Logic without Model Theory

Robert Kowalski

Abstract

Arguably, model theory serves two main functions: (1) to explain the relationship between language and experience, and (2) to specify the notion of logical consequence. In this paper I shall propose the notion of 'knowledge assimilation', the assimilation of new information into a knowledge base, as an alternative understanding of the way in which a knowledge base formulated in logic relates to externally generated input sentences that describe experience. I shall argue that the notion of logical consequence can also be understood within a knowledge assimilation framework, in terms of sentences that must hold no matter what stream of input sentences might arise in the future.

Classical model theory can be understood as dealing with static relationships among individuals. It leads naturally therefore to possible world semantics and modal logic, in which models are understood as related to one another by accessibility relations. I shall argue in favour of a non-model-theoretic alternative to possible world semantics, an alternative which employs a syntactically rich vocabulary of terms representing time, events, situations and theories.

Similarly to the way in which possible worlds can be viewed as arising from classical models, situations which cut across time and space in situation semantics can be viewed as arising from possible worlds. I shall argue for representing situations syntactically as theories and amalgamating object language and metalanguage as an alternative to situation semantics.

1 Introduction

Logic is an important object of study in such diverse disciplines as mathematics, philosophy, psychology, linguistics, computing, artificial intelligence and law. It is used informally in every other intellectual discipline: in the natural sciences, social sciences and humanities. Despite the all-pervasive nature of logic, however, there is little agreement among experts and lay people alike about whether or not humans are truly logical; and, if they

are, about what that logic might be like. Even worse, there seems to be little communication between experts in logic working in different fields.

In this paper I will outline a computational approach to logic that has proved useful for building non-trivial applications in computing, artificial intelligence and law. I will argue that such a logic can also be used to understand human reasoning in both computational and logical terms. For ease of reference, I will call this computational logic 'CL'.

The computational logic, CL, is not entirely well-defined. It is an evolving system of logic, which has its basis in the clausal and logic programming (LP) forms of logic, but which is undergoing continual refinement and reinterpretation. In this paper I will not be concerned with the theoretical foundations of CL, but rather with the practical characteristics which make it useful for building complex applications and for modelling human reasoning. These characteristics include the use of a rich vocabulary of terms and the combining of object language and meta-language, to represent and reason about time, events, states of affairs, and theories.

Johnson-Laird and Byrne [16], in their book on human deduction, dismiss clausal logic and by implication LP, as psychologically implausible and propose instead a model-theoretic account of human reasoning. They argue that their 'mental model' theory explains human performance on reasoning tasks better than proof-theoretic approaches that assume humans reason by applying rules of inference.

The mental model view of human reasoning is similar to the model-theoretic approach to databases, in which a database is regarded as a model-theoretic structure and a closed query is evaluated by determining its truth value in the database. The model-theoretic approach is also common in modal logic, where possible world semantic structures are often used directly as temporal databases or 'knowledge bases'. It is, perhaps, taken to its greatest lengths in situation semantics [2], where all information is directly associated with situations, viewed as extra-linguistic semantic structures.

Situation semantics, motivated largely by problems in linguistics, and mental models, developed in the field of psychology, share a common assumption that human logical thinking is based upon the processing of certain kinds of semantic structures. In this paper I will present and defend the contrary view that both computer and human reasoning are better viewed proof-theoretically as reasoning with sentences formulated in an internal, 'mental' language.

I will begin in section two by considering the two main goals of model theory: (1) to explain the relation between language and experience, and (2) to specify the notion of logical consequence. In section three, I will present an alternative, more pragmatic approach to the first goal—one based upon the proof-theoretic assimilation of observational sentences into

a knowledge base of sentences formulated in a language such as CL. Whereas in model theory 'truth' is a relationship between language and reality, in the alternative approach 'truth' is a relationship between sentences of the knowledge base and observational sentences.

The language considered in section three is the mental language of an agent that is forced to make sense of a continuous stream of experience formulated in terms of observational sentences. In section four, I consider the case where the observations which need to be assimilated are utterances of natural language sentences, and I argue that the *meaning* of such sentences is best understood syntactically as the result of translating them into other sentences of the agent's mental language.

In section five, I consider the second goal of model theory, to specify the notion of logical consequence, and I outline an alternative specification based upon the hypothetical consideration of all possible, complete input streams of observational sentences. The alternative specification differs from the orthodox model-theoretic specification by making no assumptions about the existence of individuals, functions and relations apart from those 'projected' by the vocabulary of the language. Although the alternative specification is entirely syntactic, it can also be understood 'pseudo-model-theoretically' in terms of Herbrand interpretations.

In sections six and seven, I consider possible world semantics and situation semantics respectively. In both cases I propose syntactically-based alternatives. In the case of temporal possible world semantics in particular, I propose the use of a rich vocabulary of terms representing time and events directly in the language. In the case of situation semantics, in sections eight and nine, I propose the use of metalanguage, in which terms represent situations regarded syntactically as theories. In section ten, I conclude.

2 What is model theory?

Arguably, model theory has two main functions. First, it aims to clarify the relationship between language and experience, by considering the concept of truth as a relationship that holds between a sentence and a semantic structure. The semantic structure, called an *interpretation* associates individuals with constant symbols of the language, functions with function symbols, and predicates or relations with predicate symbols. An atomic, variable-free sentence is said to be *true* in the interpretation (and the interpretation is said to be a *model* of the sentence) if and only if the individuals associated with the terms appearing in the sentence stand in the relation associated with the predicate symbol of the sentence. As an explanation of the relationship between language and experience, model-theoretic seman-

tic structures incorporate an assumption that experience is caused by an independently existing reality, and they serve as mathematical idealisations of that reality.

The other main purpose of model theory is to specify the relationship of *logical consequence* between a set of sentences T and a sentence P as holding if and only if P is true in every interpretation in which all the sentences of T are true. The model-theoretic definition of logical consequence formalises the intuitive understanding that P holds whenever T holds, no matter what meaning is associated with the symbols in T and P.

I shall argue that model theory fails in its first goal, of giving a good explanation of the relationship between language and experience; and it is only partially successful in its second goal, of specifying the notion of logical consequence. In both cases, the assumption of model theory that there exists a reality composed of individuals, functions and relations, separate from the syntax of language, is both unnecessary and unhelpful.

3 A more pragmatic view of the relationship between language and experience

Computer systems that interact with the world and that use logic to reason in the context of those interactions suggest a very different and more pragmatic view of the relationship between language and experience. In this view, the symbolic representations which are constructed and manipulated by the computer can be thought of as sentences of a '*knowledge base*' formulated in an internal '*mental language*'.

Here the notion of mental language needs to be understood liberally, analogously to the way in which computer languages are understood in Computing. Computer programs written in a high-level language can be compiled into lower-level languages and even into hardware, in such a way that their high-level origins can be practically unrecognisable. Nonetheless, to understand and reason about the behaviour and 'semantics' of such programs it is generally useful to view them as though they were still written in the high-level language and as though they were executed on a high-level 'virtual machine' appropriate to the high-level language. Even programs implemented directly at lower-levels are often better understood by 'decompiling' them and imagining them to have been written in a high-level language.

The fundamental thesis of logic programming is that appropriate forms of logic can serve as a high-level programming language. It follows that such forms of logic can also be used to understand and reason about computations which actually take place at a lower software, hardware or even biological level.

Logic programs can also function as deductive databases or knowledge bases. But databases, like programs, can usefully be understood at several different levels. The *external level* is what the user sees—perhaps a natural language, graphical, menu-driven or forms interface. The *physical level* is what the computer 'sees'—typically a collection of obscure data structures and complicated algorithms that exploit the physical properties of the computer to achieve efficiency. The *conceptual level* is what the designers and implementers of the database system see when they want to understand and reason about the intended behaviour of the database. It is at this conceptual level that the database can be understood as a logical theory—some collection of sentences in logical form.

It is also at the conceptual level that an intelligent computing agent can be understood as representing beliefs about its interactions with the world in the form of a theory or 'knowledge base' of sentences formulated in a 'mental', logical language.

Such theories or knowledge bases are really 'theory presentations' from which logical consequences are derived, both in order to solve problems and in order to assimilate new 'information'. Logically equivalent theories, which entail the same logical consequences, can have very different pragmatic characteristics, in the same way that different, but equivalent programs can have very different computational properties.

In the general case, the knowledge base of an agent consists both of observational *sentences*, which record inputs and which correspond directly to experience, and of theoretical *sentences*, which do not have direct counterparts in experience. Observational sentences are *ground* (i.e. variable-free) atomic sentences, which identify individuals, classify them, and record both their attributes and their relationships with other individuals. With the aid of theoretical sentences, other ground atomic sentences can be derived from input observational sentences; and these derived sentences can be compared with previous and future observational sentences. A ground atomic sentence might be regarded as 'true' if it corresponds exactly with some such past or future input observation.

For example, the knowledge base might record an input observation that

> there is smoke coming from the kitchen.

