarrow$ gives a-logic a flavour of a 'logical theory of rewriting'.[7] Rewriting itself is traditionally intensional on the syntax of terms (we look at a term's syntax to decide whether to rewrite), or completely abstract (we just take any relation on a set, but the only properties of its elements are the relation so elements are closed, in the sense that instantiation is absent) [3,28]. In some sense **at** provides just enough to give a point of contact between these two worlds and it may be possible to make this idea useful for the theory of rewriting.

How else can the λ-calculus (and friends) be modelled in a formal system and how does a-logic relate to them?

We can use a higher-order system to begin with (build the system as an extension of a λ-calculus). The λ-calculus cannot recognise a variable symbol as such, so to use it to model the λ-calculus requires deep embedding which model variables by closed terms (in essence, you have to write a Turing machine).

First-order logic also cannot recognise a variable symbol. We can take a two-level hierarchy with object-level variables modelled by constants. We cannot quantify over constants so a hallmark of such approaches is an infinity of axioms, one for each constant (the a-logic axiomatisations look similar, but are finite). This approach is followed for example by Cylindric Algebras [8] (which may assume only finitely many variable symbols so as to obtain a finite theory), by Salibra's Lambda Abstraction Algebras [24], and by some treatments of Structural Operational Semantics (SOS) [1], see for example Bernstein's SOS axiomatisations of the π-calculus, the lazy λ-calculus, and CHOCS [6].

We can do away with variables entirely, for example using Boehm trees, Scott domains, or combinators [4]. The translations are only for closed terms (hence, not compositional) and combinators may suffer exponential blow-up [7]. This is a bit off-topic for us, because we are interested not just in a 'naked denotation' but also in the formal system it is constructed in.

We can suppose a two-level hierarchy of variables and meta-variables and assume β-reduction for the former; computation/reasoning is by instantiation of meta-variables following by reduction. One of several examples of this style is Combinatory Reduction Systems (CRS). Note that in CRS meta-variables vary over closed elements — the higher-order elements are used to feed arguments to where they are used. In a-logic unknowns may be substituted for open terms. So for example $a\#b$ states that (whatever b is) it should not depend on a. This is not expressible in a CRS. Being a variable or meta-variable is an intensional property in CRS, whereas in a-logic the difference is expressed by **at** and can be mixed with implication and the rest of the logic. Finally, in CRS β-reduction is built-in. Although we certainly concentrate on substitutions and the λ-calculus in this paper, other theories are possible so a-logic is probably more flexible.

[7] To our surprise, we could find no reference in the literature for a logic-based-on a transitive reflexive not-necessarily-symmetric congruence — that is, for a logic of rewriting. We would be happy to hear of such a reference. Rewriting Logic [19] seems different, informally treating terms as formulae.

Nominal logic [22] has a notion of atom but keeps this separate from the notion of variable; this is because nominal logic was designed for reasoning *on* syntax — in our terminology, reasoning on a deep embedding where there is a clear distinction between variable symbols of the object-level language, called atoms because they behave like atomic entities, and variables of the meta-level language which range over unknown elements of the domain. A shallow embedding of (for example) the λ-calculus in nominal logic is not possible. We also believe, but have not proved, that a-logic is consistent with arbitrary choice. Nominal logic is not [22], so a-logic may have some interesting technical advantages.

a-logic may be closer to Miller and Tiu's logic of Generic Judgements (GJ) [20], where 'being a variable symbol' is managed by scoping for a dedicated quantifier ∇ instead of a dedicated predicate **at** . GJ is significantly different in being higher-order and having a 'definitional' equality [25] suited more to logic-programming than model theory; we suspect it is unsuited for our applications.

Kit Fine's 'Reasoning with arbitrary objects' [12] axiomatises dependence between elements. Perhaps our semantics, which contain variables, may be related.

References

[1] L. Aceto, W. Fokkink, and C. Verhoef. Structural operational semantics. In *Handbook of Process Algebra*. Elsevier, 1999.

[2] A. Azurat and I.S.W.B. Prasetya. A survey on embedding programming logics in a theorem prover. Technical Report 7, Institute of information and computing sciences, Utrecht University, 2002.

[3] Franz Baader and Tobias Nipkow. *Term rewriting and all that*. Cambridge University Press, Great Britain, 1998.

