Knowledge Representation for Syntactic/Semantic Processing

Robert J. Bobrow

Bolt Beranek and Newman Inc.
50 Moulton St.
Cambridge, Mass. 02238

Bonnie L. Webber

Department of Computer Science
The Moore School of Elelectrical Engineering D2
University of Pennsylvania
Philadelphia, Pa. 19104

ABSTRACT

This paper describes the RUS framework for natural language processing, in which a parser incorporating a substantial ATN grammar for English interacts with a semantic interpreter to simultaneously parse and interpret input. The structure of that interaction is discussed, including the roles played by syntactic and semantic knowledge. Several implementations of the RUS framework are currently in use, sharing the same grammar, but differing in the form of their semantic component. One of these, the PSI-KLONE system, is based on a general object-centered knowledge representation system, called KL-ONE. The operation of PSI-KLONE is described, including its use of KL-ONE to support a general inference process called "incremental description refinement." The last section of the paper discusses several important criteria for knowledge representation systems to be used in syntactic and semantic processing.

- - - - - - - - - - - -

1. Introduction

This paper discusses some theoretical implications of recasting parsing and semantic interpretation as a type of inference process which we call incremental description refinement.* It draws upon our recent experience with RUS, a framework for natural language processing developed at BBN and in use in several different natural language systems across the country (for details see [1], [2] and [7]). RUS is a very practical system that is as efficient as a semantic grammar like the SOPHIE parser [6] and as flexible and extensible as a modular syntactic/semantic processor like LUNAR [10]. It achieves this combination of efficiency and flexibility by cascading [12] syntactic and semantic processing: producing the semantic interpretation of an input utterance incrementally during the parsing process, and using it to guide the operation of the parser.

Because RUS provides a very clean interface between syntactic and semantic processing, it has been possible to experiment with a variety of knowledge representations in the different implementations noted above. The most recent such implementation uses the KL-ONE formalism [3], [4], to represent the knowledge needed for incremental processing. (This implementation has been dubbed PSI-KLONE, for "Parsing and Semantic Interpretation using KL-ONE".) KL-ONE is a uniform object-centered representational scheme based on the idea of structured inheritance in a lattice-structured taxonomy of generic knowledge. As we shall discuss later, PSI-KLONE takes advantage of KL-ONE's taxonomic lattice [11] which combines the properties of an inheritance network with those of a discrimination net.

The next section of this paper describes the syntactic/semantic cascade in general terms, and then gives a short example of its operation in the PSI-KLONE implementation. We then define the concept of an incremental description refinement (IDR) process to use as a paradigm for understanding the operation of the semantic component of the cascade. This introduces the last section of the paper, which discusses the requirements for a general frame-like knowledge representation if it is to be capable of supporting such an IDR process.

2. The Syntactic/Semantic Cascade

Within the RUS framework, the interaction between the parser and the semantic interpreter (the interpreter) takes place incrementally as the parser scans the input string from left to right, one word at a time. The semantic interpretation of each syntactic constituent is produced in parallel with the determination of its syntactic structure. Knowledge developed in the course of producing the interpretation is used to control further action by the parser. Parsing supports the processes of semantic interpretation and discourse inference (not discussed in this paper) by finding the constituents of each phrase, determining their syntactic structure, and labelling their functional relationship* to the phrase as a whole (the

*The research reported in this paper was supported in part by the Advanced Research Projects Agency, and was monitored by ONR under Contract No. N00014-77-C-0378.

*We use an extended notion of functional relation here that includes surface syntactic relations, logical syntactic (or shallow case structure) relations, and relations useful for determining discourse structures such as primary focus.

matrix). These labels are proposed purely on the basis of syntactic information, but are intended to reflect a constituent's functional role in the matrix, and not simply its internal syntactic structure.* We will refer to these labels as functional or syntactic labels for constituents.

The parser and interpreter engage in a dialogue consisting of transmissions from the parser and responses from the interpreter. A transmission is a proposal by syntax that some specific functional relation holds between a previously parsed and interpreted constituent and the matrix phrase whose parsing and interpretation is in progress. The proposal takes the form of a matrix/label/constituent triple. The interpreter either rejects the proposal or accepts it and returns a pointer to a KL-ONE data structure which represents its knowledge of the resulting phrase. (This pointer is not analyzed by the parser, but is rather used in the description of the matrix that syntax includes in its next proposal (transmission) to extend the matrix.) The parser is implemented as an ATN [9], and transmissions occur as actions on the arcs of the ATN grammar. The failure of an arc because of a semantic rejection of a transmission is treated exactly like the failure of an arc because of a syntactic mismatch; alternative arcs on the source state are attempted, and if none are successful, a back-up occurs.