Appropriate 'mental constants' would be used as symbolic representations of 'the smoke' and 'the kitchen'. These might be constants already occurring in the knowledge base, in the case of individuals about which there already exists some previous 'knowledge', or they might be new constants for new individuals. The record might use a predicate symbol to represent the relationship 'coming from'. The time of the observation might

be recorded explicitly by some form of mental time-stamping (in the manner of CL) or implicitly by a modal operator. The entire record of the observation might then take the form

isa (smoke$_1$, smoke)
isa (kitchen$_1$, kitchen)

and either

coming-from (smoke$_1$, kitchen$_1$, time$_1$), or
coming-from (smoke$_1$, kitchen$_1$)

where in the first case the third argument place of 'coming-from' indicates that 'time$_1$' is the time of the happening, whereas in the second case there is an implicit modality indicating that the event took place at the present time.

Theoretical sentences in the knowledge base (in one form or other) might represent such beliefs as

whenever and wherever there is smoke, there was an earlier event of ignition which happened and which caused the smoke

and

whenever and wherever an event of ignition happens, there is a state of fire soon afterwards.

With the aid of such theoretical sentences it would be possible to derive the conclusion that there is, or soon will be, fire in the kitchen. This conclusion can be compared with other observational sentences coming from other observations made at the same or other times. Once observations have been recorded in the mental language of the knowledge base, these comparisons between derived and input sentences are purely syntactic (relative to the mental language). A record of observing fire in the kitchen would confirm (the 'truth' of) the derived conclusion. A record of observing a smoke machine would probably refute it.

That part of the knowledge base, which includes observational sentences and those theoretic sentences which can be used to derive conclusions that can be compared with observational sentences, is often referred to as a *world model*. This use of the term is potentially confusing because the notion is completely syntactic and quite different from the notion of model in model theory.

World models are tested by comparing the conclusions that can be derived from them with other sentences that record inputs, which are observational sentences extracted from experience. In the idealised case, where

observational sentences are assumed to be faultless, they serve as the standard against which the world model can be tested. Thus, a ground atomic sentence derivable from the 'model' can be regarded as '*true*' if it is identical to an input observational sentence. Moreover, such an input need not be added to the 'model' because it is already derivable and therefore redundant. The negation of a ground atomic sentence derivable from the 'model' can be regarded as '*false*' if it is the negation of an input observational sentence. In such a case assimilation of the input would involve (perhaps non-deterministically) removing or modifying some sentence in the knowledge base that leads to the derivation of the false conclusion, and adding the 'true' input to the knowledge base.

In the idealised and unrealistic case where the observational sentences constitute a complete and correct description of all experience, then the '*truth*' or '*falsity*' of all sentences in the world model can be determined. Thus, for example, given such a complete and faultless set of observation sentences O a sentence of the form

$$\forall X p(X)$$

would be '*true*' *relative to* O, if every ground instance

$$p(t)$$

is '*true*' relative to O, where t is any ground term occurring in O, and it would be '*false*' otherwise.

Moreover, a negative sentence

$$\neg P$$

is '*true*', relative to O, if and only if P is not in O. Thus it is the assumption that the observations are complete that warrants concluding $\neg P$ if P cannot be validated by the observations. This is similar to the assumption of completeness used to justify negation as failure in logic programming [5].

Obviously, the notion of an idealised, faultless and complete set of observational sentences has much in common with the notion of model in model theory. In model theory, there is a real world, consisting of real individuals, functions and relations. In the more pragmatic theory, however, there is only an inescapable, constantly flowing input stream of observational sentences, which the agent is forced to assimilate. To inquire into the source of this input stream and to speculate about the nature of the source is both unnecessary and unhelpful. For all the agent can ever hope to determine, the source might just as well be some form of *virtual reality*.

Hallucinations can be explained as a lack of coherence among the input sentences themselves, rather than as any lack of correspondence between

input sentences and reality. Identifying an appropriate record of experience to be rejected or otherwise modified, as such an hallucination, can be a non-deterministic process, like any other process of restoring consistency to an inconsistent set of sentences.

In model theory, truth is a static correspondence between sentences and a given state of the world. In the computationally inspired, pragmatic theory, however, what matters is not so much 'truth' and correspondence between language and experience, but the appropriate *assimilation* of an inescapable, constantly flowing input stream of observational sentences into an ever changing knowledge base. Correspondence between an input sentence and a sentence that can be derived from the knowledge base is only a limiting case. In other cases some weaker form of coherence may be all that can be obtained. In the most extreme form of incoherence, which arises in the case of inconsistency, assimilation of an input might require a non-deterministic revision of the knowledge base.

A related process of *belief revision* was considered by Gärdenfors [12] and by Alchourrón, Gärdenfors and Makinson [1], who formulated a number of postulates about the relationship between a given state of a knowledge base, an input, and the resulting successor state of the knowledge base. These postulates embody a number of idealised assumptions, about the knowledge base containing all its logical consequences and about there being a unique successor state, which are not computationally feasible. Perhaps, more importantly though, the belief revision theory shares with the knowledge assimilation theory presented here the property that model-theoretic considerations are unnecessary. Moreover, Gärdenfors [12] shows how belief revision can give an alternative account of the semantics of logic, somewhat in the spirit of the account presented later in section five of this paper.

The process of *knowledge assimilation*, proposed in [18], was intended as a computationally feasible account of how input sentences might be assimilated into a given set of sentences constituting a 'theory'. The proposal was intended to include such diverse applications as updating a database, understanding a natural language discourse, enlarging and testing a scientific theory, and assimilating observations into the knowledge base of an intelligent, computing agent. A related, computationally-oriented theory of human cognition in general and of human communication in particular has been developed by Sperber and Wilson [33].

In knowledge assimilation, the relationship between a given state T of a theory, an input sentence P, and a successor state T' of the theory is determined by resource-constrained deduction. There are four cases:

1. P is a logical consequence of T.

2. Part of T is logically implied by P together with the other part of T,

i.e. $T = T_1 \cup T_2$ and T_2 is a logical consequence of $T_1 \cup \{P\}$.

3. P is inconsistent with T.

4. None of the relationships (1)–(3) hold.

Input sentences, P, occur as items in a constantly flowing input stream. Normally there is little time available to process one input before the next already appears. Although it is sometimes possible to interrupt the processing of the first input, process the second and return to the first, most inputs need to be assimilated 'on-line' in the relatively small gap between that input and the next. Thus detecting any of the logical relationships (1)–(3) outlined above is generally subject to severe limitations on the processing time available.

To make the best use of the limited computational resources, proof procedures for detecting logical consequences need to be as efficient as possible. For this reason they need to avoid generating obvious redundancies and irrelevancies. One way to reduce the generation of redundancies is to avoid explicitly putting them there in the first place. This is the purpose of cases (1) and (2).

One way to reduce the generation of irrelevancies is to focus on the input. This can be done by reasoning forward from P in cases (2) and (3), so that every conclusion generated depends non-trivially on P; and, similarly, to reason backwards from P in cases (1) and (2). Another way is to avoid unnecessary and computationally unmotivated use of the thinning rule, which derives

$$A \vee B \text{ from } A.$$

Such strategies for improving the efficiency of deduction in classical logic have been developed in the field of automated reasoning. Many of these strategies are based upon some restricted use of the resolution rule of inference [29]. These strategies implement classical logic, but derive only *relevant* conclusions, without using relevance logic. Related restrictions on the deductive processing of information have been proposed by Sperber and Wilson [33].

The successor state T' , which results from processing an input sentence P in a given state T of a theory, depends upon what logical relationships can be determined between T and P within the limited computational resources available.

In case (1) the input is determined to be redundant, and the theory does not need to change, i.e. $T' = T$. However, although the input does not contain any new logical (or 'semantic') information, it does have pragmatic value. It identifies some subtheory T^* of T used to derive P. This subtheory

can be most easily determined by reasoning backward from the input P. To the extent that all the sentences in T^* are *relevant* to the derivation of P, the input P lends support to the sentences in T^*. The increased support given to T^* could be recorded in the form of metalogical *labels* [10] which somehow measure the *degree of confirmation* or *utility* of the sentences in T^*.

The term 'degree of confirmation' comes from philosophy of science, where it indicates the extent to which an hypothesis conforms to observational evidence. Here, I use the term more in the sense of Gärdenfors' [12] *epistemic entrenchment* and Sperber and Wilson's [33] *strength* of an assumption, to indicate the extent to which a sentence has proved useful for the deductive processing of other sentences in the past or the extent to which is expected to prove useful in the future.

In case (2), the input provides useful information, which renders part, T_2 of T redundant. Therefore the input can replace T_2 in the successor state of the theory, i.e. $T' = T_1 \cup \{P\}$. Assuming that T itself is the result of a previous sequence of knowledge assimilation steps, and therefore that it contains no 'obvious' redundancies or inconsistencies, the generation of T_2 can be performed by reasoning forward from P, thereby restricting the conclusions contained in T_2 to ones in which the contribution of P is *relevant*. Moreover, if degree-of-confirmation labels are associated with sentences in the theory, then the labels associated with sentences in the set T^*, used to derive conclusions in T_2, can be revised to record a higher degree of confirmation.

