Formal Reasoning and Language Understanding Systems

Raymond Reiter
Department of Computer Science
University of British Columbia

1. Introduction

Computational studies in linguistics have led to a variety of proposals for semantic representations of natural language. To a first approximation these all have a number of features in common. First, there is some formal language onto which, with the aid of a grammar, surface forms are mapped. Secondly, there is a formal language (usually, but not necessarily, the same as the first) for the representation of world knowledge and which is used to perform inferences necessary for integrating the surface form into the knowledge structure, and/or for answering questions. Finally, there is, or should be [5,18] a specification of the semantics of these formal languages.

There seem to be three dominant proposals for semantic representations:

(1) Procedural semantics [16,17] where the underlying representation consists of procedures in some executable language.

(2) Network structures [11,13,14] which represent knowledge by appropriate graphical data structures.

(3) Logical representation [3,7,12] which express world knowledge by formulae in some formal calculus.

These distinctions are not nearly as clear as one might like. Both logical and network representations often appeal to procedural components, networks appear to be representable as logical formlae via fairly direct mappings [15], while logical formulae have straight-forward procedural representations [6].

In this paper I shall discuss mechanisms for formal reasoning within logical representations. I shall make the (gross) assumption that surface forms have aleady been mapped onto some form of predicate calculus representation. In particular, I make no claims about the role or nature of the inferences required in mapping from surface structures to a logical deep structure. Neither do I take any position on the primitives of this deep structure. They may derive from a case oriented grammar, conceptual dependency theory etc. Ultimately, of course, the extent to which the choice of these primitives facilitates inference will be a factor affecting this choice. I take it as self evident that no semantic representation can explicitly contain all of the information required by a language understanding system so there is a need for inferring new knowledge from that explicitly represented. In this connection it is worth observing that, contrary to some prevailing opinions, formal reasoning does not preclude fuzzy or imprecise reasoning. There are no a priori reasons why notions like "probably", "possibly", etc. cannot be formalized within a logical calculus and new imprecise knowledge deduced from old by means of prefectly definite and precise rules of inference.

In the remainder of this paper I discuss two paradigms for formal reasoning with which I have worked - resolution and natural deduction - and argue in favour of the latter approach. I also indicate how other semantic representations - procedures and networks - might fit into this paradigm. Finally, I discuss some problems deriving from computational linguistics which have not been seriously considered by researchers in formal inference but which I think might fruitfully be explored within a logical framework.

2. Paradigms for Formal Reasoning

A. Resolution [10]

The resolution principle is based on five key concepts, two of which (the elimination of quanifiers through the introduction of Skolem functions, unification) are of particular relevance to problems in the representation of linguistic deep structures.

1) The elimination of quantifiers

One miht choose to assign to the statement "Every animal has a nose" a logical representation of the form

$$(x)(\exists y)[ANIMAL(x) \supset HAS\text{-}AS\text{-}PART(x,y) \wedge NOSE(y)] \quad (1)$$

As is well known, the sequence of quantifiers at the head of this formula is critical to its interpretation - interchanging them assigns a totally different meanng to the formula. Hence each quantifier is assigned a scope which, roughly speaking, is the maximal part of the formula governed by that quantifier. Unfortunately, the representation of quantifiers and their scope leads to some complexity in processing this information. (Anyone who has faced this problem in semanic net representations is well aware of the difficulties.) An elegant solution is to replace each existentially quantified variable (y in (1)) by a new Skolem function (which in (1) we can call "nose") whose arguments are all of the universally quantified variables (x in (1)) in whose scope the existential variable lies. (Thus y is replaced by nose(x) in (1)). Next, all of the quantifiers are deleted. The resulting formula is logically equivalent to the original. The quantifier-free formula of (1) is

$$ANIMAL(x) \supset HAS\text{-}AS\text{-}PART(x,nose(x)) \wedge NOSE(nose(x)) \quad (2)$$

The reduction of formulae to quanifier-free form also admits a primitive form of inference by pattern matching (unification).