2.1. The role of the semantic interpreter in a cascaded system

The PSI-KLONE interpreter must perform two related tasks:

1. provide feedback to the parser by checking the semantic plausibility of syntactic labels for proposed constituents of a phrase, and

2. build semantic interpretations for individual phrases.

The mechanism for performing both these tasks is based on the idea of mapping between the (syntactic) functional labels provided by the parser and a set of extended case frame or semantic relations (defined by the interpreter) that can hold between a constituent and its matrix phrase. The mapping of functional labels to semantic relations is clearly one to many. For example, the logical subject of a clause whose main verb is "hit" might be the agent of the act (e.g. "The boy hit ...") or the instrument (e.g. "The brick hit ..."). A semantic relation (or semantic role), on the other hand, must completely specify the role played by the interpretation of the constituent in the interpretation of the matrix phrase.

*For example, a noun phrase (NP) can serve various functions in a clause, including logical subject (LSUBJ), logical object (LOBJ), surface subject (SSUBJ), and first NP (FIRSTNP).

The task of the interpreter is to determine which, if any, semantic relation could hold between a matrix phrase and a parsed and interpreted constituent, given a functional label proposed by the parser. This task is accomplished with the aid of a set of pattern-action relation mapping rules (RMRULES) that specify how a given functional label can be mapped into a semantic relation. An RMRULE has a pattern (a matrix/label/constituent triple) that specifies the conditions in which it applies, in terms of:

- o the syntactic shape of the matrix (e.g. "It is a transitive clause whose main verb is 'run'."), and the interpretation and semantic role assigned to other constituents (e.g. "The logical subject must be a person* and be the Agent of the clause"),

- o the proposed functional label, and

- o the interpretation of the constituent to be added.

The action of the RMRULE is to map the given functional label onto a semantic relation.

A proposed syntactic label is semantically plausible if its proposal triple matches the pattern triple(s) of some RMRULE(s). KL-ONE is a good language for describing structured objects such as phrases built up out of constituents, and for representing classes of objects such as the matrix/label/constituent triples that satisfy the constraints given by RMRULE patterns. In PSI-KLONE, each RMRULE pattern is represented as a KL-ONE structure called a Generic Concept (see section 2.2). These Concepts are arranged in a taxonomy that is used as a discrimination net to determine the set of patterns which match each triple. We refer to this as the taxonomy of syntactic/semantic shapes; note that it is generally a lattice and not simply a tree structure.

Associated with each semantic relation is a rule (an IRULE) that specifies how the interpretation of the constituent is to be used in building the interpretation of the matrix phrase. When all the constituents of a matrix have been assigned appropriate semantic relations, the interpretation of a phrase is built up by executing all of the IRULEs that apply to the phrase. The separation of RMRULEs from IRULEs allows PSI-KLONE to take full advantage of the properties of the syntactic/semantic cascade. As each new constituent is proposed by the parser, the interpreter uses the RMRULEs to determine which IRULEs apply to that constituent; but it does not actually apply them until the parser indicates that all constituents have been found. This buys

*That is, the constituent labelled LSUBJ must be interpretable as a person.

efficiency by rejecting constituent label assignments which have no hope of semantic interpretation, while deferring the construction of an interpretation until the syntactic well-formedness of the entire phrase is verified.

2.2. An example of the cascade

As a simplified example of the parser-interpreter interaction, and the use of the KL-ONE taxonomy of syntactic/semantic shapes in this interaction, we will briefly describe the process of parsing the clause "John ran the drill press." The simplified ATN grammar we use for this example is shown in Fig. 2-1.

