13

Progress in Natural Language Processing

As mentioned previously, the problems of understanding, generating, and translating material in ordinary human (rather than computer) languages fall under the heading of natural language processing. During the "early explorations" phase of AI research, some good beginnings were made on NLP problems. In the subsequent phase, the late 1960s to early 1970s, new work built on these foundations, as I'll describe in this part of the book.

13.1 Machine Translation

W. John Hutchins, who has written extensively about the history of machine translation (MT), has called the period 1967 to 1976, "the quiet decade."[1] Inactivity in the field during this period is due in part to the ALPAC report, which, as I have already said, was pessimistic about the prospects for machine translation. Hutchins claimed "The influence of the ALPAC report was profound. It brought a virtual end to MT research in the USA for over a decade and MT was for many years perceived as a complete failure. . . . The focus of MT activity switched from the United States to Canada and to Europe."[2]

One exception to this decade-long lull in the United States was the development of the Systran (System Translator) translating program by Petr Toma, a Hungarian-born computer scientist and linguistics researcher who had worked on the Georgetown Russian-to-English translation system. In 1968, Toma set up a company called Latsec, Inc., in La Jolla, California, to continue the Systran development work he had begun earlier in Germany. The U.S. Air Force gave the company a contract to develop a Russian-to-English translation system. It was tested in early 1969 at the Wright-Patterson Air Force Base in Dayton, Ohio, "where it continues to provide Russian–English translations for the USAF's Foreign Technology Division to this day."[3] Systran has evolved to be one of the main automatic translation systems. It is marketed by the Imageforce Corporation in Tampa, Florida.[4]

How well does Systran translate? It all depends on how one wants to measure performance. Margaret Boden mentions two measures, namely, "intelligibility" and "correctness." Both of these measures depend on human judgement. For the first, one asks "Can the translation be generally understood?" For the second, one asks "Do human 'post-editors' need to modify the translation?" Boden states that "in the two-year period from 1976 to 1978, the intelligibility of translations generated by Systran rose from 45 to 78 percent for [raw text input] . . . " She also notes that human translations score only 98 to 99 percent, not 100 percent. Regarding correctness,

https://doi.org/10.1017/CBO9780511819346.016 Published online by Cambridge University Press

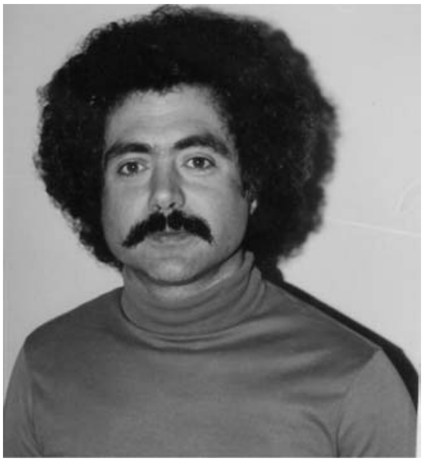
Figure 13.1. Terry Winograd. (Photograph courtesy of Terry Winograd.)

Boden states that in 1978 “only 64 percent of the words were left untouched by human post-editors. Even so, human post-editing of a page of Systran output took only twenty minutes in the mid-1980s, whereas normal (fully human) translation would have taken an hour.”[5]

13.2 Understanding

Although the late 1960s and early 1970s might have been a “quiet decade” for machine translation, it was a very active period for other NLP work. Researchers during these years applied much more powerful syntactic, semantic, and inference abilities to the problem of understanding natural language. Typical of the new attitude was the following observation by Terry Winograd, an MIT Ph.D. student during the late 1960s:[6]

If we really want computers to understand us, we need to give them ability to use more knowledge. In addition to a grammar of the language, they need to have all sorts of knowledge about the subject they are discussing, and they have to use reasoning to combine facts in the right way to understand a sentence and to respond to it. The process of understanding a sentence has to combine grammar, semantics, and reasoning in a very intimate way, calling on each part to help with the others.

