# Natural Language Processing Using a Propositional Semantic Network with Structured Variables*

Syed S. Ali and Stuart C. Shapiro
Department of Computer Science
State University of New York at Buffalo
226 Bell Hall
Buffalo, NY 14260
{syali, shapiro}@cs.buffalo.edu

**Abstract**

We describe a knowledge representation and inference formalism, based on an intensional propositional semantic network, in which variables are structured terms consisting of quantifier, type, and other information. This has three important consequences for natural language processing. First, this leads to an extended, more "natural" formalism whose use and representations are consistent with the use of variables in natural language in two ways: the structure of representations mirrors the structure of the language and allows re-use phenomena such as pronouns and ellipsis. Second, the formalism allows the specification of description subsumption as a partial ordering on related concepts (variable nodes in a semantic network) that relates more general concepts to more specific instances of that concept, as is done in language. Finally, this structured variable representation simplifies the resolution of some representational difficulties with certain classes of natural language sentences, namely, donkey sentences and sentences involving branching quantifiers. The implementation of this formalism is called ANALOG (A NAtural Logic) and its utility for natural language processing tasks is illustrated.

## 1 Introduction

This work is based on the assumption that the kind of knowledge to be represented and its associated goals have a profound effect on the design of a knowledge representation and reasoning (KRR) system. In particular, we present a KRR system for the representation of knowledge associated with natural language dialog. We will argue that this may be done with minimal loss of inferential power and will result in an enriched representation language capable of supporting complex natural language descriptions, some discourse phenomena, standard first-order inference,

*This paper will appear in the journal *Minds and Machines* in a special issue on Knowledge Representation for Natural Language Processing. It is also available as technical report 93-16, from the Department of Computer Science, SUNY at Buffalo. A postscript version is available for anonymous ftp from `ftp.cs.buffalo.edu` in the file `/pub/tech-reports/93-16.ps`.

inheritance, and terminological subsumption. We characterize our goals for a more natural logic and its computational implementation in a knowledge representation and reasoning system below.

The mapping from natural language sentences into the representation language should be as direct as possible. The representation should reflect the structure of the natural language (NL) sentence it purports to represent. This is particularly evident in rule sentences, such as "small dogs bite harder than big dogs", where the representation takes the form of an implication.

$$\forall x, y((\text{small}(x) \wedge \text{dog}(x) \wedge \text{large}(y) \wedge \text{dog}(y)) \Rightarrow \text{bites-harder}(x, y)) \tag{1}$$

This is in contrast with the predicate-argument structure of the original sentence. By comparison, the representation of *Fido bites harder than Rover* is more consistent with the structure of the original sentence,

$$\text{bites-harder(Fido, Rover)} \tag{2}$$

This is so, despite the intuitive observation that the two sentences have nearly identical syntactic structure, and similar meaning.

The subunits of the representation should be what we term *conceptually complete*. By this we mean that any component of the representation of a sentence should have a meaningful interpretation independent of the entire sentence. For example, for the representation of the sentence as in (1) above, we might ask what the meaning of $x$ or $y$ is? Presumably, some thing in the world. Note that the original sentence mentions only dogs. We suggest that a better translation might be:

$$\text{bites-harder(all}\, x \text{ such that small-dog}(x), \text{all}\, y \text{ such that large-dog}(y))$$

where the variables, $x$ and $y$, would have their own internal structure that reflects their conceptualization. Note that we are suggesting something stronger than just restricted quantification (simple type constraints can certainly be felicitously represented using restricted quantifiers). Complex internalized constraints (that is, other than simple type) and internalized quantifier structures characterize this approach to the representation of variables. Thus representation of the sentence: *Every small dog that is owned by a bad-tempered person bites harder than a large dog* should reflect the structure of the representation of (2).

A high degree of structure sharing should be possible. In language, multi-sentence connected discourse often uses reduced forms of previously used terms in subsequent reference to those terms. This can be reflected in the representation language by structure sharing and corresponds to the use of pronouns and some forms of ellipsis in discourse. An example of this phenomenon is the representation of intersentential pronominal reference to scoped terms, e. g.,

> Every apartment had *a dishwasher*. In some of them *it* had just been installed.
> Every chess set comes with *a spare pawn*. *It* is taped to the top of the box.

(examples from [21]). The structures that are being shared in these sentences are the variables corresponding to the italicized noun phrases. Logical representations can only model this "sharing" by combining multiple sentences of natural language into one sentence of logic. This method is unnatural for at least two reasons. First, when several sentences must be combined into one sentence, the resulting logical sentence, as a conjunction of several potentially disparate sentences, is overly complex. Second, this approach is counter intuitive in that a language user can re-articulate the original sentences that he/she represents. This argues for some form of separate representations of the original sentences. The problem with logic in this task is that logic requires the complete specification of a variable, corresponding to a noun phrase, and its constraints in the scope of some quantifier. This difficulty is not restricted to noun phrases; indeed, it is frequently the case that entire subclauses of sentences are referred to using reduced forms such as "too", e. g.,

John *went to the party.* Mary did, *too.*

A language-motivated knowledge representation formalism should model this sort of reference, minimally by structure sharing.

Finally, collections of logical formulas do not seem to capture the intuitive use of concepts by people. This representation for knowledge is unstructured and disorganized. What is missing is that first-order predicate logic does not provide any special assistance in the problem of what Brachman called "knowledge structuring" [6], that is, the specification of the internal structure of concepts in terms of roles and interrelations between them and the inheritance relationships between concepts. Any computational theory must incorporate knowledge-structuring mechanisms, such as subsumption and inheritance of the sort supported in frame-based and semantic-network-based systems. For example, a taxonomy provides "links" that relate more general concepts to more specific concepts. This allows information about more specific concepts to be associated with their most general concept, so information can filter down to more specific concepts in the taxonomy via inheritance. More general concepts in such a taxonomy *subsume* more specific concepts with the subsumee inheriting information from its subsumers. For atomic concepts, subsumption relations between concepts are specified by the links of the taxonomy. A clear example of subsumption in natural language is the use of descriptions such as *person that has children* subsuming *person that has a son.* If one were told: *People that have children are happy,* then it follows that *People that have a son are happy.* The intuitive idea is that more general descriptions should subsume more specific descriptions of the same sort, which in turn inherit attributes from their more general subsumers.

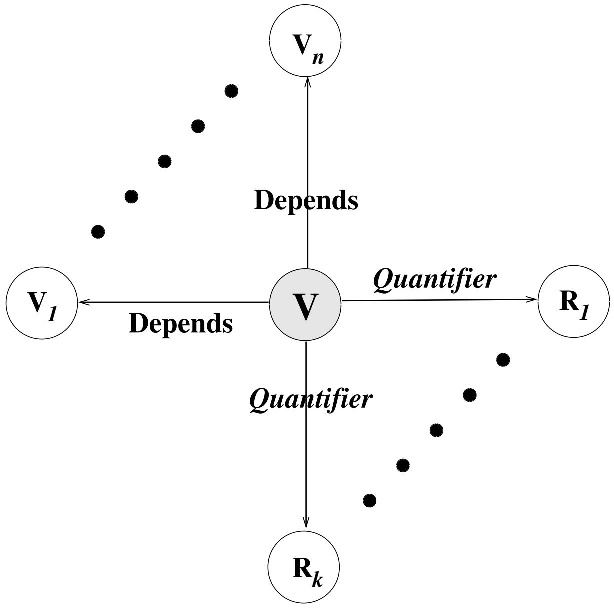
We have presented some general arguments for considering the use of a more "natural" (with respect to language) logic for the representation of natural language sentences. We have also presented some characteristics of natural language that a knowledge representation and reasoning system should support. In the remainder of this exposition, we will clarify the motivations for this work with specific examples, present an alternative representation for simple unstructured variables, and reify some aspects of the logic of these variables.

## 2 Structured Variables

We are attempting to represent variables as a "bundle" of constraints and a binding structure (quantifier). We term these bundles "structured variables" because variables, in this scheme, are non-atomic terms. The implemented language of representation is a semantic network representation system called ANALOG (A NAtural LOGic), which is a descendant of SNePS [41, 42].

Figure 1 gives a case frame proposed for the representation of variables in ANALOG. The shaded node labelled `V` is the structured variable. The restrictions on the variable are expressed by nodes $\mathtt{R}_1, \ldots, \mathtt{R}_k$. Scoping of existential structured variables (with respect to universal structured

**SYNTAX:**
If $V, V_1, \ldots, V_n$ $(n \geq 0)$ are distinct variable nodes, and if $R_1, \ldots, R_k$ $(k \geq 0)$ are proposition nodes, and all the $R_i$ dominate[a] $V$, then




is a network (actually a class of networks) and $V$ is a structured variable node.

**SEMANTICS:**
$[\![ V ]\!]^b$ is an arbitrary *quantifier*-constrained ($\forall$- or $\exists$-constrained) individual, dependent on $[\![ V_1 ]\!], \ldots, [\![ V_n ]\!]$, such that all the restriction propositions $[\![ R_1 ]\!], \ldots, [\![ R_k ]\!]$ hold for that arbitrary individual.

[a] One node dominates another if there is a path of directed arcs from the first node to the second node.
[b] $[\![ V ]\!]$ is the intensional individual denoted by $V$.

Figure 1: Caseframe for Structured Variables

variables) is expressed by the `depends` arcs to universal structured variable nodes $\mathtt{V}_1, \ldots, \mathtt{V}_n$. An example of a structured variable (the node labelled `V1`) is given in Figure 2 (described below). We note that the semantics of structured variables is left largely unspecified here, due to space limitations. However, it is an augmented (by the addition of arbitrary individuals) semantic theory based on [15, 16, 17, 41] and described in [1]. The representations of sentences, shown as examples in this paper use the case frames specified by [41].

# 3 Advantages of Structured Variables

## 3.1 Natural Form

We suggest that the representation of numerous types of quantifying expressions, using structured variables, is more "natural" than typical logics, because the mapping of natural language sentences is direct. We give an example in Figure 2 and defer the formal specification of nodes to Section 5.1.4. Node `M1!` corresponds to the asserted proposition that *all men are mortal*. Node `V1` is the structured variable corresponding to *all men*. Finally, node `M2` is the restriction proposition that corresponds to the arbitrary man being a member of the class of men. The `member-class` case frame is the representation for the proposition that an object is a member of a class. This representation is more natural in that the top-level proposition is one of class membership, rather than a rule-like if-then proposition. Note that any member-class proposition of the form "$X$ is $Y$" would be represented as a similar network structure (for example, the representation of *All rich young men that own a car are mortal* is given in Figure 19).