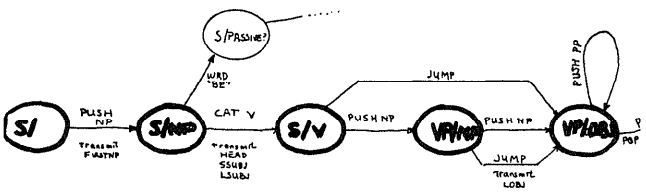

Figure 2-1: A simplified ATN

For readers unfamiliar with KL-ONE, we will explain three of its major constructs as we note the information represented in the simple taxonomy shown in Fig. 2-2. In KL-ONE, Generic Concepts (ovals in the diagram, **boldface** in the text) represent

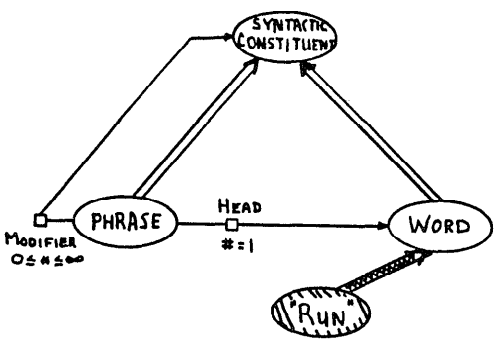

Figure 2-2: A simple KL-ONE network

description templates, from which individual descriptions or Individual Concepts (shaded ovals, also **boldface** in text) are formed. In Fig. 2-2, the most general description is **SYNTACTIC-CONSTITUENT**, which is specialized by the two descriptions, **PHRASE** and **WORD**. All KL-ONE descriptions are structured objects. The only structuring device of concern here is the Role. A Role (drawn as a small square, and underlined in text) represents a type of relationship between two objects, such as the relation between a whole and one of its parts. Every Role on a Generic Concept indicates what type of object can fill the Role, and how many distinct instances of the relation represented by the Role can occur. The restriction on fillers of a Role is given by a pointer to a Generic Concept, and the number of possible instances of the Role is shown by a number facet (indicated in the form "M < # < N" in the diagram). In our diagram we indicate that every **PHRASE** has a **WORD** associated with it which fills the Head Role of the **PHRASE**, and may have 0 or more other **SYNTACTIC-CONSTITUENTS** which are Modifiers. The double arrow or SuperC Cable between **PHRASE** and **SYNTACTIC-CONSTITUENT** indicates that every instance of **PHRASE** is thereby a **SYNTACTIC-CONSTITUENT**.

The simplified taxonomy* for our example is given in Fig. 2-3. This indicates that any **CLAUSE** whose Head is the verb "run"

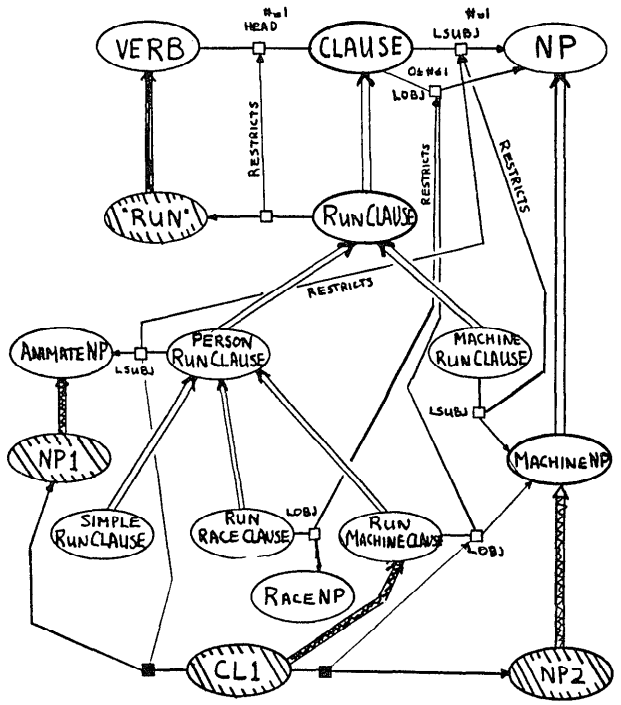

Figure 2-3: A simple KL-ONE Syntactic Taxonomy

(independent of tense and person/number agreement) is an example of a **RunCLAUSE**. There are two classes of **RunCLAUSE**s represented in the taxonomy - those whose LSUBJ is a person (the **PersonRunCLAUSE**s), and those whose LSUBJ is a machine (the **MachineRunCLAUSE**s). The class of **PersonRunCLAUSE**s is again sub-divided, and its subclasses are **RunMachineCLAUSE** (in which the LOBJ must be a machine), **RunRaceCLAUSE** (in which the LOBJ is a race), and **SimpleRunCLAUSE** (which has no LOBJ).

If we get an active sentence like "John ran the drill press", the first stage in the parsing is to PUSH for an NP from the CLAUSE network. For simplicity we assume that the result of this is to parse the noun phrase "John" and produce a pointer to **NP1**, an Individual Concept which is an instance of the Generic pattern **PersonNP**. This is the result of interaction of the parser and interpreter

*To reduce clutter, several superC cables to the Concept **NP** have been left out.

at a lower level of the ATN.

