Preface

STUART C. SHAPIRO
Department of Computer Science and Center for Cognitive Science, State University of New York at Buffalo, Buffalo, New York 14260-2000, U.S.A. (email: shapiro@cs.buffalo.edu)

Knowledge Representation (KR) is the subarea of Artificial Intelligence (AI) that deals with techniques for representing information that "intelligent" programs will deal with. It is not enough just to describe the form of the representation – one must also show how the information will be used and the ways in which the form of the representation facilitates that use. Since the use of the information in AI is often reasoning – deriving new explicit information from existing explicit information – the field has recently become known as Knowledge Representation and Reasoning (KRR).

In the early days of AI, many papers we would now recognize as being about KRR were classified as being about Natural Language Processing (NLP). Natural languages are those that people use in their day-to-day activities, those for which some population of people are or were native speakers, as opposed to formal languages such as mathematics, symbolic logic, or programming languages. The field of NLP is the subarea of AI concerned with formulating computational models of how people understand and use natural languages, and with creating computer programs that use some natural language. Those early KRR papers were concerned with the KRR underpinings for NLP programs. That is, the use to which the represented knowledge would be put would be the analysis and generation of NL utterances.

More recently, workers in KRR have gone in two directions: treating KRR formalisms, independently of any computer implementation (but certainly with an eye on implementability), more or less as a branch of formal logic; and in looking at applications of KRR systems to non-NL domains, such as software engineering, general engineering design, diagnosis systems, and database systems. Indeed, in a recent workshop on implemented KRR systems, only a small minority of implementors described their systems as being designed to support NLP. Moreover, when I recently started to prepare a talk on KRR systems for NLP, I had a very hard time finding concrete examples of NL texts and their representations in KRR formalisms. Therefore, when the editor of Minds and Machines asked me to guest edit a special issue, I seized on the opportunity of collecting some current work on KRR for NLP, complete with concrete examples.

In my call for papers for this issue, I said

> The guest editor's intention is to fill a perceived gap in the literature of KR for NL. Each paper should show some text that an actual implemented system can understand, how the information contained in that text is represented, what background information is used by the system, and how that information

Minds and Machines **3**: 377–380, 1993.

is represented, how the system processes the knowledge to do interesting things (such as answering interesting questions about the text), and how the information is processed into answers. Reports on projects whose purpose is to simulate human understanding of texts will be preferred over projects whose purpose is to provide NL interfaces to databases or to pragmatically oriented knowledge bases.

The six papers included in this issue, while not completely fulfilling this desire, did so more than most KRR papers of the last several years. They also struck another theme. Each paper describes a KRR formalism that is grounded in a formal analysis, yet which is strongly and clearly motivated by its intended use for NLP, and which is implemented as a running computer system.

In the first paper, Hwang and Schubert introduce Episodic Logic (EL), a "comprehensive framework for narrative understanding." The most distinctive features of EL are the use of episode/situation terms, and the two binary episodic operators $*$ and $**$, each of which takes a formula Φ and an episode η, $[\Phi * \eta]$ meaning that Φ is true in, or partially describes η, and $[\Phi ** \eta]$ meaning that Φ characterizes, or completely describes η. For example, the EL representation of "Little Red Riding Hood chased a butterfly" is

$$(\exists e_1 : [e_1 \text{ before } \text{Now}_1][(\exists x : [x \text{ butterfly}][\text{LRRH chase } x]) ** e_1])$$

EL contains many other features, including generalized quantifiers, lambda abstraction, sentence and predicate modifers, sentence and predicate reification, intensional predicates, and unreliable generalizations.

As shown above, EL uses restricted quantifiers. That is, instead of a formula of the form $(\forall x)[\Phi(x) \Rightarrow \Psi(x)]$ or $(\exists x)[\Phi(x) \wedge \Psi(x)]$, EL would have $(\forall x : \Phi(x)\Psi(x))$ or $(\exists x : \Phi(x)\Psi(x))$ Ali and Shapiro take this idea further in their network formalism, called ANALOG. ANALOG keeps together the properties of and propositions about an entity that are contained in the noun phrase that describes that entity. For example in standard FOPC, the generic statement "Small dogs bite harder than big dogs" would be represented as

$$\forall x \forall y((\text{small}(x) \wedge \text{dog}(x) \wedge \text{large}(y) \wedge \text{dog}(y)) \Rightarrow \text{bitesharder}(x, y))$$

and in EL it might be represented as

$$(\forall x : (\text{small}(x) \wedge \text{dog}(x))(\forall y : (\text{large}(y) \wedge \text{dog}(y))\text{bitesharder}(x, y))$$

in ANALOG it would be represented in a network version of

bitesharder(any x such that small(x) $\wedge$ dog(x), any y such that large(y) $\wedge$ dog(y))

Similarly, the question "What small dog bites harder than Sampson?" would be represented in FOPC as

$$\exists x(\text{small}(x) \wedge \text{dog}(x) \wedge \text{bitesharder}(x, \text{Sampson}))?$$

and in ANALOG in a network version of

$$\text{bitesharder}(\text{any } x \text{ such that } \text{small}(x) \wedge \text{dog}(x), \text{Sampson})?$$

ANALOG's benefits over standard FOPC include: analysis and generation of NL sentences are easier; it is easier to retrieve generic facts as answers to questions; it is easier to share representational structure; it is easier to define and implement subsumption between entities and propositions; there is a clearer semantics of variables and terms and propositions that contain them.