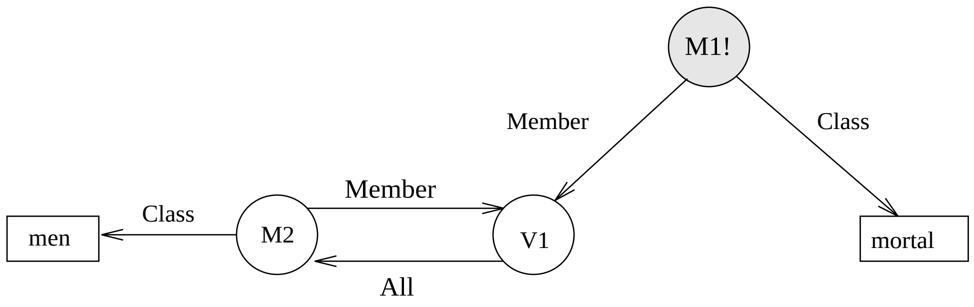



Figure 2: Structured Variable Representation of *All men are mortal.*

The representation of structured variables suggested here can represent most first-order quantifying expressions directly. Also, we can represent general quantifying expressions directly (although their semantics needs to be detailed). In general, there is a direct mapping from natural language quantifying expressions into structured variable representations, as structured variables correspond directly to noun phrases with restrictive relative clause complements. Figure 3 shows the fragment of the generalized augmented transition network (GATN) grammar that processes noun phrases corresponding to structured variables [37]. Note that in this NP subnetwork all restrictions on a noun phrase corresponding to a structured variable can be locally collected and the corresponding structured variable built. This includes restrictive relative clauses, whuch are parsed in the RREL subnetwork.

## 3.2 Conceptual Completeness

In typical logics, terms in one sentence are not referenced in other sentences. In general, re-using a term involves re-writing the term in the new formula. Ideally, we would like to re-use exactly

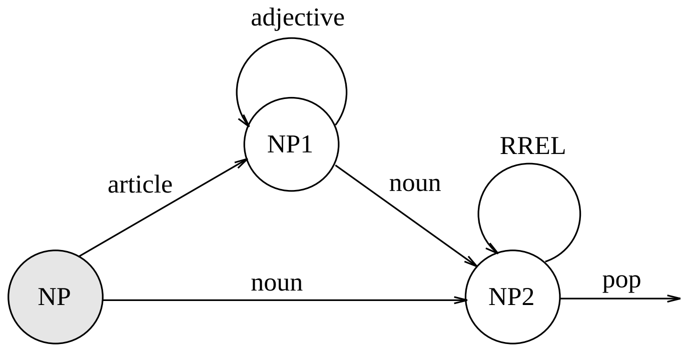



Figure 3: GATN NP subnetwork for noun phrases corresponding to structured variables.

the same terms in different sentences (in much the same way that language re-uses noun phrases), and we would want this re-use to result in closed sentences. This requires that variables (typically corresponding to noun phrases, or anaphora) in such terms be meaningful, independent of the sentence(s) that use or define them. We call such variables *conceptually complete*, and they may be shared by multiple sentences. This is an issue in the representation of multisentential dialog, where intersentential reference to sententially scoped objects frequently occurs. For example, *All cars come with a spare tire. It is not a full-sized tire*, can only be represented, in standard logics, by re-writing the terms corresponding to the shared variables in all representations of sentences that use them or combining several sentences into one representation. To see the difficultly, consider a representation of the example:

$$\forall x[\text{car}(x) \Rightarrow \exists y(\text{spare-tire}(y) \wedge \text{in}(y, x))]$$
$$\forall w\ \forall z[(\text{in}(w, z) \wedge \text{car}(z) \wedge \text{spare-tire}(w)) \Rightarrow \neg\text{full-sized}(w)]$$

In these cases, there is a clear advantage to a conceptually complete (closed) variable representation as well as the structure sharing associated with a semantic network representation. We would like the representation of the example to be:

There is a spare tire, $y$, in every car $x$

$y$ is not full-sized

using two distinct sentences; however, in the second sentence, $y$ is a free variable. With structured variables that contain their own binding structures (quantifiers), no open sentences are possible. Thus, in the example, the variable $y$ would not be free. Further, with structure sharing, distinct sentences may share constituents which would have been open sentences in a logical representation.

### 3.3 Quantifier Scoping

Any representation must account for the expression of quantifier scoping, at least as simply as the first-order predicate logic (FOPL) linear notation. The linear notation implicitly expresses a partial ordering of quantifiers and terms, by the order and bracketing of the quantifiers. With structured variables, quantifier scoping is expressed explicitly by the presence or absence of dependency arcs. However, the nonlinearity of the representation introduces a new problem in specifying limited quantifier scopings normally expressed by bracketing. For example, the differences in the sentences $\forall x\ \forall y\ (P(x,y) \Rightarrow Q(x))$ and $\forall x\ ((\forall y\ P(x,y)) \Rightarrow Q(x))$ are associated with bracketing and cannot be expressed (directly) in the nonlinear representation suggested here. The first sentence can be represented using structured variables; the second requires augmenting the representation, as its antecedent states a property of the collection of all $y$s. This should not be seen as a major shortcoming of this representation as several solutions to the problem are possible. Partitioned semantic networks [22, 23] or a representation for collections [11] would allow the representation of all possible quantifier scopings.

#### 3.3.1 Branching Quantifiers

There is a class of natural language sentences that are not expressible in any linear notation [3, 14, 33, 34]. The standard example is: *Some relative of each villager and some relative of each townsman hate each other.* A linear notation (as in standard logics) requires that one existentially quantified relative (of the townsman or villager) scope inside both universally quantified townsman and villager. This forces a dependency that should not be there, since the relative of the villager depends only on the particular villager and the relative of the townsman depends only on the particular townsman. Examples of these types of quantifiers are called *branching quantifiers*, because expression of their scoping required a tree-like notation (the Henkin prefix [24]). For example, the branched quantifier sentence could be expressed as in Figure 4.

Since the dependency arcs associated with structured variables can specify any partial order of the variables in a sentence, we may express any sort of dependency, including those of the type associated with branching quantifiers. Figure 5 illustrates how this is done for the relative-villager sentence. Nodes V1 and V3 represent the existential relatives that are scope dependent on nodes V2 (each villager) and V4 (each townsman), respectively.

$$\begin{array}{lll} \forall x - \exists y & & \\ & \backslash & \\ & & [(\text{villager}(x) \wedge \text{townsman}(z)) \Rightarrow (\text{relative}(x,\ y) \wedge \text{relative}(z,\ w) \wedge \text{Hates}(y,\ w))] \\ & / & \\ \forall z - \exists w & & \end{array}$$

Figure 4: Branching quantifier representation for *Some relative of each villager and some relative of each townsman hate each other.*

#### 3.3.2 Donkey Sentences

Another class of sentences that are difficult for first-order logics are the so-called *donkey sentences* [18]. These are sentences that use pronouns to refer to quantified variables in closed subclauses outside of the scope of the subclauses. In the example sentence *Every farmer who owns a donkey*

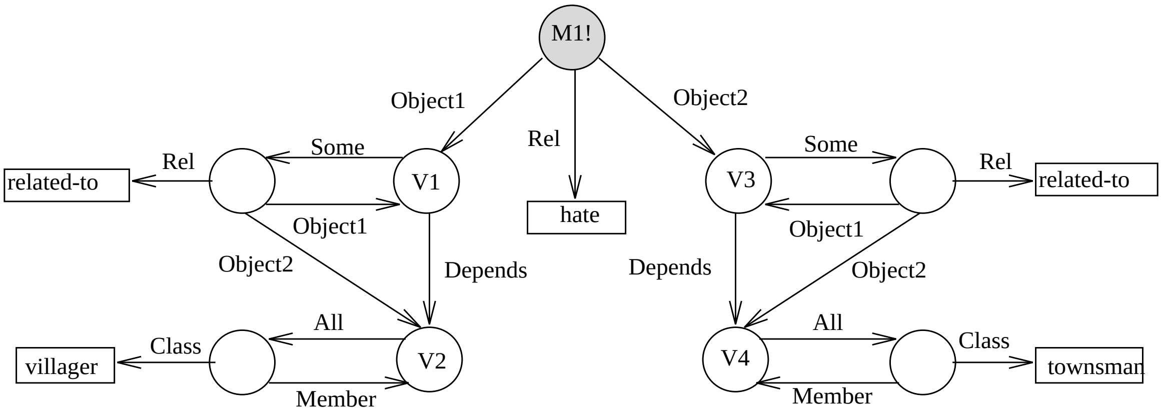



Figure 5: Representation for the Branched Quantifier Sentence: *Some relative of each villager and some relative of each townsman hate each other.*

*beats it*, the noun phrase *a donkey* is a variable inside the scope of a universally quantified variable (*every farmer*) and is referred to pronominally outside the scope of the existentially quantified donkey. Consider some attempts to represent the above sentence in FOPL:

(a) $\forall x$ (farmer($x$) $\Rightarrow \exists y$ (donkey($y$) & owns($x, y$) & beats($x$, $y$)))
(b) $\forall x$ (farmer($x$) $\Rightarrow$ (($\exists y$ donkey($y$) & owns($x, y$)) $\Rightarrow$ beats($x$, $y$)))
(c) $\forall x\ \forall y$ ((farmer($x$) & donkey($y$) & owns($x, y$)) $\Rightarrow$ beats($x$, $y$)))

Representation (a) says that *every* farmer owns a donkey that he beats, which is clearly more than the original sentence intends. Representation (b) is a better attempt, since it captures the notion that we are considering only farmers who own donkeys; however, it contains a free variable. Representation (c) fails to capture the sense of the original sentence in that it quantifies over *all* farmers and donkeys, rather than just farmers that own donkeys. To see this, consider the case of the farmer that owns two donkeys and beats only one of them. Clearly, the donkey sentence can apply to this case, but interpretation (c) does not.

Since there are no open subformulas, it is possible for formulas to use constituent variables at any level, including at a level which would result in an unscoped variable in an FOPL representation. Figure 6 shows the representation for *Every farmer who owns a donkey beats it.* Note that `M1!` denotes the proposition *Every farmer who owns a donkey beats it* and `V1` and `V2` are *every farmer that beats a donkey he owns* and *a beaten donkey that is owned by any farmer*, respectively.