(ii) Unification

In effect the unification algorithm answers questions like "Is formula A an

nstance (special case) of formula B?" or Is there a special case common to both A nd B?" Unification is simply consistent attern matching i.e. if a variable in one osition matches an expression, it must atch the identical expression in some other osition. Thus ANIMAL(x) unifies with NIMAL(fritz) under the substitution ritz|x. HAS-AS-PART(fritz,nose(fritz)) nifies with HAS-AS-PART(x,nose(x)) again ith fritz|x. P(x,f(x),y) unifies with (z,f(a),b) under the substitution |z,a|x,b|y, but fails to unify with (b,f(a),b).

iii) Canonical form for formulae

The resolution paradigm requires that a uantifier-free formula be converted to lausal form, i.e. a conjunct of disjuncts alled clauses. The conversion algorithm is uite straightforward involving Boolean ransformatons of the form $A \supset B \rightarrow \bar{A} \vee B$, $\bar{B} \rightarrow \bar{A} \vee \bar{B}$, $A \vee BC \rightarrow (A \vee B)(A \vee C)$ etc. The ormula (2) has two clauses in its canonical orm:

$\overline{\text{ANIMAL}}(x) \vee \text{HAS-AS-PART}(x, \text{nose}(x))$
$\overline{\text{ANIMAL}}(x) \vee \text{NOSE}(\text{nose}(x))$ (3)

iv) The resolution rule of inference

There is but one rule of inference in esolution theory: If $L1 \vee \alpha$ and $L2 \vee \beta$ are two lauses such that (a) L1 and L2 are omplementary literals. (A literal is a redicate symbol together with its rguments, or the negation of same. Two iterals are complementary if they have the ame predicate symbol, and one is unnegated hile the other is negated.)
b) The argument list of L1 is unifiable with hat of L2 under a substitution , then one an infer the new clause $(\alpha \vee \beta)\sigma$. For xample, if we know tha cats are animals, or n clausal form

$\overline{\text{CAT}}(y) \vee \text{ANIMAL}(y)$

hen by unifying ANIMAL(y) on its omplementary literal ANIMAL(x) in (3), we an infer

$\overline{\text{CAT}}(y) \vee \text{HAS-AS-PART}(y,\text{nose}(y))$
$\overline{\text{CAT}}(y)$ NOSE(nose(y)) (4)

.e. cats have noses. If in addition it is nown that CAT(fritz), then by unifying this n CAT(y) in (4), we can deduce the two lauses

HAS-AS-PART(fritz,nose(fritz)) (5.1)
NOSE(nose(fritz)) (5.2)

v) Completeness

Resolution is a refutation logic i.e. f T is some statement to be proved, the lausal form of its negation is added to the lauses representing the knowledge base, and n attempt is made to derive a contradiction y means of the single resolution inference ule. For exmple, to prove that Fritz has a ose i.e.
$\exists z)[\text{NOSE}(x) \wedge \text{HAS-AS-PART}(\text{fritz},z)]$
irst negate, yielding
$z)[\overline{\text{NOSE}}(z) \vee \overline{\text{HAS-AS-PART}}(\text{fritz},z)]$,
hen remove the universal quantifier which ields the clause
$\overline{\text{OSE}}(z) \vee \text{HAS-AS-PART}(\text{fritz},z)$. Resolving ith (5.1) yields $\overline{\text{NOSE}}(\text{nose}(\text{fritz}))$ which ontradicts (5.2).

Resolution is also complete. This means that if T is indeed logically valid (T is true under all possible interpretations in which the knowledge base is true) then there is a refutation proof of T with resolution as the sole rule of inference. There are two observations one can make here. The first is that resolutin is very much a competence model for formal inference. By no stretch of the imagination can it be construed as a performance model, in part because of its canonical representatin for formulae, in part because of its "unnatural" rule of inference. Secondly, by virtue of its completeness resolution is provably adequate as a competence model, in contrast with linguistic competence models for which the adequacy of any proposed theory is largely an empirical question.

It is the combination of representational security deriving from completeness and theoretical elegance deriving from the simplicity of the underlying logic that has led to so much intensive research into resolution. In particular, attempts to deal with the gross inefficiency of the theory have been largely syntactic, designed to constrain the possible inferences that can be made, but without sacrificing the completeness security blanket. Very little research has been devoted to the representation and use of domain knowledge, primarily, I think, because the ways in which humans use such knowledge have no correspondents within the resolution paradigm.

B. Natural Deduction Systems[1,8,9]

These can best be characterized as attempts to define a performance model for logical reasoning, in contrast to resolution as a competence model. In particular, any such model must make use of all of the domain specific "non-logical" knowledge available to a human, and make use of it in corresponding ways. Among the features of such systems are the following:

(i) Formulae are quantifier-free, but remain in their "natural" form. Thus, (1) is represented in the form (2), not as (3).

