KA: Situating Natural Language Understanding in Design Problem Solving

Justin Peterson, Kavi Mahesh, Ashok Goel, and Kurt Eiselt
College of Computing
Georgia Institute of Technology
Atlanta, Georgia 30332-0505
Contact: goel@cc.gatech.edu

Abstract

In this paper, we investigate the interaction between linguistic and non-linguistic processes by considering the role of functional reasoning in understanding design specifications written in natural language. We describe KA, an experimental model-based interpretation and design system which understands English language descriptions of the design problems it solves, and examine whether KA's problem-solving capabilities help i) ascertain the relevance of ambiguous design specifications and ii) identify unspecified relations between design requirements. Our results demonstrate that augmenting language processing with the ability to reason about function along the lines suggested in KA provides effective solutions to these problems in particular as well as to other problems in natural language understanding.

Introduction

The modularity of "mind" is a contentious issue in Cognitive Science. It has long been recognized that language understanding requires cognitive abilities far beyond what pure linguistic knowledge permits. It is unclear, however, in what manner, if any, linguistic and non-linguistic processes interact. Advocates for the modularity of "mind" have argued for a very limited form of interaction (Fodor, 1983; Jackendoff, 1987). Others have contended that the interaction is so open-ended as to make any boundaries between linguistic processing and the other cognitive processes insignificant (Marslen-Wilson & Tyler 89).

This issue also arises in computational models that integrate language processing with other cognitive tasks. The "strong modularity" approach has been to make the other cognitive tasks (such as path planning or expert decision making) the main task and add a natural language front-end to the system. This front-end works in service of the rest of the system and has limited abilities to translate the natural language inputs into another representation comprehensible to the rest of the system (e.g., (Hayes & Simon, 1974)). Others have opted for an "open-ended" integration, endowing their language processing systems with a variety of domain and world knowledge as well as a range of inference, explanation, and reasoning capabilities (e.g., BORIS (Lehnert et al., 1983)).

In our work, we take a third approach. We propose a modular processing architecture that contains both separate language understanding and problem solving components. However, these models interact in at least two significant ways. They share common knowledge, and they communicate the results of their reasoning to each other. The language understander, for instance, uses the results of problem solving operations such as case retrieval and model-based adaptation to resolve ambiguities. The problem solver in turn uses the decisions made by the language understander to direct the course of its own problem decomposition and solving process. A major difference between this approach to integrating language with other cognitive tasks and previous approaches is that the language understander does not need to possess either the knowledge or the reasoning abilities to solve problems or to explain the workings of physical devices. Nor does the problem solver need to know how to solve linguistic problems. All that the two need is to solve their own problems partially, be able to communicate their decisions and results with each other, and cooperate in an integrated architecture to arrive at a negotiated solution to the overall problem.

In this paper, we present our research on the integration of natural language understanding and problem solving capabilities in the context of the design of physical devices. We describe an experimental integrated system called KA that illustrates some of the benefits of building an integrated theory of multiple cognitive tasks focusing on language understanding and its interaction with design problem solving. We show how our work on KA imposes constraints on the target representation of natural language understanding and how the integrated approach redefined classical problems in language processing such as ambiguity and underspecification in terms of the overall goals of the KA system. Language understanding imposes constraints, in return, on the task structure of the design problem solver.

The Problem

As argued in (Pittges et al., 1993), design and diagnosis of physical devices have been construed narrowly as problem solving tasks, ignoring other cognitive tasks such as language understanding, learning, and visual simulation that are part of the overall task of design. In our work, we look at the "real" design task which involves understanding written design requirements, acquiring knowledge from books and manuals, and understanding feedback from users and other experts. In our view, neglecting the role of natural language understanding in the overall task of design produces underconstrained theories, design systems that communicate in artificial languages, and the hand-coding of design knowledge available in books and manuals. Similarly, treating the understanding of design specifications as a pure language understanding problem in the absence of design problem solving also results in underconstrained theories with ill-defined output representations.

In our previous work (Goel, 1991a), we viewed device design as involving memory, comprehension and learning processes in addition to problem solving in the form of device modeling, simulation, and redesign. We represented a designer's understanding of the functioning of devices in terms of structure-behavior-function (SBF) models. SBF models specify the internal causal behaviors that compose the functions of the structural elements of a device into its functions. We showed both how SBF device models support memory and learning processes in addition to problem solving, and how these processes impose additional constraints on the models and on model-based reasoning (E.g., Goel, 1991b). We also showed how these tasks impose constraints on the semantics of SBF device models and model-based functional reasoning.