# 4 Related Work

This paper suggests a hybrid approach to knowledge representation and reasoning for natural language processing by combining semantic network representations and structured object rep-

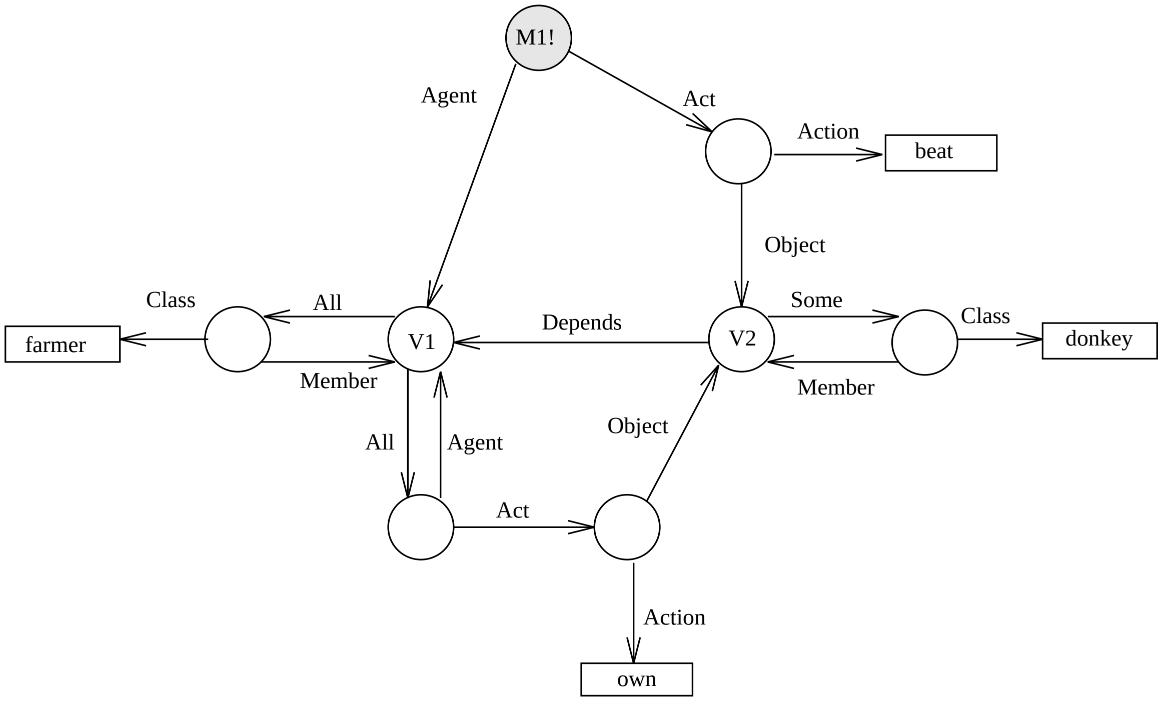



Figure 6: Representation for the Donkey Sentence: *Every farmer that owns a donkey beats it.*

resentations (corresponding to structured variables) to produce a KR formalism that addresses the previously outlined goals of knowledge representation for natural language processing. There is a large body of work that addresses similar issues. For comparative purposes, it is useful to categorize this related work into three groups: structured variable representations, atomic variable representations, and, hybrid variable representations.

There is a large body of work in structured object representation that characterizes very complex structured variables as frames, scripts, and so on. Frame-based systems such as KL-ONE and KRL use highly structured concept representations to express the "meaning" of concepts [5, 9, 45]. These concept representations are constructed using structural primitives. These highly structured objects, which typically consist of slots and fillers (with other mechanisms, such as defaults), can be viewed as complex structured variables that bind objects with the appropriate internal structure (although they are not, typically, so viewed). Their use of structural primitives allows the specification of a subsumption mechanism between concepts. A difficulty with these representations is that structured representations correspond directly to predicates in the underlying logic. Thus, constituents of a structured concept are not available as terms in the logic. In the donkey sentence, the donkey in *farmer that owns a donkey* cannot be used as a term.

An alternative to the frame-based, highly structured object representations is that of a logical form representation. The general motivation for a logical form is the need for a mediating representation between syntactic and meaning representations, usually in the context of determining quantifier scoping. Representative examples of logical-form-based approaches include [25, 27, 43]. Logical form representations resemble this work in that, typically, quantifiers are bundled with typed variables that are "complete" in the manner described here. Additionally, the utility of such representations lies in the relative ease of mapping natural language into a representation, which is also, clearly, a goal of this work. However, logical form is not a meaning representation, unlike the other representational work considered here. In general, logical form representations provide a "weakest" ambiguous interpretation that is subject to further computation before its meaning is apparent. It is possible to view this work as an "improved" logical form that has the advantage of having a "natural" mapping from language to representation. We improve on logical form, however, by including clear specification of some types of difficult quantifier scoping, incorporated into a propositional semantic network system allowing structure sharing (of nodes and terms) and cyclic structures of the sort seen in English, which are not easily represented in a linear notation. Further, we provide a subsumption mechanism not typically present in logical form work.

Previous work using atomic (that is, unstructured) variable representation has been primarily based on FOPL. In the work of Schubert et al. [35, 10], which uses a semantic-network-based formalism, variables are atomic nodes in the network. Type (and other) restrictions are specified by links to the variable nodes. There are no explicit universal or existential quantifiers. Free variables are implicitly universally quantified; Skolem arcs specify existentially quantified variable nodes. Because this is a nonlinear notation, sentences with branching quantifiers can be represented. However, the separation of variables from their constraints causes the representation to be not "natural" relative to the original natural language. Moreover, since restrictions on possible fillers for variables appear to be simple type restrictions, there is no representation for noun phrases with restrictive relative clause complements and, consequently, no representation for donkey sentences. Fahlman's [13] representation of variables is more general (potentially variables have complex structure) but has similar shortcomings.

An alternative atomic variable representation theory is that of Discourse Representation Theory [26]. DRT is a semantic theory that (among other things) accords indefinite noun phrases the status of referential terms rather than the standard quantified variables, and definite noun phrases the status of anaphoric terms. These terms are scoped by discourse representation structures (DRSs), and the theory provides rules to expand these DRSs based on the discourse being represented, as well as rules for interpreting the DRSs. DRT was directly motivated by the difficulties in the representation of donkey sentences and deals with them by making the scope of terms (variables) be the DRS rather than the sentence (proposition). DRSs themselves may scope inside other DRSs, creating a hierarchy of DRSs and scoped terms. The approach is similar to that of Hendrix's [22, 23] partitioned semantic networks. As with all the atomic variable representations, there is a separation of constraints from variables, and the form of DRSs is not "natural" in the same sense that a proposition that represented a sentence would be. Further, the rules of construction of DRS explicitly prohibit the representation of intersentential pronominal reference to scoped terms. Variables are still scoped (by the DRS) and not conceptually complete, although all their constraints are associated with the DRS in whose scope they lie.

An important philosophically motivated attempt to represent the semantics of natural language is Montague grammar [12, 31, 32]. Montague grammar is a theory of natural language processing in that it is a complete formal specification of the syntax, semantics, and knowledge representation for natural language understanding. Montague grammar mimics the syntactic structure of the surface sentence in specifying the mapping to logic and interpretation. In that, it resembles this work, but its coverage is far more ambitious than, and exceeds the scope of, the work in this paper. However, as a compositional semantic theory based on a higher-order intensional logic, it provides no inherent facility for the description of discourse relations and anaphoric connections [20]. Further, it suffers from the same (previously described) problems that all logic-based variable representations do. A related body of work is that of Barwise and Cooper [4] on *generalized quantifiers* in natural language. They attempt to represent and provide semantics for more general types of quantified natural language sentences (e. g., *many, most*) and specify a translation of a fragment of English using phrase structure rules. Their discussion of semantic issues related to these generalized quantifiers (which are, typically, manifested as noun phrases) forms a productive basis for any attempt to specify the semantics of quantified noun phrases.

Hybrid variable representations accord variables potentially complex internal structure. A representative system is the work of Brachman with KRYPTON [7, 8]. KRYPTON is a KR system that supports (and separates) two kinds of knowledge: terminological (in the TBox) and assertional (in the ABox). Since KRYPTON can represent complex descriptions in the TBox, in principle, general structured variables with arbitrary restrictions are possible in the TBox, with logic-based assertional representations in the ABox. Descriptions in the TBox are used in the ABox (syntactically as unary predicates or as binary relations). The form of the representation is more natural than FOPL, since restrictions on objects, which can take the form of complex terminological constraints in the TBox, are simple predicates (in the ABox) on those objects. However, variables are still atomic in the ABox and since sentences of the ABox are FOPL-based, KRYPTON cannot represent branched quantifiers or donkey sentences. Additionally, constituents of complex terms (concepts) in the TBox are not available in the ABox, so donkey sentences cannot be represented.

The Ontic system of McAllester [29] also provides the ability to define structured variables using a combination of type expressions and functions that reify these types into sets. Ontic is a system for verifying mathematical arguments, and, as such, the selection of type expressions and functions is limited to the mathematical domain. Additionally, Ontic is first-order and set-theoretic with quantification over terms (which may be variables of complex type). In principle, one could represent natural language in Ontic; however, the type system would have to be enriched, and it would still suffer from the disadvantages outlined for KRYPTON. In later work, McAllester has addressed natural language issues [19, 30] similar this work, particularly the natural form issue.

# 5 The Knowledge Representation Formalism

## 5.1 Syntax and Semantics of the Formalism

In this section, we provide a syntax and semantics of a logic whose variables are not atomic and have structure. We call these variables *structured variables.* The syntax of the logic is specified by a complete definition of a propositional semantic network representation formalism (an augmentation of [39]). By a propositional semantic network, we mean that all information, including propositions, "facts", etc., is represented by nodes. The implemented system, ANALOG, is used here, for convenience, to refer to the logical system.

### 5.1.1 Semantics

As a propositional semantic network formalism, any theory of semantics that ascribes propositional meaning to nodes can be the semantics used in ANALOG. In this paper, examples and representations are used that follow the case frame semantics of [41, 40] which provide a collection of propositional case frames and their associated semantics based on an extended first-order predicate logic. We augment that logic further with arbitrary individuals (for the semantics of structured variables) in a manner similar to the semantic theory of [15, 16, 17]. We will provide semantics for nodes, in this paper, as necessary. For a complete specification of the semantics of ANALOG, see [1, 2].

### 5.1.2 The Domain of Interpretation

ANALOG nodes are terms of a formal language. The interpretation of a node is an object in the domain of interpretation, called an entity. Every ANALOG node denotes an entity, and if $n$ is an ANALOG node, then $[\![n]\!]$ denotes the entity represented by $n$. It is useful, for discussing the semantics of ANALOG networks, to present them in terms of an "agent". Said agent has beliefs and performs actions, and is actually a model of a cognitive agent.

