Consistency and Completeness of OMEGA, a Logic for Knowledge Representation

Giuseppe li and Maria Simi

Massachusetts Institute of Technology
545 Technology Square
Cambridge, Mass. 02139

Istituto di Scierue deirinforrnqzione
Corso Italia 40
1-56100 Pisa, Italy

Abstract

Omega is a description system for knowledge embedding which incorporates some of the attractive modes of expression in natural language such as descriptions, inheritance, quantification, negation, attributions and multiple viewpoints. It combines mechanisms of the predicate calculus, type systems, and pattern matching systems. *Description abstraction* is a construct which we introduce to extend the expressive power of the system. The logic foundations for the basic constructs of Omega are investigated. Semantic models are provided, an axiomatization is derived and the consistency and completeness of the

I Introduction

Omega is a description system developed for knowledge representation and reasoning (10). Omega is a *calculus of Ascriptions* rather then a ceteufua of *predicetes* as ordinary logic. The concept of description in logic, and in particular of *definite descriptions,* can be traced back to the works by Frege and by Russell (15]. Logicians have always been bothered by the semantic problems raised by definite descriptions when none or more then one individual meets the description. Therefore they have favored their elimination by showing how descriptions can be contextually replaced by means of other constructs (16).

In Omega we deal with *indefinite descriptions* such as the description "(eMan)". Omega then, has more the flavor of a naive set theory. Omega however aNows many ways to build new objects, rather than the single set formation construct of set theory. Omega is a type free system, in the sense that a single logical type is admitted, the type of descriptions. With Omega we achieve the goal of an intuitively sound and consistent theory of classes which permits unrestricted abstraction within a powerful logic system. *Description abstraction* is a construct provided in Omega similar to set abstraction. Abstractions add considerable expresaive power to a language, nevertheless such powerful constructs are likely to lead to inconsistencies or paradoxes. The rules for abstraction we present have been studied to avoid those problems. The proof of consistency that is provided in the paper is therefore significant to establish this claim.

The main goal of this paper is to present the logic theory Omega, to introduce its models, derive an axiomatization and finally to show the consistency and completeness of the system.

We consider the study of s knowledge representation system as a logic system to be of fundamental importance. In thie way we isolate the basic deductive mechanisms from the intricacies of specific programming languages or implementations. The fundamental results obtained in this way can help us understand the basic mechanisms of reasoning on a knowledge base, and insure us of the soundness of the system.

As an alternative to symbolic logic, many knowledge representation systems use *semantic networks* (14) as their basic formalism. The term "semantic" in the name "semantic network" refers to the fact that the formalism was originally developed to represent the semantic information necessary to understand natural language. Despite their name, the semantics of semantics network has *never* been satisfactorily developed. In the work by Brachman [4]. the ambiguities and inadequacies of semantics networks are investigated. A typical problem deals with the meaning and use of inheritance links.

In most cases the semantics of semantic networks is determined by the way the processing routines of application programs manipulate the information represented by nodes and links of the network. Many authors consider semantic networks as embedded into first order predicate calculus (9,18], so the reasoning mechanisms are directly drawn from predicate logic. Semantic networks can then be considered a convenient notation (i.e. an alternative *syntax) tor* expressing variable-free assertions of predicate calculus. The notation is convenient because it suggests an organization of the data base of assertions, highly suitable for mechanical manipulation. Kowalski in [7] adopts this point of view, and also proposes an extension to the notation of semantic networks in order to express a more general class of formulas, including universally quantified variables. The class he suggests is the class of *Horn clauses,* i.e. formulas of the kind:

A i A A j - A n - e - A ^ ,

where all A, are positive atomic formulas. For this class of formulas a uniform proof procedure based on resolution performs reasonably well. However the expressive power is still limited, mostly because of the restriction in the use of negation.

By drawing such connections between semantic networks and predicate logic, one can argue that a semantics has been provided for semantic networks. However, many of the fundamental aspects of semantic networks, such as inheritance, remain unexplained.

Other knowledge representation systems provide deductive mechanisms of their own. most often procedurally defined [8], [3], [5], (19). Such deductive mechanisms are not formally investigated, so their logical soundness can be questioned. It is in fact the case that some of these systems perform unwanted or incorrect deductions. An example of this kind of problems is the so-called "copy confusion" effect reported by Fehlman about his NETL system.

In comparison with other knowledge representation systems, Omega has increased expressive power deriving from including:

- *variables:* this allows description of complex general relations, rather then isolated assertions;
- *quantification:* both over individuals and classes (descriptions):

This resaerch *was* supported in part *through a* grant tamOftvt* to t* MMcttinitSlatnQt Laboratory of MIT.

Tdt second author hat boon twpportod in part ay a talowanto of tha Cowlato Ntfonato -»-■«-««.. — . . .

- *negation:* this removes the major limitation of systems like PROLOG, based on the use of Horn clauses;

- *abstraction:* allows description of classes of individuals in terms of their properties. It is a very powerful construct, which allows for instance to define the concept of converse of a relation, that had to be given as primitive in [10].

We believe that recent proposals of parallel problem solving systems such as Ether [12] will be suitable for implementing the reasoning mechanisms demanded by a rich system such as Omega. For instance the ability to process concurrently *proponents* and *opponents* [12] of a same goal, is what is needed to appropriately deal with sentences containing negations. Conversely, it is the provision of negative facts what makes the proponents/opponents metaphor profitable and effective. So far the limitations on the use of negation seem to have been dictated mainly by considerations related to sequential proof algorithms.