Since it is not yet clear what role **NP1** plays in the clause (i.e. because the clause may be active or passive), the parser must hold onto **NP1** until it has analyzed the verb. Thus the first transmission from the parser to the interpreter at this level is the proposal that "run" (the root of "ran") is the Head of a **CLAUSE**. The interpreter accepts this and returns a pointer to a new Individual Concept **CL1** which it places as an instance of **RunCLAUSE**.*

Since the parser has by now determined that the clause is a simple active clause, it can now transmit the proposal that **NP1** is the LSUBJ of **CL1**. Because **NP1** is an instance of a **PersonNP**, the interpreter can tell that it satisfies the restrictions on the LSUBJ of one of the specializations of **RunCLAUSE**, and thus it is a semantically plausible assignment. The interpreter fills in the LSUBJ Role of **CL1** with **NP1**, and connects CL1 to **PersonRunCLAUSE**, since that is the only subConcept of **RunCLAUSE** which can have a **PersonNP** as its LSUBJ.

Finally, the parser PUSHes for an NP, resulting in a pointer to **NP2**, an instance of **MachineNP**. This is transmitted to the interpreter as the LOBJ of CL1. Since CL1 is a **PersonRunCLAUSE**, the taxonomy indicates that it can be either an instance of a **RunRaceCLAUSE** or a **RunMachineCLAUSE**, or a **SimpleRunCLAUSE**. Since NP2 has been classified as an instance of **MachineNP**, it is not compatible with being the LOBJ of a **RunRaceCLAUSE** (whose LOBJ must be interpretable as a race). On the other hand **NP2** is compatible with the restriction on the filler of the LOBJ Role of **RunMachineCLAUSE**.

We assume that the taxonomy indicates all the acceptable subcategories of **PersonRunCLAUSE**. Thus it is only semantically plausible for **NP2** to fill the LOBJ Role of **CL1** if **CL1** is an instance of **RunMachineCLAUSE**. This being the case, the interpreter can join **CL1** to **RunMachineCLAUSE** and fill its LOBJ Role with NP2, creating a new version of CL1 which it returns to the parser.

At this point, since there are no more words in the string, the parser transmits a special message to the interpreter, indicating that there are no more constituents to be added to CL1. The interpreter responds by finding the IRULEs inherited by CL1 from **RunMachineCLAUSE**, **PersonRunCLAUSE**, etc. and using the actions on those IRULEs to create the interpretation of **CL1**. It associates that interpretation with **CL1** and returns a pointer to **CL1**, now a fully parsed and interpreted clause, to the parser.

---

*Actually, the interpreter creates a Generic subConcept of **RunCLAUSE**, in order to facilitate sharing of information between alternative paths in the parse, but we will ignore this detail in the remainder of the example.

3. Incremental Description Refinement

We view the cascaded interaction of syntactic analysis and semantic interpretation as implementing a recognition paradigm we refer to as incremental description refinement. In this paradigm we assume we are initially given a domain of structured objects, a space of descriptions, and rules that determine which descriptions apply to each object in the domain.* As an example, consider the domain to be strings of words, the structured descriptions to be the parse trees of some grammar, and say that a parse tree applies to a string of words if the leaves in the tree correspond to the sequence of words in the string. In general we assume each description is structured, not only describing the object as a whole, but having components that describe the parts of the object and their relationship to the whole as well.

We consider a situation that corresponds to left-to-right parsing. A machine is presented with facts about an object or its parts in some specified order, such as learning the words in a string one by one in a left-to-right order. As it learns more properties of the object the machine must determine which descriptions are compatible with its current knowledge of the properties of the object and its parts.

Incremental description refinement (IDR) is the process of:

- o determining the set of descriptions compatible with an object known to have a given set of properties, and
- o refining the set of descriptions as more properties are learned.

More precisely, for every set of properties P = {p1,...,pn} of some object O or its parts, there is an associated set of descriptions C(P), the descriptive cover of P. The descriptive cover of P consists of just those descriptions which **might possibly be applicable** to O, given that O has the properties P1,...,Pn; that is, the set of descriptions which apply to **at least one** object which has all the properties in P.