(ii) There are many (not just one) rules of inference, each corresponding to some observable inference mechanism used in human reasoning. Some examples: (grossly simplified. In particular the role of unification is suppressed.)
(a) Generalized modus ponens. If $A \wedge B \wedge C \supset D$ is a known fact, and if A,B and C are all known facts, then D may be deduced as a new known fact. If one of A,B or C is not known, no deduction is made.
(b) Back-chaining. If the current subgoal is to prove D, and if $W \supset D$ is known, then a possible next subgoal is to prove W.
(c) Case analysis. If $A \vee B$ is known, generate two cases, one a context in which A is assumed true, the other a context in which B is true, and proceed with the proof for each context.
(d) Splitting conjunctive subgoals. If

the current subgoal is to prove A∧B, first prove A, then prove B.

(e) Implicative subgoals. If the current subgoal is to prove A⊃B, update the current context with A, and prove B.

Quite a number of additional inference rules are possible. I have given a few examples only to indicate the flavour of the approach, and its naturalness. Some observations. First, the logic yields direct proofs, each of which must be provable assuming that its ancestor is provable. This property turns out to be critical for the application of domain specific knowledge for reducing search. (See (iii) below.) I know of no resolution logic with this property. Thirdly, the search for a proof proceeds by decomposing a problem into simpler problems as in rules (c), (d) and (e). Finally, there is an explicit representation of local contexts which prevents irrelevant formulae in adjacent contexts from polluting the local search. By way of contrast, resolution systems operate in a single global context.

(iii) Central to the natural deduction approach is it emphasis on the representation and appropriate use by the logic of domain specific knowledge. Examples of such knowledge are models, counterexamples, special cases etc. The fact that, as noted in (ii), each subgoal W must be provable provides the logic with a handle on how to use such knowledge. For if W or some special case of W is false in a model, or if there is a known counterexample to W, then there is no point in trying to prove it. If W is true in some model, or if it is possible to derive consequences of W which are known to be true, then there is additional evidence to warrant trying to prove it.

In some approaches [9] formulae in the knowledge base may have associated with them domain specific knowledge indicating how best to use that formula in the search for a proof. For example, in view of the enormous number of possible animals, there would be associated with CAT(y)⊃ANIMAL(y) the advice: If you are trying to prove that something is an animal and you don't currently know it to be a cat, don't try to prove it is a cat. The representation of this kind of knowledge clearly derives from the exhortations of the proceduralist [6].

(iv) Natural deduction systems are incomplete. This seems to be a necessary consequence of their emphasis on generating subgoals each of which must be provable. There are serious questions as to whether this is a satisfactory state of affairs. A facile argument has it that humans are necessarily incomplete (because of natural time and space bounds) so there is no need for computational logic to concern itself with this issue. However, for a logic to qualify as a performance model, it must be incomplete in precisely the ways that we are. The fact is that we overcome some of the limitations to time and space bounds by appealing to a variety of "non-logical" processes. Typical of these processes is the inspired guess which one encounters in mathematics whenever an induction hypothesis is proposed, or some obscure expressin is somehow pulled out of a hat to make a proof go through. One thing is certain. Neither the induction hypothesis, nor the expression was discovered by any process of pattern directed (via unification) search using the rules of inference of a logic, despite the fact that completeness guarantees the ultimate success of such a search. The difficulty with formulating an appropriate notion of completeness for a performance model is precisely in characterizing these non-logical processes and how they function in "completing" the underlying logically incomplete rules of inference. One of the virtues of natural deduction systems is that this distinction between logical and non-logical processes is made, and that it is possible in some fairly general situations for the logic to recognize when to invoke appropriate external routines [9].

3. The Two Cultures - Future Prospects

It is safe to say that there has been little communication between researchers in computational linguistics and formal inference. The justification seems to be that the former are concerned with performing shallow inference on large knowledge bases, whereas the latter focus on deep inference over relatively small domains. I believe this distinction is a superficial one, and that each discipline has much to gain from the problems and proposed solutions of the other. As an example of how a logical paradigm can be relevant to current ideas in computational linguistics, consider the relationship between semantic nets and logical representations.

Almost all of the question-answering systems that I know of use semantic nets for their inferencing component despite the fact that

(a) their semantics is by no means clear [18]

(b) there are serious difficulties in representing and processing quantifiers and their scopes

(c) no methods have been proposed for computing on a net which yield inferencing capabilities even remotely approximating those of a natural deduction system - capabilities which we know humans possess.