In this paper we present and develop the description system Omega, as a logic system. The system we develop has some similarities with the one presented by R. Martin in [13], even though we started with different aims and motivations. Martin's system is proposed as a system of mathematical foundation, as an alternative to the classical theory of sets. However no result of completeness or consistency for that logic is presented.

A sound logical theory for Omega is necessary as a formal concise specification for the algorithms that will carry out the reasoning process. Results such as the correctness or the completeness of such algorithms can be established only with respect to the theory. Properties and theorems about the theory can be exploited in the design of the proof procedure, as it was the case for the resolution algorithm in First Order Predicate Logic.

We are leaving out from the present discussion other features of Omega such as the calculus of attributions and of metadescriptions.

A version of Omega has been implemented on the MIT LISP machine by the authors and has been used to describe the base of active knowledge supporting an experimental system of office forms [1]. A second implementation of Omega is currently under development by Jerry Barber. A subset of Omega is being used to describe two dimensional objects within the SBA system by Peter de Jong [6].

Our experience has proven that the axiomatization provides an extremely useful guideline in the process of implementation.

II Descriptions and Predications

This section is intended as an informal introduction to the theory Omega. For more exhaustive presentation and examples we refer to [10].

The simplest kind of description is the individual description, like:

Boston

or

Paul

Here the names Boston and Paul are names describing individual entities.

An *instance description* is a way to describe a collection of individuals. For instance

(•Man)

represents the collection of individuals in the class of men.

The most elementary sentence in Omega is a predication. A predication relates a *subject* to a *predicate* by the relation *is.* For instance the predication

Paul *is* (a Man)

is meant to assert that the individual named Paul belongs to the class of man. Predication can be used to relate arbitrary deecriptione. For instance tha sentence:

(a Man) it (a Mortal)

states the fact that any Individual of claaa man is also an individual of class mortal. A fundamental property of the relation Ia is rransWvrty, that allows tor instance to conclude that

Socrates/* (a Mortal)

from

Socrates /a(a Man) and (a Man) *1$* (a Mortal)

The description operators **nd%* or and *not* allow us to build mora complex descriptions, like in the following example:

(a Boolean) *Is (true or M$o)*

More complex statements can be built by combining statements with the logical connectives $\wedge$, $\vee$, $\neg$ and $\Rightarrow$, as in:

Jean *is* ((*a* Man) *or* (*a* Woman)) $\wedge$ $\neg$ (Jean *is* (*a* Man))
$\Rightarrow$ Jean *is* (*a* Woman)

Note the difference between description operators and statement connectives in the following examples:

(*true* $\wedge$ *false*) *is false*
(*true and false*) *is Nothing*

where *Nothing* is our notation for "the null entity".

III Syntax

The language of the theory Omega is presented hare using the kind of notation which has become standard in logic and denotational semantics. We list first the syntactic categories of the language. For each of them we show our choice of metavariables ranging on the elements of that category, that we will uae in the rest of the expoaition.

I: individual constants ($I, I_1, I_2, I_3 \ldots$), including the constants *true* and *false*

V: variables ($v, v_1, v_2, v_3 \ldots$)

C: class constants ($c, c_1, c_2, c_3 \ldots$)

S: statements ($\sigma, \sigma_1, \sigma_2, \sigma_3 \ldots$)

Δ: descriptions ($\delta, \delta_1, \delta_2, \delta_3 \ldots$)

Descriptions and statements are built from constants and variables according to the following syntax:

Descriptions	Statements
I, *Nothing*, *Something*	*true*, *false*
v	v
(a c)	(δ_1 *is* δ_2)
(*Any* v *such that* σ)	$\forall v \,.\, \sigma$
(δ_1 *or* δ_2)	($\sigma_1 \vee \sigma_2$)
(δ_1 *and* δ_2)	($\sigma_1 \wedge \sigma_2$)
(*not* δ)	($\neg \sigma$)
σ	($\sigma_1 \Rightarrow \sigma_2$)

Nothing and *Something* are two special constants.

IV Semantics

We have followed in this work the method of defining validity in a semantic way. First we characterize a dees of models Ior Omega and define the notion of truth in a model. This gives an intuitive and immediate semantic interpretation for our theory. Then we look for en axiomatization for valid formulas of the theory. The models defined earlier ere useful in this stage for providing a guideline and a criterion for the suitability of the axiomatization.

By proving the completeness theorem, we show that the axiomatization is adequate. Furthermore the completeness result telle that our theory is consistent if end only if it has a model. Therefore the existence of models presented in this section implies the consistency of Omega.

The structure of the interpretation is

$$\mathcal{A} = \langle D, \mathfrak{I}, C \rangle$$

where 0 Is the domain of interpretation, a nonempty set of individuals, 3 is a mapping from individual constants into elements of D

$$\mathfrak{I} : I \rightarrow D$$

and Cieamapping from class constants into subsete of D

$$C : C \rightarrow 2^D$$

A. Definition of Value of a Description

The value of a description can be defined as a mapping from descriptions *and* environments into subsets of P.

$$\Upsilon : \Delta \rightarrow E \rightarrow 2^D$$

An environment defines an association between variables end subsets of D.