In case (3), the test for inconsistency can be performed by reasoning forward from P, on the assumption that the search for inconsistency can be restricted to proofs in which the contribution of P is relevant [30]. The derivation of an inconsistency identifies a subset T^* of $T \cup \{P\}$, containing P, which needs to be revised in order to avoid the inconsistency. In general, this can be done in many different ways, and therefore the choice of successor state T' will be non-deterministic. i.e. different choices of T' will have the desired effect of avoiding the inconsistency. In some domains, such as database updates, it is common simply to ignore the input, and so $T' = T$. In other domains, where the input can be regarded as recording 'true' observations, some other way of restoring consistency needs to be found. In these and other cases, degree-of-confirmation labels can help to identify candidate sentences to be removed or otherwise modified. In any case, it may not be possible to identify a unique successor state T', in which case the agent may need to explore alternative successor states, whether in sequence or in parallel. Notice, moreover, that exploring alternative, mutually incompatible successor states in parallel might give an external observer the misleading impression that the agent is irrationally committed to holding simultaneously incompatible beliefs.

In case (4), where none of the other logical relationships can be determined in the time (and space) available, it may be necessary to add the input to the theory, obtaining $T' = T \cup \{P\}$. In many domains, however, it is more appropriate (and more coherent) to determine an *abductive explanation* Δ such that $T \cup \Delta$ implies P and to let $T' = T \cup \Delta$. Other constraints that are normally imposed upon Δ include that (a) $T \cup \Delta$ be *consistent*, (b) Δ be *minimal*, i.e. no strict subset Δ' of Δ is such that $T \cup \Delta'$ implies P, and (c) Δ be *basic*, i.e. not derivable (by deduction or abduction) from $T \cup \Delta'$, where $\Delta' \neq \Delta$. As in case (3), the choice of T' will often be non- deterministic. Again, degree-of-confirmation labels can help to compare different derivations and to choose a Δ such that the relevant subset T^* of T used with Δ to derive P has a greater (or at least no worse) degree of confirmation than other relevant subsets used with other abductive explanations. As in case (1), the generation of the *relevant* set T^* needed to derive P can be performed by reasoning backward from the input P.

Adding an abductive explanation in place of the input to obtain a successor state of the knowledge base violates the Alchourrón–Gärdenfors–Makinson rationality postulates, but accords well with the Sperber–Wilson Relevance Theory. A survey of the extension of logic programming to incorporate abduction is given by Kakas, Kowalski and Toni [17].

For the sake of efficiency, cases (1) and (4) can be combined, using resolution to reason backward from the goal P, reducing it either to the empty set of subgoals (case 1) or to a set of abducible subgoals (case 4). Cases (2) and (3) can also be combined, using resolution to reason forward from the assertion P, either to derive conclusions already in T (case 1) or to derive an obvious inconsistency (case 3). Furthermore, the test (case 4) that an abductive explanation Δ is consistent with T, can be subsumed by treating Δ as a new set of inputs to be assimilated.

Compared with knowledge assimilation, model theory can be regarded as dealing with the special and limiting case where the input sentences constitute a correct and complete description of the world, and are given entirely in advance. Cases (1)–(4) of knowledge assimilation are roughly analogous to the recursive definition of truth in model theory. The big difference, however, is that model theory assumes the existence of semantic structures containing individuals, properties and relationships, separate from the syntactic structures of the language. Knowledge assimilation, on the other hand, assumes only that there is a constant stream of input sentences that need to be assimilated.

In the normal case, the input sentence to be assimilated is an observational sentence—for example, some record of a natural language utterance. Such observational sentences typically have the form of ground atomic sentences. However, an agent might also generate its own hypothetical inputs,

as in the case of abduction, induction, or theory formation more generally. The four cases of knowledge assimilation apply also to such hypothetical input sentences, in which case (2) assumes a special importance, because it indicates the *explanatory power* of the hypothesis. The greater the number (and degree-of-confirmation) of sentences T_2 derivable from the hypothesis, the greater the explanatory power and utility of the hypothesis.

In summary, knowledge assimilation provides a syntactic and pragmatic alternative to model theory as an account of the relationship between language and experience. It assumes that experience takes the form an inescapable stream of input sentences, which needs to be assimilated into a constantly changing knowledge base. The knowledge base serves to organise and provide efficient access to useful information. Not only do its consequences need to correspond as much as possible with experience, but ideally they need to provide coherent explanations as well.

In the next two sections, I will discuss natural language processing and logical consequence in knowledge assimilation terms. In the following sections I will discuss syntactic alternatives to possible world semantics and situation semantics.

4 Natural language processing

Natural language understanding can also be understood in knowledge assimilation terms—but with two complicating factors: The first concerns the relationship between language and thought. Presumably, there is or there ought to be some close relationship between the structure of a natural language utterance and the structure of some sentence in the mental language of the communicator. In the simplest case, the utterance conveys the communicator's thought as directly and as simply as possible. In other cases, the utterance may be ambiguous or misleading. In yet others, it might attempt to articulate a new thought, as part of the communicator's process of assimilating a new hypothesis into its own knowledge base. In many cases, the correspondence between a natural language utterance and its intended meaning might be very imperfect indeed.

The second complication concerns the difference in the meaning of an utterance as understood by the communicator compared with its meaning as understood by the recipient. For the recipient, this means that the utterance needs to be understood in terms of both the recipient's own point of view as well as the recipient's understanding of the communicator's point of view.

Consider, for example, the process of attempting to understand and assimilate the consecutive sentences in a text such as the one constituting this paper. Arguably, the reader has a two-fold task. The first is to

understand the text in its own terms, assessing the extent to which the presumed meanings of individual sentences cohere with the previously determined meanings of the sentences which preceded them. The second is to understand the significance of the text for the reader's own beliefs, assessing the extent to which the meanings conveyed can be coherently assimilated into the reader's own knowledge base. In the simplest case, understanding a straight-forward account of some historical event, for example, the two tasks might collapse into one if the recipient has sufficient faith in the communicator. In a more complicated case, however, the recipient might not only decide to reject the information, but also to conclude that the communicator is using the communication for some ulterior motive. An example, where the recipient would have benefited from reasoning in such a way, is the crow in Aesop' s fable of the fox and crow.

Notice that throughout the preceding discussion I have implicitly assumed that the '*meaning*' (and by implication the '*semantics*') of natural language sentences is be obtained by translating such natural language sentences into other sentences of a mental language. This is a kind of '*correspondence theory*' of meaning—not a correspondence between natural language utterances and actual states of affairs, but rather a correspondence between natural language utterances and mental language sentences.

In the standard account of natural language understanding, the observational sentences that are input to a recipient record only the syntactic form of the utterance. The recipient needs to process this syntactic form to generate a representation of its 'semantics'.

Consider, for example, the natural language sentence

'All humans are mortal.'

A typical natural language processing program would first generate an internal representation of its syntax, for example a list such as

['All', 'humans', 'are', 'mortal', '.']

and then a representation of its meaning, e.g.

$$\forall X(h(X) \rightarrow m(X))$$

The program might usefully record the source and context of the input by means of appropriate metalevel sentences such as

said (c_o, john, ['All', 'humans', 'are', 'mortal', '.'])
said-that (c_o, john, '$\forall X(h(X) \rightarrow m(X))$')

where the second sentence would be derived from the first. Here the first argument, c_o, is some representation of the context—possibly a time indicator, an event identifier, or even a situation in the spirit of situation

semantics. By reference to this argument, the agent trying to assimilate the input can gain access to previous inputs in the same discourse.

Having obtained a metalevel representation of the presumed meaning of the input, the agent would then attempt to assimilate this metalevel sentence into its knowledge base, perhaps deriving such object level conclusions as

$$\forall X(h(X) \rightarrow m(X)) \qquad \text{or}$$

$$\text{logician (john)} \vee \text{psychologist (john)}.$$

In the first case, the conclusion might be derived by means of a metalevel sentence[1]

$$\text{believes } (X, Y) \leftarrow \text{said-that } (X, Y) \wedge \text{ trustworthy } (X)$$

together with a scheme[2] that combines an object level conclusion with a metalevel condition

$$Y \leftarrow \text{believes } (X, \text{ `}Y\text{'}) \wedge \text{ wise } (X)$$

and with object level sentences that express that John is both trustworthy and wise.

In the second case, the conclusion might be derived from other, quite different assumptions in the knowledge base, e.g.

$$\text{logician } (X) \vee \text{ psychologist } (X) \leftarrow \text{said-that } (X, Y) \wedge \text{ logic-example } (Y)$$

In both cases what is assimilated is not an object level sentence expressing that all humans are mortal, but rather a metalevel sentence expressing that john said that all humans are mortal.

In general it is important to distinguish between information coming from direct experience and information coming from communication. Although both kinds of information are appropriately expressed by means of observational sentences, information which comes from direct experience is most naturally expressed in object level form, e.g.

$$h \text{ (john)}.$$

a record of an observation that john is human, whereas information which is communicated is most naturally expressed in metalevel form, e.g.

[1] Upper case symbols are used here and elsewhere in this paper for variables. Any variable occurring in a sentence is assumed to be universally quantified, even when the quantifiers are not written explicitly. Note also that I use '$p \rightarrow q$' and '$q \leftarrow p$' interchangeably.