In our current work, we have developed an experimental model-based interpretation and design system called KA which embodies the expansive view of reasoning about physical devices that includes the comprehension of texts and dialogues on the design of physical devices as well as the design, diagnosis, and redesign of devices. KA understands English language descriptions of the design problems it attempts to solve. In doing so, it addresses well-known problems of natural language understanding such as resolving ambiguity, interpreting indirect statements, and inferring unspecified information. For quite some time, the conventional wisdom has been that these problems are best addressed by constraint-based methods that employ a knowledge of natural language's distributional structure and rules of combination. KA diverges radically from this conventional wisdom, offering a fresh approach to oft-studied problems. In KA, functional and causal knowledge contained in SBF models as well as model-based methods for case retrieval and adaptation (i.e., not just problem-solving knowledge but the very results of the performance of problem solving) are used to understand natural language. The very same knowledge and methods in KA's design problem solving also bias linguistic decisions for ambiguity resolution, provide the insight necessary to interpret indirect statements, and supply the information absent in underspecified texts. KA demonstrates the merits of integrating design problem solving with language understanding by providing partial solutions to some of the most difficult problems in language understanding as well as design problems.

Reformulating Linguistic Problems

Because research in natural language understanding has so decidedly separated the problems of linguistic analysis and sentence understanding from the other problems that must be resolved in the meaningful interpretation of texts, the resources that have been applied in most text understanding systems have been severely limited. Real world tasks such as designing from written requirements specifications provide a context which refocuses many of the linguistic problems that have been central to the field, allowing us to consider novel solutions to time-worn yet unresolved problems.

In taking this approach to our work, we have found that linguistic problems and the problems of large texts that are inherent to written design requirements actually become problems which require reasoning about the design of the device in question. Requirements specifications are notoriously confusing and incomplete, providing poor articulations of the design requirements. So far, the KA project has encountered several problems which we illustrate with the use of sample Text 1[1] shown in Figure 1. Its corresponding SBF description appears in Figure 2.

Text 1: The system shall consist of two computer elements interfaced to each other over an xxxx link. Computer A shall send a K byte request packet to Computer B every M seconds. In response to the request packet Computer B shall send an L byte response packet back to Computer A. Packet encoding is N bit ASCII.

Figure 1: Sample Design Specification

Input:	*Substance:* Request-Message
	Size: K Byte
Output:	*Substance:* Response-Message
	Size: L Byte
Structure:	*Component:* Computer A
	Component: Computer B
	Component: Link
	Baud-Rate: Z

Figure 2: Functional Specification for Text 1

Ambiguity

Typically, the natural language surface form of written functional specifications indirectly refers to a design requirement. The writers of such requirements specifications can usually point out specific statements about the inputs and the outputs of the components that indicate the general functions of the device, but in no way are these requirements indicated in the natural language surface form. The system must be able to use the indirect statements given to infer the design requirements because, failing to do so, it would be unable to pursue a design solution.

For example, to successfully understand Text 1 as a design problem, one must be able determine the function of the device being described, its inputs, and its outputs. However, none of these characteristics are explicitly described in the text. The text describes the device (referred to as "the system") in terms of its components (e.g., " computer A", "computer B"), their connectivity and types of information they transmit (e.g., "request packet"). Nothing is stated about the function of "the system". Its inputs and outputs are not even referred to. To understand the English description in Text 1, KA must be able to generate functional requirements from alternate information sources, using the information provided in the text (e.g., K byte request packet, L byte response packet) as constraints on generation. Before KA can use this information to constrain generation, however, it must determine its relevance to the function of the device. In the text, it is unclear whether a "request packet" and a "response packet" are related to the function of the device or to its internal workings. An understanding of "requests" and "responses" reveals, however, that these information packets are the inputs and outputs

[1] Some design details have been masked to protect our sponsor's proprietary information.