$$E : V \rightarrow 2^D$$

1. Value of constant descriptions:

a. $\Upsilon[\![i]\!]\rho = \{\mathfrak{I}[\![i]\!]\}$

b. $\Upsilon[\![\mathit{Nothing}]\!]\rho = \{\}$

c. $\Upsilon[\![\mathit{Something}]\!]\rho = D$

2. $\Upsilon[\![v]\!]\rho = \rho(v)$

3. $\Upsilon[\![(a\ c)]\!]\rho = C[\![c]\!]\rho$

4. $\Upsilon[\![(\mathit{Any}\ v\ \mathit{such\ that}\ \sigma)]\!]\rho = \{x \in D \mid \models_{\mathcal{A}} [\![\sigma]\!]\rho[\{x\}/v]\}$

5. $\Upsilon[\![(\delta_1\ \mathit{or}\ \delta_2)]\!]\rho = \Upsilon[\![\delta_1]\!]\rho \cup \Upsilon[\![\delta_2]\!]\rho$

6. $\Upsilon[\![(\delta_1\ \mathit{and}\ \delta_2)]\!]\rho = \Upsilon[\![\delta_1]\!]\rho \cap \Upsilon[\![\delta_2]\!]\rho$

7. $\Upsilon[\![(\mathit{not}\ \delta)]\!]\rho = D - \Upsilon[\![\delta]\!]\rho$

8. $\Upsilon[\![\sigma]\!]\rho = \Upsilon[\![\mathit{true}]\!]\rho$ if $\models_{\mathcal{A}} [\![\sigma]\!]\rho$, $\Upsilon[\![\mathit{false}]\!]\rho$ if $\not\models_{\mathcal{A}} [\![\sigma]\!]\rho$

The interpretation of the description abstraction Any Is the set of individuals which, substituted for v in the statement a, make the statement true. The notation

$$\models_{\mathcal{A}} [\![\sigma]\!]\rho$$

means the! the statement a in the environment p is true relative to the structure *JL,* and it is completely defined in the next section.

B. Definition of Truth Valve

1. $\models_{\mathcal{A}} [\![\mathit{true}]\!]\rho$, $\models_{\mathcal{A}} [\![\neg \mathit{false}]\!]\rho$

2. Variables

a. $\models_{\mathcal{A}} [\![v]\!]\rho$ iff $\rho(v) = \Upsilon[\![\mathit{true}]\!]\rho$

b. $\models_{\mathcal{A}} [\![\neg v]\!]\rho$ iff $\rho(v) = \Upsilon[\![\mathit{false}]\!]\rho$

3. $\models_{\mathcal{A}} [\![(\delta_1\ \mathit{is}\ \delta_2)]\!]\rho$ iff $\Upsilon[\![\delta_1]\!]\rho \subseteq \Upsilon[\![\delta_2]\!]\rho$

4. $\models_{\mathcal{A}} [\![(\forall v\ .\ \sigma)]\!]\rho$ iff for all $x \in 2^D$, $\models_{\mathcal{A}} [\![\sigma]\!]\rho[x/v]$*

5. $\models_{\mathcal{A}} [\![(\sigma_1 \vee \sigma_2)]\!]\rho$ iff $\models_{\mathcal{A}} [\![\sigma_1]\!]\rho$ or $\models_{\mathcal{A}} [\![\sigma_2]\!]\rho$

6. $\models_{\mathcal{A}} [\![(\sigma_1 \wedge \sigma_2)]\!]\rho$ iff $\models_{\mathcal{A}} [\![\sigma_1]\!]\rho$ and $\models_{\mathcal{A}} [\![\sigma_2]\!]\rho$

7. $\models_{\mathcal{A}} [\![\neg \sigma]\!]\rho$ iff not $\models_{\mathcal{A}} [\![\sigma]\!]\rho$

8. $\models_{\mathcal{A}} [\![(\sigma_1 \Rightarrow \sigma_2)]\!]\rho$ iff $\models_{\mathcal{A}} [\![\sigma_1]\!]\rho$ implies $\models_{\mathcal{A}} [\![\sigma_2]\!]\rho$

The truth of a statement relative to a structure $\mathcal{A}$ is defined as:

Definition 1: (Truth) $\models_{\mathcal{A}} \sigma$ iff $\forall \rho \models_{\mathcal{A}} [\![\sigma]\!]\rho$

Given the definition of truth, validity is defined as follows:

Definition 2: (Validity) $\models \sigma$ iff $\forall M \models_M \sigma$

V Axiomatization

The method that we followed in deriving the axiomatization, is based on noting rules that preserve validity of formulae and making them into axioms or inference rules.

A. Axiom Schemata

We present the axioms of the system in the form of natural deduction inference rules [11]. As a matter of notation we will use a double bar to indicate that a rule can be applied in both directions. The bar is omitted if the set of premises is empty.

Axioms for Statements

S1: true

S2: $\neg\ \mathit{false}$

S3: $\sigma \Rightarrow \mathit{true}$

S4: $\mathit{false} \Rightarrow \sigma$

S5: $\mathit{true} \Rightarrow \sigma \vee \neg \sigma$

S6: $\sigma \wedge \neg \sigma \Rightarrow \mathit{false}$

S7: $$\frac{\sigma_1 \Rightarrow \sigma_2,\ \sigma_1 \Rightarrow \sigma_3}{\sigma_1 \Rightarrow \sigma_2 \wedge \sigma_3}$$

S8: $$\frac{\sigma_1 \Rightarrow \sigma_3,\ \sigma_2 \Rightarrow \sigma_3}{\sigma_1 \vee \sigma_2 \Rightarrow \sigma_3}$$

S9: $$\frac{\sigma_1 \Rightarrow \sigma_2}{\neg\sigma_2 \Rightarrow \neg\sigma_1}$$

S10: $\sigma \Rightarrow \neg(\neg\sigma)$

S11: $\neg(\neg\sigma) \Rightarrow \sigma$

S12: $\sigma \Rightarrow \sigma\ \mathit{is\ true}$

S13: $\neg\sigma \Rightarrow \sigma\ \mathit{is\ false}$

---

*$\rho[x/v]$ is the same environment as ρ but for variable v to which it associates x.