As one learns more about some object, the set of descriptions consistent with that knowledge shrinks. Hence, the basic step of any IDR process is to take (1) a set of properties P, and (2) its cover C(P), and (3) some extension of P into a set P', and to produce C(P') by removing inapplicable elements from C(P). The difficulty is that it is usually impractical, if not impossible, to represent C(P) extensionally: in many cases C(P) will be infinite. (For example, until the number of words in a string is learned, the number of parse trees in C(P) remains infinite no matter how many words in the string are known.) Thus, the

---

*We assume that at least one description applies to each object in the domain.

covering set must be represented intensionally, with the consequence that "removing elements" becomes an inference process which determines the intensional representation of C(P') given the intensional representation of C(P). Note that just as any element of C(P), represented extensionally, may be structured, so may the intensional representation of C(P) be structured as well.

The trick in designing an efficient and effective IDR process is to choose a synergistic inference process/intensional representation pair. One example is the use of a discrimination tree. In such a tree each terminal node represents an individual description, and each non-terminal node represents the set of descriptions corresponding to the terminals below it. Every branch indicates a test or discrimination based on some property (or properties) of the object to be described. Each newly learned property of an object allows the IDR process to take a single step down the tree, as long as the properties are learned in an order compatible with the tree's structure. Each step thus reduces the set of descriptions subsumed.

Another IDR process is the operation of a constraint propagation system. In such a system an object is described by a set of nodes, each of which bears a label chosen from some fixed set. The nodes are linked into a network, and there is a constraint relation that specifies which pairs of labels can occur on adjoining (i.e. linked) nodes. The facts learned about an object are either links between previously known nodes, or label sets which specify the possible labels at a single node. A descriptive cover is simply the cross-product of some collection of node label sets. The refinement operation consists of (1) extending the analysis to a new node, (2) removing all incompatible labels from adjacent nodes and (3) propagating the effects. Unlike the use of a discrimination net, constraint propagation does not require that information about nodes be considered in some *a priori* fixed order.

As mentioned earlier, in the RUS framework we are attempting to refine the semantic description of an utterance in parallel with determining its syntactic structure. The relevant properties for this IDR process include the descriptions of various constituents and their functional relations to their matrix (cf. Section 2). Unfortunately, surface variations such as passive forms and dative movement make it difficult to assume any particular order of discovery of properties as the parser considers words in a left to right order. However, the taxonomic lattice of KL-ONE can be used as a generalization of a discrimination tree which is order independent. The actual operation used in PSI-KLONE involves an extended notion of constraint propagation operating on nodes in the taxonomic lattice, and thus the resulting system has interesting analogies to both simpler forms of IDR processes.

The complete algorithm for the IDR process in PSI-KLONE is too complex to cover in this paper, and will be described in more detail in a forthcoming report. However, the reader is urged to return to the example in Sec. 2.2 and reconsider it as an IDR process as we have described above. Briefly, we can view the KL-ONE taxonomy of syntactic/semantic shapes as a set of discrimination trees, each with a root labelled by some syntactic phrase type. At each level in a tree the branches make discriminations based on the properties of some single labelled constituent (LC), such as the LSUBJ of a CLAUSE.

The parser first proposes a phrase type such as CLAUSE, and the IDR process determines which tree has a root with that label. That root becomes the current active node in the IDR process. All further refinement is done within the subtree(s) dominated by an active node. As the parser proposes and transmits new LC's to the IDR, the IDR may respond in one of two ways:

1. it may reject the LC because it is **not compatible** with any branch below the currently active node(s), or

2. it may accept the LC, and replace the current active node(s) with the (set of) node(s) which can be reached by branches whose discriminations are compatible with the LC.

4. The IDR Process and Knowledge Representation

We have identified four critical characteristics of any general representation scheme that can support an IDR process in which descriptions are structured and covering descriptions are represented intensionally. In such a scheme it must be possible to efficiently infer from the representation:

1. what properties of a structured object provide sufficient information to guarantee the applicability of a description to (some portion of) that object - i.e., criteriality conditions,

2. what mappings are possible between classes of relations - e.g. how functional relationships between syntactic constituents map onto semantic relationships

3. which pairs of descriptions are **mutually incompatible** - i.e., cannot both apply to a single individual

4. which sub-categorizations of descriptions are exhaustive - i.e., at least one of the sub-categories applies to anything to which the more general description applies.

Our analysis of the assumptions implicit in the current implementation of PSI-KLONE has led us to an understanding of the importance of these four points in a IDR. By making these four points explicit in the next implementation we expect to be able to deal with a larger class of phenomena than the current system handles. In the following sections we illustrate these four points in terms of the behavior of the current version of PSI-KLONE and the improvements we expect to be able to make

with more explicit use of these types of information.