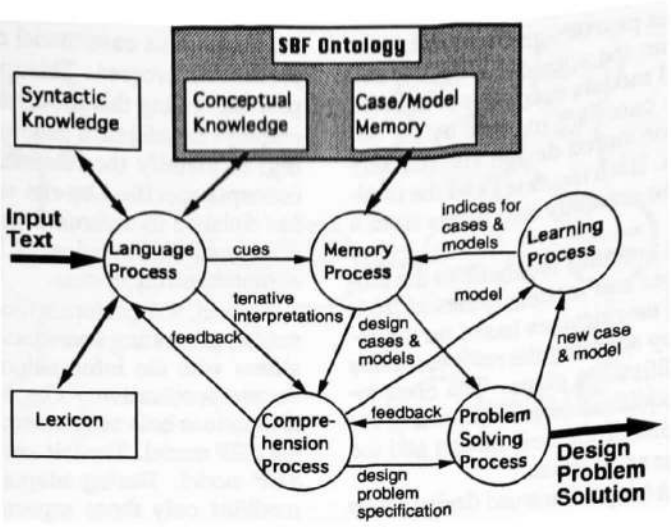

Figure 3: KA System Architecture

of the device. If KA is to make use of such indirect statements about functional requirements, it must use its specific knowledge about computing networks to infer that a *request* message passed from one computer to another is the input to the system, and the message sent in *response* is the output.

Underspecification

The natural language surface form does not indicate the relationships between design requirements. It identifies detailed device requirements without articulating how these requirements relate to one another. For example, although baud rate, size of an information packet, and frequency of transmission have a well-defined relationship to one another in a computer network, requirements specifications for computing devices such as the one in Text 1 rarely, if ever, mention this relationship. A superficial analysis of the natural language surface form would produce three separate requirements (one for the baud rate, one for information packet size, and one for the frequency of transmission), entailing an extremely inefficient problem decomposition. If the system is to pursue designs efficiently, it must combine these disparate requirements into a coherent specification of the design. For example, in Text 1, using the frequency of transmission in combination with the size of the information packets, KA can infer the appropriate baud rate for the 'xxxx' link between the system's two computers.

In order to map requirement specifications to useful functional descriptions, KA must effectively resolve ambiguity, fill in missing details, identify the relevance of indirect statements, and combine related information. To do so efficiently, KA uses memory, comprehension, and problem solving processes in addition to pure language processes. In this way, these "background" processes in KA provide a robust context in which effective communication in natural language becomes feasible.

The KA architecture

KA accepts a requirements specification written in English and produces a design expressed as a structure-behavior-function (SBF) model which meets the design requirements. The functional architecture for KA is illustrated in Figure 3. It consists of several knowledge sources containing syntactic, conceptual, and episodic knowledge and employs memory, comprehension, problem solving, and learning processes in addition to a language process. Each of KA's components uses a different knowledge sources to bring a unique capability to the system.

The **language** process uses syntactic and conceptual knowledge (i.e., a knowledge of concepts and the relations between them) to generate cues for the memory processes as well as tentative interpretations for the comprehension processes. If the KA system is to effectively communicate in natural language, it must be able to resolve the different types of ambiguities (e.g., lexical and structural ambiguities) that arise in written texts. The language process in KA uses an early-commitment processing strategy with robust error-recovery to resolve word sense ambiguities (Eiselt, 1989). This mechanism has proved itself to be quite effective. Its early-commitment strategy provides the system with the ability of pursuing a tentative interpretation of the discourse. This allows the system to discover the entailments of this line of interpretation, bringing other processes on-line early in the course of language understanding. In situations where the early decision is incorrect, the error-recovery mechanisms may use feedback from the comprehension process (or problem solving) to reactivate a previously retained alternative interpretation.

The language process consists of two components, a parser which produces syntactic structures and a semantic network that produces conceptual interpretations. Consistent with the early-commitment processing strategy, the semantic network resolves word-sense ambiguities by considering processing choices in parallel, selecting the alternative that is consistent with the current context, and deactivating but retaining the unchosen alternatives for as long as space and time resources permit. If some later context proves the initial decision to be incorrect, retained alternatives are reactivated without reaccessing the lexicon or reprocessing the text.

The **memory** process retrieves past design cases and case-specific SBF device models from the episodic memory, and stores newly-acquired cases and models in memory. In order to ensure effective retrieval, the cases are indexed by the device structure and function of the stored design and the SBF models are indexed by the cases. Both the cases and the models are represented in a common ontology that arises from a qualitative physics.

The **comprehension** process provides feedback to the language process based on the cases and models retrieved from memory, and forms a model of new devices based on the input from the language process by adapting the retrieved cases and models using generic modification plans. The comprehension process selects these modification plans by using the differences between the functions of the new device and the functions of the retrieved design as an index.