### 5.1.3 Metapredicates

To help formalize this description we introduce the metapredicates *Conceive*, *Believe*, and =. If $n, n_1, n_2$ are metavariables ranging over nodes, and $p$ is a metavariable ranging over proposition nodes, the semantics of the metapredicates listed above are:

- $Conceive(n)$ Means that the node is actually constructed in the network. $Conceive(n)$ may be true without $[\![n]\!]$ being known to be true or false.
- $Believe(p)$ Means that the agent believes the proposition $[\![p]\!]$.
- $n_1 = n_2$ Means that $n_1$ and $n_2$ are the same, identical, node.

Belief implies conception, as specified in axiom one.

**Axiom 1:** $Believe(p) \Rightarrow Conceive(p)$

### 5.1.4 Definition of Nodes

Informally, a node consists of a set of labeled (by relations) directed arcs to one or more nodes. Additionally, a node may be labeled by a "name" (e.g., `BILL, M1, V1`) as a useful (but extra-theoretic) way to refer to the node. This naming of a rule or proposition node is of the form M$n$, where $n$ is some integer. A "!" is appended to the name to show that the proposition represented by the node is believed. However, the "!" does not affect the identity of the node or the proposition it represents. Similarly, variable nodes are labeled V$n$ where $n$ is some integer, and base nodes are named B$n$ where $n$ is some integer (additionally, base nodes may be named for the concept they represent, e. g., man). More formally a node is defined as follows:

**Definition 1:** There is a none-empty collection of labelled atomic nodes called **base nodes**. Typically, base nodes are labelled by the entity they denote. *Example:* bill is a base node.

**Definition 2:** A **wire** is an ordered pair $<r, n>$, where $r$ is a relation, and $n$ is a node. Metavariables $w, w_1, w_2, \ldots$ range over wires. *Example:* <member, john> is a wire.

**Definition 3:** A **nodeset** is a set of nodes, $\{n_1, \ldots, n_k\}$. Meta-variables $ns, ns_1, ns_2, \ldots$ range over nodesets. *Example:* {john, bill} is a nodeset if john and bill are nodes.

**Definition 4:** A **cable** is an ordered pair $<r, ns>$, where $r$ is a relation, and $ns$ is a non-empty nodeset. Meta-variables $c, c_1, c_2, \ldots$ range over cables. *Example:* <member, {john, bill}> is a cable.

**Definition 5:** A **cableset** is a non-empty set of cables, $\{<r_1, ns_1>, \ldots, <r_k, ns_k>\}$, such that $r_i = r_j \iff i = j$. Meta-variables $cs, cs_1, cs_2, \ldots$ range over cablesets. *Example:* {<member, {john, bill}>, <class, {man}>} is a cableset.

**Definition 6:** Every node is either a base node or a cableset. *Example:* bill is a base node, {<member, {john, bill}>, <class, {man}>} is a cableset.

**Definition 7:** We overload the **membership** relation "$\in$" so that $x \in s$ holds just under the following conditions:

1. If $x$ is a node and $s$ is a nodeset, $x \in s \iff \exists y\ [y \in s \wedge Subsume(y, x).]$
   *Example:* M1 $\in$ {M1, M2, M3}

2. If $x$ is a wire such that $x = <r_1, n>$, and $s$ is a cable such that $s = <r_2, ns>$, then $x \in s \iff r_1 = r_2 \wedge n \in ns.$
   *Example:* <member, john> $\in$ <member, {john, bill}>

3. If $x$ is a wire and $s$ is a cableset, then $x \in s \iff \exists c[c \in s \wedge x \in c]$.
   *Example:* <member, john> $\in$ {<member, {john, bill}>, <class, {man}>}

Because we need more definitions before *Subsume* can be defined, we defer its definition to Figure 8.

**Definition 8:** A **variable** node is a cableset of the form {<all, $ns$>} (**universal variable**) or {<some, $ns_1$>, <depends, $ns_2$>} (**existential variable**). A variable node is further restricted in the form it may take in the following ways:

1. If it has the form {<all, $ns$>}, then every $n \in ns$ *must* dominate it.

2. If it has the form {<some, $ns_1$>, <depends, $ns_2$>}, then every $n \in ns_1$ *must* dominate it *and* every $n \in ns_2$ *must* be a universal variable node.

3. Nothing else is a variable node.

*Example:* `V1` = `{<all, {{<member, {V1}>, <class, {man}>}}>}` is the variable node corresponding to *every man*. The variable label `V1` is just a convenient extra-theoretic method of referring to the variable.

We define two selectors for variable nodes:

$$rest(v) = \begin{cases} ns & \text{if } v = \{<all, ns>\} \\ ns_1 & \text{if } v = \{<some, ns_1>, <depends, ns_2>\} \end{cases}$$

$$depends(v) = \ ns_2 \quad \text{if } v = \{<some, ns_1>, <depends, ns_2>\}$$

Informally, *rest*($v$) is the set of restriction propositions on the types of things that may be bound to the variable node $v$. *Depend*($v$) is the set of universal variable nodes on which an existential variable node, $v$, is scope-dependent.

**Definition 9:** A **molecular** node is a cableset that is *not* a variable node. *Example:* `{<member, {john, bill}>, <class, {man}>}` is a molecular node, since it is a cableset but not a variable node.

**Definition 10:** An **nrn-path** from the node $n_1$ to the node $n_{k+1}$ is a sequence,

$$n_1, r_1, \ldots, n_k, r_k, n_{k+1}$$

for $k \geq 1$ where the $n_i$ are nodes, the $r_i$ are relations, and for each $i$, $<r_i, n_{i+1}>$ is a wire in $n_i$. *Example:* If `M1` = `{<member, {john, bill}>, <class, {man}>}`, then `M1,member,john` and `M1,class,man` are some nrn-paths.

**Definition 11:** A node $n_1$ **dominates** a node $n_2$ just in case there is an nrn-path from $n_1$ to $n_2$. The predicate *dominate*($n_1$, $n_2$) which is true if and only if $n_1$ dominates $n_2$. *Example:* If `M1` = `{<member, {john, bill}>, <class, {man}>}`, then `M1` dominates `john`, `bill`, and `man`.

**Definition 12:** A **rule** node is a molecular node that dominates a variable node that does not, in turn, dominate it.

*Example:* `V1` = `{<all, {M1}>}`

`M1` = `{<member, {V1}>,<class, {man}>}`

`M2` = `{<member, {V1}>, <class, {mortal}>}`

`M2` is a rule node since `M2` dominates `V1`, which does not, in turn, dominate `M2`. `M1` is *not* a rule node because, while it dominates `V1`, it is also dominated by `V1`. The non-rule nodes that dominate variable nodes correspond to restrictions on binders of those same variable nodes.

### 5.1.5 The ANALOG model

**Definition 13:** An **ANALOG model** is a tuple ($A$, $B$, $M$, $R$, $U$, $E$, $\Gamma$) where $A$ is a set of relations, $B$ is a set of base nodes, $M$ is a set of non-rule molecular nodes, $R$ is a set of rule nodes, $U$ is a set of universal variable nodes, and $E$ is a set of existential variable nodes, and $\Gamma \subseteq M \cup R$. $B$, $M$, $R$, $U$, and, $E$ are disjoint. $\Gamma$ consists of believed propositions. Note that the metapredicates *Believe* and *Conceive* are, by definition:

$$Believe(n) \iff n \in \Gamma.$$
$$Conceive(n) \iff n \in M \cup R$$

### 5.1.6 Reduction

We follow [38, 39] in arguing for a form of reduction inference (defined in axioms 2 and 3 below) as being useful. This is a form of structural subsumption [44], peculiar to semantic network formalisms,s which allows a proposition to "reduce" to (logically imply) propositions whose wires are a subset of the wires of the original proposition. Figure 7 gives an example of a proposition expressing a brotherhood relation among a group of men. Node `M1` represents the proposition that `bill, john, ted,` and `joe` are brothers. By reduction subsumption, all proposition nodes (such as `M2` and `M3`) involving fewer brothers follow.

In the following model $(A,\ B,\ M,\ R,\ U,\ E,\ \Gamma)$:

$A =${`relation, arg`}
$B =${`bill, john, ted, joe`}
$M =${`M1, M2, M3`}
$\Gamma =${`M1, M2, M3`}

where:

```
M1 = {<relation, {brothers}>, <arg, {bill, john, ted, joe}>}
M2 = {<relation, {brothers}>, <arg, {john, ted, joe}>}
M3 = {<relation, {brothers}>, <arg, {bill, john}>}
```

Some reductions:

```
reduce(M2, M1) = T
reduce(M3, M1) = T
```

Figure 7: Example of Subsumption by Reduction for a Particular Model

However, we must restrict the use of reduction inference to precisely those propositions and rules which are reducible through the use of the *IsReducible* metapredicate.

**Axiom 2:** $Reduce(cs_1, cs_2) \iff (\forall w[w \in cs_2 \Rightarrow w \in cs_1] \wedge IsReducible(cs_1))$.

Note that the semantics of the metapredicate *IsReducible* will be specified in terms of the particular case frames used in a representation. Propositions like `M1` are clearly reducible, but not all propositional case frames are reducible. For example,

$$\forall x((\text{man}(x) \wedge \text{rich}(x)) \Rightarrow \text{happy}(x))$$

should not allow the reduction (involving fewer constraints on $x$):

$$\forall x(\text{man}(x) \Rightarrow \text{happy}(x))$$

as the latter does not follow from the former. Note that reduction is appropriate when the constraints in the antecedent of the rule are disjunctive.

A proposition that is a reducible reduction of a believed proposition is also a believed proposition. Since nodes are cablesets we state this as in axiom 3.

**Axiom 3:** $(Reduce(n_1, n_2) \wedge Believe(n_1)) \Rightarrow Believe(n_2)$

### 5.1.7 Types of Nodes

We have defined four orthogonal types of nodes: base, molecular, rule, and variable nodes. Informally, base nodes correspond to individual constants in a standard predicate logic, molecular nodes to sentences and functional terms, rule nodes to closed sentences with variables, and variable nodes to variables. Note that syntactically all are terms in ANALOG, however.

### 5.1.8 The Uniqueness Principle

No two non-variable nodes in the network represent the same individual, proposition, or rule.

**Axiom 4:** $n_1 = n_2 \iff [\![n_1]\!] = [\![n_2]\!]$

This is a consequence of the intensional semantics. A benefit of this is a high degree of structure-sharing in large networks. Additionally, the network representation of some types of sentences (such as the donkey sentence) can reflect the re-use of natural language terms expressed by pronouns and other reduced forms.

## 5.2 Subsumption

Semantic network formalisms provide "links" that relate more general concepts to more specific concepts; this is called a *taxonomy*. It allows information about concepts to be associated with their most general concept, and it allows information to filter down to more specific concepts in the taxonomy via inheritance. More general concepts in such a taxonomy *subsume* more specific concepts, the subsumee inheriting information from its subsumers. For atomic concepts, subsumption relations between concepts are specified by the links of the taxonomy. To specify subsumption, some additional definitions are required.