[2] A simpler representation of this scheme as a combined object-level, metalevel sentence without quotation will be given in section 8.

said (c_o, john, ['All', 'humans', 'are', 'mortal', '.'])

Of course, such metalevel observational sentences are also, in a sense, object level sentences which record direct experiences of the communication itself.

The example shows how difficult it is to test whether or not a computing agent processes information logically, and, if it does, what kind of logic the agent employs. A psychologist, for example, who poses logic puzzles in natural language, can not simply assume that the agent receiving the communication assimilates the information communicated directly in object-level form as though it were the result of its own direct experience. To determine whether or not the agent reasons logically, the interrogator would need to know the contents of the agent's knowledge base and understand how the agent assimilates the communication (not the information communicated!) into that knowledge base.

5 The specification of logical consequence

I believe that the considerations presented in the previous two sections call into question the usefulness of model theory in providing a useful account of the relationship between language and experience. Model theory helps to explain neither the relationship between mental language and the world nor the relationship between natural language and its meaning.

But perhaps the more significant achievement of model theory is its providing a specification of logical consequence, which is arguably more compelling than simply providing a proof procedure. In this respect, model theory (non-constructively) specifies the notion of logical consequence in much the same way that a program specification specifies a program. In contrast, proof theory provides a non-deterministic, but constructive definition of logical consequence, which is analogous to a non-deterministic program.

Model theory formalises the intuitive specification of logical consequence, which can be put informally in the form

> a set of sentences T logically implies a sentence P
> if and only if
> for every interpretation I of the language L
> in which T and P are formulated,
> P holds in I if all sentences in T hold in I.

Model theory formalises the notion of interpretation in this informal specification in terms of set theoretic or algebraic structures, which have a different nature from the linguistic structures of the language L. I have

already argued that, for the purpose of understanding the relationship between language and experience, such extra linguistic semantic structures are neither necessary nor useful. I have argued instead that the notion of assimilating a dynamically changing input stream of observational sentences can give a better account both of the relationship between mental language sentences and experience and of the relationship between natural language sentences and their meanings. I shall now argue that a similar, purely syntactic notion, in which interpretations are understood as idealised, complete and faultless input streams of observational sentences, can be used to formalise the specification of logical consequence. This notion is, in fact, similar to the notion of Herbrand interpretation in model theory.

The syntactic specification of logical consequence is especially transparent in the case of sentences formulated in clausal form, which is the basis for both resolution and LP. Clausal form is normally considered as an implementation of classical first-order logic (FOL). However, although clausal form has some disadvantages compared with FOL, it can also be considered as a knowledge representation formalism in its own right [18]. Its main advantage is its simplicity and the fact that trivial syntactic differences between sentences are avoided by the use of a canonical form. For example, a conjunction of sentences, $A \wedge B$, is represented as a set $\{A, B\}$; double negation, $\neg\neg A$, is automatically eliminated; and negation is only applied to atomic formulae. Existential quantifiers are avoided by introducing 'skolem' constants and function symbols, which make existential commitments more explicit than existential quantifiers. For example, the sentence

$$\forall X \exists Y \text{ father } (Y, X)$$

becomes a clause

$$\forall X \text{ father } (\text{dad}(X), X),$$

where the function symbol 'dad' is distinct from any other function symbol used elsewhere.

In general, a *clause* can be written either as an universally quantified disjunction of literals or as an universally quantified implication of the form

$$\forall X_1, \ldots, X_k (A_1 \wedge \ldots \wedge A_n \rightarrow B_1 \vee \ldots \vee B_m)$$

where the A_i and B_j are atoms and $X_1, \ldots, X_l$ are all the variables occurring in the A_i and B_j. In LP, the number of conclusions m is always less than or equal to one. If m is zero, then the clause is equivalent to a denial

$$\forall X_1, \ldots, X_k \neg [A_1 \wedge \ldots \wedge A_n].$$

Because all variables are universally quantified it is usual to omit explicit universal quantifiers.

The semantics of clausal form is normally defined in terms of Herbrand interpretations, which are sets of ground atoms. I shall show that this semantics can also be understood in knowledge assimilation terms. Given a set of clauses S, a *Herbrand interpretation* of S is any set of ground atoms constructed from the vocabulary of predicate symbols, function symbols, and constant symbols occurring in S. Thus Herbrand interpretations can be understood purely syntactically as a complete and faultless set of possible observational sentences, where every ground term of the language is regarded as the name of a 'conceivably' observable individual and every predicate symbol as the name of an observable predicate or relation.

To show that a set of clauses T *logically implies* a sentence P, the denial, $\neg P$, of P is converted into a set of clauses P^* and it is shown that $T \cup P^*$ is *inconsistent*. This is done by showing that every Herbrand interpretation of $T \cup P^*$ falsifies some clause in $T \cup P^*$. A Herbrand interpretation I *falsifies* a clause if it falsifies some ground instance

$$A_1 \wedge \ldots \wedge A_n \rightarrow B_1 \vee \ldots \vee B_m$$

of the clause. Such a variable-free clause is *falsified* by I if all of $A_1, \ldots, A_n$ belong to I, but none of $B_1, \ldots, B_m$ belong to I.

It is possible to execute the semantics of clausal form directly, to determine whether a set, S, of clauses is inconsistent, using the method of semantic trees [21], originally developed to prove the completeness of resolution. The semantic tree procedure can be viewed as an idealised process of assimilating all possible input streams of complete observations and showing that each such stream falsifies some clause in the set of clauses S. There is a one-to-one correspondence between such input streams and Herbrand interpretations of S.

The process of assimilating all possible input streams of complete observations can be formulated as a process of growing a binary-branching tree of partial input streams, and terminating the growth of a branch when the observations recorded on that branch already falsify some clause in S. If S is inconsistent, the process will terminate after only a finite number of steps.

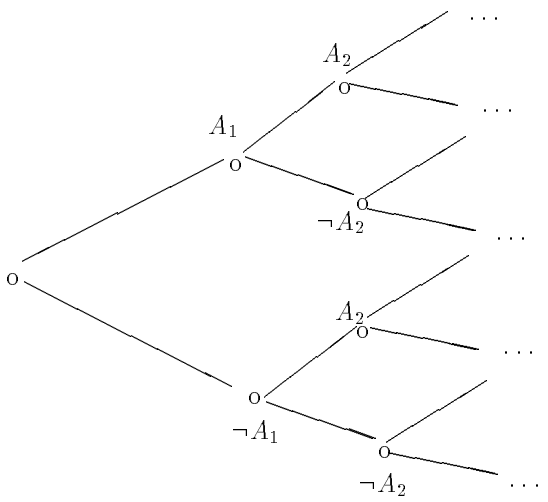
To be more precise, given S, first some procedure for enumerating all ground atoms

$$A_1, \ldots, A_n, \ldots$$

constructible from the vocabulary of S is defined. This determines a one-to-one correspondence between Herbrand interpretations I of S and sequences

$$l_1, \ldots, l_n, \ldots$$

where each l_i is either A_i, if A_i belongs to I, or $\neg A_i$, if A_i does not belong to I. The collection of all such sequences can be presented in the form of a binary tree:



When a new node l_n of the tree is generated, the new information l_n is assimilated into the theory consisting of S together with the earlier part of the branch $l_1, \ldots, l_{n-1}$. If $S \cup \{l_1, \ldots, l_n\}$ is inconsistent (i.e. the information on the branch so far already falsifies some clause in S), then the branch is terminated, and all possible input streams extending this branch are eliminated from further consideration.

The semantic tree procedure has all the hallmarks of a program specification. Although it is executable, it is inefficient, especially if the enumeration of atoms determining the growth of branches is not dependent on the structure of the clauses in S. Its efficiency can be greatly improved both by choosing an enumeration which is sensitive to the structure of the set of clauses and by appropriately choosing different enumerations on different branches.

Arguably, the semantic tree procedure is also a good semantics, because it considers all possible (syntactically characterised) meanings of the vocabulary occurring in a given set of clauses. It is not, however, model-theoretic, because it makes no assumptions about the existence of possible individuals, functions, and relations independently from the syntax of the language.

The semantic tree procedure can also be viewed as a form of reasoning by means of hypothetical cases, somewhat similar in spirit to the way in which a lawyer might argue in favour of a general principle by appealing

to imaginary cases. Such reasoning by means of cases is an important characteristic of legal reasoning and of practical reasoning in general. It may be tempting to understand such hypothetical, case-based reasoning in model-theoretic terms. The semantic tree procedure and the example of legal reasoning show that it can also be understood more simply in purely linguistic terms.

Curiously, the semantic tree procedure also has a purely proof-theoretic interpretation. Extending a branch by creating two new successor nodes can be interpreted as the application of a rule

$$\frac{\{A\} \cup S \vdash \bot \quad \{\neg A\} \cup S \vdash \bot}{S \vdash \bot}$$

Recognising that a partially constructed branch already falsifies some clause can be interpreted as the application of a complex, premise-free rule

$$\frac{}{\{A_1, \ldots, A_n, \neg B_1, \ldots, \neg B_m, C\} \cup S \vdash \bot}$$

where $A_1 \wedge \ldots \wedge A_n \rightarrow B_1 \vee \ldots \vee B_m$ is a ground instance of C.