These are all non-problems for an appropriate logical system. Nevertheless, there are definite virtues to semantic nets as knowledge representations, especially their use in forming associations among concepts and their explicit representaion of superset links. It seems to me that there would be definite advantages to interfacing a natural deductive system with a semantic net, each component doing what it does best. In its simplest realization, imagine a net all of whose nodes denote nominal concepts and all of whose links denote "subset" or "superset". Within the logic, each variable and function symbol occurring in a formula is assigned a type which is the domain over

ich the varible is meant to range or the nge of the function symbol. Each such pe has a corresponding node in the net. r example, (2) would be represented as

HAS-AS-PART(x{ANIMAL},nose{NOSE}(x{ANIMAL} (6)

The general fact that cats are animals s no representation in the logical mponent, but is represented in the net by propriately linked CAT and ANIMAL nodes. w the question "Does Fritz have a nose?" anslates to an attempt to prove S-AS-PART(fritz{CAT}, y{NOSE}). If we uld unify this with (6) the question would answered. However, a term (in this case cannot unify with another term (fritz) less their types are compatible. To termine compatibility the unifier calls on e semantic net processor to check whether path of superset links connects node CAT node ANIMAL. In this case there is such path, so the unificaton succeeds.

Notice how each component benefits from e presence of the other. The logic nefits by processing fewer, and nsiderably more compact formulae than uld otherwise be necessary. (Compare (6) th (2)). In particular, compactification iminates many logical connectives, which s the effect of reducing the number of plications of rules of inference in riving a result. This is so because these les are "connective driven". Since search largely a matter of the nondeterministic plication of rules of inference, the arch space is reduced. Notice also that ie unifier is now responsible for some iference beyond that of simple pattern tching. From a search strategic point of ew there are sound reasons for encouraging iis transfer of logical power from the iles of inference to the unifier. Thus, ie unifier should also be responsible for aling generally with transitive and flexive relations by appealing to mputations on appropriate data structures iich represent these relations. The neral point of view here is that as much the inferencing as possible should be fected computationally rather than gically, leaving the logic to deal with difficult" problems. Given this view, a mantic net is just one of a whole class of ssible data structures which facilitate mputation as a substitute for certain nds of deduction. Assuming that it is ssible to isolate "what nets do best" the signer of a net is free to tune its presentation and procedures with respect a few well defined tasks without concern r its general inferencing abilities (or ack thereof).

Finally, it must be admitted that there e a host of problems deriving from inguistic considerations which have not ven been considered by researchers in rmal inference. Many of these problems, particular most of the "fuzzy" kinds of easoning described in [2], probably cannot e nicely incorporated in any paradigm for rmal inference. Nevertheless, there emain many interesting questions worth exploring within a logical framework.

(i) Other quantifiers. Logic has been content to deal with just two quantifiers - "there exists" and "for all". Natural language invokes a whole spectrum of quantifiers - "most of", "many of", "seven of", "a few of", etc. There is no difficulty in augmenting the syntax of a logical formalism with new quantifiers corresponding to these. The difficulty is in defining their semantics, and in specifying appropriate new rules of inference. It is possible, for example, to define "most-of" in some set theoretic formalism which effectively says "more than 80%", but I find this approach unsatisfying. A differenct approach, borrowing on the successful treatment of "there exists" in logic, might define "most-of" as a Skolem function with certain properties peculiar to our understanding of the meaning of "most of". Thus, one property of the "Skolem function" most-of is that it unifies with any term of the same type as the argument to most-of; the unifier returns the atom "probably". Thus, "Most dogs bark" becomes something like BARK(most-of(x{DOG})), and "Does Fido bark?" translates to BARK(fido{DOG}). Unification succeeds and we conclude something like PROBABLY(BARK(fido{DOG})). Clearly there are plenty of problems here not least what we mean by "probably", but the example gives the flavour of a possible logical approach, as well as an indication how certain kinds of "fuzzy" reasoning might be modeled in an extended logic.

(ii) Different levels of memory - contexts for wanting, needing etc. Consider representing "x wants P" in some logical formalism, where P is an arbitrary proposition. In specifying the properties of "WANT" we shall need (among other things) some kind of schema of the form

$WANTS(x,P) \wedge Q \supset$
WANTS(x, anything derivable from P and Q) (7)

where Q is an arbitrary proposition. This is unlike anything that researchers in formal inference have had to deal with. One possible approach, deriving from the context mechanism in natural deduction systems, is to maintain a variety of contexts, one containing formulae assumed universally true (the knowledge base), and for each individual x who wants something a context of all the formulae representing what x wants. Notice that within a want-context there is no commitment to the truth value of a formula - x may want a unicorn. The role of the schema (7) is assumed by the logic which knows which intercontextual inferences are legal.