**Definition 14:** A **binding** is a pair $v/u$, where either $u$ and $v$ are both structured variables of the same type (universal or existential), or $u$ is a universal SV and $v$ is any node.

*Examples:* `V1/V2`
`JOHN/V1`

**Definition 15:** A **substitution** is a (possibly empty) set of bindings, $\{t_1/v_1, \ldots, t_n/v_n\}$.

*Examples:* `{V1/V2, JOHN/V3}`
`{B1/V1, M1/V2}`

**Definition 16:** The result of **applying** a substitution, $\theta = \{t_1/v_1, \ldots, t_m/v_m\}$, to a node $n$ is the instance $n\theta$ of $n$ obtained by simultaneously replacing each of the $v_i$ dominated by $n$ with $t_i$. If $\theta = \{\}$, then $n\theta = n$.

*Example:* If M1 = {<member, {V1}>, <class, {MAN}>} then:
M1{JOHN/V1} = {<member, {JOHN}>, <class, {MAN}>}

**Definition 17:** Let $\theta = \{s_1/u_1, \ldots, s_n/u_n\}$ and $\rho = \{t_1/v_1, \ldots, t_m/v_m\}$ be substitutions. Then the **composition** $\theta \cdot \rho$ of $\theta$ and $\rho$ is the substitution obtained from the set

$$\{s_1\rho/u_1, \ldots, s_n\rho/u_n, t_1/v_1, \ldots, t_m/v_m\}$$

by deleting any binding $s_i\rho/u_i$ for which $u_i = s_i\rho$.

*Example:* $\theta = \{$V1/V2, V4/V3$\}$
$\rho = \{$V2/V1, JOHN/V4$\}$
$\theta \cdot \rho = \{$JOHN/V3, JOHN/V4$\}$

**Definition 18:** A substitution, $\theta$, is **consistent** iff neither of the following hold:

$$\exists u, t, s[t/u \in \theta \wedge s/u \in \theta \wedge s \neq t]$$
$$\exists u, v, t[t/u \in \theta \wedge t/v \in \theta \wedge u \neq v]$$

A substitution that is not consistent is termed **inconsistent**. The motivation for the second constraint (called the unique variable binding rule, UVBR) is that in natural language, users seldom want different variables in the same sentence to bind identical objects [38]. For example, *Every elephant hates every elephant* has a different interpretation from *Every elephant hates himself.* Typically, the most acceptable interpretation of the former sentence requires that it not be interpreted as the latter. UVBR requires that within an individual sentence that is a rule (has bound variables), any rule use (binding of variables) must involve different terms for each variable in the rule to be acceptable.

*Examples:* {JOHN/V2, BILL/V2} is inconsistent.
{JOHN/V1, JOHN/V2} is inconsistent.
{JOHN/V1, BILL/V2} is consistent.

**Definition 19:** The predicate *occurs-in*$(x, y)$ where $x$ is a variable is defined:

$$\textit{occurs-in}(x, y) \iff \textit{dominate}(y, x).$$

*occurs-in* enforces the standard occurs check of the unification algorithm (and is just a more perspicacious naming of *dominate*) [28].

In ANALOG, we specify subsumption as a binary relation between arbitrary nodes in the network. We define subsumption between two nodes $x$ and $y$ in Figure 8. This definition of subsumption includes subsumption mechanisms that Woods classifies as *structural, recorded, axiomatic,* and *deduced* subsumption [44]. In Figure 8, case (1) corresponds to identical nodes (a node, obviously, subsumes itself). Case (2) is the reduction inference case discussed in section 5.1.6. Case (3)

*Subsume*$(x, y)$ in a model $(A,\ B,\ M,\ R,\ U,\ E,\ \Gamma)$ if any one of:

1. $x = y$.
2. *Reduce*$(x, y)$
3. For $x \in U$ and $y \in B \cup M \cup R$, if not *occurs-in*$(x,\ y)$ and there exists a substitution $S$ such that

$$\forall r[r \in rest(x),\ \ \Gamma \vdash r\{y/x\} \cdot S].$$

Logical derivation is here denoted by "$\vdash$." Substitution and substitution application with respect to a node is here denoted by "$r\{y/x\} \cdot S$."

4. For $x \in U$ and $y \in U \cup E$, if

$$\forall r[r \in rest(x) \Rightarrow \exists s[s \in rest(y) \wedge Subsume(r, s)]]$$

5. For $x, y \in E$, if all of the following hold:

$$\forall s[s \in rest(y) \Rightarrow \exists r[r \in rest(x) \wedge Subsume(r, s)]]$$
$$\forall r[r \in rest(x) \Rightarrow \exists s[s \in rest(y) \wedge Subsume(r, s)]]$$
$$\forall d[d \in depends(y) \Rightarrow \exists c[c \in depends(x) \wedge Subsume(c, d)]]$$

Otherwise, fail.

Figure 8: Subsumption Procedure

applies when a universal structured variable node subsumes another node. This corresponds to a description like *any rich man* subsuming *John* if *John* is known to be a man and rich. Such a variable will subsume another node if and only if every restriction on the variable can be derived (in the current model) for the node being subsumed. Subsumption, consequently, requires derivation which is defined in [1]. For the examples shown here, standard first-order logical derivation will be assumed. Case (4) allows a more general universal variable node to subsume a less general existential variable node. For this to happen, for every restriction in the universal variable node there must be a restriction in the existential variable node, and the former restriction must subsume the latter restriction. For example, the variable node corresponding to *every rich girl* would subsume *some rich happy girl* (but not *some girl*). Case (5) allows one existential variable node to subsume another. The requirement for this case is, essentially, that the variables be notational variants of each other. This is because it is not, in general, possible for any existential variable to subsume another except when they are structurally identical. The reason this case is needed (rather than just excluding it entirely) is that for a rule node corresponding to *every boy loves some girl* to subsume *every rich boy loves some girl*, the existential variable node corresponding to the *some girl* in the first rule node must subsume the existential variable node corresponding to the *some girl* in the second rule node (see Section 6 for numerous examples of this sort of subsumption in natural language).

The commonsense nature of the subsumption cases can, most easily, be illustrated by examples, some of which, for conciseness, are given in natural language in Figure 9. The examples illustrate

Case 1: $Subsume$(`john`, `john`)

Case 2: $Subsume$({`<member`, {`john, bill`}`>`, `<class`, {`man`}`>`},
{`<member`, {`john`}`>`, `<class`, {`man`}`>`})
$Subsume$(All rich men are happy,
All young rich men that own a car are happy)

Case 3: $Subsume$(All dogs, Fido), Provided $\vdash$ dog(Fido).

$Subsume$(All things have a mass, Fido has a mass)

Case 4: $Subsume$(Every girl, Every pretty girl)
$Subsume$(Every pretty girl, Every pretty girl that owns a dog)
$Subsume$(Every girl, Some pretty girl)
$Subsume$(Every pretty girl, Some pretty girl that owns a dog)

Case 5: $Subsume$(Every boy that loves a girl,
Every boy that loves a girl and that owns a dog)

Figure 9: Examples of Subsumption (cases refer to cases of Figure 8)

the idea that more general quantified descriptions should subsume less general quantified descriptions of the same sort. In Figure 10, a more detailed example for a particular model is given. Node `M2` represents the proposition that *all men are mortal*, `M3` the proposition that `Socrates is a man`, and `M4` the proposition that `Socrates is mortal`. `V1` is the structured variable representing any man. It then follows that `M4` is a special case of `M2` directly by subsumption, since `V1` subsumes `Socrates`. Note that the restrictions on subsumption involving variables is stricter than *Reduce*, which only requires that the wires of one node be a subset of the other.

As with reduction (Axiom 3), a proposition that is subsumed by a believed proposition is also a believed proposition. This can be stated as a more general form of Axiom 3.

**Axiom 5:** $(Subsume(n_1, n_2) \wedge Believe(n_1)) \Rightarrow Believe(n_2)$

### 5.3 Summary

We have formally specified the subsumption mechanism in the ANALOG system. The subsumption mechanism takes advantage of the conceptual completeness of the structured variable representation to allow the kinds of common sense description subsumption relationships that are pervasive in natural language.

In the following model ($A$, $B$, $M$, $R$, $U$, $E$, $\Gamma$):

$A$ ={`member, class, all`}
$B$ ={`man, mortal, Socrates`}
$M$ ={`M1, M3, M4`}
$R$ ={`M2`}
$U$ ={`V1`}

where:

```
M1 = {<member, {V1}>, <class, {man}>}
M2 = {<member, {V1}>, <class, {mortal}>}
M3 = {<member, {Socrates}>, <class, {man}>}
M4 = {<member, {Socrates}>, <class, {mortal}>}
V1 = {<all, {M1}>}
```

The resulting subsumption:

```
subsume(M2, M4) = T
```

Figure 10: Example of Subsumption for a Particular Model

# 6 ANALOG for Natural Language Processing

So far, we have motivated some aspects of the logic underlying the ANALOG KRR system and formalized some important concepts, such as subsumption, associated with the logical system. At this junction, we will attempt to illustrate the utility of the system in the context of specific examples of natural language processing.

ANALOG includes a generalized augmented transition network (GATN) natural language parser and generation component linked up to the knowledge base (based on [37]). A GATN grammar specifies the translation/generation of sentences involving complex noun phrases into/from ANALOG structured variable representations.

We present three demonstrations of the NLP component of ANALOG. The first illustrates the representation and use of complex noun phrases, the second illustrates the use of non-linear quantifier scoping and structure sharing, and the last is a detailed presentation (with most of the underlying ANALOG representations) of a demonstration that illustrates the use of rules and valid and useful answers to questions. The last two demonstrations also have examples of subsumption.

