Report Number: WUCS-88-21

1988-08-01

A Parallel Distributed Approach to Parsing Natural Language Deterministically

Stan C. Kwasny

The Determinism Hypothesis (Marcus, 1980) has given rise to much debate. The hypothesis makes explicit the idea that Natural Language interpretation need not depend in any fundamental way on the use of pseudo-parallelism or backtracking. We are exploring the consequences of this hypothesis in attempting to develop approaches to parsing which integrates current work in parallel distributed adaptive networks. We follow the basic approach of "Wait-and-See" parsing (WASP) which has shown the Natural Language interpretation of all but some varieties of "garden-path" sentences can be deterministically performed using a stack, a buffer for sentence constituents, and partitioned packets of... **Read complete abstract on page 2.**

A Parallel Distributed Approach to Parsing Natural Language Deterministically

Stan C. Kwasny

Complete Abstract:

The Determinism Hypothesis (Marcus, 1980) has given rise to much debate. The hypothesis makes explicit the idea that Natural Language interpretation need not depend in any fundamental way on the use of pseudo-parallelism or backtracking. We are exploring the consequences of this hypothesis in attempting to develop approaches to parsing which integrates current work in parallel distributed adaptive networks. We follow the basic approach of "Wait-and-See" parsing (WASP) which has shown the Natural Language interpretation of all but some varieties of "garden-path" sentences can be deterministically performed using a stack, a buffer for sentence constituents, and partitioned packets of rules. Specifically, we replace the rule packets with a single neural network and train the network with an appropriate training set. Training sets derive from either examples of existing WASP grammars or from traces of sentence processing.

A PARALLEL DISTRIBUTED APPROACH TO PARSING NATURAL LANGUAGE DETERMINISTICALLY

Stan C. Kwasny

WUCS-88-21

August 1988

Department of Computer Science
Washington University
Campus Box 1045
One Brookings Drive
Saint Louis, MO 63130-4899

Abstract

The Determinism Hypothesis (Marcus, 1980) has given rise to much debate. The hypothesis makes explicit the idea that Natural Language interpretation need not depend in any fundamental way on the use of pseudo-parallelism or backtracking.

We are exploring the consequences of this hypothesis in attempting to develop approaches to parsing which integrates current work in parallel distributed adaptive networks. We follow the basic approach of "Wait-and-See" parsing (WASP) which has shown that Natural Language interpretation for all but some varieties of "garden-path" sentences can be deterministically performed using a stack, a buffer for sentence constituents, and partitioned packets of rules. Specifically, we replace the rule packets with a single neural network and train the network with an appropriate training set. Training sets derive from either examples of existing WASP grammars or from traces of sentence processing.

This work was supported in part by the National Science Foundation under grant No. DCR-8518725. Partial support was also received form the Center for Intelligent Computer Systems at Washington University.

A Parallel Distributed Approach to Parsing Natural Language Deterministically[1]

Stan C. Kwasny

Department of Computer Science
Washington University
St. Louis, Missouri 63130

WUCS-88-21

1. Introduction

Mitchell Marcus (1980) has proposed the Determinism Hypothesis which makes explicit the idea that Natural Language interpretation need not depend in any fundamental way on the use of pseudo-parallelism or backtracking.

The consequences of this hypothesis are being explored here through the development of parsing techniques which integrate Marcus' work with current work in parallel distributed adaptive networks. The basic approach of "Wait-and-See" parsing (WASP) is being faithfully pursued. This approach has provided convincing evidence that Natural Language interpretation for all but some varieties of "garden-path" sentences can be deterministically performed with a stack, a buffer for sentence constituents, and partitioned packets of rules. Specifically in our work, the collection of rule packets has been replaced with a single neural network and trained with an appropriate training set. Training sets derive either from examples of existing WASP grammars or from traces of sentence processing.

Recently, language processing formalisms based on neural networks have been proposed by several researchers (e.g., Cottrell (1985a, 1985b), Selman & Hirst (1985), Waltz & Pollack (1985)). Fanty (1985) implements the Cocke-Younger-Kasami algorithm for parsing context-free languages as a connectionist network in which an arbitrary (15 word) limit is imposed on the lengths of input sentences. All of these attempts share many of the same advantages and disadvantages., but none of the models allow sentence "streams" to be processed iteratively, nor do they offer an explanation of "garden path"

---

[1] This work was supported in part by the National Science Foundation under grant No. DCR-8518725. Partial support was also received from the Center for Intelligent Computer Systems at Washington University.

phenomena. We have considered the adaptability of several well-known conventional natural language processors to implementation as neural networks. Deterministic parsing showed the most promise since it offers insight into garden path sentences and does not require backtracking nor pseudo-parallelism to work. Furthermore, it is consistent with standard neural network learning techniques.