Clearly, there is a one-to-one correspondence between a semantic tree demonstration that a set of clauses S is inconsistent and a proof of $S \vdash \bot$ using these deduction rules. Moreover, this correspondence between semantic tree demonstrations and deduction rule proofs is obviously very like the correspondence between the semantic and proof-theoretic interpretations of the semantic tableau procedure.

The examples of the semantic tree and semantic tableau procedures are not unique. Many other proof procedures are ambiguous and can be interpreted both semantically and proof-theoretically. For example, the model-elimination procedure [23], as its name implies, was originally understood in purely model-theoretic terms. Today this interpretation has largely been forgotten. On the other hand, the clausal theorem-prover SATCHMO [24] and its parallel variant MGTP [9] are still commonly regarded as model-theoretic procedures, even though, in my opinion, it is more useful to understand them purely proof-theoretically.

In the light of such examples, as Wilfred Hodges has observed [14], it is easy to understand how Johnson-Laird might argue that human beings reason in model-theoretic rather than in proof-theoretic terms.

6 From model theory to possible worlds

The model-theoretic approach to logic leads naturally both to the notion of possible worlds and to the enrichment of FOL by means of modal operators. The more pragmatically-oriented, CL approach, on the other hand,

leads to the use of a rich vocabulary of terms representing such entities as time, events, situations, and theories without leaving FOL and without introducing new logical operators.

Given the seemingly static view of the world inherent in the semantic structures of classical model theory, it is natural to view time as transforming one static state of the world into another. Thus the possible world semantics of modal logic replaces the simple model-theoretic interpretations of FOL by complexes of interpretations (possible worlds) connected by accessibility relations. In the simplest and most commonly occurring case, the concepts of time, events, states, and state transitions are present in the possible worlds semantic structures, but are absent from the language itself. In their place, the modal language contains modal operators, which are used to form sentences whose truth values are determined by reference to the accessibility relation between possible worlds. Thus, for example

> a sentence **future** P is true in a possible world I
> in a possible worlds structure W
> if and only if P is true in some possible world I'
> accessible from I in W.

The notion that a modal theory T logically implies a conclusion P is defined, analogously to the case for FOL, as holding if and only if

> for every world structure W and every
> possible world I in W, if T is true in I in W
> then P is true in I in W.

Modal logic and its possible world semantics seem to be adequate for reasoning about change in an unchanging environment. This is the case, for example, when reasoning about general properties of programs. In such a case, possible worlds correspond to possible program execution paths; and sentences true in all possible world structures correspond to program properties, such as correctness and termination, that hold no matter what execution path the program might take. That the modal language does not allow explicit reference to specific states of computation is not a limitation, because properties of specific states are not needed for proving general program properties.

The situation is quite different, however, when modal logic is used to record incoming observations that change over the course of time. For example, an observation that

> mary is at work

might be recorded by a simple sentence of the form

location (mary, work).

But, immediately after the observation has been completed, the further sentence

past location (mary, work)

would need to be added to the knowledge base. Both sentences would need to be retained until mary leaves work, at which time the first sentence would need to be deleted.

Maintaining modal sentences becomes even more complicated when they express expectations about the future. For example, should a sentence such as

future location (mary, home)

be deleted when the observation

location (mary, home)

is first recorded? Or, rather, should the expectation be retained as part of a default strategy which expects facts to persist until they are explicitly terminated? If the latter, then how should such default rules be formulated?

Because of such problems of maintaining modal sentences in a changing environment, many applications of modal logic abandon the notion of logical consequence and use possible world structures directly as temporal databases or knowledge bases instead. Incoming information about the present state is input into a possible world representing the present state. Information about the past or future is appropriately recorded in past or future possible worlds. As time changes, the possible world representing the present state changes accordingly.

Using possible world structures as knowledge bases is a natural extension of using classical models as relational databases. In the relational database case, the database consists only of ground atomic sentences. Sentences more general than ground atoms can occur only as queries or integrity constraints. They are evaluated by determining their truth values in the database regarded as a model-theoretic structure. By assuming that the database is complete (i.e. the 'closed world assumption'), a ground negative literal $\neg A$ is assumed to be true if A does not belong to the database.

By analogy with relational databases, the most obvious way to use possible world structures as databases or knowledge bases is to associate only ground atoms with possible worlds and to restrict more general sentences to queries and integrity constraints. Unfortunately, not only does this greatly restrict the kind of information that can be stored in a knowledge base, but it also involves enormous duplication. Facts, such as

location(mary, work)

that hold in several possible worlds must be included in all the worlds in which they hold. Moreover, general statements, such as

whenever mary stays late at work,
she drives the car to work in the morning,

do not fit into such a possible world structure.

Many authors have argued for using deductive theories rather than model-theoretic structures as databases. The most obvious advantage is that the database can then contain sentences more general than just ground atoms. Moreover, a relational database can be considered as a special case in which the deductive theory is a set of ground atoms augmented with a completeness assumption [11].

Similar arguments apply to possible world structures. A possible world structure can be regarded as a special case of a non-modal theory in which the fact that a relationship

$\langle\langle r, a_1 \ldots a_n \rangle\rangle$ holds in a possible world s

is represented by a sentence such as

holds $(\langle\langle r, a_1 \ldots a_n \rangle\rangle, s)$.

Such a non-modal approach to the representation of modal concepts is exemplified by McCarthy's situation calculus, Allen's interval logic, the event calculus of Kowalski and Sergot and many other formalisms for the representation of temporal information both in database systems and in artificial intelligence. Moreover, Gabbay [36] has developed a general methodology, called '*labelled deductive systems*' by means of which possible world structures can be translated directly into classical logic.

7 From possible worlds to situation semantics

Situation semantics can be viewed as arising from possible world semantics in a similar way to that in which possible world semantics can be viewed as arising from model theory. Whereas a possible world, in the context of temporal reasoning, can be viewed as representing an instantaneous time slice of an entire world state, a *situation* is most naturally viewed as representing partial information that cuts across time and space. Typical situations might include, for example,

> s_1, the situation consisting of all of Mary's activities at work on 1 April 1993, and s_2, the situation consisting of all the information I have about Mary.

Situations '*support*' items of information, which are semantic entities called *infons*, similar to the way in which relationships between individuals hold in the semantic structures of classical model theory and possible world semantics. In addition to recording 'ordinary' relationships between individuals, infons also record locations, times and polarities. A polarity of 1 indicates that a relationship holds; a polarity of 0 that it doesn't. For example

$$\sigma_1 = \langle\langle \text{location,mary, work, 1 April 1993, } 1\rangle\rangle$$
$$\sigma_2 = \langle\langle \text{location,mary, work, 1 April 1993, } 0\rangle\rangle$$

where the first argument, 'location', is the relation name, the third argument, 'work' or 'home', names the actual location and the fourth argument, '0' or 'l', indicates whether or not the relationship actually holds.

Compound infons can be constructed by means of conjunction, disjunction, universal and existential quantification. The only negation allowed in most treatments of situation semantics is that provided by the polarity '0', which can be associated with basic (non-compound) infons. (This restricted use of negation is similar to the restricted use of negation in the clausal form of logic.) However, there is normally no connective for constructing compound infons by means of implication (as in the standard treatment of clausal form, but not in the treatment presented in section 5).

To a first approximation [8], (the external form of) the mental state of an agent can be understood as a collection (or knowledge base) of *propositions* of the form

$$s \vDash \sigma$$

expressing that a situation s *supports* an infon σ. (The distinction between the external and internal form of a belief is discussed below in section 9.) Information that persists over time can be recorded, by means of a proposition such as

$$s_1 \vDash \forall T \in [10:00,\ 10:50]\langle\langle \text{lecturing, mary, work}, T, 1\rangle\rangle$$

for example. Situation semantics can also relate information about one situation to information about another, by means of 'constraints' between situation types. Such constraints serve the function of implication as a logical connective in ordinary FOL. For example, let S and T be the *situation types* (rather like sets)

$$S_1 = [\mathbf{s}_1 \mid \mathbf{s}_1 \vDash \langle\langle \text{location, mary, work, } \mathbf{t}, 1\rangle\rangle \wedge \langle\langle \text{late, } \mathbf{t}\rangle\rangle \wedge \langle\langle \text{day, } \mathbf{t}, \mathbf{d}\rangle\rangle]$$

$$S_2 = [\mathbf{s}_2 \mid \mathbf{s}_2 \vDash \exists T(\langle\langle \text{drives, mary, work, T, } 1\rangle\rangle \wedge T < \mathbf{t} \wedge \langle\langle \text{day}, T, \mathbf{d}\rangle\rangle)]$$

Then the proposition

$$w \vDash S_1 \Rightarrow S_2$$

expresses, as part of the information supported by the world state, w, that, for every situation in which mary works late, there exists a situation in which she drives to work earlier in the same day.

More generally, a *constraint* of the form

$$S \Rightarrow S'$$

expresses that if s is any situation of type S then there is a corresponding situation s' of type S', possibly extending s.