4.1. Criteriality Conditions

The point here is an obvious one, but bears repeating. If a taxonomy is to be used for recognition, then there must be some way, based on partial evidence, to get into it at the right place for the recognition (IDR) process to begin. That is, for any ultimately recognizable phrase there must be at least one criterial condition, i.e. a collection of facts which is sufficient to ensure the applicability of some particular description. In the syntactic/semantic taxonomy, the criterial condition is often, for a phrase, the properties of belonging to a particular syntactic category (e.g., noun phrase, clause, etc.) and having a particular lexical item as head. Recalling the example given in Section 2.2, the evidence that the input had the shape of a **CLAUSE** and had the verb "run" as its head constituted sufficient conditions to enter the taxonomy at the node **RunCLAUSE** - i.e., a specialization of **CLAUSE** whose head is filled by the verb "run". Without the notion of criterial properties, we cannot ensure the applicability of any description and therefore have no way of continuing the recognition process.

4.2. Mapping Syntactic to Semantic Relations

In RUS, the parser intermittently sends messages to the interpreter asking whether it is semantically plausible for a constituent to fill a specified functional role. The interpreter's ability to answer this question comes from its RMRULEs and their organization. This is based on the assumption that a potential constituent can fill some functional role in the matrix phrase if and only if it also fills a semantic role compatible with:

- o that functional role
- o the interpretation of that constituent
- o the head of that matrix phrase
- o the roles filled by the other constituents of that phrase
- o other syntactic/semantic properties of that phrase and its constituents.

With respect to the first of these points, one effective way of representing the compatibility restrictions between syntactic and semantic relations derives from the fact that each purely syntactic relation can be viewed as an **abstraction** of the syntactic properties shared by some class of semantic relations (i.e., that they have syntactically identical arguments). If

1. a general frame-like system is used to represent the system's syntactic/semantic knowledge,
2. possible syntactic and semantic relations are represented therein as "slots" in a frame, and
3. there is an abstraction hierarchy among slots (the Role hierarchy in KL-ONE), as well as the more common IS-A hierarchy among frames (the SUPERC link between concepts in KL-ONE),

then the interpreter can make use of this abstraction hierarchy in answering questions from the parser.

As an example, consider a question from the parser, loosely translatable as "Can the PP 'on Sunday' be a PP-modifier of the NP being built whose head is 'party'?" (cf. Fig. 4-1).

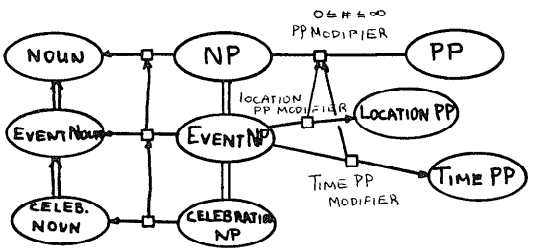

Figure 4-1: A Simple NP Taxonomy

We assume the NP headed by "party" has already been classified by the interpreter as a **CelebrationNP**. As indicated in Fig. 4-1 this concept inherits from **EventNP** two specializations of the general PP-modifier relation applicable to NP - location-PP-modifier and time-PP-modifier. Thus "on Sunday" can be one of its PP-modifiers **iff** it can be either its location-PP-modifier or its time-PP-modifier. The next section will discuss how this decision can be made. The point here is that there must be some indication of which syntactic relations can map onto which semantic ones, and under what circumstances. An abstraction hierarchy among Roles provides one method of doing so.

4.3. Explicit compatibility/incompatibility annotations

As noted above, the semantic interpreter must be able to decide if the interpretation assigned to the already parsed constituent is compatible with the type restrictions on the arguments of a semantic relation. For example, the PP "on Sunday" can be a PP-modifier of an NP whose Head is "party" if it is compatible with being either a **time-PP**, and hence capable of instantiating the relation time-PP-modifier, or a **location-PP** and hence instantiating the relation location-PP-modifier.* There are two plausible strategies for formalizing the somewhat informal notion of compatibility:

---

*In this, as in Fig. 4-1, we assume for simplicity that only these two semantic relations are consistent with the syntactic relation PP-modifier for an NP whose head is "party".

1. a constituent is judged compatible with a restriction if its syntactic/semantic shape (and hence interpretation) **guarantees consistency** with the type restrictions, or

2. it is judged compatible if its interpretation **does** not **guarantee** inconsistency.