Our engineering goal in this work is to demonstrate that an appropriately designed neural network can capture the same generalities as the WASP rule packets. This goal has been achieved for a small grammar and is being evaluated for larger grammars.[2]

Another goal is one of economy. Can the effort of designing WASP grammars be minimized? In our system, the "grammar" (i.e., network) is derived automatically from the training sequences, typical problems of rule coordination and debugging of rule packets will not occur. Since the rules themselves serve little useful purpose, in many cases, beyond their function in the system, there seems to be no real advantage in having laboriously-constructed rule packets.

Using the network, experiments can be performed easily with various grammar learning strategies. We are examining both deductive and inductive learning strategies. Two types of training are being pursued: "deductive" training in which the training sequence is derived from the rules of a WASP grammar and "inductive" training in which the training sequence is derived from the states of sentence processing.

We hope to exploit the robustness properties of neural networks in order to show the value of this approach in developing robust grammars for "ill-formed" inputs (e.g., terse messages, elliptical inputs). Networks support soft-constraint satisfaction which make them well-suited for this type of task. Traditionally, this goal has been pursued through the addition to a conventional parser of mechanisms such as relaxation techniques and meta-rules which are specifically designed to deal with those problems.

2. The WASP Advantage

WASP systems process input sentences primarily left-to-right and deterministically. The absence of backtracking is an important advantage in developing a neural network-based parser, particularly since training occurs with sample inputs and their individual processing steps.

---

[2] In related work on rule-based expert systems, Gallant (1988) found that it was easy and advantageous to re-designing such systems around neural networks.

A single processing step consists of selecting a rule that can fire from the appropriate rule set. The rule set is determined by the current (top-level) node of the structure being built and conflicts are resolved from the static ordering of rules within the rule set. Once selected, the rule is fired and its action is performed. The action effects changes on the stack and buffer. After a series of processing steps, a termination rule will fire terminating the processing and leaving the parse structure on the top of the stack.[3]

The work of Mitch Marcus has provided the groundwork for a large covering grammar of English in the style of PARSIFAL rules. The rules provide an "existence proof" that proper processing mechanisms exist when training a network to perform tasks identical to those reflected in the rules. Unlike Marcus, however, the network approach to deterministic parsing does not require that large numbers of rules be tediously constructed for the successful performance of the system. One complaint often made against neural networks is that the resultant network of weights and connections do not produce revealing insights into various phenomena under study. One could argue that a system of often obscure rules, understood by perhaps only the creator himself, can be critiqued similarly. Thus, even with rules, the situation may not be transparent if the rules or the interaction of the rules is complicated.

Several researchers have found rule-based, WASP systems valuable in exploring various aspects of language processing. PARAGRAM (Charniak, 1983) considers all rules in parallel and "scores" each test performed on the left-hand side of a rule according to predefined weights. The rule with the best score is fired. Thus, some rule always fires and parsing never fails due to a collapse in the rules. This is providing a context in which to explore ill-formedness, among other phenomena. The essence of this part of PARAGRAM is subsumed within our PDP framework.

Other work on wait-and-see parsing that relates to this research includes Fidditch (Hindle, 1983) which extends Marcus' parser to non-fluencies by providing for multiple stack entries (i.e., multiple sentence fragments) as a legitimate parse, and LPARSIFAL (Berwick, 1982) which attempts to posit PARSIFAL rules from examples. This latter work may eventually also be substantially subsumed in our system.

In fairness, not everyone agrees with the primary arguments in favor of the Determinism Hypothesis. Briscoe (1983) provides examples which argue that Marcus has not accurately distinguished garden-path sentences from others. In a companion paper,

[3] This is an over-simplified view of the processing involved, but accurately reflects Winston's discussion. A more accurate view, including a discussion of attention-shifting (AS) rules, rule priorities, etc., can be obtained from Marcus' thesis.