Note that this axiom set ia not minimal, since for instance S1 -82.83-S4 and"S6S6 are pairwiae derivable from one another. (A similar remark applies to the axioms for descriptions presented below).

The notion of "sameness" between descriptions is defined as follows:

Definition 3: (Sameness)

$(\delta_1\ \textit{same}\ \delta_2)$ iff $((\delta_1\ \textit{is}\ \delta_2) \wedge (\delta_2\ \textit{is}\ \delta_1))$

We will also need the following abbreviation:

Definition 4: (Individuality) **Individual**$[\delta]$ stands for $\neg(\delta\ \textit{is Nothing}) \wedge \forall v\,.\,(v\ \textit{is}\ \delta) \Rightarrow (\delta\ \textit{is}\ v) \vee (v\ \textit{is Nothing})$

Axioms for Descriptions

D1: $\delta_1\ \textit{is}\ \delta_2 \Leftrightarrow \forall v.\ \text{Individual}[v] \wedge v\ \textit{is}\ \delta_1 \Rightarrow v\ \textit{is}\ \delta_2$

D2: $\text{Individual}[I]$

D3: $\delta\ \textit{is Something}$

D4: $\textit{Nothing is}\ \delta$

D5: $\textit{Something is}\ (\delta\ \textit{or}\ (\textit{not}\ \delta))$

D6: $(\delta\ \textit{and}\ (\textit{not}\ \delta))\ \textit{is Nothing}$

D7: $$\frac{\delta_1\ \textit{is}\ \delta_2,\ \delta_1\ \textit{is}\ \delta_3}{\delta_1\ \textit{is}\ (\delta_2\ \textit{and}\ \delta_3)}$$

D8: $$\frac{\delta_1\ \textit{is}\ \delta_3,\ \delta_2\ \textit{is}\ \delta_3}{(\delta_1\ \textit{or}\ \delta_2)\ \textit{is}\ \delta_3}$$

D9: $$\frac{\delta_1\ \textit{is}\ \delta_2}{\textit{not}\ \delta_2\ \textit{is not}\ \delta_1}$$

D10: $\textit{not}\ (\textit{not}\ \delta)\ \textit{is}\ \delta$

D11: $\delta\ \textit{is not}\ (\textit{not}\ \delta)$

The notion of individual, as it ia formally defined, correeportde in our interpretation to the aet consisting of a single element, or singleton. Axiom D1 then corresponds to the following version of the axiom of extentionality for sets:

$$A \subseteq B \Leftrightarrow \forall x \in A\,.\,\{x\} \subseteq A \Rightarrow \{x\} \subseteq B$$

Axiom D2 states the fact that individual constants are individuals.

We have choaen thia aet of axioms so that there ia an almost complete symmetry between the axioms for statements and the axioms for descriptions. In thia way any theorem about descriptions has a corresponding dual theorem about statements. A single proof procedure will work for both descriptions and statements.

There ia another strong correspondence between statements and descriptions, which ia expressed in the following

Lemma S:

1. *Nothing —am (Any* v *such ttmt* flefae)

2. *Something* feme *(Any* v such *that try)*

3- (61 or *ty —nr (Any* v eucJi *thmt* (v fa «,) V (v Ia <*))

4. $(\delta_1\ \textit{and}\ \delta_2)\ \textit{same}\ (\textit{Any}\ v\ \textit{such that}\ (v\ \textit{is}\ \delta_1) \wedge (v\ \textit{is}\ \delta_2))$

5. $(\textit{not}\ \delta)\ \textit{same}\ (\textit{Any}\ v\ \textit{such that}\ \neg\ (v\ \textit{is}\ \delta))$

Actually these statements could have as well been given as an alternative axiomatization for description operators. We prefer however the set of axioms presented above since they do not involve explicitly description abstraction. The inference rules for description abstraction are rather complicated. With the axioms we have chosen, the use of abstraction rules and extentionality (axiom D1) can often be avoided.

B. Inference Rules

It is easy to verify that these two *transitivity* rules preserve the truth according to our interpretation:

$$\frac{\sigma_1 \Rightarrow \sigma_2,\ \sigma_2 \Rightarrow \sigma_3}{\sigma_1 \Rightarrow \sigma_3} \qquad \frac{\delta_1\ \textit{is}\ \delta_2,\ \delta_2\ \textit{is}\ \delta_3}{\delta_1\ \textit{is}\ \delta_3}$$

The ordinary rule of Modus Ponens can be derived from transitivity by letting σ_1 be *true*.

Standard rules can be used for $\Rightarrow$ introduction, $\forall$ introduction (generalization) and $\forall$ elimination (instantiation).