Consider the problem of rejecting "on Sunday" as a location-PP-modifier. Conceivably one could reject it on the grounds that "Sunday" doesn't have a syntactic/semantic shape that **guarantees** that it is a **location-NP**. This is essentially the strategy followed by the current version of PSI-KLONE. More specifically, the PSI-KLONE system searches along the superC cables of a constituent to find just those semantic relations which are guaranteed to be compatible with the interpretation of the constituent and matrix.

However, that strategy would have to reject "birthday present" as being compatible with **apparel-NP** (thereby rejecting "Mary wore her birthday present to New York"), **vehicle-NP** (thereby rejecting "Mary drove her birthday present from Boston to Philadelphia"), **animate-NP** (thereby rejecting "Mary fed her birthday present some Little Friskies"), etc. Thus, we believe that future systems should incorporate the second strategy, at least as a fall-back when no interpretation is found using only the first strategy. This strategy also makes it easier for the system to handle pronouns and other semantically empty **NPs** (e.g. "thing", "stuff", etc.) whose syntactic/semantic shapes **guarantee** almost nothing, but which are compatible with many semantic interpretations.

The implication here for both language processing and knowledge representation is that:

1. incompatibility must be marked explicitly in the representation, and

2. the most useful strategy for determining compatibility involves not being able to show explicit incompatibility.

One caveat and one further observation: this strategy is not by itself effective in certain cases of metonymy, which Webster's defines as "the use of the name of one thing for that of another associated with or suggested by it." For example, semantics would reject "the hamburger" as the subject of a clause like "the hamburger is getting impatient"* which might occur in a conversation between a waiter and a short-order cook. However, the taxonomy would be able to provide information needed to resolve the metonymy, since it would indicate that "the hamburger" is possibly being used metonymously to refer to some discourse entity which is both describable by an **animate-NP** and associated with some (unique) hamburger.

The observation concerns the way in which semantic interpretation was done in LUNAR [10], which was to judge semantic compatibility solely on the basis of positive syntactic/semantic evidence. A semantic interpretation rule could only be applied to a fragment of a parse tree if the rule's left-hand side - a syntactic/semantic template - could be matched against the fragment. The only kinds of semantic constraints actually used in the LUNAR templates were predicates on the head of some tree constituent -- e.g. that the head of the NP object of a PP constituent were of class element, rock, etc. Given this restriction, LUNAR would not be able to handle an utterance like "give me analyses of aluminum in NASA's gift to the Royal Academy", where clearly "gift to the Royal Academy" is not incompatible with rock.

4.4. Explicit marking of exhaustive sub-categorization in the taxonomy

The algorithm we have developed for incremental description refinement requires that the IDR process be able to distinguish exhaustive from non-exhaustive sub-categorization in the taxonomy of syntactic/semantic shapes. Exhaustiveness marking plays a role similar to that played by inclusive or in a logical framework. That is, it justifies the application of case-analysis techniques to the problem of determining if a proposed constituent is compatible with a given syntactic role. The interpreter is justified in rejecting a proposed label for a constituent only if it has considered all possible ways in which it can correspond to a semantic relation.

Exhaustiveness marking also make it possible to infer positive information from negative information as was done in the example in section 2.2. There, the interpreter inferred that the clause was a **RunMachineCLAUSE**, because it was known to be a **PersonRunCLAUSE** and the proposed LOBJ was incompatible with it being a **RunRaceCLAUSE**. Such reasoning is justified only if the subcategories **RunMachineCLAUSE**, **SimpleRunCLAUSE** and **RunRaceCLAUSE** exhaust the possibilities under **PersonRunCLAUSE**.

These types of inference do not always come up in systems that are primarily used to reason about inherited properties and defaults. For example, as long as one knows that DOG and CAT are both specializations of PET, one knows what properties they inherit from PET. It is irrelevant to an inheritance process whether there are any other kinds of PET such as MONKEY, BOA-CONSTRICTOR or TARANTULA.

Many formalisms, including KL-ONE, do not require the sub-categorization of a node to be exhaustive. So there are two options vis-a-vis the way exhaustiveness can be indicated. A recognition algorithm can act as if every node were exhaustively sub-categorized - a type of Closed World Assumption [8] - this is essentially the way

---

*If we assume something like "hamburger" being an instance of **definite-food-NP**, which is marked as incompatible with **animate-NP**, the restriction on the subject of "impatient".

the current PSI-KLONE system operates. Unfortunately, there are other uses of KL-ONE in the natural language system in which concepts are subcategorized but it is clear that an exhaustive subcategorization has not been made. If the meaning of the links in the representation scheme is to be well-defined, it must be possible to distinguish exhaustive from non-exhaustive sub-categorization. The implication for both knowledge representation and inference is that some clear stand must be taken vis-a-vis the representation of exhaustive sub-categorizations.