The **problem solving** process adapts retrieved design cases to solve new design problems. Repair plans are used to perform the adaptations to the old design's structure. The new design is verified by a qualitative simulation of its SBF model and produced as a solution to the design problem. Finally, the solution is sent to the learning process for later reuse.

The **learning** process learns indices to new design cases and device models. An index specifies multiple dimensions of generalizations that pertain to a design's various functions. The learning process generates these generalizations by a process described in (Bhatta & Goel, 1992).

KA at Work

In the current investigation, we examined whether KA's SBF models and ability to reason about function could help infer the relevance of indirect statements as well as identify relationships between design details underspecified in the natural language surface form. Our approach to these problems relied extensively on KA's memory of design cases, case-specific SBF models, and model-based adaptation. Our results indicate that:

- Using case-specific SBF models as the starting point for the interpretation of a requirements specification enables the language process to identify the relevance of statements that, on the surface, appear to be irrelevant to the design requirements.
- Model-based adaptation prevents missing the big picture by fashioning a functional specification from a disparate set of requirements that do not directly make statements about the function of the device to be designed.
- Using KA's SBF models and diagnosis capability ensures that critical relationships between design details that are left unarticulated in the written requirements are identified and that these relations impact the structural specification extracted from the text.

The Process

In this investigation, we focused on extracting the critical features from ill-specified texts such as Text 1. Briefly, KA processes Text 1 in the following manner. First, using its memory of past design cases and case-specific SBF models, KA employs a complete SBF model as a baseline from which the relevance of indirect statements about the function of the device can be inferred. The memory process extracts a relevant model from its case/model memory and sends it to the comprehension process. This model is fed back to the language process. Using this model as baseline, the language process employs its inference generation capability (i.e., marker passing) to identify the relations between the feedback and the concepts specified by the text. Once the language process has finished its inference generation, it produces a tentative functional specification of the design which is sent to the comprehension process.

Second, KA performs model-based adaptation on the SBF model, generating a new case-specific SBF model that is consistent with the information provided in the tentative functional specification. The comprehension process identifies distinctions between the tentative functional specification and the SBF model. Then, it uses these distinctions to modify the SBF model. During adaptation, the comprehension process modifies only those aspects of the stored model that conflict with the tentative specification. This leaves a significant number of design details unaffected. In effect, design details are transferred from the stored SBF model to the new device model.

Third, during adaptation, KA identifies those distinctions that require changes to the new device's structure and adapts the tentative design specification accordingly. The resulting new case is stored in KA's memory of case-specific models for later reuse. Below, we discuss of these steps in detail.

Inferring the Relevance of Indirect Statements The memory process begins by sending a relevant SBF model to the comprehension process which feeds it back to the semantic network. The semantic network activates the model's corresponding concepts and conceptual relations. For example, in the subsection of the semantic network displayed in Figure 4, the concepts **Old-Device**, **Y Byte**, **Response message**, and **Response** and the primitive conceptual relations that relate these concepts (e.g., *parameter*, *instance*, *part*) are activated by feedback from the Comprehension process.

The input is then parsed and the content words of each sentence are passed to the semantic network which initiates marker passing at each word's corresponding concept. Using the feedback as a bridge, the semantic network identifies conceptual relations between the concepts activated by the text and constructs a new set of inferences. The new inferences relate concepts specified in the text to the functional specification of the new device. Finally, a tentative functional specification of the new device is produced from these new inferences and sent to the comprehension process.

To see how the feedback acts as a bridge between the concepts activated by the text, consider the subsection of the semantic network displayed in Figure 4. In this semantic network, the concept **L Byte** is activated by the appearance of the words "L byte" in the input text, **Response** is activated by the appearance of "response", and **New-Device** is activated by the appearance of "system". Using only these active concepts, the semantic network would be *unable* to identify critical conceptual relations such as that between **L byte** and the **Output** of the **New Device** because the active concepts **L byte** and **New Device** are only distantly related to each other. Basing its decision on the length of the path between the two concepts, the semantic network would deem it unlikely that the text intends to relate these concepts without further evidence.