(iii) Computation vs. deduction. This is a general problem involving the trade-off between the generality of deduction with its attendant inefficiency, and the use of highly tuned procedural specialists. My particular bias is that one cannot entirely do away with deduction, but that the logic should recognize if and when a deduction is best done procedurally, call the right specialist, and know what to do with the

results returned. This point of view is reflected in my earlier suggestion that one possible role for a semantic net is as a specialist for checking compatibility of types. Similarly, work in procedural semantics (e.g.[17]) can be viewed as complementary to deduction, not as an antithetical paradigm.

Ideally, what we want is "search-free" inference i.e. an appropriate collection of procedural specialists together with some supervisory system which knows which specialist to call, and when. If the specialists are "factored out" there is no logic left. The possibility of realizing this ideal seems to me remote, if only because mathematics is a human activity which does require formal inference and hence search. Consequently, it is important to better understand this trade-off between computation and deduction (or the particular and the general) and we can hope that in the future researchers in formal reasoning will clarify some of the issues. In this connection it is worth remarking that the distinction between computation and deduction is by no means clear [4].

REFERENCES

[1] Bledsoe, W.W., Boyer, R.S. and Henneman, W.H., Computer proofs of limit theorems, Artificial Intelligence, 3 (1972), pp. 27-60.

[2] Carbonell, J.R. and Collins, A.M., Natural semantics in artificial intelligence, Proc. Third IJCAI, Stanford University, Stanford, CA (1973), pp.344-351.

[3] Coles, L.S., An on-line question-answering system with natural language and pictorial input, Proc. ACM 23rd Natl. Conf. (1968), pp.157-167.

[4] Hayes, P.J., Computation and deduction, Proc. Symposium on the Mathematical Basis of Computation, Czech. Academy of Sciences (1973).

[5] Hayes, P.J., Some problems and non-problems in representation theory, Proc. AISB Summer Conf., University of Sussex,Brighton, U.K. (1974), pp.63-79.

[6] Hewitt, C., Description and theoretical analysis (using schemata) of PLANNER: A language for proving theorems and manipulating models in a robot, AI TR-258 (1972), AI Lab., M.I.T.

[7] McCarthy, J. and Hayes, P., Some philosophical problems from the standpoint of artificial intelligence, Machine Intelligence 4, Meltzer and Michie (eds), pp.463-502 (American Elsevier, NYC 1969).

[8] Reiter, R., A semantically guided deductive system for automatic theorem-proving, Proc. Third IJCAI, Stanford University, Stanford CA (1973), pp.41-46.

[9] Reiter, R., A paradigm for formal reasoning, Dept. of Computer Science, Univ. of British Columbia (forthcoming).

[10] Robinson, J.A., A machine oriented logic based on the resolution principle, J. ACM, 12 (1965), pp.23-41.

[11] Rumelhart, D.E. and Norman, D.A., Active semantic networks as a model of human memory, Proc. Third IJCAI, Stanford University, Stanford CA (1973), pp.450-457.

[12] Sandewall, E.J., Representing natural language information in predicate calculus, Machine Intelligence 6, Meltzer and Michie (eds), pp. 255-277.

[13] Schank, R.C., Identification of conceptualizations underlying natural language, Computer Models of Thought and Language, Schank and Colby (eds), pp.187-247, (W.H. Freeman and Company, San Francisco CA, 1973).

[14] Simmons, R.F., Semantic networks: their computation and use for understanding English sentences, Computer Models of Thought and Language, Schank and Colby (eds), pp.63-113.

[15] Simmons, R.F. and Bruce, B.C., Some relations between predicate calculus and semantic net representations of discourse, Proc. Second IJCAI, The British Computer Society, London (1971), pp.524-530.

[16] Winograd, T., Understanding Natural Language, Cognitive Psychology, 3, (1972).

[17] Woods, W.A., Procedural semantics for a question-answering machine, AFIPS Conf. Proc., FJCC, 33 (Part I), (1968), pp.457-471.

[18] Woods, W.A., What's in a link: Foundations for semantic networks, Representation and Understanding, Bobrow and Collins (eds), Academic Press (forthcoming).