a-logic

Murdoch J. Gabbay

Joint work with Michael J. Gabbay

University of Amsterdam, Amsterdam, 22/11/2005

Thanks to Johan van Benthem and Olivier Roy for inviting me

Syntax of a-logic

a-logic is First-Order Logic, predicates inductively defined by:

$$\phi, \psi ::= \bot \mid p(t_1, \ldots, t_n) \mid \forall x.\, \phi \mid \phi \supset \phi$$

p are atomic predicate symbols (with arity n). Term language is arbitrary; suppose it has numbers.

a-logic has one distinguished unary atomic predicate symbol $\mathbf{at}$, read 'atom'.

Intuitively:

$\mathbf{at}\,(x)$ is 'true' $\qquad$ $\mathbf{at}\,(3)$ is 'false'.

Sequents of a-logic

A context Γ is a finite set of predicates. So is a cocontext Δ. A sequent is a pair $\Gamma \vdash \Delta$. Valid sequents are inductively defined by (the usual rules and):

$$\frac{(s \text{ not a variable symbol})}{\Gamma, \mathbf{at}\, s \vdash \Delta}(\mathbf{at}\, L) \qquad \frac{\Gamma,\ \mathbf{at}\, a \vdash \Delta}{\Gamma \vdash \Delta}(Fresh)$$

(In $(Fresh)$, $a \not\in \Gamma, \Delta$.)

Example derivations

$$\frac{}{\mathbf{at}\,(3)\ \vdash}\,(\mathbf{at}\,L)$$

. . . is valid, since 3 is not a variable symbol.

$$\frac{}{\mathbf{at}\,(x)\ \vdash}\,(FALSE)$$

. . . is not valid, since x is a variable symbol.

$(Fresh)$ says 'an atom exists', more on that later. . .

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

Substitution lemma

Necessary for cut-elimination, and important correctness result stating 'variables really do represent unknown terms':

Lemma:

$$\Gamma \vdash \Delta \text{ derivable implies } \Gamma[a{\mapsto}u] \vdash \Delta[a{\mapsto}u] \text{ is derivable.}$$

Proof (sketched):

Observe that the property of not being a term is invariant under substitution. Therefore if

$$\frac{}{\Gamma,\ \mathbf{at}\ s \vdash \Delta}(\mathbf{at}\ L)$$

is valid, then so is

$$\frac{}{\Gamma[a{\mapsto}u],\ \mathbf{at}\ s[a{\mapsto}u] \vdash \Delta[a{\mapsto}u]}(\mathbf{at}\ L).$$

a-logic as a First-Order Logic theory

$(\mathbf{at}\ L)$ is equivalent to $\forall \overline{x}.\ \neg \mathbf{at}\ f\overline{x}$ for each term-former f.

$(Fresh)$ is equivalent to $\exists a.\ \mathbf{at}\ a$.

Easy-peasy!

a-logic axioms of substitution

$\langle$`suspension of disbelief`$\rangle$

Let explicit substitution $u\langle a\mapsto x\rangle$ be a ternary term-former.

Then

$$a\#u \quad \text{is sugar for} \quad \mathbf{at}\, a \ \wedge\ \forall x.\, u\langle a\mapsto x\rangle = u$$

and expressed 'a is not free in u'.

$\langle$`/suspension of disbelief`$\rangle$

a-logic axioms of substitution

$$\mathbf{at}\,a \supset u\langle a\mapsto a\rangle = u \qquad \mathbf{at}\,a \supset a\langle a\mapsto x\rangle = x$$

$$\mathbf{at}\,a \wedge \mathbf{at}\,b \supset (a \neq b \Leftrightarrow a\#b)$$

$$\mathbf{at}\,a \wedge b\#u \supset u\langle a\mapsto b\rangle\langle b\mapsto y\rangle = u\langle a\mapsto y\rangle$$

$$\mathbf{at}\,a \wedge a\#x \supset a\#u\langle a\mapsto x\rangle$$

$$\mathbf{at}\,b \wedge a\#b \wedge a\#y \supset u\langle a\mapsto x\rangle\langle b\mapsto y\rangle = u\langle b\mapsto y\rangle\langle a\mapsto x\langle b\mapsto y\rangle\rangle$$

Provably closed

Write

$$\bullet s \quad \text{for} \quad \forall a.\, \mathbf{at}\, a \supset a\#s$$

This says ‘s is (provably) closed’.