Constraints can also convey information about a single state. For example, the constraint

$$S_3 \Rightarrow S_4$$

where

$$S_3 = [\mathbf{s} \mid \mathbf{s} \vDash \langle\langle \text{kisses}, \mathbf{a}, \mathbf{b}, \mathbf{l}, \mathbf{t}, 1\rangle\rangle]$$
$$S_4 = [\mathbf{s} \mid \mathbf{s} \vDash \langle\langle \text{touches}, \mathbf{a}, \mathbf{b}, \mathbf{l}, \mathbf{t}, 1\rangle\rangle]$$

can be understood as expressing that if a person **a** kisses a person **b** at location **l** and time **t** in situation **s** then **a** touches **b** at location **l** and time **t** in the same situation **s**.

A knowledge base consisting of propositions in situation semantics is analogous to a possible worlds structure used directly as a temporal knowledge base. In both cases, logical consequence and proof procedures play no role. In both cases, the alternative is to use an ontologically rich vocabulary of terms representing time, events, and theories, to use theories (sets of sentences) as knowledge bases, and to use proof procedures to deduce logical consequences. In this alternative approach, situations are represented by theories, infons by ordinary sentences and the 'supports' relation by a metapredicate

$$\text{demo}\ (T, P)$$

which expresses that the conclusion named P can be demonstrated from the theory named T.

8 Combining object language and metalanguage

Metaprogramming is a powerful, commonly used technique for implementing expert systems, natural language processing systems, theorem-provers, interpreters and compilers in Prolog and in other logic programming languages. Its use has also been proposed for theory construction [4] (including

the construction and manipulation of modules viewed as theories), knowledge assimilation [18, 3] and the representation of knowledge and belief in multi-agent systems [22]. Many of these applications require that object language and metalanguage be combined, similarly to the way in which modal logic combines sentences with and without modal operators.

Following the results of Tarski [34], who showed that inconsistencies can arise when object language and metalanguage are combined in an unrestricted manner, it has been generally held that object language and metalanguage should be separated in an hierarchical fashion, so that self-reference can not occur. Moreover, Montague [25] and Thomason [35] showed that object language and metalanguage can not be combined consistently in the more restricted manner of modal logic. However, more recent studies (e.g. [26, 27, 7, 31]) indicate ways in which object language and metalanguage can be combined, provided that appropriate restrictions are imposed.

At least two other objections have been raised against systems that combine object language and metalanguage. One is that the naming conventions necessary to distinguish between object level expressions and their metalevel names are syntactically cumbersome. The other is that using syntactic expressions to represent intensional concepts, such as knowledge and belief, is too fine grained, in the sense that it distinguishes, as different beliefs, logically equivalent sentences that are trivial variants of one another.

One possible approach to the first objection is to abandon naming conventions altogether and to allow syntactic expressions to function as terms which name themselves. This is the approach taken informally in much Prolog programming practice (including the so-called non-ground naming of variables [13]) and more formally in the micro-Prolog programming language [6]. Semantic foundations for using syntactic expressions as their own names were laid by Richards [28] and Gabbay [10] and have been further developed by Jiang [15].

The use of syntactic expressions as their own names allows combined object-level metalevel sentences such as

$$\forall X, Y(X \leftarrow \text{ believes } (Y, X) \wedge \text{ wise } (Y)).$$

In common with other universally quantified sentences, such sentences can be understood as standing for the set of all their ground instances, e.g. for such instances as

likes (john, mary) $\leftarrow$ believes (john, likes (john, mary)) $\wedge$ wise (john).

The second objection, that the use of syntactic expressions to represent intensional concepts is too fine grained, has been partly addressed by

the discussion in section 5, where it was pointed out that clausal form (and other canonical forms) eliminates trivial syntactic distinctions between otherwise identical sentences. In this respect, the relationship between clausal form and the standard form of FOL might be regarded as similar to the relationship between the 'deep structure' which expresses the 'meaning' of natural language sentences and the 'surface structure' exhibited by the sentences themselves.

Nonetheless, syntactic representations of intensionality (even in canonical form) are much finer grained than modal representations. Thus, for example, the two sentences

$$\begin{array}{ll} \text{believes (john,} & \forall X \text{ (human } (X) \rightarrow \text{mortal } (X)) \wedge \text{human (john))} \\ \text{believes (john,} & \forall X \text{ (human } (X) \rightarrow \text{mortal } (X)) \wedge \text{human (john)} \\ & \wedge \text{ mortal (john))} \end{array}$$

are logically equivalent in modal logic, where 'believes' is a modal operator, because the two sentences

$$\forall X \text{ (human } (X) \rightarrow \text{mortal } (X)) \wedge \text{human (john)}$$
$$\forall X \text{ (human } (X) \rightarrow \text{mortal } (X)) \wedge \text{human (john)} \wedge \text{mortal (john)}$$

are logically equivalent. However, they are not equivalent in metalogic, where 'believes' is a metapredicate, unless they become so as a consequence of non-logical axioms such as

$$\text{believes } (T, P) \leftarrow \text{believes } (T, P \leftarrow Q) \wedge \text{believes } (T, Q)$$
$$\text{believes } (T, P \wedge Q) \leftarrow \text{believes } (T, P) \wedge \text{believes } (T, Q)$$

As Konolige [37] observes, the finer granularity of syntactic representations of belief potentially avoids the omniscience problem of conventional modal representations: that if an agent holds a belief then it holds all logical consequences of that belief. In fact, however, Konolige treats belief as a modal operator, but gives it a syntactic interpretation, in which an agent is regarded as holding a belief if (and only if) the agent can prove that belief from its 'knowledge base'. That the agent **a** can prove **p** is determined by an 'attachment rule', which associates a knowledge base $Kb_{\mathbf{a}}$ and inference system $\vdash_{\mathbf{a}}$ with **a** and shows that

$$Kb_{\mathbf{a}} \vdash_{\mathbf{a}} p$$

by directly applying the inference rules of $\vdash_{\mathbf{a}}$ to $Kb_{\mathbf{a}}$.

In metalogic programming it is natural to interpret the 'believes' predicate as a two argument proof predicate

$$\text{demo } (T, P)$$

where the first argument T names a theory (the knowledge base of an agent) and the second argument P names a sentence which is believed by the agent because it can be derived (or 'demonstrated') from T. The 'demo' predicate can be defined by such non-logical axioms as

(d1) $\text{demo}(T, P) \leftarrow \text{demo}(T, P \leftarrow Q) \wedge \text{demo}(T, Q)$

(d2) $\text{demo}(T, P \wedge Q) \leftarrow \text{demo}(T, P) \wedge \text{demo}(T, Q)$

identical (except for the different predicate symbol) to those for 'believes' given before. Alternatively, it can be implemented by means of an attachment (or reflection) rule, similar to that of the modal language of Konolige.

In CL, the theory parameter T of the 'demo' predicate names a set of clauses. A finite set of clauses can be represented either by a list or by a conjunction of the clauses in the set. The two representations are identical for conjunctions

$$C_1 \wedge \ldots \wedge C_{n-1} \wedge C_n$$

written in the canonical form

$$C_1 \wedge (\ldots \wedge (C_{n-1} \wedge (C_n \wedge \text{ true})) \ldots)$$

where '$\wedge$' functions as an infix list constructor and 'true' as a list terminator or empty list. That a sentence is provable from a set of sentences because it belongs to the set can be expressed by the non-logical axioms

(d3) $\text{demo}\ (P \wedge Q, P)$

(d4) $\text{demo}\ (P \wedge Q, R) \leftarrow \text{demo}\ (Q, R)$

similar to the axioms defining list membership.

Thus for example

$$\text{demo}\ ((p \leftarrow q) \wedge (q \wedge \mathit{true}), p)$$

can be proved by using (d3) and (d4) to show

$$\text{demo}\ ((p \leftarrow q) \wedge (q \wedge \text{ true}), p \leftarrow q)$$
$$\text{demo}\ ((p \leftarrow q) \wedge (q \wedge \text{ true}), q)$$

and then using (d1).

An alternative and often more useful way of representing theories and other syntactic objects is by means of constants [19]. Membership of a sentence in a set of sentences constituting a theory, represented by a constant,

can be expressed by means of appropriate non-logical axioms. Thus, if the constant c names the theory

$$\{c_1, \ldots, c_n, \ldots\}$$

then membership in the set can be defined, for example, by enumeration

$$\begin{array}{l} \text{demo}(c, c_1) \\ \quad \vdots \\ \text{demo}(c, c_n) \\ \quad \vdots \end{array}$$

The use of constants as names of theories can even be used for infinite sets of sentences. For example, the infinite set of ground, object level clauses of unbounded length

prime (2) ← true
prime (3) ← ¬ divides (2,3) ∧ true
⋮
prime (N+1) ← ¬ divides (N, N+1) ∧(...∧ (¬ divides (2, N+1) ∧ true)...)
⋮

can be named by a constant, say 'prime', and membership in the infinite set can be defined by the three clauses

demo (prime, prime (N+1) ← X) ← conditions (N,N+1, X)
conditions (1, N, true)
conditions (M+1, N, ¬ divides (M+1, N) ∧ X)← conditions (M, N, X)

Naming theories by constants is especially useful when the 'demo' predicate represents belief. In such cases it is not realistic to name a knowledge base by an explicit conjunction of sentences, either because the knowledge base is too large or because its complete contents are not known.