13.2.1 *SHRDLU*

Perhaps the NLP achievement that caused the greatest excitement was the SHRDLU natural language dialog system programmed by Terry Winograd (1946– ; Fig. 13.1) for his Ph.D. dissertation (under Seymour Papert) at MIT.[7]

SHRDLU was able to carry on a dialog about what Winograd called a “micro-world,” a world consisting of toy blocks and a “gripper” for moving them about. Unlike the the world of real blocks used in previous MIT and Stanford hand–eye research, Winograd’s blocks world was simulated on a DEC PDP-10 computer and viewed on a DEC 340 black-and-white graphics display such as that shown in

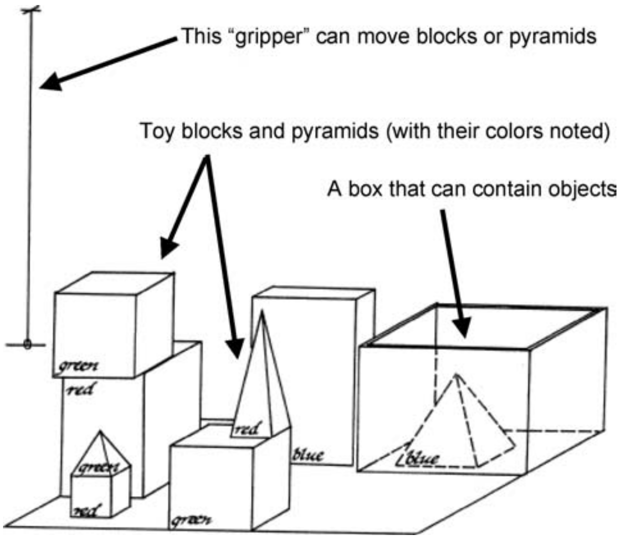
This "gripper" can move blocks or pyramids
Toy blocks and pyramids (with their colors noted)
A box that can contain objects
green
red
green
red
red
green
blue
blue

Figure 13.2. SHRDLU's world. (Illustration used with permission of Terry Winograd.)

Fig. 13.2. SHRDLU could be given commands, typed in ordinary English, to manipulate the blocks using its gripper. It could also be asked questions about the state of its world. It could "understand" those commands and questions, carry out requested manipulations, and ask the user clarifying questions in English about questions or commands it did not understand.

Before describing something about SHRDLU's design, let us look at an example of part of a dialog.[8] In the following, the user's input is numbered and shown in lowercase letters, and SHRDLU's response is in all caps, Winograd's commentary follows in the same typeface, and mine is in parentheses in normal type:

```
1. pick up a big red block.

In order to pick up the red block, it had to clear it off by
finding a space for the green one and moving the green one away.

OK.

The system answers "OK" when it carries out a command.
```

(I won't bother to show what the block configuration looks like as the dialog proceeds, assuming you can use your own mental "model" to keep track of things. The gripper is now holding the red block.)

```
2. grasp the pyramid.

There are three pyramids in the scene, so the program does
nothing.

I DON'T UNDERSTAND WHICH PYRAMID YOU MEAN.

3. find a block which is taller than the one you are holding and
put it into the box.
```

(The user has now apparently changed his mind about pyramid grasping.)

```
The system has to figure out what is meant by “one” and “it,”
as well as using deduction to decide which block it is holding.

BY “IT,” I ASSUME YOU MEAN THE BLOCK WHICH IS TALLER THAN THE
ONE I AM HOLDING.

Printed out to keep us informed.

The hand has to set down the red block to pick up the taller blue
one.

OK.

4. what does the box contain?

THE BLUE PYRAMID AND THE BLUE BLOCK.

5. what is the pyramid supported by?

Earlier the system responded to “the pyramid” by saying
“I don’t understand which pyramid you mean.” now, it knows
that we must mean the one which was just mentioned.

THE BOX
```

This dialog illustrating SHRDLU’s capabilities goes on for forty-two more user inputs and system responses.[9]

To engage in dialogs like this, SHRDLU needed “knowledge.” It needed to know about the syntax of that part of English it used, and it needed to know what various words and word combinations mean in the context of their use. It also needed to know about its blocks world – how blocks can be manipulated and what it means for an object to be inside of the “box.” It needed to keep track of the dialog so that it could decide to which object mentioned previously a word such as “it” referred.

All of this needed knowledge was represented in LISP programs, or “procedures,” as Winograd called them. Knowledge about syntax was represented as a collection of procedures based on the principles of “systemic grammar.”[10]Knowledge about the meanings of words in context was represented in procedures that could refer to a dictionary of word meanings, to other parts of the sentence in which the word was used, and to the discourse. Knowledge about the blocks world was represented in two ways: There was a model that gave the locations of all of the objects and there were procedures that could infer the predicted effects (in the model) of manipulations by the gripper on the various objects. The object-moving procedures had information both about the preconditions and about the effects of these manipulations. These procedures were encoded in a version of Hewitt’s PLANNER language, which, as mentioned previously, bore some resemblance to STRIPS operators. Additional procedures in the PLANNER language were used for other types of inference needed by the system. Logical rules were expressed as programs, which were capable of making both forward and backward deductions.

