Logical Justification in Distributed Authorization

Andreas Blass

University of Michigan
Ann Arbor, MI 48109

ablass@umich.edu
and
Microsoft Research
Redmond, WA 98052

DKAL

DKAL, the Distributed Knowledge Authorization Language, is being developed at Microsoft Research.

Contributors to DKAL and related logical considerations include(d)

- Yuri Gurevich
- Itay Neeman
- Lev Beklemishev
- Michał Moskal
- Guido de Caso
- Nikolaj Bjørner
- Carlos Cotrini
- Yuri Savateev
- Arnab Roy
- Nikhil Swamy
- Juan Chen
- Jean-Baptiste Jeannin
- Roy D'Souza
- Artem Melentyev
- Sergio Mera
- me.

What is DKAL?

Consider a distributed system of independent, asynchronous, communicating agents, or *principals*.

Each has a database of information, called its *substrate*, which it can read and update.

Each also has an *infostrate*, consisting of information known to it, either

- initially, or
- as a result of communication, or
- as a result of deduction, or
- by reading the substrate, or
- as a matter of policy.

DKAL is a formalism in which to express the *policies* of principals.

Infons are items of information that can be known and communicated by principals.

General Assumptions

Communications are encrypted in a way that assures the recipients that the message is genuine and comes from the actual sender.

Each principal acts in discrete rounds, applying its policy (as of the beginning of the round), using its substrate and mailbox (as of the beginning of the round) to decide on

- information to be added to or deleted from its infostrate,
- updates to its substrate, and
- messages to be sent to other principals.

All these actions are carried out at the end of the round.

Infons and Justification

Infons look like formulas, built as in multi-sorted first-order logic (or a fragment) plus the "said" construction:

If p is a (term of type) principal and α is an infon, then

$$p \text{ said } \alpha$$

is an infon.

Currently, the first-order part of infon logic includes $\top$, $\bot$, $\wedge$, $\rightarrow$, $\forall$, and $\forall$ cannot be nested inside other operators.

When an infon is sent from one principal to another, it may be accompanied by a *justification*.

An infon of the form "p said α"
or "$\beta \rightarrow p$ said α"
can be justified by the signature of (the principal denoted by) p.

Other infons can be justified by deductions from these.

Examples

X: principal; Y: file; K string

upon request(Y) from X
    if $\big($boss said $(\forall Z : \text{file})\,\text{ok}(X, Z)\big)\wedge$
      asInfon key(Y, K)
    then say to X use (K).

Note the difference between

if α then send to P (me said β)

and

send to P ($\alpha \rightarrow$ me said β).

In the former, the sender checks whether (it knows) α.
In the latter, the recipient's knowledge of α is relevant.

Infon Logic

What logic should be used by principals?
DKAL is modular; one could insert any desired logic.
Different principals could use different logics privately.
For justified communication, one wants a shared logic to be used in the justifications.
What should that logic be?
Trade-off between power and efficiency.
Theoretically, intuitionistic or even classical first-order logic would be appropriate.
Efficiency dictates weaker fragments.
Currently allow only universal quantification, and only over whole infons, not nested within connectives.
Connectives are only $\top, \bot, \wedge, \rightarrow$.
And only part of the logic of $\rightarrow$ is available.
Disjunction and part of its logic are about to be added.

Primal Infon Logic

Modify the intuitionistic logic of $\top, \bot, \wedge, \rightarrow$ by replacing implication introduction

$$\frac{\Gamma, \alpha \vdash \beta}{\Gamma \vdash (\alpha \rightarrow \beta)}$$

with the weaker form

$$\frac{\Gamma \vdash \beta}{\Gamma \vdash (\alpha \rightarrow \beta).}$$

Also, extend all the rules to work inside "said" contexts.

The proposed partial introduction of disjunction would have the introduction rules

$$\frac{\alpha}{\alpha \vee \beta} \qquad \frac{\beta}{\alpha \vee \beta}$$

(also within "said" contexts) but not the elimination rule.

Reasons for the changes from intuitionistic rules:

- sufficient for (currently envisioned) applications
- efficient computability

Is there a mathematically neat reason?