If an agent has a unique knowledge base (or set of beliefs), then the agent's name can conveniently double as the name of the knowledge base. Thus the metasentences

demo (john, $p \leftarrow q$)
demo (john, q)

can be interpreted as expressing that john both believes $p \leftarrow q$ and also believes q. From these sentences it is possible to derive the conclusion

demo (john, p)

using the clause (d1).

9 Situations as theories

From a purely formal point of view, much of situation semantics can be formalised in metalogic by using theories to represent situations, sentences to represent infons, and the 'demo' predicate to represent the 'supports' relation. There is even a formal correspondence between the definition of the 'demo' predicate and certain properties of the 'supports' relation, including for example such properties as

$$s \vDash \sigma_1 \wedge \sigma_2 \text{ iff } s \vDash \sigma_1 \text{ and } s \vDash \sigma_2$$

which is formally like the clause (d2) of the definition of 'demo'.

Unlike the use of constraints between situation types to represent conditional statements in situation semantics, CL uses ordinary implication instead. Thus a conditional statement such as

> 'If a person kisses a person at a
> time and a location, then the first
> person touches the second person at the
> same time and the same location'

can be formulated as an ordinary object level sentence

$$\text{kisses } (A, B, T, L) \rightarrow \text{touches } (A, B, T, L).$$

It can also be expressed at the metalevel, either in the form

(m1) $\text{demo } (S, \text{ kisses } (A, B, T, L)) \rightarrow \text{demo } (S, \text{ touches } (A, B, T, L))$

or in the form

(m2) $\text{demo } (S, \text{ kisses } (A, B, T, L) \rightarrow \text{touches } (A, B, T, L))$.

Whereas the first of these metalevel formulations is analogous to the formulation by means of a constraint in situation semantics, the second is analogous to the prohibited use of implication to construct compound infons. (m1) can be derived from (m2) using (d1).

Constraints between different situation types can also be formalised in CL by means of metalevel implications. For example, the constraint that

> for every situation in which
> Mary works late there exists a
> situation in which she drives to work
> early that same day

can be formulated by means of the metalevel statement

[demo(earlier(S), drive(mary, work, before(T)))
$\wedge$ before $(T) < T \wedge$ day (before(T), D)]
$\leftarrow$ demo (S, location (mary, work, T) $\wedge$ late $(T) \wedge$ day (T, D)).

Here the 'Skolem' function symbol 'earlier' constructs a name for the situation earlier(S) which exists as a function of the universally quantified variable S, thereby eliminating the need for an existential quantifier. Similarly, the function symbol 'before' avoids the use of an existential quantifier for the time before T which exists as a function of T.

In situation semantics, an infon can occur as part of a meaningful statement only in the context of a situation which supports it. In CL, on the other hand statements can be formulated at either the object level or the metalevel, as is most appropriate. Thus it would be simpler and possibly also more appropriate to formulate the connection between Mary' s working late and driving to work by the purely object level statement

[drive (mary, work, before $(T)) \wedge$ before $(T) < T \wedge$ day (before(T), D)]
$\leftarrow$ location (mary, work, T) $\wedge$ late $(T) \wedge$ day (T, D).

The possibility of formalising situation semantics in the metalogical component of CL glosses over an important philosophical difference between the two approaches. Situation semantics views and represents mental states of an agent and their relationship to the world objectively from an external 'theoretician's' point of view. CL, on the other hand, is conceived of as a mental language in which an agent subjectively constructs internal representations of its experience and beliefs and uses those representations to derive logical consequences.

Thus, for example, situation semantics would represent the external content of John's belief that it is raining by the proposition

$$s \vDash \langle\langle \text{raining}, t_0, 1 \rangle\rangle$$

as seen externally by the 'theoretician', where s is John's immediate environment at the time t_0 that he holds the belief. Devlin [8, p. 165], in discussing this example, denotes the internal structure of John' s belief by

$$\langle \text{Bel}, \text{-}, \text{raining}\sharp, \text{now}\sharp, 1 \rangle$$

where raining$\sharp$ is John's notion of raining, now$\sharp$ is John's notion of present time, and the dash in the second argument indicates that the belief is 'situated', i.e. does not itself involve a notion of the situation s that figures in the external content of John' s belief. This internal structure, which is neither an infon nor a proposition, is not of direct concern to situation theory.

In CL there is no 'theoretician' and no external content of beliefs, only agents and their internal representations of their own beliefs, as well as

their internal representations of other agents' beliefs. Thus, John might represent his own belief, that it is raining, in the object level form

$$\text{raining } (t_0)$$

where t_0 records the time of the event, or in the more informative form

$$\text{raining } (l_0, t_0)$$

where l_0 records the location of the event. For John's internal purposes these two parameters, l_0 and t_0, alone are likely to constitute an adequate indication of the situation in which the raining takes place.

Another agent, say Mary, might have her own representation of John's belief, perhaps in the form

$$\text{demo (john, raining } (l_0, t_0)).$$

This representation, while external to John, would be internal to Mary.

Agents may be inclined to associate objective status to their beliefs, regarding them as objectively 'true'. They may be similarly inclined to regard other agents' beliefs as 'true' if they accord with their own beliefs.

Thus, for example, if Mary believes

$$\begin{array}{l}\text{human (john)}\\ \text{mortal } (X) \leftarrow \text{human } (X)\\ \neg\ (\text{superhuman } (X) \wedge \text{ human } (X))\\ \text{demo (john,}\forall X\ (\text{mortal } (X) \leftarrow \text{human } (X))\\ \text{demo (john, superhuman (john))}\end{array}$$

then she will believe

$$\neg \text{ superhuman (john)}$$

as a logical consequence of her beliefs. Moreover, she will probably regard John' s belief that he is superhuman as false. Of course, John himself might not actually believe that he is superhuman. So, from John's point of view, Mary' s belief that John believes that he is superhuman would be false.

10 Conclusion

In this paper I have outlined an agent-centred, computationally-oriented, and purely syntactic account of the relationship between language and experience. In this account, an agent interacts with its environment through a constant stream of inputs, which it assimilates in the form of observational sentences into an evolving knowledge base of beliefs. Both the knowledge

base and the inputs are formulated as sentences in the agent's internal mental language.

The assimilation of inputs is constrained by the computational resources available. Consequently the agent' s knowledge base should be structured to make the best use of the limited computational resource. For the sake of efficiency, redundant derivations of the same conclusion should be avoided. For the sake of more effective problem-solving, beliefs which are more useful should be easier to derive than beliefs which are less useful.

The resource-constrained nature of an agent' s ability to derive logical consequences from its knowledge base is an essential aspect of its 'pragmatics', because what matters in practice is not whether a consequence follows from the knowledge base in the ideal case, but rather whether it follows in the case at hand. Thus, for example, a logically inconsistent knowledge having many useful consequences might well be more 'logical' than a consistent one which gives access to only few useful consequences, especially if the inconsistency is inaccessible in practice or if it can be prevented from polluting the rest of the knowledge base if and when it is found.

In the knowledge assimilation account of the relationship between language and experience, it is unnecessary and unhelpful to be concerned about the existence and the nature of the 'world' which generates the input stream. In this respect, the knowledge assimilation account diverges from model theory, which posits the existence of an external reality having a 'semantic' structure which is analogous to the syntactic structure of the language of the knowledge base.

I have argued also that the notion of logical consequence can be specified in purely syntactic, knowledge-assimilation terms, without the extra-syntactic structures of model theory. Not only is the syntactic specification executable, but it leads directly to more efficient and more conventionally defined proof procedures.

The model-theoretic view of logic leads naturally to possible world semantics and potentially to situation semantics. The knowledge assimilation view, on the other hand, leads to the employment of a syntactically rich language with a vocabulary of terms representing such objects as time, events, and theories. For this purpose it is necessary to combine object level and metalevel in the same language. Moreover, for the sake of simplicity and naturalness of expression, it is useful to allow syntactic objects to be named both by themselves and by constant symbols or other ground terms. The use of constant symbols as names of theories is especially useful for representing situations and other agents' knowledge bases as theories in the combined object-level and metalevel language.

This paper was written partly in reaction to Johnson-Laird' s theories about the model-theoretic nature of human deduction. His work and that of his colleagues have two parts: an experimental part which establishes

certain empirical data, and a theoretical part which attempts to explain the empirical results. I regret that I have not had time to investigate the extent to which the computationally-oriented CL approach to deduction and knowledge assimilation might provide an alternative explanation for the same empirical results. Nonetheless, I hope that I have drawn attention to some of the difficulties involved in assessing whether or not an agent understands natural language logically, and that I have raised some doubts about whether seemingly model-theoretic reasoning is truly model-theoretic and not simply proof theory in disguise.

I am aware of other holes I have left in my argument. I have said very little, for example, about how an agent might generate outputs which affect its environment and which have a subsequent affect on its own and other agents' future inputs. Clearly, such an output will normally be generated by some plan formation process in the context of the agent's 'resident goals'. The agent will record the output, predict its expected effect on the environment using its 'world model', and compare its expectations against its later observations. The relationships between inputs and outputs and between goals and actions, within the knowledge assimilation framework outlined in this paper, undoubtedly requires further investigation.