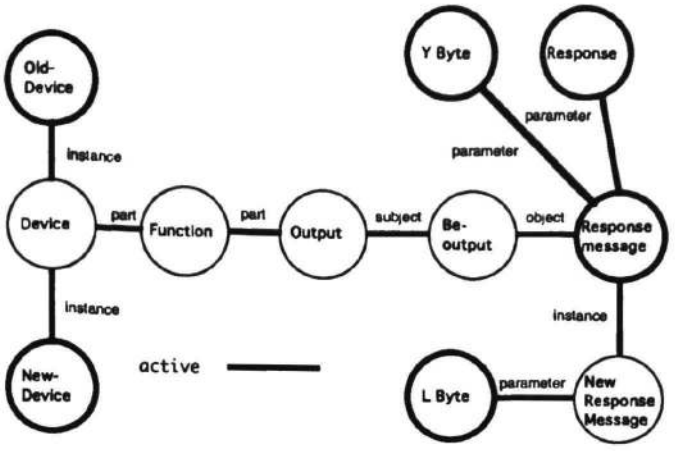

Figure 4: Identifying the Relevance of *response* and *L Byte*

However, when the semantic network begins with feedback-activated concepts such as **Response message**, conceptual relations such as that between **L byte** and **New-Device** as well as that between **Response** and **New-Device** can be identified. The concepts activated by the text are more closely-related to the concepts activated by feedback than they are to each other, so the semantic network can identify conceptual relations between the text-activated concepts and feedback-activated concepts (as indicated by the active paths in Figure 4). This produces inferences that serve to relate the text-activated concepts, inferences that identify the relevance of concepts such as **Response** and **L byte** to the function of the **New-Device** .

Generating a New Functional Specification Given a tentative and underspecified functional specification produced by the language process, the comprehension process compares the SBF model and this underspecified functional specification to determine the distinctions between the two device descriptions. It notes distinctions that are extremely significant such as the distinction between the size of the response package (*L bytes* versus *J bytes*) and those that are less significant such as the difference in the names of the components (*A* versus *C*).

Once all of these distinctions have been collected, the comprehension process begins adapting the stored SBF model. It modifies the component names such that they are consistent with the new functional specification, changes the sizes of the response package and request package, etc. In doing so, it transfers a large amount of the stored SBF model to the new SBF model. For example, it transfers the types of the components in the old device to components of the new device. The result is that all of the design details are filled in, and a significant number of assumptions are made. The comprehension process assumes, for example, that the new device has the same behavioral descriptions as the stored device and the same structural description.

Identifying Relationships between Design Details During the adaptation of the stored SBF model, the assumption that the structural specifications of the new and stored designs are equivalent is examined. The comprehension process considers each of the differences it has identified between the new specification and the stored specification, looking for those differences that may require modifications to the device structure. Differences that are particularly relevant are differences in device inputs and outputs. For example, in this example, the distinction between the size of the new design's output and the stored design's output (i.e., *L bytes* versus *J bytes*) imposes new constraints on the structure of the new design. The comprehension process collects these difference and orders them with respect to their priority.

Examining them in order of their priority, the comprehension process retrieves generic modification plans that rectify the differences between the new design specification and the stored design specification by adapting the stored SBF model. Generic modification plans are indexed in memory by the type of differences they reduce and are made available by the memory process. In achieving their ends, generic modification plans manipulate, delete, and augment device structure.

After the comprehension process has received the generic modification plans from the memory process, it begins to diagnose the new model's failure, in this particular example, its failure to produce the desired output. While investigating the causes of the new design's failure, the comprehension process identifies the relationship between the frequency of transmission (i.e., *every M seconds*), the size of the response packet (i.e., *L bytes*) and the baud rate of the link component. Using the qualitative relations specified in the stored SBF model, it notes that baud rate of the link component limits the amount of information that can be transferred at a particular frequency. It concludes the baud rate of the current link component is to low and that increasing the baud rate of this component would provide for the size of the response packet

in the new design and the desired frequency of transmission.

Given the diagnosis and the generic modification plans, the comprehension process "repairs" the structural specification of the new design, replacing the link component in the stored SBF model with a link component that has a higher baud rate.

Discussion and Conclusions

In this work, we believe that we have found partial answers to some of the issues that arise in integrating design problem solving and natural language understanding:

- **The meaning of understanding language:** A piece of text describing a design specification is *understood* if it can be represented in the target (SBF) language and if it results in a successful design of the specified device. In general, the success of language understanding is determined in terms of the success of the overall cognitive task.
- **Feedback to NLP:** We have identified three ways in which the problem solving task can provide feedback to the language process. First, previous experiences and other knowledge structures retrieved from the problem solver's memory provide appropriate contexts for language understanding. Second, the cost of problem solving for alternative interpretations of texts gives a good decision metric for resolving the ambiguities between interpretations. Finally, the ontology of the task determines correctness of an interpretation thereby providing feedback to the language process.
- **Constraints imposed on NLP:** The overall cognitive task affects the NLP task by redefining classical problems in NLP such as ambiguity resolution. It also suggests new ways of solving those problems. In addition, the task demands that the representations and the output of language understanding be expressive enough to capture the distinctions that are significant in the ontology of the task.
- **Constraints imposed by NLP:** The integration of NLP with other tasks requires the task to be able to process incomplete and incremental input and provide immediate feedback to the language process. It demands an iterative/cooperative solution to the overall task. It also requires that the overall system must have problem solving ability required to solve the linguistic problems: for instance, it must have a case available in memory for resolving a particular ambiguity even if that case is never used in design problem solving.

What lessons regarding the modularity of 'mind', even tentative ones, can be drawn from our work on KA? KA certainly is modular, but the nature of the modularity depends on the level at which it is analyzed. Modularity in KA can be viewed at the levels of task, process and knowledge, and representation. At the task level, 'language processing' and 'problem solving' are distinct modules, characterized by the types of information they take as input and give as output. At the next level, some of the processes are task-specific but others are shared. Language processing and problem-solving, for example, are both informed by the same memory processes which retrieve episodic and conceptual information. Similarly, some of the knowledge is task-specific and some of it is shared. Only the language processes use lexical and syntactic knowledge, and only the problem-solving processes use knowledge of the primitive structural elements out of which devices are composed (e.g., types of structural components, substances, and relations among them). On the other hand, both the language and problem solving processes both employ functional and causal knowledge of devices. Finally, at the level of representation, the language and problem-solving processes share the same vocabulary for representing conceptual knowledge. Thus, from the viewpoint of KA, the issue of modularity is much more complex than either the orthodox 'modularists', such as Fodor and Jackendoff, or the 'nonmodularists', such as Marlen-Wilson and Tyler, suggest.

Acknowledgements

This work has been supported by the National Science Foundation (research grant IRI-92-10925), the Office of Naval Research (research contract N00014-92-J-1234), and Northern Telecom (research gift). The authors would like to thank their colleagues Sambasiva Bhatta, Andres Gomez de Silva Garza, Jeff Pittges, and Eleni Stroulia.

References

Bhatta, S. & Goel, A. (1992). Use of mental models for Constraining Index Learning in Experience-Based Design. In *Proceedings of AAAI workshop on Constraining Learning with Prior Knowledge* (pp. 1-10). San Jose, CA.

Eiselt, K. (1989). Inference Processing and Error Recovery in Sentence Understanding. Doctoral Dissertation, Irvine, CA: University of California, Irvine, Department of Information and Computer Science.

Fodor, J. (1983). *Modularity of Mind.* Cambridge, MA: MIT Press.

Goel, A. 1991a. A Model-Based Approach to Case Adaptation. In *Proceedings of the Thirteenth Annual Conference of the Cognitive Science Society* (pp. 143-148). Hillsdale, NJ: Lawrence Erlbaum Associates.

Goel, A. 1991b. Model Revision: A Theory of Incremental Model Learning. In *Proceedings of the Eighth International Conference on Machine Learning* (pp. 605-609). San Mateo, CA: Morgan Kaufmann Publishers.

Hayes, J. R. & Simon, H. A. (1974). Understanding Written Problem Instructions. In L. W. Gregg (Ed.), *Knowledge and Cognition* (pp. 167-200). Potomac, MD: Lawrence Erlbaum Associates.

Jackendoff, R. (1987). *Consciousness and the Computational Mind.* Cambridge, MA: MIT Press.

Lehnert, W. G., Dyer, M. G., Johnson, P. N., Yang, C. J., & Harley, S. (1983). BORIS - An Experiment in In-Depth Understanding of Narratives. *Artificial Intelligence*, 20(1), 15-62.

Marslen-Wilson, W. & Tyler, L. (1989). Against Modularity. In J. Garfield (Ed.), *Modularity in Knowledge Representation and Natural Language Understanding.* Cambridge, MA: MIT Press.

Pittges, J., Eiselt, K., Goel, A., Gomez, A., Mahesh, K., & Peterson, J. (1993). KA: Integrating Natural Language Processing and Problem Solving. In *Proceedings of the Fifteenth Annual Conference of the Cognitive Science Society* (pp. 818-823). Hillsdale, NJ: Lawrence Erlbaum Associates.