## 6.1 Representation of Complex Noun Phrases

One of the most apparent advantages of the use of structured variables lies in the representation and generation of complex noun phrases that involve restrictive relative clause complements. The restriction set of a structured variable typically consists of a type constraint along with property constraints (adjectives) and other more complex constraints (restrictive relative clause complements).

```
(parse -1)
ATN parser initialization...
Input sentences in normal English orthographic convention.
Sentences may go beyond a line by having a space followed by a <CR>
To exit the parser, write ^end.
: Every man owns a car
I understand that every man owns some car.
: Every young man owns a car
I understand that every young man owns some car.
: Every young man that loves a girl owns a car that is sporty
I understand that every young man that loves any girl owns some sporty car.
: Every young man that loves a girl that owns a dog owns a red car that is sporty
I understand that every young man that loves any girl that owns any dog owns
some red sporty car.
: Every young man that loves a girl and that is happy owns a red sporty car that wastes gas
I understand that every young happy man that loves any girl owns some
sporty red car that wastes gas.
: ^end
ATN Parser exits...
```

Figure 11: Examples of complex noun phrase use that correspond to structured variables

In Figure 11, user input is italicized, the text at the beginning is a standard message and will be omitted from the remaining figures. Figure 11 shows example sentences with progressively more complex noun phrases being used. These noun phrases are uniformly represented using structured variables. Parsing and generation of these noun phrases is simplified because structured variables collect all relevant restrictions on a variable into one unit, a structured variable. The parser parses the user's sentence and builds an ANALOG representation for the user input. The resulting representation is then passed to the generation component, which generates the output response (sometimes prefixed by the canned phrase `I understand that`). If constraints on variables corresponding to the complex noun phrases were represented using FOPL, then it would be difficult to generate natural language noun phrases corresponding to these variables. This is because the constraints on variables would, likely, be well-separated from the variables in the antecedents of rules involving these variables. This is not the case in a structured variable representation.

## 6.2 Non-Linear Quantifier Scopings and Structure Sharing

Since this representational formalism is grounded in an inherently nonlinear notation (semantic networks), the representation of tree-like quantifier scopings is straightforward. Thus, sentences involving branching quantifiers can be represented. In addition, the structure of ANALOG's semantic network representation allows structure-sharing (and, indeed, may require it) to a high degree. One of the initial goals was the representation of co-reference and structure sharing in multisentential dialog. Because the parser/generator maintains a very simple discourse model, we cannot illustrate complex examples (such as the branching quantifier sentence) of this structure sharing in natural language (although such representation may readily be built). A simpler example is the donkey sentence, where a scoped constituent of a noun phrase (the *donkey* in *Every farmer who owns a donkey beats it*) is used in the main clause of the sentence. Figure 12 illustrates a dialog involving questions about the donkey sentence.

: *Every man that owns some donkey beats it*
`I understand that every man that owns some donkey beats some donkey.`
: *Fred is a boy*
`I understand that Fred is a boy.`
: *Every boy is a man*
`I understand that every boy is a man.`
: *Doc is a donkey*
`I understand that Doc is a donkey.`
: *Dumbo is a donkey*
`I understand that Dumbo is a donkey.`
: *Fred owns Doc*
`I understand that Fred owns Doc.`
: *Fred owns Dumbo*
`I understand that Fred owns Dumbo.`
: *Does Fred beat Doc*
`I don't know.`
: *Does any man beat some donkey*
`Yes, every man that owns some donkey beats some donkey.`
: *Every man that owns any donkey beats it*
`I understand that every man that owns any donkey beats every donkey.`
: *Does Fred beat Doc*
`Yes, Fred beats Doc.`
: *Does any man beat some donkey*
`Yes, every man that owns some donkey beats some donkey.`
: *Does any man beat any donkey*
`Yes, Fred beats Doc and every man that owns any donkey beats every`
`donkey and every man that owns some donkey beats some donkey.`
: *Who beats a donkey*
`Fred beats Doc and every man that owns any donkey beats every donkey`
`and every man that owns some donkey beats some donkey.`

Figure 12: Example of structure sharing in donkey sentence.

In Figure 12 note that the system is initially unable to determine whether Fred beats Doc or Dumbo. This is because the initial rule (`every man that owns some donkey beats it`) is satisfied in a model where only one of the donkeys is being beaten. After the system is told that *all* such donkeys are beaten, it does determine that Fred beats Doc. Note that this determination also requires that the description `every man` subsume `Fred` who is a boy (and consequently a man). This is an example of derived subsumption. Also note that the answers to many questions are often rules themselves (e. g., `Who beats a donkey` has as one answer `Every man that owns some donkey`). This is discussed in the next section.

| | |
|---|---|
| : *Every man is mortal* | *(1)* |
| `I understand that every man is mortal.` | *(2)* |
| : *Who is mortal* | *(3)* |
| `Every man is mortal.` | *(4)* |
| : *Is any rich man mortal* | *(5)* |
| `Yes, every rich man is mortal.` | *(6)* |
| : *John is a man* | *(7)* |
| `I understand that John is a man.` | *(8)* |
| : *Is John mortal* | *(9)* |
| `Yes, John is mortal.` | *(10)* |
| : *Who is mortal* | *(11)* |
| `John is mortal and every rich man is mortal and every man is mortal.` | *(12)* |
| : *Are all rich young men that own some car mortal* | *(13)* |
| `Yes, every young rich man that owns some car is mortal.` | *(14)* |
| : *Any rich young man that owns any car is happy* | *(15)* |
| `I understand that every young rich man that owns any car is happy.` | *(16)* |
| : *Is John happy* | *(17)* |
| `I don't know.` | *(18)* |
| : *Young rich John owns a car* | *(19)* |
| `I understand that mortal rich young John owns some car.` | *(20)* |
| : *Who owns a car* | *(21)* |
| `Mortal rich young John owns some car.` | *(22)* |
| : *Is John happy* | *(23)* |
| `Yes, mortal rich young John is happy.` | *(24)* |

Figure 13: Examples of questions that have rules as answers.

### 6.3 Rules as Answers to Questions

Because the structure of the representation of rules is "flat", that is, there is not the artificial antecedent-consequent structure associated with first-order logic-based representations, it is possible to frame questions whose answers are rules and not just ground formulas. Since the structure of the question will mirror the structure of the rule, any rule that is subsumed by a question is an answer to that question. Figure 13 gives a sample dialog involving questions whose answers are ground propositions (e. g., *Is John mortal*) as well as questions whose answers are rules (e. g., *Who is mortal*). This dialog also illustrates the uses of subsumption. Since we told the system *Every man is mortal*, it follows that any more specifically constrained man (e. g., *Every rich young man that owns some car*) must also be mortal. Note that this answer (a rule) follows directly by subsumption from a rule previously told to the system. This is another way in which rules may be answers to questions.

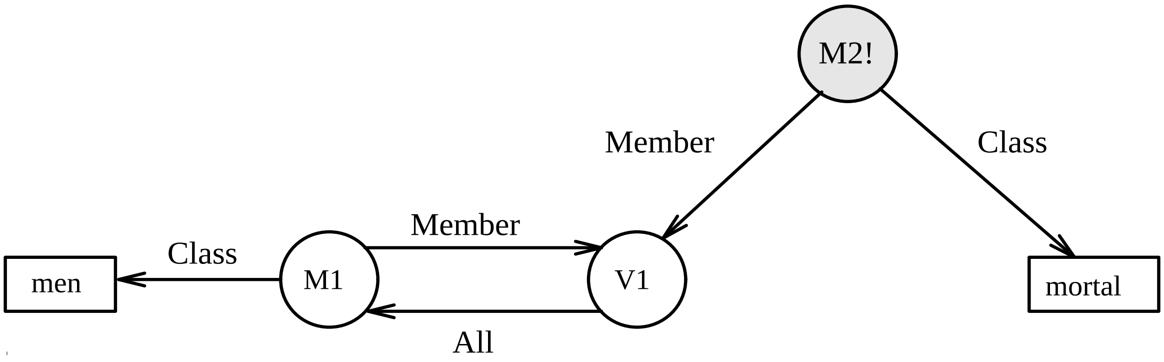



Figure 14: Representation of sentence (1) *Every man is mortal.*

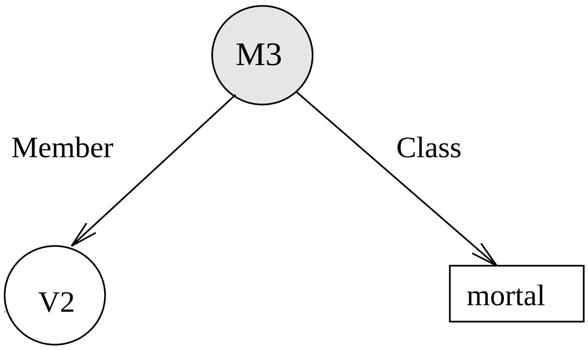



Figure 15: Representation of sentence (3) *Who is mortal?*

#### 6.3.1 A Detailed Demonstration Examination

In this section we present the representations and processing associated with the last demonstration in detail. All references to sentences will be to the numbered sentences in Figure 13. The representation for sentence (1) is that of Figure 14. Sentence (3) then asks who is mortal. In a standard FOPL-based system, no answer could be given because there are, as yet, no instances of men in the knowledge base. This is contrary to the commonsense answer of sentence (4), which reiterates the rule of sentence (1). This is possible in ANALOG because the structure of the representation of the question (*Who is mortal*) is similar to that of any of its answers. Thus, any asserted proposition that is subsumed by the question is a valid answer (including rules).

Sentence (5) is an example of a question about a rule. Since *every man is mortal* is believed (the system was told this in sentence (1)) it follows that any more restricted sort of man is also mortal. The subsumption procedure specifies this explicitly. The representation of sentence (5) in Figure 16 is a less general form of sentence (1) in Figure 14, since `V1` (any man) subsumes `V3` (any rich man). Since the rule in Figure 16 is subsumed by a believed node (that of sentence (1)), it follows by Axiom 5 that sentence (5) is believed (thus, the representation of the question itself is a believed proposition) and the system answers yes to the question. Sentence (7) asserts that *John is a man*, and the result is the representation of Figure 17. At this point, the system knows *all men are mortal* and *John is a man*. When the question of sentence (9) (whose representation