SHRDLU’s processes for language understanding can be divided into three parts, namely, syntax, semantics, and inference, but doing so is somewhat misleading because the interplay among these parts was a key feature of the system. As Winograd stated, “Since each piece of knowledge can be a procedure, it can call on any other piece of knowledge of any type.” For example, Winograd wrote, “As it finds each piece of the syntactic structure, it checks its semantic interpretation, first to see if it is plausible, then (if possible) to see if it is in accord with the system’s knowledge of the world, both specific and general.”

Winograd’s procedural representation of knowledge (together with Hewitt’s PLANNER language for encoding such representations) can be contrasted with McCarthy’s use of logical formulas to represent knowledge declaratively. The success of SHRDLU fueled a debate among AI researchers about the pros and cons of these two knowledge representation strategies – procedural versus declarative. Actually, the use of LISP to represent procedures blurs this distinction to some extent because, as Winograd pointed out, “LISP allows us to treat programs as data and data as programs.” So, even though SHRDLU’s knowledge was represented procedurally, it was able to incorporate some declarative new knowledge (presented to it as English sentences) into its procedures.

SHRDLU’s performance was indeed quite impressive and made some natural language researchers optimistic about future success.[11] However, Winograd soon abandoned this line of research in favor of pursuing work devoted to the interaction of computers and people. Perhaps because he had first-hand experience of how much knowledge was required for successful language understanding in something so simple as the blocks world, he despaired of ever giving computers enough knowledge to duplicate the full range of human verbal competence. In a 2004 e-mail, Winograd put SHRDLU’s abilities in context with those of humans:[12]

> There are fundamental gulfs between the way that SHRDLU and its kin operate, and whatever it is that goes on in our brains. I don’t think that current research has made much progress in crossing that gulf, and the relevant science may take decades or more to get to the point where the initial ambitions become realistic. In the meantime AI took on much more doable goals of working in less ambitious niches, or accepting less-than-human results (as in translation).

13.2.2 *LUNAR*

On their return from the first manned moon landing, the Apollo 11 astronauts brought back several pounds of moon rocks for scientific study. Various data about these rocks were stored in databases that could be accessed by geologists and other scientists. To make retrieval of this information easier for lunar geologists, NASA asked William A. Woods, a young computer scientist at BBN, about the possibility of designing some sort of natural-language “front end” so that the databases could be queried in English instead of in arcane computer code. Woods had just completed his Ph.D. research at Harvard on question-answering systems.[13]

Sponsored by NASA’s Manned Spacecraft Center, Woods and BBN colleagues Ron Kaplan and Bonnie Webber developed a system they called “LUNAR” for answering questions about the moon rocks.[14] LUNAR used both syntactic and semantic processes to transform English questions into moon rock database queries. Syntactic analysis was performed by using “augmented transition networks” (ATNs), a methodology developed by Woods during his Harvard Ph.D. research. (I’ll describe what ATNs are all about shortly.) The semantic component, guided by the ATN-derived parse trees, transformed English sentences into what Woods called a "meaning representation language" (MRL). This language was a logical language (like that of the predicate calculus) but extended with procedures that could be executed. MRL was originally conceived by Woods at Harvard and further developed at BBN.

LUNAR was able to "understand" and answer a wide variety of questions, including, for example,

"What is the average concentration of aluminum in high alkali rocks?"
"How many breccias contain olivine?"
"What are they?" (LUNAR recognized that "they" referred to the breccias named as answers to the last question.)

LUNAR was the first question-answering system to publish performance data. It was able to answer successfully 78% of the questions put to it by geologists at the Second Annual Lunar Science Conference held in Houston in January 1971. Reportedly, 90% would have been answerable with "minor fixes" to the system.

In a June 2006 talk[15] about LUNAR, Woods mentioned some of its limitations. The following dialog illustrates one shortcoming:

User: What is a breccia?
LUNAR: S10018.
User: What is S10018?
LUNAR: S10018.

Woods said, "LUNAR simply finds referents of referring expressions and gives their names. There is no model of the purpose behind the user's question or of different kinds of answers for different purposes."