[4] H.P. Barendregt. *The Lambda Calculus: its Syntax and Semantics (revised ed.)*. North-Holland, 1984.

[5] John Bell and Moshé Machover. *A course in mathematical logic*. North-Holland, 1977.

[6] Karen L. Bernstein. A congruence theorem for structured operational semantics of higher-order languages. In *Logic in Computer Science*, pages 153–164, 1998.

[7] Martin W. Bunder. Some improvements to Turner's algorithm for bracket abstraction. *J. Symb. Log.*, 55(2):656–669, 1990.

[8] S. Burris and H. Sankappanavar. *A Course in Universal Algebra*. Springer, 1981.

[9] CIAO. CIAO Prolog.
`www.clip.dia.fi.upm.es/Software/Ciao/ciao_html/ciao_toc.html`.

[10] Marcel Crabbé. On the notion of substitution. *Logic Journal of the IGPL*, 12(2):111–124, 2004.

[11] Maribel Fernández and Murdoch J. Gabbay. Nominal rewriting. *Information and Computation*, 205:917–965, 2007.

[12] Kit Fine. *Reasoning with Arbitrary Objects*. Blackwell, 1985.

[13] Dov Gabbay and Franz Günthner, editors. *Handbook of philosophical logic*. Number 166 in Synthese Library. D.Reidel Publishing company, 1986.

[14] Murdoch J. Gabbay and J. Cheney. A sequent calculus for nominal logic. In *Proc. 19th IEEE Symposium on Logic in Computer Science (LICS 2004)*, pages 139–148. IEEE Computer Society, 2004.

[15] Murdoch J. Gabbay and Michael J. Gabbay. a-logic. In *We Will Show Them: Essays in Honour of Dov Gabbay*, volume 1. College Publications, 2005.

[16] Murdoch J. Gabbay and Aad Mathijssen. Capture-avoiding substitution as a nominal algebra. In *ICTAC'2006: 3rd International Colloquium on Theoretical Aspects of Computing*, volume 4281 of *LNCS*, pages 198–212, 2006.

[17] Murdoch J. Gabbay and Aad Mathijssen. Capture-avoiding substitution as a nominal algebra. *Formal Aspects of Computing*, 2008. Available online.

[18] Murdoch J. Gabbay and Andrew M. Pitts. A new approach to abstract syntax with variable binding. *Formal Aspects of Computing*, 13(3–5):341–363, 2002.

[19] Narciso Martí-Oliet and José Meseguer. Rewriting logic: roadmap and bibliography. *Theor. Comput. Sci.*, 285(2):121–154, 2002.

[20] Dale Miller and Alwen Tiu. A proof theory for generic judgments: An extended abstract. In *Proc. of LICS 2003*, pages 118–127. IEEE, 2003.

[21] Larry Paulson. *The Isabelle reference manual.* Cambridge University Computer Laboratory, 2001.

[22] Andrew M. Pitts. Nominal logic, a first order theory of names and binding. *Information and Computation*, 186(2):165–193, 2003.

[23] Antonino Salibra. On the algebraic models of lambda calculus. *Theoretical Computer Science*, 249(1):197–240, 2000.

[24] Antonino Salibra. Lambda calculus: models and theories. In *AMiLP-2003*, number 21 in TWLT Proceedings, pages 39–54, 2003.

[25] Peter Schroeder-Heister. Definitional reflection and the completion. In *Extensions of Logic Programming*, volume 798 of *Springer Lecture Notes in Artificial Intelligence*, pages 333–347, 1993.

[26] Mark R. Shinwell and Andrew M. Pitts. On a monadic semantics for freshness. *Theoretical Computer Science*, 342(1):28–55, 2005.

[27] Joseph E. Stoy. *Denotational Semantics: The Scott-Strachey Approach to Programming Language Theory.* MIT Press, Cambridge, MA, USA, 1977.

[28] Vincent van Oostrom. *Confluence for Abstract and Higher-Order Rewriting.* PhD thesis, Vrije Universiteit, Amsterdam, 1994.

[29] Martin Wildmoser and Tobias Nipkow. Certifying machine code safety: Shallow versus deep embedding. In *TPHOLs*, volume 3223 of *LNCS*, pages 305–320. Springer, 2004.

Acknowledgements. We acknowledge the support of British Academy grant PDF/2006/509 and grant RYC-2006-002131 at the Polytechnic University of Madrid.