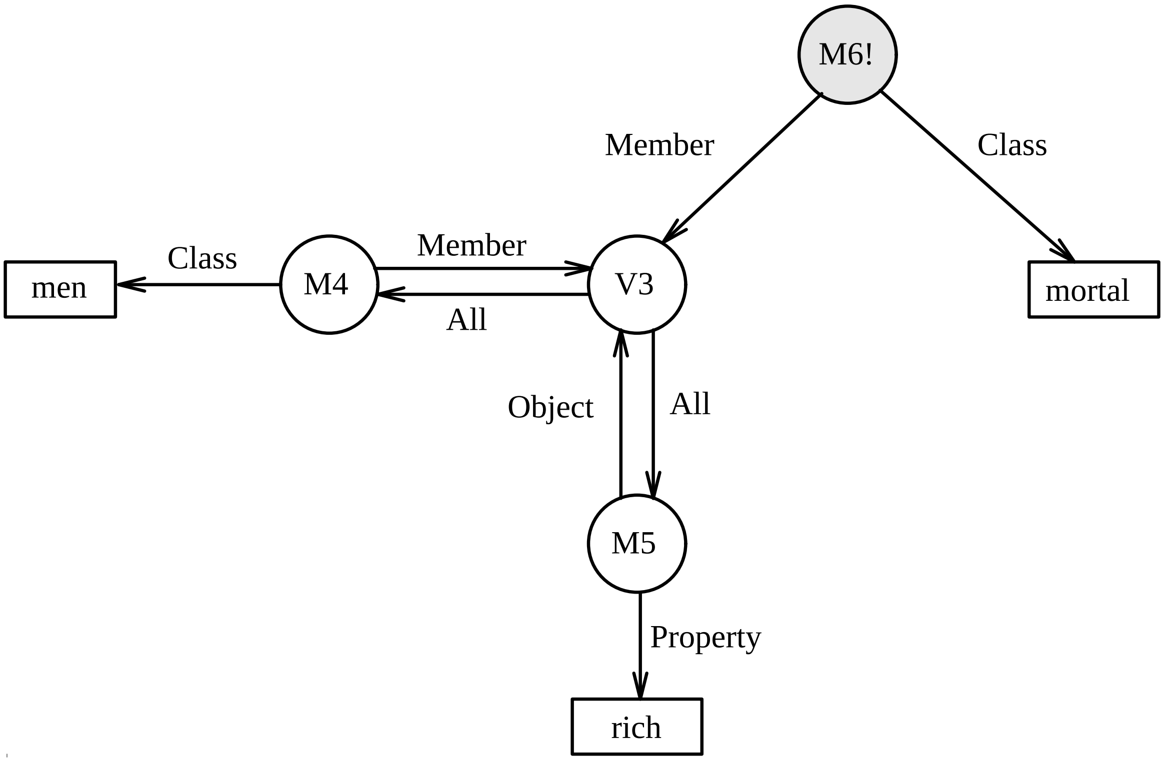



Figure 16: Representation of sentence (5) *Is any rich man mortal?*

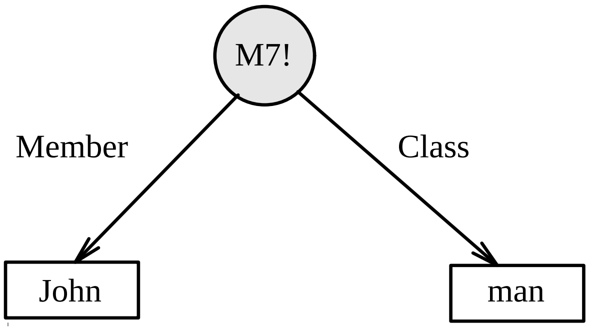



Figure 17: Representation of sentence (7) *John is a man.*

is in Figure 18) is asked, the system finds the rule of sentence (1) and determines that it subsumes sentence (9) because John is a man, and again by axiom 5 the result follows. However, note that in this derivation the result is a ground formula rather than a rule. Sentence (11) illustrates the retrieval of the system's information about who is mortal; note the additional believed propositions. Sentence (13) is an example of a more complex noun phrase in a rule. The representation of (13) is in Figure 19 and is subsumed by that of sentence (1) or (5) leading to the yes answer. In Figure 19, `V4` represents the arbitrary rich young man that owns some car, and `V5` represents some owned

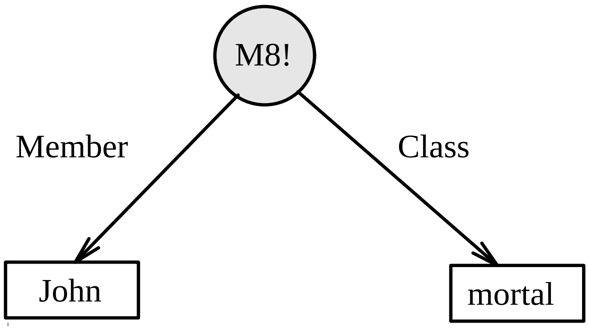



Figure 18: Representation of sentence (9) *Is John mortal?*

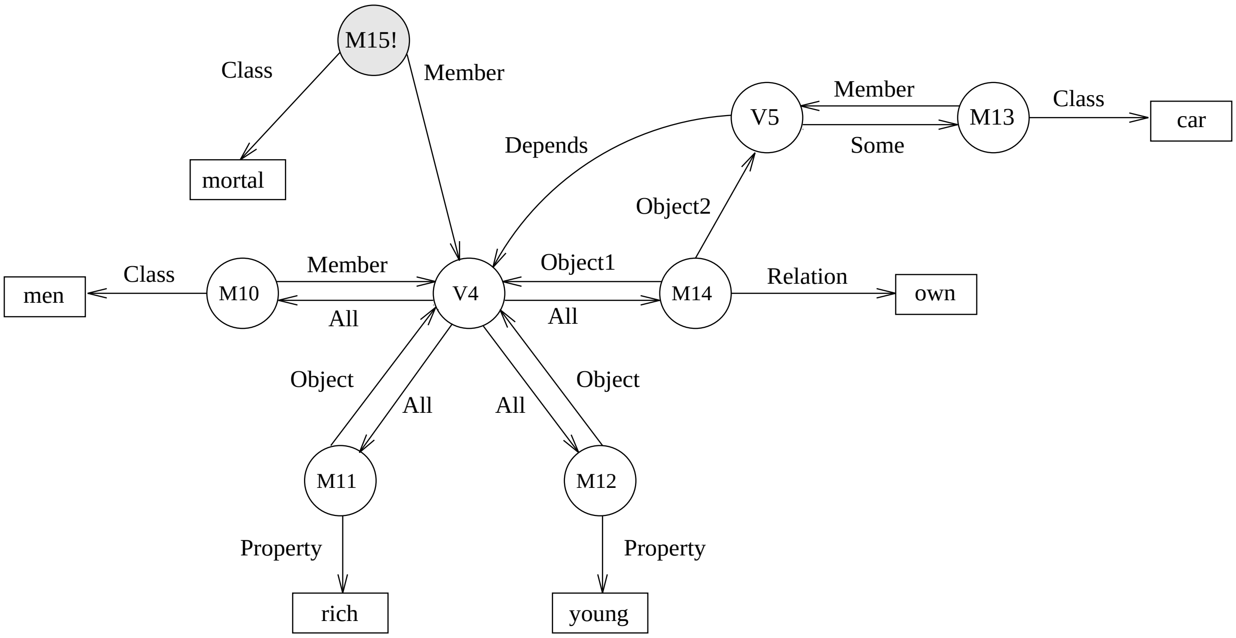



Figure 19: Representation of sentence (13) *Are all rich young men that own some car mortal?*

car of `V4`. In sentence (15) (Figure 20), a new rule about rich young car-owning men (`V6`) being happy (`M21`) is introduced. The question of sentence (17) (*is John happy*) cannot be answered, because the system cannot determine that node `V6` subsumes `John`. This is because, while `John` is a man, he is not known to be young, rich, and owning a car (requirements for this subsumption). Sentence (19) informs the system of these requirements the systems understanding is verified by question (21), whose representation is shown in Figure 22. Note that the structure of the question involving two variables (*Who* and *a car*) is identical to that of the structure of its answer, which would not be the case if constraints were separated from variables in the antecedents of rules (as is done in typical logics). The question is asked again and, because the subsumption relationship can be determined, is answered in the affirmative.

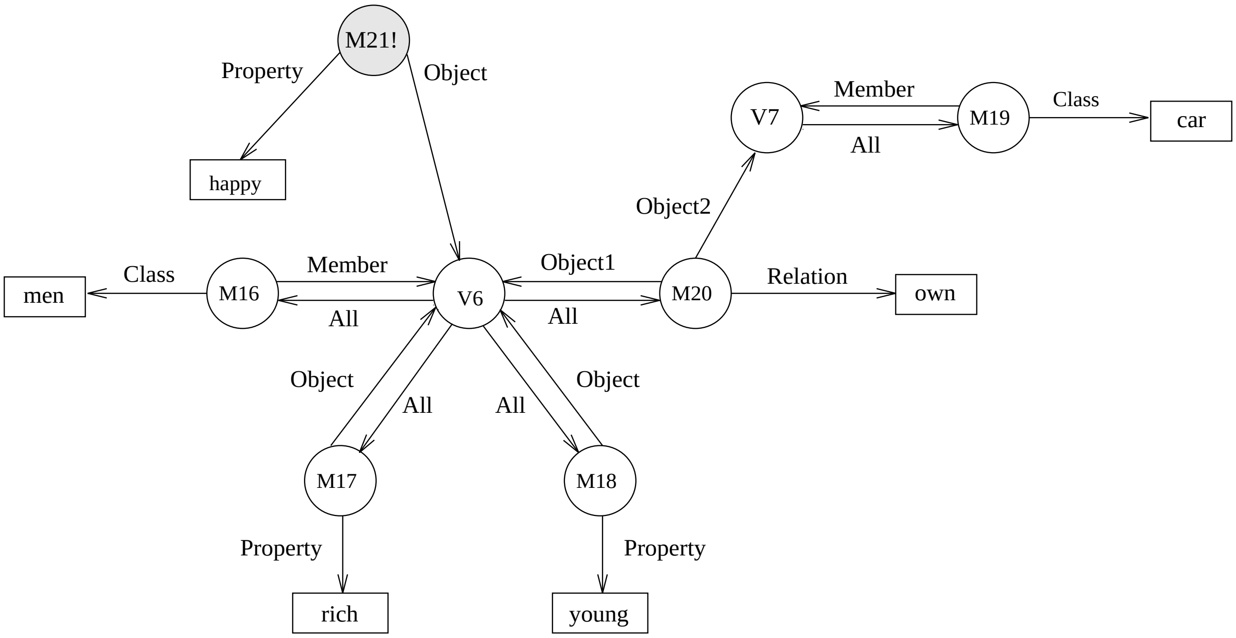



Figure 20: Representation of sentence (15) *Any rich young man that owns any car is happy.*

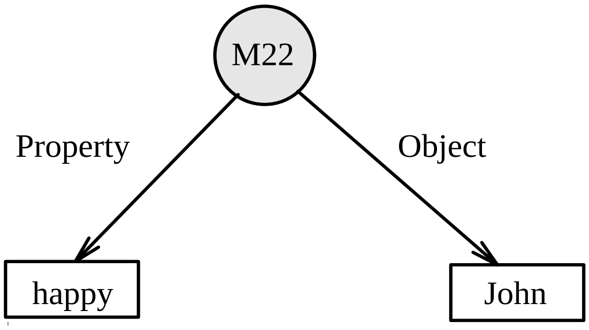



Figure 21: Representation of sentence (17). *Is John happy?*

# 7 Summary

We initially presented some broad goals for a knowledge representation and reasoning system for natural language processing. We have described, in some detail, such a system. ANALOG is a propositional semantic-network-based knowledge representation and reasoning system that supports many aspects of NLP, in particular, the representation and generation of complex noun phrases, the representation of various types of quantified variable scoping, a high degree of structure sharing, and subsumption of the sort typically associated with ordinary natural language use. We have presented examples of natural language dialog and their associated representations in the ANALOG system that illustrate the utility of this formalism for NLP.