Although LUNAR could recognize several different ways of phrasing essentially the same question, Woods claimed that "there are other requests which (due to limitations in the current grammar) must be stated in a specific way in order for the grammar to parse them and there are others which are only understood by the semantic interpreter when they are stated in certain ways."[16]

13.2.3 *Augmented Transition Networks*

Many people realized that context-free grammars (like the ones I discussed earlier) were too weak for most practical natural language processing applications. For example, if we were to expand the illustrative grammar I described in Section 7.1 so that it included (in addition to "threw" and "hit" and "man") the present-tense verbs "throw," "throws," and "hits" and the plural noun "men," then the strings "the men hits the ball" and "the man throw the ball" would be inappropriately accepted as grammatical sentences. To expand a context-free grammar to require that nouns and verbs must agree as to number would involve an impractically large collection of rules. Also, allowing for passive sentences, such as "the ball was hit by the men," would require even further elaboration. Clearly, the sorts of sentences that geologists might ask about moon rocks required more powerful grammars – such as the augmented transition networks that Woods and others had been developing.

In Chomsky's 1957 book[17] he had proposed a hierarchy of grammatical systems of which context-free grammars were just one example. His more powerful grammars had a "transformational component" and were able, for example, to parse a sentence such as "the ball was hit by the man" and give it the same "deep structure" as it would give the sentence "the man hit the ball." Augmented transition network grammars could also perform these kinds of transformations but in a more computationally satisfying way.

An augmented transition network is a maplike graphical structure in which the nodes represent points of progress in the parsing process, and the paths connecting two nodes represent syntactic categories. We can think of parsing a sentence as traversing a path through the network from the start node (no progress at all yet) to an end node (where the sentence has been successfully parsed). Traversing the path builds the syntactic structure of the sentence in the form of a parse tree. Analysis of a sentence involves peeling off the words in left-to-right fashion and using them to indicate which path in the network to take.

Syntactic analysis could begin by peeling off a single word and finding out from a lexicon whether it was a noun, a determiner, an auxiliary (such as "does"), an adjective, or some other "terminal" syntactic category. Or it could begin by peeling off a group of words and checking to see whether this group was a noun phrase, a verb phrase, a prepositional phrase, or what have you. In the first case, depending on the category of the single word, we would take a path corresponding to that category leading out from the start node. To accommodate the second case, there would be possible paths corresponding to a noun phrase and the other possible higher level syntactic categories.

But how would we decide whether or not we could take the noun-phrase path, for example? The answer proposed by Woods and others was that there would be additional transition networks corresponding to these higher level categories. We would be permitted to take the noun-phrase path in the main transition network only if we could successfully traverse the noun-phrase network. And because one path in the noun-phrase network might start with a prepositional phrase, we would have to check to see whether we could take that path (in the noun-phrase network) by successfully traversing a prepositional-phrase network. This process would continue with one network "calling" other networks in a manner similar to the way in which a program can fire up (or "call") other programs, possibly *recursively*. (You will recall my discussion of recursive programs: programs that can call versions of themselves.) For this reason, assemblages of networks like these are called recursive transition networks.

The first networks of this kind were developed at the University of Edinburgh in Scotland by James Thorne, Paul Bratley, and Hamish Dewar.[18] Later, Dan Bobrow and Bruce Fraser proposed a transition network system that elaborated on the Scottish one.[19] Both of these systems also performed auxiliary computations while traversing their networks. These "augmentations" allowed the construction of a "deep structure" representation of the sentence being analyzed. Woods's work

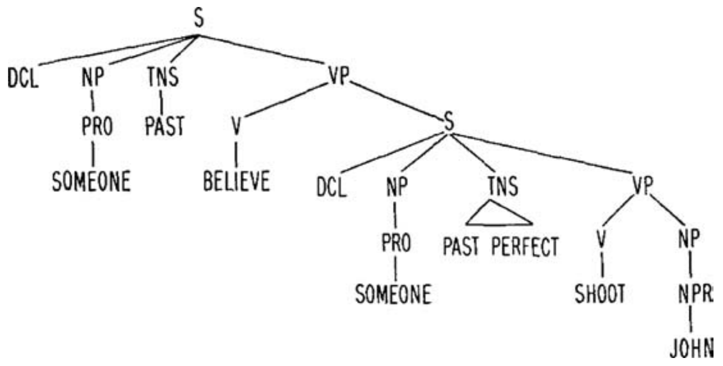

Figure 13.3. A parse tree obtained for the sentence "John was believed to have been shot." (From William A. Woods, "Transition Network Grammars for Natural Language Analysis," *Communications of the ACM*, Vol. 13, No. 10, pp. 591–606, 1970.)