I have said very little, too, about the characteristics of CL which support such pragmatically important properties as relevance and even paraconsistency. Here I will mention again only that we should look for such properties to emerge as the result of the need for efficiency which arises from resource-constrained deduction. Thus, to make the best use of the limited resources available, both redundancies and irrelevancies have to be avoided as much as possible. In the case of resolution-based proof procedures,, irrelevancies are avoided both by focusing on the input and by eliminating the thinning rule, which allows $A \vee B$ to be derived from B. The elimination of thinning does not introduce a new logic, but simply makes classical logic more efficient. Resource limitations also mean that inconsistencies can exist without being detected and without leading to the derivation of arbitrary and irrelevant consequences. Such paraconsistency does not require a new logic, but simply emerges as a property of classical logic when it is used in a practical context.

I began this paper by referring to the multitude of different disciplines in which logic plays a central role. In this paper, I have explicitly considered only computing and artificial intelligence to any significant extent, and linguistics and psychology to a much lesser degree. However, two other disciplines, philosophy of science and law, have also contributed implicitly to the approach presented here. The notion of knowledge assimilation, in particular, owes much to the concepts of observational and theoretical sentences, confirmation, falsification, and explanation developed in philosophy of science. On the other hand the idea of a canonical language, CL, based

on resolution and logic programming, and combining object language with metalanguage has been greatly supported by investigations of legal reasoning and the formalisation of legislation [32, 20].

I also set out as my ultimate goal, in the introduction of this paper, to outline an approach to logic that could be used to explain human reasoning in both logical and computational terms. This goal was deliberately ambiguous with respect to explaining competence or performance, where *competence* is concerned with how humans ought to reason, and *performance* with how they actually reason in practice. I chose not to distinguish between these two goals because in the case of designing an artificial agent there is no reason why the two kinds of reasoning should be distinguished. Moreover, in the case of a human agent, it seems to me that the theory is applicable to both goals.

A performance theory of human reasoning would be interesting for scientific reasons. But from a purely practical point of view, a competence theory would be even more important, because it could be used by people to improve their own natural reasoning skills. This, after all, is the original and ultimate goal of logic, viewed as a discipline in its own right. It would be a pleasant irony if computationally-oriented logics, originally developed for use by machines, should also prove convenient for use by human beings.

Acknowledgements

I am grateful to Jon Barwise and Dov Gabbay for helpful discussions and for comments on an earlier draft of this paper. This work was supported both by the ESPRIT Basic Research project Compulog and by Fujitsu Research Laboratories.

References

[1] Alchourrón, C.E., Gärdenfors, P. and Makinson, D. On the Logic of Theory Change: Partial meet functions for contraction and revision. *Journal of Symbolic Logic*, **50**, 510–530, 1985.

[2] Barwise, J. and Perry, J. *Situations and Attitudes.* MIT Press, 1983.

[3] Bowen, K. A. and Kowalski, R. A. Amalgamating Language and Metalanguage in Logic Programming, in *Logic Programming* (Clark, K.L. and Tärnlund, S.-A., editors), Academic Press, pp. 153–173, 1982.

[4] Brogi, A., Mancarella, P., Pedreschi, D., Turini, F. Composition operators for logic theories, in *Proc. Symposium on Computational Logic*, Springer-Verlag, 1990.

[5] Clark, K. L. Negation by failure, in *Logic and Databases*, Gallaire, H. and Minker, J. [eds], Plenum Press, pp. 293–322, 1978.

[6] Clark, K.L. and McCabe, F.G. *Micro-Prolog: Programming in Logic*. Prentice Hall, 1984.

[7] des Rivires, J. and Levesque, H. J. The Consistency of Syntactical Treatments of Knowledge, in *Proceedings of the 1986 Conference on Theoretical Aspects of Reasoning about Knowledge*, (Halpern, J. editor), pp. 115–130, 1986.

[8] Devlin, K. *Logic and Information*. Cambridge University Press, 1991.

[9] Fujita, M., Hasegawa, R. Miyuki, K. and Fujita, H. Model Generation Theorem Provers on a Parallel Inference Machine. In *Proc. Fifth Generation Computer Systems 1992*, pp. 357–375, 1992.

[10] Gabbay, D. Metalevel features in the object level. In *Intensional Logics for Programming*, L, Fariñas del Cerro and M. Penttonen, eds. Oxford University Press, pp. 85–123, 1992.

[11] Gallaire, H., Minker, T. and Nicolas, J.-M. Logic and Databases: A Deductive Approach, *ACM Computing Serveys*, **16**, pp. 153–185, 1984.

[12] Gärdenfors, P. *Knowledge in Flux: Modelling the dynamics of epistemic states*. A Bradford Book, MIT Press, 1988.

[13] Hill, P. M. and Lloyd, J. W. Analysis of metaprograms, In *Metaprogramming in Logic Programming*, (H.D. Abramson and M.H. Rogers, editors), MIT Press, pp. 23–52, 1989.

[14] Hodges, W. The Logical Content of Theories of Deduction. Queen Mary College, London, 1993.

[15] Jiang, Y. On the semantics of real metalogic programming - a preliminary report, Technical report, Imperial College, London, 1993.

[16] Johnson-Laird, P.N. and Byrne, R.M.J. *Deduction*. Lawrence Erlbaum Associates Ltd. Hove, East Sussex, 1991.

[17] Kakas, A., Kowalski, R. and Toni, F. Abductive Logic Programming, *Journal of Logic and Computation*, **2**, 719–770, 1992.

[18] Kowalski, R. *Logic for Problem Solving*, North Holland Elsevier, 1979.

[19] Kowalski, R. Problems and Promises of Computational Logic, in *Proceedings Symposium on Computational Logic*, Springer-Verlag, pp. 1–36, 1990.

[20] Kowalski, R.A. Legislation as Logic Programs: in *Logic Programming in Action*. Springer-Verlag, pp. 203-230, 1992.

[21] Kowalski, R. and Hayes, P. J. Semantic Trees in Automatic Theorem-Proving, in *Machine Intelligence 4*, (eds. B. Meltzer and D. Michie), Edinburgh University Press, pp. 87–101, 1969. Also in *Anthology of Automated Theorem-Proving Papers*, Vol. 2, Springer-Verlag, pp 217–232, 1983.

[22] Kowalski, R. and Kim, J.S. A Metalogic Programming Approach to Multi-Agent Knowledge and Belief, in *Artificial Intelligence and Mathematical Theory of Computation* (V. Lifschitz, ed.) Academic Press, 1991.

[23] Loveland, D. *Automated Theorem Proving: A Logical Basis*, North Holland, 1978.

[24] Manthey, R. and Bry, F. SATCHMO: A theorem prover implemented in Prolog, in *Proc. Ninth Int. Conf. on Automated Deduction (CADE)*, 1988.

[25] Montague, R. Syntactical Treatments of Modality, with Corollaries on Reflection Principles and Finite Axiomatizability, *Acta Philosophic Fennica*, **16**, 153–167, 1963.

[26] Perlis, D. Languages with Self-Reference I : Foundations. *J. of Artificial Intelligence*, **25**, 301–322.

[27] Perlis, D. Language with Self-Reference II: Knowledge, Belief and Modality, *Artificial Intelligence*, **34**, 179–212.

[28] Richards, B. A Point of Reference. *Synthese*, **28**, 431–445, 1974.

[29] Robinson, J.A. A Machine Oriented Logic Based on the Resolution Principle. *J. ACM*, **12**, 23–41, 1965.

[30] Sadri, F. and Kowalski, R. A. A theorem proving approach to database integrity, In *Foundations of deductive databases and logic programming*, J. Minker, editor, Morgan Kaufmann, pp. 313–362, 1987.

[31] Sato, T. Metaprogramming through a truth predicate. In *Proc. JICSLP*, MIT Press, pp 526–540, 1992.

[32] Sergot, M. J., Sadri, F., Kowalski, R. A., Kriwaczek, F., Hammond, P. and Cory, H. T. The British Nationality Act as a logic program, *CACM*, **29**, 370–386, 1986.

[33] Sperber, D. and Wilson, D. *Relevance: communication and cognition*, Basil Blanckwell Ltd., Oxford, U.K, 1986.

[34] Tarski, A. Der Wahrheitsbegriff in den formalisierten Sprachen, *Studia Philosophia*, **1**, 261–405, 1936. English translation "The concept of truth in formalised languages" in A. Tarski *Logic, Semantics, and Mathematics*, Oxford, 1956.

[35] Thomason, Richmond H. A note on syntactic treatments of modality, *Synthese*, **44**, 391–395, 1980.

[36] Gabbay, D. Classical versus non-classical logic. In *Handbook of Logic in Artificial Intelligence and Logic Programming*, (D. Gabbbay, C. Hogger and J.A. Robinson, eds.) Oxford University Press, 1993.

[37] Konolige, K. *A Deduction Model of Belief*, Pitman Research Notes in Artificial Intelligence, Pitman, 1986.