Discharge Only Trivialities

A general pattern for making rules of inference feeble:
Wherever any deducibility is presupposed, require it to be trivial, in the sense that the deduced formula is simply one of the hypotheses.

An innocuous example, $\wedge$-introduction

$$\frac{\Gamma \vdash \alpha \qquad \Gamma \vdash \beta}{\Gamma \vdash (\alpha \wedge \beta).}$$

For the DOT version, we assume that α and β are not merely deducible from Γ but elements of Γ.
So the rule becomes

$$\Gamma, \alpha, \beta \vdash (\alpha \wedge \beta).$$

In view of structural rules, this could be written simply as

$$\alpha, \beta \vdash (\alpha \wedge \beta),$$

and it is as good as th original intrduciton rule.

More Unimportant Examples

The same process converts the $\wedge$-elimination rules

$$\frac{\Gamma \vdash (\alpha \wedge \beta)}{\Gamma \vdash \alpha} \quad \text{and} \quad \frac{\Gamma \vdash (\alpha \wedge \beta)}{\Gamma \vdash \beta}$$

to

$$(\alpha \wedge \beta) \vdash \alpha \quad \text{and} \quad (\alpha \wedge \beta) \vdash \beta,$$

which are equivalent to the original.

The (introduction) rule for $\top$ becomes the equivalent $\vdash \top$.

Similarly, the elimination rule for $\bot$, in the form

$$\frac{}{\Gamma, \bot \vdash \alpha}$$

becomes $\bot \vdash \alpha$.

And you get the same result if you start with the elimination rule in the form

$$\frac{\Gamma \vdash \bot}{\Gamma \vdash \alpha.}$$

Key Example 1

For implication, the elimination rule

$$\frac{\Gamma \vdash (\alpha \to \beta) \qquad \Gamma \vdash \alpha}{\Gamma \vdash \beta}$$

becomes the equivalent

$$(\alpha \to \beta), \alpha \vdash \beta.$$

But the introduction rule

$$\frac{\Gamma, \alpha \vdash \beta}{\Gamma \vdash (\alpha \to \beta)}$$

becomes much weaker. DOT requires that β be either an element of Γ or identical to α. So we get the two rules

$$\beta \vdash (\alpha \to \beta) \qquad \text{and} \qquad \vdash (\alpha \to \alpha).$$

The first of these is the $\to$-introduction rule for primal logic.

The second seems to be a harmless addition to primal logic.

Key Example 2

For disjunction, the usual introduction rules yield, under DOT,

$$\alpha \vdash (\alpha \vee \beta) \qquad \text{and} \qquad \beta \vdash (\alpha \vee \beta),$$

which are equivalent to the usual rules. The elimination rule, in either the form

$$\frac{\Gamma, \alpha \vdash \theta \qquad \Gamma, \beta \vdash \theta}{\Gamma, (\alpha \vee \beta) \vdash \theta}$$

or the form

$$\frac{\Gamma, \alpha \vdash \theta \qquad \Gamma, \beta \vdash \theta \qquad \Gamma \vdash (\alpha \vee \beta)}{\Gamma \vdash \theta,}$$

yields the far weaker

$$(\alpha \vee \alpha) \vdash \alpha.$$

Quantifiers

The DOT process can be applied also to the usual quantifier rules.

The results are equivalent to the usual $\exists$-introduction and $\forall$-elimination rules. We get

$$\varphi(t) \vdash \exists x\, \varphi(x) \qquad \text{and} \qquad \forall x\, \varphi(x) \vdash \varphi(t)$$

under the usual requirement of substitutability of t for x in φ.

But for the other two rules, we get only the far weaker

$$\exists x\, \psi \vdash \psi \qquad \text{and} \qquad \psi \vdash \forall x\, \psi$$

when x is not free in ψ.

That is, $\exists$-elimination and $\forall$-introduction work only for dummy quantifiers.

Another Appearance of DOT

Jan von Plato's "general elimination rules" for $\wedge, \to, \forall$ are connected to the usual rules by DOT.