5. Conclusion

The approach we have taken in RUS is midway between completely decoupled syntactic and semantic processing and the totally merged processing that is characteristic of semantic grammars. RUS has already proven the robustness of this approach in several different systems, each using different knowledge representation techniques for the semantic component. The RUS grammar is a substantial and general grammar for English, more extensive than the grammar in the LUNAR system [10]. Although the grammar is represented as an ATN, we have been able to greatly reduce the backtracking that normally occurs in the operation of an ATN parser, allowing RUS to approach the performance of a "deterministic" parser [2]. With the aid of a "grammar compiler" [5] this makes it possible to achieve parsing times on the order of .25 CPU seconds, on a DEC KL10, for twenty word sentences.

In this paper we have focused on the latest embodiment of the RUS framework in the PSI-KLONE system -- in particular on the nature of its cascaded syntactic/semantic interactions and the incremental description refinement process they support. We believe that what we are learning about such cascaded structures and IDR processes in building PSI-KLONE is of value for the design of both natural language systems and knowledge representation systems.

ACKNOWLEDGEMENTS

Our work on the PSI-KLONE system has not been done *in vacuo* - our colleagues in this research effort include Ed Barton, Madeleine Bates, Ron Brachman, Phil Cohen, David Israel, Hector Levesque, Candy Sidner and Bill Woods. We hope this paper reflects well on our joint effort.

The authors wish to thank Madeleine Bates, Danny Bobrow, Ron Brachman, David Israel, Candy Sidner, Brian Smith, and Dave Waltz for their helpful comments on earlier versions of this paper. Our special thanks go to Susan Chase, whose gourmet feasts and general support made the preparation of this paper much more enjoyable than it might otherwise have been.

REFERENCES

[1] Bobrow, R. J.
The RUS System.
BBN Report 3878, Bolt Beranek and Newman Inc., 1978.

[2] Bobrow, R. J. & Webber, B. L.
PSI-KLONE - Parsing and Semantic Interpretation in the BBN Natural Language Understanding System.
In CSCSI/CSEIO Annual Conference. CSCSI/CSEIO, 1980.

[3] Brachman, R. J.
On the Epistemological Status of Semantic Networks.
In Findler, Nicholas V. (editor), Associative Networks - The Representation and Use of Knowledge in Computers, . Academic Press, New York, 1979.

[4] Brachman, R. J.
An Introduction to KL-ONE.
In Brachman, R.J., et al. (editors), Research in Natural Language Understanding, Annual Report (1 Sept. 78 - 31 Aug. 79), pages 13-46. Bolt Beranek and Newman Inc, Cambridge, MA, 1980.

[5] Burton, R. & Woods, W. A.
A Compiling System for Augmented Transition Networks.
In COLING 76. Sixth International Conference on Computational Linguistics, Ottawa, Canada, June, 1976.

[6] Burton, R., Seely Brown, J.
Semantic Grammar: A Technique for Constructing Natural Language Interfaces to Instructional Systems.
BBN Report 3587, Bolt Beranek And Newman Inc., May, 1977.

[7] Mark, W. S. & Barton, G. E.
The RUSGrammar Parsing System.
GMR 3243, General Motors Research Laboratories, 1980.

[8] Reiter, R.
Closed World Data Bases.
In Gallaire, H. & Minker, J. (editor), Logic and Data Bases, . Plenum Press, 1978.

[9] Woods, W. A.
Transition Network Grammars for Natural Language Analysis.
CACM 13(10), October, 1970.

[10] Woods, W. A., Kaplan, R. M. & Nash-Webber, B.
The Lunar Sciences Natural Language Information System: Final Report.
BBN Report 2378, Bolt Beranek and Newman Inc., June, 1972.

[11] Woods, W. A.
Taxonomic Lattice Structures for Situation Recognition.
In Theoretical Issues in Natural Language Processing - 2. ACL and ACM/SIGART, July, 1978.

[12] Woods, W. A.
Cascaded ATN Grammars.
Amer. J. Computational Linguistics 6(1), Jan.-Mar., 1980.