Ritchie (1983) takes issue with some of the implementation details of PARSIFAL, and with what he perceives as an over-commitment to determinism. This, Ritchie concludes, severely inhibits the grammar-writer in that no non-deterministic behavior (e.g., handling multiple word senses) is possible.

Being fully aware of the potential limitations involved, we argue that WASP systems have proven themselves as viable research tools and objects of study. This work attempts to further generalize along the lines of this tradition.

3. The PDP Advantage

Parallel Distributed Processing (PDP), connectionist networks, or neural networks offer many significant and important advantages in building intelligent systems which could easily be exploited in language processing. These advantages can be maintained in a properly-constructed PDP language processing system.

The soft-constraint satisfaction property offered by PDP is well known. Networks degrade gracefully when presented with input cases that are similar to, but not precisely like the training cases. Visual models based on neural networks have the ability to reconstruct an entire image from an incomplete image in *completion tasks* (Rumelhart & McClelland, 1986). It seems clear from these experiments that neural nets are capable of exploiting those features of a situation which make it unique from other situations. The knowledge required in such tasks is not attached to a single unit in the network, but distributed across many units, perhaps the entire network. Graceful degradation occurs as a natural by-product of such a network model.

Our research is leading toward an examination of robust language processing using a deterministic PDP language system such as the one described here. In some sense, robustness is the most important reason for considering this approach. Attempts to process ill-formed inputs using conventional (symbolic) means, though successful in limited ways, have generally resulted in somewhat ad hoc methodologies that are tedious to use and have there own "sharp edges" in performance.[4] In the small grammars we have examined at this point in the research we are just beginning to derive results on this problem and hesitate to report anything or make any sort of claims until those results have been fully studied.

---

[4] See, for example, Kwasny & Sondheimer, (1981); Weischedel & Sondheimer, (1983); Granger, (1983); Weischedel & Ramshaw, (1987).

Language processors that need to interface closely with semantic, pragmatic, and other processing components can potentially do so in more natural ways through properly interconnected specialized networks, assuming such components are themselves neural networks. Work has progressed in developing such components faster than language models have been developed that use such components and thus the increased importance of such language processing components.

The PDP approach attempts to simulate aspects of the behavior of neurons or clusters of neurons as nodes and their interactions with other neurons or clusters in a relatively large inter-connected network of units. Several characteristics of this view are attractive in attempting to resolve some of the issues mentioned above.

4. Example

A simple grammar has been implemented to test the basic mechanisms of grammar specification and learnability. In this simple example, no attempt was made to examine its ability to work on ill-formed input, nor with a semantic component. This is consistent with the approach in WASP. Input elements were coded according to features important to the grammar.

The grammar used in this experiment is taken from the simple, pedagogical example in Winston (1984). The rules are organized according to the current (top-most) node of the structure being built. Figure 1 shows the grammar in rule form. Simple tests are performed on the stack and buffer of the current state of processing to determine which rule to fire. In the original system by Marcus, each rule has a priority rating which is used to determine the order in which to attempt the rules and noun phrases are processed by "attention shifting rules." These complexities are not reflected in this example.

Figure 2 shows the same grammar rules coded as training data for a PDP run. An attempt has been made to keep the coding scheme in line with the rule version, but at the expense of a very sparsely coded network and one that consequently takes more training cycles. In later versions, we expect coding to be more dense and the correspondence between symbols in the original grammar and input units to be less direct.

The rule partitioning of the PARSIFAL rules is realized directly in the coding of the PDP data and therefore no partitioning is required. Attachments to nodes on the stack are also part of the coding. Note that finite limitations exist for the length of the buffer (as in PARSIFAL), the number of items that can be stacked, and the number of attachments that each item can have.

Rule	S1	If	first item is a noun-phrase node
		then	attach the first item
Rule	S2	If	first item is a verb-phrase node
		then	attach the first item
Rule	S3	If	first item is a verb
			current node has a noun phrase attached
		then	create a verb-phrase node
Rule	S4	If	the buffer is empty
		then	stop, reporting a successful parse
Rule	VP1	If	first item is an auxiliary verb
			the second item is a verb
		then	attach the first item
Rule	VP2	If	first item is a verb
		then	attach the first item
Rule	VP3	If	first item is a noun-phrase node
		then	attach the first item
Rule	VP4	If	first item is a preposition
			the second item is a noun-phrase node
			current node has a verb and a noun phrase attached
		then	create a prepositional-phrase node
Rule	VP5	If	first item is a prepositional-phrase node
		then	attach the first item
Rule	VP6	If	the buffer is empty
		then	drop the current node into the buffer
Rule	PP1	If	first item is a preposition
		then	attach the first item
Rule	PP2	If	first item is a noun phrase
		then	attach the first item
Rule	PP3	If	the buffer is empty
		then	drop the current node into the buffer