on "augmented, recursive transition networks" built on and refined these ideas and introduced an elegant network definition language.[20]

As an example, Woods described how one of his networks analyzed the sentence "John was believed to have been shot."[21] After all of the calls to subsidiary networks and all of the auxiliary computations were performed the parse tree shown in Fig. 13.3 was obtained. We can observe two things about this parse tree. First, note the occurrence of "PAST" and "PAST PERFECT" as tense markers. Second, note that the form of the original passive-voice sentence has been transformed to an active-voice sentence using a presumed pronoun "SOMEONE." As Woods notes, the structure can be paraphrased as "Someone believed that someone had shot John." Network grammars get at the "deep structure" of sentences by transforming them into a standard form.

13.2.4 *GUS*

Dan Bobrow and Ronald Kaplan, two of the researchers doing NLP work at BBN, moved to the newly formed Palo Alto Research Center (PARC) of Xerox in the mid-1970s. One of the projects that they and other researchers worked on there was GUS (an acronym for Genial Understander System), which was "intended to engage a sympathetic and highly cooperative human in an English dialog, directed towards a specific goal within a very restricted domain of discourse."[22] The restricted domain was that of a travel agent engaged in "a conversation with a client who wants to make a simple return trip to a single city in California." The following transcript of an actual dialog shows off some of its abilities:

GUS: Hello. My name is GUS. I can help you plan a simple trip by air. Where do you want to go?
(1) *Client*: I want to go to San Diego on May 28.
GUS: What time do you want to leave?

(2) *Client*: I must be in San Diego before 10 am.
GUS: Would you like P.S.A. flight 102 that arrives at 9.15 am?
(3) *Client*: What is the next flight?
GUS: Air California flight 310 that leaves at 8.30 am and arrives at 10.00 am.
(4) *Client*: I'll take the first one.
GUS: In what name should I make a reservation, please?
(5) *Client*: Dan Bobrow.
GUS: I have confirmed the following flight: P.SA. flight 102 on Wednesday May 28 from San Jose to San Diego that leaves at 7.30 am and arrives at 9.15 am. What date do you want to return on?
(6) *Client*: On Friday in the evening.
GUS: Would you like the flight that leaves at 7.45 pm?
(7) *Client*: That's fine.
GUS: I have confirmed the following flight: P.S.A. flight 307 on Friday May 30 from San Diego to San Jose that leaves at 7.45 pm and arrives at 9.30 pm. Thank you for calling. Goodbye.

Although the abilities of the system were certainly limited, GUS was able to deal with a number of problems. One of these involves what NLP researchers call "resolving anaphora," by which they mean deciding on the objects or events to which various words or phrases in a dialog refer. Several examples, keyed to the numbered sentences in the dialog above, are mentioned in the paper about GUS:

At line (3), for example, the client's query refers to the flight mentioned in GUS's immediately preceding utterance. In (4) there is a reference to the flight mentioned earlier in the conversation, [following line (2)]. Note that "next flight" in (3) was to be interpreted relative to the order of flights in the airline guide whereas "first one" in (4) refers to the order in which the flights were mentioned. Another implicit referent underlies the use of "Friday" to specify a date in (6). Resolution of this reference requires some complicated reasoning involving both the content and the context of the conversation. Since May 28 has been given as the departure date, it must presumably be the following Friday that the client has in mind. On the other hand, suppose that the specifications were reversed and Friday had been given as the departure date at line (1). It would then be most readily interpretable as referring to the Friday immediately following the conversation.

GUS was a combination of several communicating subsystems, a morphological analyzer for dealing with word components, a syntactic analyzer for generating parse trees, a "reasoner" for figuring out a user's meanings and intentions, and a language generator for responding. Controlling these components was done by using an "agenda" mechanism. As the authors explain,

GUS operates in a cycle in which it examines this agenda, chooses the next job to be done, and does it. In general, the execution of the selected task causes entries for new tasks to be created and placed on the agenda. Output text generation can be prompted by reasoning processes at any time, and inputs from the client are handled whenever they come in. There are places at which information from a later stage (such as one involving semantics) are fed back to an earlier stage (such as the parser). A supervisory process can reorder the agenda at any time.