The rules for description abstraction require more careful examination. The most obvious formulation of the abstraction rule for descriptions would be:

$$\frac{\sigma[\delta]}{\delta\ \textit{is}\ (\textit{Any}\ v\ \textit{such that}\ \sigma[v])} \qquad \textbf{A1}$$

and the corresponding concretion rule:

$$\frac{\delta\ \textit{is}\ (\textit{Any}\ v\ \textit{such that}\ \sigma[v])}{\sigma[\delta]} \qquad \textbf{C1}$$

Unfortunately, these unrestricted forms for the abstraction and concretion rules lead to contradiction. Consider for example

John *is* (*a* **Man**)

With (*a* **Man**) for δ, by A1, we get:

(*a* **Man**) *is* (*Any* v *such that* (**John** *is* v))

but if also

Paul *is* (*a* **Man**)

then by transitivity:

Paul is (*Any* v *such that* (**John** *is* v))

and by C1:

(**John** *is* **Paul**)

which might not be true.

A restricted form of the abstraction principle that avoids such problems is the following:

$$\frac{\sigma[\delta] \wedge \text{Individual}[\delta]}{\delta\ \textit{is}\ (\textit{Any}\ v\ \textit{such that}\ \sigma[v])} \qquad \textbf{A2}$$

Similar problems arise with an unrestricted concretion rule. Let $\sigma[v] = \text{Individual}[v]$ and suppose for instance:

Individual[John]
Individual[Paul]

then by A2

John *is* (*Any* v *such that* Individual[v])
Paul *is* (*Any* v *such that* Individual[v])

From the axiom D6, which introduces the *or* description, we get:

(John *or* Paul) *is* (*Any* v *such that* Individual[v])

and from this and C1:

Individual[John *or* Paul]

We solve this problem by restricting C1 analogously to what we did for A1:

$$\frac{\delta \textit{ is } (\textit{Any } v \textit{ such that } \sigma[v]) \wedge \text{Individual}[\delta]}{\sigma[\delta]} \qquad \text{C2}$$

With this restriction to individuals, it is easy to verify that the abstraction and concretion rules are sound.

Note that with this form of the concretion principle Russell's paradox is avoided. Suppose we call z the following description:

$z \equiv$ (*Any* v *such that* $\neg$ (v *is* v))

If we allow C1, since z *is* z is true by reflexivity of *is*, we would derive the paradoxical consequence:

$\neg$(z *is* z)

We can show however that:

z *same Nothing*

because

(*Any* v *such that* $\neg$ (v *is* v)) *same* (*Any* v *such that false*) *same Nothing*

Therefore rule C2 cannot be applied because *Nothing* is not an individual.

The complete set of inference rules is summarized in the following table.

Inference Rules

Statements

$$\frac{\begin{matrix}[\sigma_1] \\ \vdots \\ \sigma_2\end{matrix}}{\sigma_1 \Rightarrow \sigma_2}$$

$$\frac{\sigma_1 \Rightarrow \sigma_2,\ \sigma_2 \Rightarrow \sigma_3}{\sigma_1 \Rightarrow \sigma_3}$$

$$\frac{\sigma}{\forall v . \sigma}$$

provided v is not free in any assumption on which σ depends

$$\frac{\forall v . \sigma[v]}{\sigma[\delta]}$$

Descriptions

$$\frac{\delta_1 \textit{ is } \delta_2,\ \delta_2 \textit{ is } \delta_3}{\delta_1 \textit{ is } \delta_3}$$

$$\frac{\sigma[\delta] \wedge \text{Individual}[\delta]}{\delta \textit{ is } (\textit{Any } v \textit{ such that } \sigma[v])}$$

$$\frac{\delta \textit{ is } (\textit{Any } v \textit{ such that } \sigma[v]) \wedge \text{Individual}[\delta]}{\sigma[\delta]}$$

As it can be seen from the formulation of the rules of abstraction and concretion, Omega cannot be finitely axiomatized in first order logic.

VI Soundness and Completeness

The first result to be proved about the axiomatization is its soundness:

Theorem 6: (Soundness)

For every closed statement A, $\Gamma \vdash A \Rightarrow \Gamma \models A$

We are using the notation $\Gamma \models A$ as an abbreviation for:

$\forall M . (\forall \sigma . \sigma \in \Gamma \wedge \models_M \sigma) \Rightarrow \models_M A$

The derivability relation $\vdash$ is defined in a standard way, so that $\Gamma \vdash \sigma$ means that the statement σ can be derived from the statements in Γ and from the axioms by applying the rules of inference.

The soundness theorem states that whenever a statement can be derived from the set of premises Γ then it is true in every model that satisfies the statements in Γ. We omit here the proof since the argumentation is straightforward.

We next turn to prove the completeness theorem for the theory presented above.

The completeness theorem gives us a measure of the adequacy of our axiomatization. In fact it asserts that the set of valid formulas coincides with the set of theorems of our theory. So it provides a bridge between what is established as semantically valid, and what is established by symbolic manipulations.

More important, this result is the fundamental step in showing the consistency of our formal theory.

The completeness theorem can be formulated as follows:

Theorem 7: (Completeness)

For every closed statement A, $\Gamma \models A \Rightarrow \Gamma \vdash A$

Proof: We follow a Goedel-Henkin argument [17]. The implication $\Gamma \vdash A \Rightarrow \Gamma \models A$ corresponds to the soundness theorem. The other direction of the implication is equivalent to saying:

$\Gamma \nvdash A \Rightarrow \exists M\ \Gamma \not\models_M A$

If A cannot be derived from Γ then there exists a model that satisfies Γ but not A. So we can prove the result by building a model $\mathcal{M}$, possibly in dependence from A, such that:

$\Gamma \nvdash A \Rightarrow \Gamma \not\models_{\mathcal{M}} A$

In order to do that, for any given closed formula A, we build a complete Henkin extension of theory Γ, called Γ_A, having the following additional properties:

$\Gamma \nvdash A \Rightarrow \Gamma_A \nvdash A$ (1)

$\Gamma_A \vdash (\sigma_1 \vee \sigma_2) \Rightarrow \Gamma_A \vdash \sigma_1$ or $\Gamma_A \vdash \sigma_2$ (2)

Since Γ_A is a complete Henkin extension of Γ, it also satisfies the following properties:

$\Gamma \subseteq \Gamma_A$ (3)

For all closed σ either $\Gamma_A \vdash \sigma$ or $\Gamma_A \vdash \neg \sigma$ (4)

For every statement $\sigma[v]$, there is a constant c, such that: (5)
$\Gamma_A \vdash \sigma[c] \Rightarrow \forall v . \sigma[v]$

Γ_A is built as the limit of an inductive sequence of theories, starting from Γ. For the details of the construction we refer to [2].

Next we prove the following lemma in order to prove that Γ_A has a model:

Lemma 8: (Main lemma) $\exists M . \forall \sigma . \Gamma_A \vdash \sigma \Leftrightarrow \Gamma_A \models_M \sigma$

The proof is outlined in the next section. Let $\mathcal{M}$ be such a model for Γ_A. Assuming now that $\Gamma \not\vdash A$, it follows that $\Gamma_A \not\vdash A$ and then $\Gamma_A \not\models_{\mathcal{M}} A$

Since a model of Γ_A is also a model of Γ, we have proved that:

$$\Gamma \not\vdash A \Rightarrow \exists M . \Gamma \not\models_M A$$

and the completeness theorem.

A. Proof of Main Lemma

We will prove the lemma by building a model $\mathcal{M}$ such that:

$$\forall \sigma\, \Gamma_A \vdash \sigma \Leftrightarrow \Gamma_A \models_{\mathcal{M}} \sigma$$

$\mathcal{M}$ will be a term model built out of syntactic material. More precisely equivalence classes of individual descriptions will be the elements of the domain of interpretation. Let us define the equivalence relation ~ as:

$$\delta \sim \delta' \equiv \Gamma_A \vdash (\delta \textit{ same } \delta')$$

We will denote as $|\delta|$ the equivalence class of δ according to ~.

We will call $\text{Ind} = \{\delta \in \Delta \mid \Gamma \vdash \text{individual}[\delta]\}$. The domain of interpretation is the quotient of the set Ind with respect to the relation ~.

The model $\mathcal{M}$ is defined as: $\mathcal{M} = \langle \text{Ind}/\sim, \mathcal{J}, \mathcal{C} \rangle$ where

$$\mathcal{J}(i) = |i|$$
$$\mathcal{C}(c) = \{|\delta| \mid \delta \in \text{Ind and } \Gamma_A \vdash (\delta \textit{ is } (a\, c))\}$$

This model has the following significant property:

$$\Upsilon[\![\delta]\!] = \{|\delta'| \mid \delta' \in \text{Ind and } \Gamma_A \vdash (\delta' \textit{ is } \delta)\}$$

This property establishes the connection between the semantics (value of descriptions) and the syntax (derivability of predications). Because Omega allows statements as special cases of descriptions, this property is all is needed to establish the main lemma, as the following result shows:

Lemma 9: If $\Upsilon[\![\sigma]\!] = \{|\delta'| \mid \delta' \in \text{Ind and } \Gamma_A \vdash (\delta' \textit{ is } \sigma)\}$ then $\Gamma_A \vdash \sigma \Leftrightarrow \Gamma_A \models_{\mathcal{M}} \sigma$

Proof: Assume first that $\Gamma_A \vdash \sigma$. By axiom S12, it is also $\Gamma_A \vdash (\sigma \textit{ is true})$. Using this fact and transitivity in the premise of the lemma, we have

$$\Upsilon[\![\sigma]\!] \subseteq \{|\delta'| \mid \delta' \in \text{Ind and } \Gamma_A \vdash (\delta' \textit{ is true})\} = \{|\textit{true}|\}$$

But by the definition of Υ:

$$\Upsilon[\![\sigma]\!] = \{|\textit{true}|\} \text{ if and only if } \Gamma_A \models_{\mathcal{M}} \sigma$$

On the other hand, if we assume that $\Gamma_A \models_{\mathcal{M}} \sigma$ then

$$\Upsilon[\![\sigma]\!] = \{|\textit{true}|\} = \{|\delta'| \mid \delta' \in \text{Ind and } \Gamma_A \vdash (\delta' \textit{ is } \sigma)\}$$

This means that $\Gamma_A \vdash (\textit{true is } \sigma)$. Since Γ_A is complete, then either $\Gamma_A \vdash \sigma$ or $\Gamma_A \vdash \neg \sigma$. But the latter case is impossible because, by applying axiom S13, we would get $\Gamma_A \vdash (\sigma \textit{ is false})$, and by transitivity $\Gamma_A \vdash (\textit{true is false})$, which contradicts the consistency assumption for Γ_A.

Lemma 10: $\Upsilon[\![\delta]\!] = \{|\delta'| \mid \delta' \in \text{Ind and } \Gamma_A \vdash (\delta' \textit{ is } \delta)\}$

Proof: The proof is done by induction on the structure of descriptions. We give the flavor of the proof presenting a few of the cases.

1. $\delta \equiv i$.

$$\Upsilon[\![i]\!] = \{|i|\} = \{|\delta'| \mid \delta' \in \text{Ind and } \Gamma_A \vdash (\delta' \textit{ is } i)\}$$

since $i \in \text{Ind}$ by axiom D2. Note that this case takes care also of the individual constants *true* and *false*.