Figure 1: Three Packets of Grammar Rules. *WASP rules taken from introductory text by Winston (1984). Rules are organized into rule packets. In this simplified presentation, conflicts are resolved through static ordering.*

Components			S-Rules				VP-Rules						PP-Rules		
			1	2	3	4	1	2	3	4	5	6	1	2	3
Top of Stack	Nodes	S	+	+	+	+	?	?	?	?	?	?	?	?	?
		NP	?	?	?	?	?	?	?	?	?	?	?	?	?
		VP	?	?	?	?	+	+	+	+	+	+	?	?	?
		PP	?	?	?	?	?	?	?	?	?	?	+	+	+
	Attachments	S	?	?	?	?	?	?	?	?	?	?	?	?	?
		NP	?	?	+	?	?	?	?	+	?	?	?	?	?
		VP	?	?	?	?	?	?	?	?	?	?	?	?	?
		PP	?	?	?	?	?	?	?	?	?	?	?	?	?
		verb	?	?	?	?	?	?	+	+	?	?	?	?	?
		prep	?	?	?	?	?	?	?	?	?	?	?	?	?
		noun	?	?	?	?	?	?	?	?	?	?	?	?	?
		adj	?	?	?	?	?	?	?	?	?	?	?	?	?
		det	?	?	?	?	?	?	?	?	?	?	?	?	?
		aux	?	?	?	?	?	?	?	?	?	?	?	?	?
Buffer	1st Item	S	?	?	?	-	?	?	?	?	?	-	?	?	-
		NP	+	?	?	-	?	?	+	?	?	-	?	+	-
		VP	?	+	?	-	?	?	?	?	?	-	?	?	-
		PP	?	?	?	-	?	?	?	?	+	-	?	?	-
		verb	?	?	+	-	?	+	?	?	?	-	?	?	-
		prep	?	?	?	-	?	?	?	+	?	-	+	?	-
		noun	?	?	?	-	?	?	?	?	?	-	?	?	-
		adj	?	?	?	-	?	?	?	?	?	-	?	?	-
		det	?	?	?	-	?	?	?	?	?	-	?	?	-
		aux	?	?	?	-	+	?	?	?	?	-	?	?	-
	2nd Item	S	?	?	?	-	?	?	?	?	?	-	?	?	-
		NP	?	?	?	-	?	?	?	+	?	-	?	?	-
		VP	?	?	?	-	?	?	?	?	?	-	?	?	-
		PP	?	?	?	-	?	?	?	?	?	-	?	?	-
		verb	?	?	?	-	+	?	?	?	?	-	?	?	-
		prep	?	?	?	-	?	?	?	?	?	-	?	?	-
		noun	?	?	?	-	?	?	?	?	?	-	?	?	-
		adj	?	?	?	-	?	?	?	?	?	-	?	?	-
		det	?	?	?	-	?	?	?	?	?	-	?	?	-
		aux	?	?	?	-	?	?	?	?	?	-	?	?	-
			→	→	→	→	→	→	→	→	→	→	→	→	→
Output	Attach (A)	A1	+	+	-	-	+	+	+	-	+	-	+	+	-
	Create (C)	C VP	-	-	+	-	-	-	-	-	-	-	-	-	-
		C PP	-	-	-	-	-	-	-	+	-	-	-	-	-
	Drop (D)	D	-	-	-	-	-	-	-	-	-	+	-	-	+
	Stop (S)	S	-	-	-	+	-	-	-	-	-	-	-	-	-

Figure 2: WASP Rules for PDP system. *Thirteen rules shown as data to the PDP network. Plusses (+) indicate the presence of a feature while minus (-) indicates its absence. Question marks (?) indicate a "don't care" item which during training gets converted randomly into either a plus or a minus.*

The finite nature of the stack, in theory, limits the processing capabilities of the system. It could be argued, however, that placing limitations on the depth of recursion of language structures (i.e., the stack) is preferred to limitations on sentence length as in other PDP models mentioned earlier.