One way in which ANALOG is restricted relative to standard FOPC is that since

$$\text{greedy}(\text{any } x \text{ such that } \text{person}(x) \wedge \text{eats}(x, \text{ any } y \text{ such that } \text{cookie}(y)))$$

represents "Any person who eats any cookie is greedy," that is,

$$\forall x(\text{person}(x) \Rightarrow \forall y((\text{cookie}(y) \wedge \text{eats}(x, y)) \Rightarrow \text{greedy}(x)))$$

it cannot represent

$$\forall x(\text{person}(x) \Rightarrow (\forall y(\text{cookie}(y) \Rightarrow \text{eats}(x, y)) \Rightarrow \text{greedy}(x)))$$

that is, "Any person who eats every cookie is greedy." This last sentence, however, can be interpreted to be about the collection of all cookies, that is, "Any person who eats all cookies is greedy," and it is the representation of collections that is the topic of the paper by Franconi.

The concept of a collection is neither the same as that of a class or category, nor the same as that of a set. Classes are used to form abstraction or generalization hierarchies. A subclass differs from its superclass by listing more detailed properties for all its members. Collections, however, as Franconi says, "are contingent aggregates of objects," and subcollections have fewer members that their supercollections. On the other hand, collections differ from sets in not obeying the extensionality axiom – that identity is determined solely by their elements. To use Franconi's example, the members of the Beatles may be the same people as the founders of the Apple Records company, yet the Beatles and the founders of the Apple Records company are different collections. Similarly, the Beatles retains its identity as a collection even across a change in drummer. (Franconi does not actually discuss this situation.) In Franconi's language, called $\mathcal{ALCS}$, classes are represented as predicates, but collections are represented as individual terms.

In a NL sentence, a collection may be given a collective reading, a distributive reading, or a cumulative reading. In a collective reading, the collection participates as a whole (e..g, "John is the leader of the Beatles."). In a distributive reading, the members of the collection each participates separately (e.g., "The

Beatles were born in Liverpool.") In a cumulative reading, the members may form several subcollections, each of which participates either collectively or distributively (e.g., "The Beatles sing 'Yesterday'.") Franconi introduces two operators, called plural quantifiers, to distinguish the distributive and cumulative readings from each other and from the collective reading.

One linguistic phenomenon that is frequently either ignored entirely, or treated only minimally is negation. Moreover, negation often defeats KR formalisms that seemed perfectly adequate when negation is ignored. Iwańska's UNO representation handles negation in a wide variety of linguistic constructs. The key to this is a representation that consists of a Boolean combination of descriptions. (Boolean in the sense of expressions using *and*, *or*, and *not* as connectives.) For example, "John doesn't love Mary a lot" would be represented as if were *Either John likes but doesn't love Mary or John loves someone but not Mary or John loves Mary but less than a lot*. The system eliminates alternative interpretations if later input clarifies the situation.

Quantz and Schmitz describe a system that uses information stored in three different representation schemes to disambiguate the anaphoric referents of pronouns for a machine translation system. The three schemes are: a surface syntactic structure based on GPSG; a "structural text representation" that combines syntactic and semantic information to represent a functional structure of sentences; and a "referential text representation" that uses a Description Logic representation language to represent the entities mentioned in the text and the relations among them conveyed by the text.

In the final paper, McDonald presents KRISP, a representation system used in an NLU system to embody the meaning of texts and their pragmatic context. KRISP is in the KL-ONE family of KR languages, but has several distinguishing features, including being grounded in the lambda calculus, and putting more emphasis than most languages in that family on the representation of particular individuals.

As a whole, I think you will find the papers in this issue clearly convey the ways that NLP constrains and influences KR, you will see some solutions to some difficult issues in NLP, and you will get a concrete idea of how computers can be programed to process natural language.

Editor's Note: Since several of the papers for this special issue were longer than expected, papers by J. Joachim Quantz and Birte Schmitz, "Knowledge-Based Disambiguation for Machine Translation", and by David D. McDonald, "'Krisp': A Representation of the Semantic Interpretation of Texts", will appear in *Minds and Machines* **4** (1994), forthcoming.