2. $\delta \equiv (\delta_1 \textit{ or } \delta_2)$.

$\Upsilon[\![\delta_1 \textit{ or } \delta_2]\!] = \Upsilon[\![\delta_1]\!] \cup \Upsilon[\![\delta_2]\!] =$ (by induction hypothesis)
$\{|\delta| \mid \delta \in \text{Ind and } \Gamma_A \vdash (\delta \textit{ is } \delta_1)\} \cup$
$\quad \{|\delta| \mid \delta \in \text{Ind and } \Gamma_A \vdash (\delta \textit{ is } \delta_2)\} =$
$\{|\delta| \mid \delta \in \text{Ind and } \Gamma_A \vdash (\delta \textit{ is } \delta_1) \text{ or } \Gamma_A \vdash (\delta \textit{ is } \delta_2)\} =$ (by prop. 2)
$\{|\delta| \mid \delta \in \text{Ind and } \Gamma_A \vdash (\delta \textit{ is } \delta_1) \vee (\delta \textit{ is } \delta_2)\} =$ (by Lemma 6.6)
$\{|\delta| \mid \delta \in \text{Ind and } \Gamma_A \vdash (\delta \textit{ is } (\delta_1 \textit{ or } \delta_2))\}$

3. $\sigma \equiv (\delta_1 \textit{ is } \delta_2)$.

$\Upsilon[\![\delta_1]\!] \subseteq \Upsilon[\![\delta_2]\!]$ (Assumption)
$\{|\delta| \mid \delta \in \text{Ind and } \Gamma_A \vdash (\delta \textit{ is } \delta_1)\} \subseteq$
$\quad \{|\delta| \mid \delta \in \text{Ind and } \Gamma_A \vdash (\delta \textit{ is } \delta_2)\}$ (by ind. hyp.)
$\delta \in \text{Ind and } \Gamma_A \vdash \delta' \textit{ is } \delta_1 \Rightarrow \Gamma_A \vdash \delta' \textit{ is } \delta_2$
$\Gamma_A \vdash \delta_1 \textit{ is } \delta_2$ (by axiom D1)
$\Gamma_A \vdash (\delta_1 \textit{ is } \delta_2) \textit{ is true}$ (by axiom S12)
$\{|\delta| \mid \delta \in \text{Ind and } \Gamma_A \vdash (\delta \textit{ is } (\delta_1 \textit{ is } \delta_2))\} \subseteq$
$\quad \{|\delta| \mid \delta \in \text{Ind and } \Gamma_A \vdash (\delta \textit{ is true})\}$ (by transitivity)
$\{|\delta| \mid \delta \in \text{Ind and } \Gamma_A \vdash (\delta \textit{ is true})\} = \{|\textit{true}|\} =$
$\Upsilon[\![\delta_1 \textit{ is } \delta_2]\!]$ (by definition of Υ).
The proof is similar under the hypothesis that: $\neg\, \Upsilon[\![\delta_1]\!] \subseteq \Upsilon[\![\delta_2]\!]$

4. $\sigma \equiv (\sigma_1 \vee \sigma_2)$.

$\Gamma_A \models_{\mathcal{M}} \sigma_1 \text{ or } \Gamma_A \models_{\mathcal{M}} \sigma_2$ (Assumption)
$\Gamma_A \vdash \sigma_1 \text{ or } \Gamma_A \vdash \sigma_2$ (by ind. hyp. and lemma 9)
$\Gamma_A \vdash (\sigma_1 \vee \sigma_2)$ (by property 3 of Γ_A)
$\Gamma_A \vdash (\sigma_1 \vee \sigma_2) \textit{ is true}$ (by axiom S12)
$\{|\delta| \mid \delta \in \text{Ind and } \Gamma_A \vdash (\delta \textit{ is } (\sigma_1 \vee \sigma_2))\} \subseteq$
$\quad \{|\delta| \mid \delta \in \text{Ind and } \Gamma_A \vdash (\delta \textit{ is true})\}$ (by transitivity)
$\{|\delta| \mid \delta \in \text{Ind and } \Gamma_A \vdash (\delta \textit{ is true})\} = \{|\textit{true}|\} =$
$\Upsilon[\![\sigma_1 \vee \sigma_2]\!]$ (by definition of Υ)
The proof is similar under the hypothesis that $\Gamma_A \not\models_{\mathcal{M}} \sigma_1$ and $\Gamma_A \not\models_{\mathcal{M}} \sigma_2$

Once lemma 10 has been established we can prove the main lemma. In fact from lemma 9 we can deduce the following:

$$\text{if } \forall \sigma . \Upsilon[\![\sigma]\!] = \{|\delta'| \mid \delta' \in \text{Ind and } \Gamma_A \vdash (\delta' \textit{ is } \sigma)\}$$
$$\text{then } \forall \sigma . \Gamma_A \vdash \sigma \Leftrightarrow \Gamma_A \models_{\mathcal{M}} \sigma$$

The main lemma follows from this and lemma 10.

VII Consistency

The consistency of Omega can be established by means of the following result:

Theorem 11: If an Omega theory Γ has a model, then it is consistent.

Proof: Suppose Γ has a model M. Then $\Gamma \not\models_M \textit{false}$. From the completeness result it follows that $\Gamma \not\vdash_M \textit{false}$ which proves the consistency of Γ.