$\bullet s$ does not imply that s is actually closed viewed as syntax.

For example, from $\bullet x$ we may derive $\bullet x$ — but x is a variable.

Similarly,

$$\begin{array}{c} \bullet x \\ \vdots \\ \forall a.\, \mathbf{at}\, a \supset \bullet(a\langle a \mapsto x\rangle) \end{array}$$

but $a\langle a \mapsto x\rangle$ clearly has free variables a and x.

a-logic axioms of the λ-calculus

$$
\begin{array}{lrl}
(\alpha) & \forall a, b, x, y. & \mathbf{at}\, a \wedge b\#x \supset \lambda a.x = \lambda b.x\langle a \mapsto b\rangle \\
(\beta) & \forall a, x, y. & \mathbf{at}\, a \wedge \bullet y \supset (\lambda a.x)y = x\langle a \mapsto y\rangle \\
(\xi) & \forall x, y. & \bullet x \wedge \bullet y \\
 & & \quad \supset (\forall z.\, \bullet z \supset xz = yz) \supset x = y \\
(\sigma\lambda) & \forall a, b, x, y. & \mathbf{at}\, b \wedge a\#y \supset (\lambda a.x)\langle b \mapsto y\rangle = \lambda a.(x\langle b \mapsto y\rangle) \\
(\sigma app) & \forall a, x, y, z. & \mathbf{at}\, a \supset (xy)\langle a \mapsto z\rangle = (x\langle a \mapsto z\rangle)(y\langle a \mapsto z\rangle) \\
(\#app) & \forall x, y.\, \exists a. & a\#x \wedge a\#y \\
 & & \quad \wedge \forall b.\, (b\#axy \Leftrightarrow (b\#a \wedge b\#x \wedge b\#y))
\end{array}
$$

(Recall that $a\#x$ implies $\mathbf{at}\, a$.)

Meaning of the axioms

- $(\alpha)\quad \forall a,b,x,y.\ \mathbf{at}\,a \wedge b\#x \supset \lambda a.x = \lambda b.x\langle a{\mapsto}b\rangle$:
  α-equivalence, obviously.

- $(\beta)\quad \forall a,x,y.\ \mathbf{at}\,a \wedge \bullet y \supset (\lambda a.x)y = x\langle a{\mapsto}y\rangle$:
  β-equivalence, but only for provably closed terms.

- $(\xi)\quad \forall x,y.\ \bullet x \wedge \bullet y \supset (\forall z.\ \bullet z \supset xz = yz) \supset x = y$:
  How do we test whether two λ-abstractions are equal? Provided they are provably closed, they can β-reduce, so apply them to equal arguments.

Meaning of the axioms

- $(\sigma\lambda)$ $\forall a, b, x, y.\ \mathbf{at}\, b \wedge a\#y \supset (\lambda a.x)\langle b{\mapsto}y\rangle = \lambda a.(x\langle b{\mapsto}y\rangle)$:
  How to distribute substitution under λ. Note works also for open terms.

- (σapp) $\forall a, x, y, z.\ \mathbf{at}\, a \supset (xy)\langle a{\mapsto}z\rangle = (x\langle a{\mapsto}z\rangle)(y\langle a{\mapsto}z\rangle)$:
  How to distribute substitution under application. Again, works also for open terms.

- $(\#app)$ $\forall x, y.\ \exists a.\ a\#x \wedge a\#y \wedge\ \forall b.\ (b\#axy \Leftrightarrow (b\#a \wedge b\#x \wedge b\#y))$:
  Believe it or not, this says “there are infinitely many atoms”.

“There are infinitely many atoms”

$(\#app)\quad \forall x, y.\, \exists a.\, a\#x \wedge a\#y \wedge\ \forall b.\, (b\#axy \Leftrightarrow (b\#a \wedge b\#x \wedge b\#y))$:

Think of axy as ‘the pair (x, y) at a’, write it $(x, y)_a$.

By $(Fresh)$ there is an atom a. We get $b\#a$ from $(a, a)_b$. Then $c\#a, b$ from $(b, (a, a))_c$, and so on. Similarly for any $\{x_1, \ldots, x_n\}$ we generate $a\#x_1$ from $(x_1, x_1)_a$, and $b\#a, x_1, x_2$ from $(x_2, (x_1, x_1)_a)_b$, and so on.

This axiomatisation is correct in a suitable formal sense:

Lemma: From a FOL model of alogic + sub + lambda we obtain a λ-model by taking $\{p \mid \bullet p\}$.