Actions are coded as output to be learned by the network. In this example, possible actions are Attach (first item), Create VP, Create PP, Drop, and Stop. More actions are added by simply introducing more output level elements in the network. For example, to permit the creation of additional nodes, new create actions must be added which create those specific nodes.

In these experiments, the three packets of rules for S, VP, and PP as presented in Winston were coded for a 3-level PDP network containing 44 input elements, 15 hidden-level elements, and 5 output elements. Using backward propagation (Rumelhart & McClelland, 1987), the network converged after 200,000 training presentations to give perfect performance for the 13 grammar rules coded.

5. Open Problems

Several open problems exist which need to be resolved before satisfactory Natural Language Processing is possible with Neural Networks. The most important of these is relates to the issue of overcoming the "finiteness" constraint on inputs. Methods need to be developed which permit unbounded streams of input to be processed by a neural network. Our approach is to apply the network iteratively in a manner identical to the rules in WASP. The iteration is managed external to the network itself and this makes our solution far from completely satisfying.

Furthermore, the action portion of the rule is left unchanged from WASP, that is, it too is external to the network. Ideally, we would like for the actions (which change the "state"[5] of the parser) to be incorporated into what is learned in the network. This form of coding the rules has disadvantages and some of these are under investigation. Each parsing state determines what rule should fire and once fired a rule produces an action which is then effected on that state to yield a new state. Requiring an action as output introduces a point of fragility just as symbolic systems lack robustness due to the fragility of their individual symbols.[6]

---

[5] The term "state" is used here loosely to mean a configuration or situation that occurs on each iteration of the parser. The state determines which rule(s) can fire and thus incorporates all that is necessary to uniquely determine that.

[6] See Derthick & Plaut (1986) for a good discussion of issues surrounding the physical symbol system hypothesis. Paul Smolensky (1987) further discusses related issues of sub-cognition. Issues pertaining to sub-symbolic representations of language are also discussed at great length in

Finally, no satisfying method of representation yet exists for unbounded structures which correspond to the result of language processing. Perhaps this is a none-problem in that, once fully investigated, language processing may turn out to be best view in its relation to more behaviorally-defined outputs. However, it is easy to show that what is commonly viewed as syntax is undoubtedly a necessary component of language processing and, therefore, should be investigated. We resolve this issue at present by requiring a large, but bounded memory for such structures according to our perception of what performance limits seem to be present in human language perception.

6. Future Work

This work is being pursued on several fronts. Larger grammars, from appendices C and D of Marcus, are being coded and additional testing is planned. Variations in the coding scheme are being evaluated.

Of course, networks of moderate to large size are required, depending on the complexity of the examples. In experimentation conducted already, a very simple grammar having 13 rules in the WASP version requires 44 input units, 15 hidden units, and 5 output units using a backward propagation learning strategy. This small network, thus, has 735 connections to be managed and from experience we know that training is slow, taking around 200,000 iterations to full convergence.

Work has begun on training a network for the grammar from appendix C of Marcus which has 25 rules and significant additional complexity. Although coding is not yet complete, the network being constructed has increased the number of input units to 66 and output units to 10. We expect to conduct trials with a hidden layer containing around 50 units, so the total number of connections should be about 3800. A later experiment with a still larger grammar is planned. The grammar in appendix D of Marcus contains 142 rules and uses 105 uniquely defined symbols. We speculate that a network capable of learning those rules should require on the order of 10,000 units and hundreds of thousands of connections.

In the experiments presented here, training was performed from the rules that were known to work for Marcus and testing was performed on actual sentences. The rules tend to have many "don't care" indicators, signified by the question mark in the appropriate feature slots of Figure 2. Sentences have few if any question marks since they represent an actual set of states in processing with actual words and structures. Two

---

a thesis proposal by Dorffner (1987).

types of training are being pursued: "deductive" training from a grammar; and "inductive" training from sentence examples. This latter training data may simply be derived from traces of processing in a conventional WASP system. Such training is expected to yield results to be compared with those of Berwick.

7. Summary

We have argued that implementation of wait-and-see parsing within a PDP network can provide a WASP system with enormous advantages, both practical and theoretic. To demonstrate feasibility, we have described a prototype effort at constructing such a system.

Although only preliminary studies have been conducted, the evidence is strongly in favor of continuing this approach toward achieving a robust natural language parser.