VIII Omega and other Formal Logic Theories

Omega is not a First Order Predicate Logic, since it includes a *variable binding operator*, namely the *Any* construct, which is not present nor expressible in First Order Predicate Logic. In this sense Omega is more similar to set theory, which includes set abstraction, besides universal and existential quantifiers.

The version of Omega presented here is a first order theory. We are investigating extending the axiomatization to higher orders. In [10] we presented examples of the use of higher order capabilities.

Omega is a Set Theory. However Omega relies on constructors for building new descriptions. In set theory, pairs for instance are built by means of set formation alone. In Omega a Pair constructor can be used to describe pairs of objects. A Pair of two individuals will be an individual itself, therefore separating the inheritance relationship from the component relationship. Omega is a constructive set theory and has no axiom corresponding to the powerset axiom of classical est theory. The constructive nature of Omega implies that It is possible to use the proof of a statement to determine a description which meets some requirements or is the answer to a question.

IX Conclusion

The results of this paper provide us with a solid base on which to build a theory of Knowledge repreeentetion.

The system presented is powerful enough to express arithmetic, by following the construction of [13]. The use of attributions as in [10] however increases the naturalness of the notation. There are a number of problems regarding attributions to be addressed such as: interaction between attributions, merging and inheritance; functional dependencies between attributions; contrasting uses of attributions (for instance attributions have been used to express part/whole relationships as wen as to express n-ary relations). In [10] a preliminary set of axioms for attributions was presented. A set theoretical model along the lines of the present work has to be defined in order to obtain a satisfactory full axiomatization of attributions. Such sxiomatization will appear in a forthcoming paper [2].

Another important construct allowed in Omega is the A-abetractkxv The axiomatizat»on for this construct is similar to the axiomatization of the A calculus. This construct interacts nicely with the inheritance mechaniam of Omega, allowing the same notation to be used for describing the type of a function aa well as for defining its values.

We have also investigated the problems arising from allowing self-reference in the language. In order to avoid logical paradoxes Mke the *Her* paradox, it is necessary to give up the rules of excluded middle (axiom S5) and contradiction (axiom 86). This allows models where the value of a sentence can be neither true nor false. Such solution follows the lines of the one suggested by Viseer in [20]. The logic can still be proved to be complete, but the proofs by hypothetical reasoning become more cumbersome.

Acknowledgments

Can Hewitt has been the leading force behind the developement of Omega and its axiomatization. Luce Cardelli and Giuseppe Congo have thoroughly discussed our ideas in the early stages of this work. William Dinger has provided constructive criticism and encouragement. We had several suggestions from Jerry Barber, who is Involved in implementing Omega, about the basic rules of deduction.

References

[1] Attardi G., Barber G. and Simi M. "Towards an Integrated Omcs Work Station". *Strvmentazione* e *Automaaiorm* (March 1060).

[2] Attardi O., Hewitt C. and Simi M. "Semantics of Inheritance and Attribution* in the Description System OMEGA". Al Memo forthcoming, MIT, 1661.

[3] Bobrow, D. G. and Winograd, T. "An Overview of KRL, a Knowledge Representation Language". *Cognitive* Science J, 1 (1877).

[4] Brachmen, R.J "A Structural Paradigm for Representing Knowledge". Report 3606, Bolt Beranek and Newman Inc., May, 1978.

[6] CiccaretK, E. and Brachmen, R.J. NKt-One Reference Manual". UPH rtepoiv

[6] De Jong, P. Private communication. February 17th, 1661.

[7] Detiyanni, A.. Kowalaki, R.A. "Logic and Semantic Networks". *Comm.* ACM 22.3(1979), 184-182.

[8] Fahlman, Scott. *"NETL A System tor Representing* and *Using Real-World* Know/edge". MIT Press, 1970.

[9] Fikes, Rand Hendrix, G. "A Network-Based Knowledge Representation and its Natural Deduction System". Proc. IJCAI-77, Cambridge, Mass.. August. 1977. pp. 236-246.

[10] Hewitt C, Attardi G. and Simi M. "Knowledge Embedding with the Description System OMEGA". Proc. of First National Annual Conference on Artificial Intelligence, Stanford University, August. 1960, pp. 157-163.

[11] Kalish and Montague. *"Logic Techniques of Formal Reasoning".* Harcourt, Brace and World, 1964.

[12] Kornfetd W. A. "ETHER -• A Parallel Problem Solving System". Proc. of 6th int. Joint Conference on Artificial Intelligence, Tokyo, 1979. pp. 490-492.

[13] Martin R. M. "A homogeneous system for formal logic". 7/ie *Journal ot Symbolic Logic 8,*1 (1943).

[14] Quillian, M. R. "Semantic Memory". In *Semantic Information Processing,* M. Minaky, Ed.. MIT Press, 1968.

[15] B. Russell. "On Denoting". *Mind 14* (1906). 479-483.

[18] Scott D. "Existence and Description in Formal Logic". In *Bertrand Russell: Philosopher ot the Century,* R. Schoenman, Ed., G. Allen ft Unwin Ltd.. London, 1969.

[17] Shoenfietd J. R. *"MathematicalLogic"* Addison Wesley, 1967.

[18] Shubert L.K. "Extending the Expressive Power of Semantic Networka". *Artificial Intelligence 7* (1878), 163-108.

[19] Steels, L. "Reasoning Modeled as a Society of Communicating Experts". Al Lab Technical Report 542, MIT. June, 1979.

[20] Visser, A. ."The Liar Paradox". Lecture Notes. ,1980.