(An ‘extensional combinatory algebra’ is a model of the FOL theory ECA

$$\forall x, y.\, \mathsf{k}xy = x \qquad \mathsf{s}xyz = (xy)(xz)$$
$$\forall x, y.\, (\forall z.\, xz = yz) \supset x = y. \quad)$$

Models of lambda to models of ECA

Pick a, b, c distinct (so $a\#b$, $b\#c$, and $a\#c$ are all derivable). Set

$$\mathsf{s} \equiv \lambda abc.(ab)(ac)$$
$$\mathsf{k} \equiv \lambda ab.a$$

We can prove:

- $\bullet\mathsf{s}$ and $\bullet\mathsf{k}$.
- $\lambda abc.(ab)(ac) = \lambda a'b'c'.(a'b')(a'c')$, so s is canonical.
- Likewise $\lambda ab.a = \lambda a'b'.a'$.
- $\mathsf{s}pqr = (pq)(pr)$ and $\mathsf{k}pq = p$.
- $\forall p, q.\ (\forall r.\ pr = qr) \supset p = q$.

The reverse direction

Just take formal sentences over the extensional λ-model, quotiented by the axioms of lambda in a suitalbe (slightly subtle) sense. Details omitted.

ECA to lambda to ECA: we get back where we started.
lambda to ECA to lambda: we may lose some equalities on open terms.

For example, consider a model such that

$$\forall a, b.\ \mathbf{at}\, a \wedge \mathbf{at}\, b \supset (\lambda a.a)b = b$$

We lose this identity.

“The map from ECA to lambda throws away the internal meta-language”.

Some observations

- Unknown λ-terms are variables.
- Unknown λ-term variables are atoms.
- ‘Real’ λ-terms are provably closed.
- Explicit substitution acts also for open term and is part of the internal meta-language.
- Rules

$$
\begin{array}{ll}
(\beta') & \forall a, x, y.\, \mathbf{at}\, a \wedge \neg\mathbf{at}\, y \supset (\lambda a.x)y = x\langle a \mapsto y \rangle \\
(\xi') & \forall x, y.\, \neg\mathbf{at}\, x \wedge \neg\mathbf{at}\, y \supset (\forall z.\, \bullet z \supset xz = yz) \supset x = y
\end{array}
$$

are reasonable; they just equate more terms of the internal meta-language.

Essentially a term

This is wrong:

$$(\beta_{FALSE}) \quad \forall a, x, y.\ \mathbf{at}\, a \supset (\lambda a.x)y = x\langle a \mapsto y\rangle$$

Sneakily did not mention ‘essentially a term’.

In order to avoid contradiction from $\mathbf{at}\, a \supset a\langle a \mapsto a\rangle = a$, we must suspend $(\mathbf{at}\ L)$ for some term-formers.

See the paper.

Interesting solution

Instead of equality, use a ‘is more defined than’ relation $\rightsquigarrow$. Then $a\langle a{\mapsto}a\rangle \rightsquigarrow a$.

Have investigated this somewhat. Suggests an interesting class of models of the λ-calculus.

One advantage of this is that it enables us to express Hilbert’s arbitrary choice operator.

Sadly, did not make it into the paper.

Semantics

A semantics of a-logic is easy to construct via its FOL axiomatisation. However a direct semantics is also possible. Variables are first-class elements of the domain, and $\mathbf{at}$ is a predicate which is true at most on variables (though not necessarily on all of them).

Essentially the same idea is a Kripke structure of 'increasingly defined' valuations on variables. $\mathbf{at}\,x$ determines whether at the present world x has been assigned a value or not.

None of this made it into the paper!

Conclusions

a-logic is a first-order logic (axiomatisation) with interesting expressivity. It suggests a class of models where ‘generic elements’ (the atoms) form part of the underlying domain.

You can do with generic elements what you do with ‘provably closed’ elements; there is some flexibility to the strength of the admissible theory of equality between elements containing generic elements.

There are also some interesting logical limits to how far that equality can be pushed while avoiding inconsistency, and interesting thoughts about how we might overcome these, if we wish.

lambda is close to Beeson’s Otter2 theorem-prover. Beeson makes a case for Otter2 (meta-language FOL, object-language untyped λ-calculus) as a good theorem-proving environment. However Otter2 terms are hard-wired; lambda axioms can be programmed in almost any current theorem-prover.

Things to be done!