The full ANALOG system has not been described here, due to space limitations, but includes

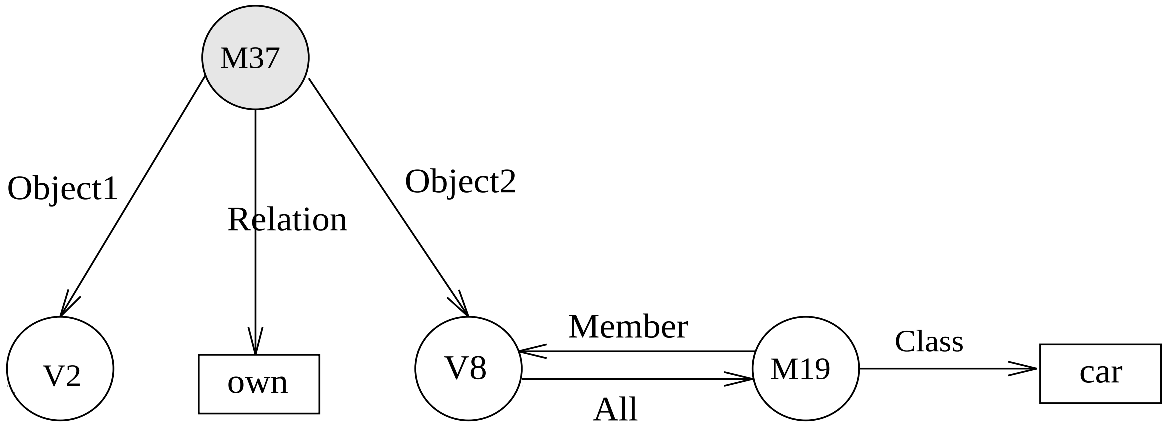



Figure 22: Representation of sentence (21). *Who owns a car?*

facilities for derivation (inference), path-based inference, and belief revision.

## Acknowledgments

Our thanks to Professor William J. Rapaport for his comments on various drafts of this paper.

## References

[1] Syed S. Ali. A Structured Representation for Noun Phrases and Anaphora. In *Proceedings of the Fifteenth Annual Conference of the Cognitive Science Society*, 1993. To appear.

[2] Syed S. Ali. Natural Language Processing Using Propositional Semantic Networks. *Minds and Machines*, 1993. Special Issue on Knowledge Representation for Natural Language Processing, (to appear).

[3] Jon Barwise. On Branching Quantifiers in English. *J. Phil. Logic*, 8:47–80, 1979.

[4] Jon Barwise and Robin Cooper. Generalized Quantifiers and Natural Language. *Linguistics and Philosophy*, 4:159–219, 1981.

[5] Daniel G. Bobrow and Terry Winograd. An Overview of KRL, a Knowledge Representation Language. *Cognitive Science*, 1(1):3–46, 1977.

[6] Ronald J. Brachman. On the Epistemological Status of Semantic Networks. In N. V. Findler, editor, *Associative Networks: Representation and Use of Knowledge in Computers*. Academic Press, New York, 1979.

[7] Ronald J. Brachman, Richard E. Fikes, and Hector J. Levesque. KRYPTON: a Functional Approach to Knowledge Representation. *IEEE Computer*, 16(10):67–73, 1983.

[8] Ronald J. Brachman, Victoria Pigman Gilbert, and Hector J. Levesque. An Essential Hybrid Reasoning System: Knowledge and Symbol Level Accounts of KRYPTON. *Proceedings IJCAI-85*, 1:532–539, 1985.

[9] Ronald J. Brachman and J. Schmolze. An Overview of the KL-ONE Knowledge Representation System. *Cognitive Science*, 9(2):171–216, 1977.

[10] Nick Cercone, Randy Goebel, John De Haan, and Stephanie Schaeffer. The ECO Family. *Computers and Mathematics with Applications*, 23(5):95 – 131, 1992. Special issue on Semantic Networks in Artificial Intelligence (Part 1).

[11] Sung-Hye Cho. Collections as Intensional Entities and Their Representations in a Semantic Network. In *Proceedings of the Second Pacific Rim International Conference on Artificial Intelligence*, pages 388–394, 1992.

[12] David R. Dowty, Robert E. Wall, and Stanley Peters. *Introduction to Montague Semantics.* D. Reidel Publishing Co., Boston, 1981.

[13] Scott E. Fahlman. *NETL: A System for Representing and Using Real-World Knowledge.* MIT Press, Cambridge, MA, 1979.

[14] G. Fauconnier. Do Quantifiers Branch. *Linguistic Inquiry*, 6(4):555–578, 1975.

[15] Kit Fine. A Defense of Arbitrary Objects. In *Proceedings of the Aristotelian Society*, volume supp. vol. LVII, pages 55–77, 1983.

[16] Kit Fine. Natural Deduction and Arbitrary Objects. *Journal of Philosophical Logic*, 14:57–107, 1985.

[17] Kit Fine. *Reasoning with Arbitrary Objects.* Basil Blackwell, Oxford, 1985.

[18] Peter Thomas Geach. *Reference and Generality.* Cornell University Press, Ithaca, New York, 1962.

[19] Robert Givan, David A. McAllester, and Sameer Shalaby. Natural Language Based Inference Procedures Applied to Schubert's Steamroller. In *Proceedings of AAAI-91*, pages 915–920, 1991.

[20] Per-Kristian Halvorsen. Natural Language Understanding and Montague Grammar. *Computational Intelligence*, 2:54 – 62, 1986.

[21] Irene Heim. Discourse Representation Theory, 1990. Tutorial material from ACL-90.

[22] Gary G. Hendrix. Expanding the Utility of Semantic Networks through Partitioning. *Proc. 4th IJCAI*, 1977.

[23] Gary G. Hendrix. Encoding Knowledge in Partitioned Networks. In N. V. Findler, editor, *Associative Networks: The Representation and Use of Knowledge in Computers.*, pages 51–92. Academic Press, New York, 1979.

[24] L. Henkin. *Some remarks on infinitely long formulas*, pages 167–183. Pergamon Press, Oxford, 1961.

[25] J. R. Hobbs and S. M. Shieber. An algorithm for generating quantifier scopings. *Computational Linguistics*, 13(1-2):47 – 63, 1987.

[26] Hans Kamp. A Theory of Truth and Semantic Representation. In Jeroen Groenendijk, Theo M. V. Janssen, and Martin Stokhof, editors, *Truth, Interpretation and Information*, pages 1–41. Forbis, Cinnaminson, 1984.

[27] L. K. Schubert and F. J. Pelletier. From English to Logic: Context-free Computation of Conventional Logical Translation. *American Journal of Computational Linguistics*, 8:165–176, 1982. Reprinted (with corrections) in B. J. Grosz, K. Sparck-Jones and B. L. Webber (eds.), *Readings in Natural Language Processing*, 293 – 311, Morgan Kaufman, 1986.

[28] J. W. Lloyd. *Foundations of Logic Programming.* Springer-Verlag, New York, 1987. 2nd Ed.

[29] David A. McAllester. *Ontic: A Knowledge Representation System for Mathematics.* MIT Press, Cambridge, MA, 1989.

[30] David A. McAllester and Robert Givan. Natural language syntax and first-order inference. *Artificial Intelligence*, 56(1):1 – 20, 1992.

[31] Richard Montague. The proper treatment of quantification in ordinary english. In J. Hintikka, J. Moravcsik, and P. Suppes, editors, *Approaches to Natural Language*, pages 221–242. Reidel, Dordrecht, 1973. Also in R. Montague, 1974, *Formal Philosophy: Selected Papers of Richard Montague*, ed. by Richard Thomason, New Haven: Yale University Press.

[32] Richard Montague. English as a Formal Language. In Richmond H. Thomason, editor, *Formal Philosophy*, pages 188–221. Yale University Press, 1974.

[33] W. V. Quine. *Ontological Relativity and Other Essays.* Columbia University Press, London and New York, 1969.

[34] W. V. Quine. *Philosophy of Logic.* Prentice-Hall, Englewood Cliffs, NJ, 1970.

[35] Lenhart K. Schubert, Randolph G. Goebel, and Nicholas J. Cercone. The Structure and Organization of a Semantic Net for Comprehension and Inference. In N. V. Findler, editor, *Associative Networks: Representation and Use of Knowledge in Computers*, pages 121–175. Academic Press, New York, 1979.

[36] S. C. Shapiro. Generalized augmented transition network grammars for generation from semantic networks. In *Proceedings of the 17th Annual Meeting of the Association for Computational Linguistics*, pages 25–29. University of California at San Diego, 1979. Superseded by 34.

[37] S. C. Shapiro. Generalized augmented transition network grammars for generation from semantic networks. *The American Journal of Computational Linguistics*, 8(1):12–25, 1982.

[38] S. C. Shapiro. Symmetric relations, intensional individuals, and variable binding. *Proceedings of the IEEE*, 74(10):1354–1363, 1986.

[39] S. C. Shapiro. Cables, Paths, and "Subconscious" Reasoning in Propositional Semantic Networks. In John F. Sowa, editor, *Principles of Semantic Networks*, pages 137–156. Morgan Kaufmann, 1991.

[40] S. C. Shapiro and W. J. Rapaport. SNePS considered as a fully intensional propositional semantic network. In N. Cercone and G. McCalla, editors, *The Knowledge Frontier*, pages 263–315. Springer–Verlag, New York, 1987.

[41] S. C. Shapiro and William J. Rapaport. SNePS Considered as a Fully Intensional Propositional Semantic Network. *Proceedings of the 5th National Conference on Artificial Intelligence*, 1:278–283, 1987.

[42] S. C. Shapiro and William J. Rapaport. The SNePS Family. *Computers and Mathematics with Applications*, 23(5):243 – 275, 1992. Special issue on Semantic Networks in Artificial Intelligence (Part 1).

[43] W. A. Woods. *Semantics and Quantification in Natural Language Question Answering*, volume 17. Academic Press, New York, 1978.

[44] William A. Woods. Understanding subsumption and taxonomy: A framework for progress. In John F. Sowa, editor, *Principles of Semantic Networks*, pages 45 – 94. Morgan Kaufmann, 1991.

[45] William A. Woods and James G. Schmolze. The KL-ONE Family. *Computers and Mathematics with Applications*, 23(5):133 – 177, 1992. Special issue on Semantic Networks in Artificial Intelligence (Part 1).