8. Acknowledgements

The author expresses gratitude to John Merrill for developing VICE, a system for implementing neural networks that learn through back-propagation, and the system in which these experiments were conducted. Thanks also to John for many thoughtful discussions on PDP and this research. Georg Dorffner contributed by changing my basic view of Linguistics and broadening my perspectives in this work. Discussions with Robert Port contributed to the ideas expressed herein in too many ways to mention here. Comments on an earlier draft by T. Dan Kimura are gratefully acknowledged. I, of course, am primarily responsible for errors.

References

Berwick, Robert C. (1982). Locality Principles and the Acquisition of Syntactic Knowledge. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA.

Briscoe, E.J. (1983). Determinism and its implementation in PARSIFAL. In K. Sparck Jones and Y. Wilks (Eds.), *Automatic Natural Language Parsing*, 61-68. (Ellis Horwood: Chichester, England).

Charniak, E. (1983). A Parser with Something for Everyone. In Margaret King (Ed.), *Parsing Natural Language*, 117-149. (Academic Press: New York).

Cottrell, G.W. (1985). Connectionist Parsing. *Proceedings of the 7th Annual Conference of the Cognitive Science Society*, Irvine, CA, 201-211.

Cottrell, G.W. (1985). A Connectionist Approach to Word Sense Disambiguation. Phd Dissertation, available as: *Technical Report 154*, Computer Science Department, University of Rochester.

Derthick, M., and D.C. Plaut (1986). Is Distributed Connectionism Compatible with the Physical Symbol Hypothesis? *Proceedings of the 8th Annual Conference of the Cognitive Science Society*, Amherst, MA, 639-644.

Dorffner, G. (1987). A Completely Distributed Network Model for Computational Linguistics. Unpublished dissertation proposal.

Fanty, M. (1985). Context-Free Parsing in Connectionist Networks. *Technical Report 174*, Computer Science Department, University of Rochester.

Gallant, Stephen I. (1988). Connectionist Expert Systems. *Communications of the ACM*, 31, 2, 152-169.

Granger, Richard H. (1983). The NOMAD System: Expectation-Based Detection and Correction of Errors during Understanding of Syntactically and Semantically Ill-Formed Text. *American Journal of Computational Linguistics*, 9(3-4): 188-196.

Hindle, Donald (1983). Deterministic Parsing of Syntactic Non-Fluencies. *Proceedings of the 21st Annual Meeting of the Association for Computational Linguistics*, 123-128.

Kwasny, S.C., and N.K. Sondheimer (1981). Relaxation Techniques for Parsing Ill-Formed Input. *American Journal of Computational Linguistics*, 7(2): 99-108.

Marcus, M. P. (1980). *A Theory of Syntactic Recognition for Natural Language*. (MIT Press; Cambridge, MA).

Ritchie, G.D. (1983). The implementation of a PIDGIN interpreter. In K. Sparck Jones and Y. Wilks (Eds.), *Automatic Natural Language Parsing*, 69-80. (Ellis Horwood: Chichester, England).

Rumelhart, David E., & James L. McClelland (1986). Parallel Distributed Processing, Vols. 1 & 2, MIT Press: Cambridge, MA.

Selman, B. & G. Hirst (1985). A Rule-Based Connectionist Parsing System. *Proceedings of the 7th Annual Conference of the Cognitive Science Society*, Irvine, CA, 212-221.

Smolensky, Paul (1987). Connectionist AI, Symbolic AI, and the Brain. *Artificial Intelligence Review*, 95-109.

Waltz, D.L. & J.B. Pollack (1985). Massively Parallel Parsing: A Strongly Interactive Model of Natural Language Interpretation. *Cognitive Science 9*, 51-74.

Weischedel, R.M. and N.K. Sondheimer (1983). Meta-Rules as a Basis for Processing Ill-Formed Input. *American Journal of Computational Linguistics*, 9(3-4): 161-177.

Weischedel, R.M. and L. Ramshaw (1987). Reflections on the Knowledge Needed to Process Ill-Formed Language. In S. Nirenburg (Ed.), *Machine Translation: Theoretical and Methodological Issues*, to appear. (Cambridge Univ Press, Cambridge, England).

Winston, P.H. (1984). *Artificial Intelligence*, second edition. Addison-Wesley, Reading, MA).