The syntactic component of GUS had "access to a main dictionary of more than 3,000 stems and simple idioms." The syntactic analyzer was based on a system developed earlier by Ronald Kaplan, which used a transition-network grammar and was called a "General Syntactic Processor."[23] Client sentences were encoded in "frames" (which are related to Minsky's frames but closer in form to semantic networks). Some frames described the sequence of a normal dialog, whereas others represented the attributes of a date, a trip plan, or a traveler. GUS's reasoning component used the content and structure of the frames to deduce how best to interpret client sentences.

Besides anaphora, the paper mentioned several other problems that GUS was able to deal with. However, it also cautioned that "it is much too easy to extrapolate from [the sample dialog] a mistaken notion that GUS contained solutions to far more problems than it did." Sample dialogs recorded between human clients and humans playing the role of a GUS revealed numerous instances in which the computer GUS would fail. The authors concluded that if users of systems like GUS departed "from the behavior expected of them in the minutest detail, or if apparently insignificant adjustments are made in their structure," the systems would act as if they had "gross aphasia" or had just simply died. The authors conceded that "GUS itself is not very intelligent, but it does illustrate what we believe to be essential components of [an intelligent language understanding] system. . . . [It] must have a high quality parser, a reasoning component, and a well structured data base of knowledge." Subsequent work on NLP at PARC and many other places sought to improve all of these components.

The systems developed by researchers such as Winograd, Woods, Bobrow, and their colleagues were very impressive steps toward conversing with computers in English. Yet, there was still a long way to go before natural language understanding systems could perform in a way envisioned by Winograd in the preface to his Ph.D. dissertation:

> Let us envision a new way of using computers so they can take instructions in a way suited to their jobs. We will talk to them just as we talk to a research assistant, librarian, or secretary, and they will carry out our commands and provide us with the information we ask for. If our instructions aren't clear enough, they will ask for more information before they do what we want, and this dialog will all be in English.

Notes

1. W. John Hutchins, "Machine Translation: A Brief History," in E. F. K. Koerner and R. E. Asher (eds.), *Concise History of the Language Sciences: From the Sumerians to the Cognitivists*, pp. 431–445, Oxford: Pergamon Press, 1995. (Also available online at http://www.hutchinsweb.me.uk/ConcHistoryLangSci-1995.pdf.) [181]
2. *Ibid.* [181]
3. W. John Hutchins, *Machine Translation: Past, Present, Future*, Chichester: Ellis Horwood, 1986. An updated (2003) version is available online at http://www.hutchinsweb.me.uk/PPF-TOC.htm. Some of the technical details of Systran's operation are described in the book. [181]
4. http://www.translationsoftware4u.com/. [181]

5. Margaret A. Boden, *Mind as Machine: A History of Cognitive Science*, p. 683, Oxford: Oxford University Press, 2006. [182]
6. This quote is from the preface of Winograd's Ph.D. dissertation. [182]
7. SHRDLU is described in Winograd's dissertation "Procedures as a Representation for Data in a Computer Program for Understanding Natural Language." It was issued as an MIT AI Technical Report No. 235, February 1971, and is available online at https://dspace.mit.edu/bitstream/1721.1/7095/2/AITR-235.pdf. The thesis was also published as a full issue of *Cognitive Psychology*, Vol. 3, No. 1, 1972, and as a book *Understanding Natural Language*, New York: Academic Press, 1972. The letters in SHRDLU comprise the second column of keys in linotype machines, which were used to set type before computers were used for that. This nonsense word was often used in *MAD* magazine, which Winograd read in his youth. Failing to think of an acceptable acronym to use to name his system, Winograd used SHRDLU. For Winograd's account, see http://hci.stanford.edu/~winograd/shrdlu/name.html. [182]
8. Taken from Section 1.3 of Winograd's thesis. [183]
9. Readers interested in the entire dialog can see it either in Winograd's thesis or on one of his Web sites at http://hci.stanford.edu/~winograd/shrdlu/. [184]
10. Winograd cites, among others, M. A. K. Halliday, "Categories of the Theory of Grammar," *Word*, Vol. 17, No. 3, pp. 241–292, 1961. [184]
11. For a short film of SHRDLU in action, see http://projects.csail.mit.edu/films/aifilms/digitalFilms/3mpeg/26-robot.mpg. [185]
12. From http://www.semaphorecorp.com/misc/shrdlu.html. [185]
13. William A. Woods, "Semantics for a Question-Answering System," Ph.D. dissertation, Harvard University, August 1967. Reprinted as a volume in the series *Outstanding Dissertations in the Computer Sciences*, New York: Garland Publishing, 1979. [185]
14. William A. Woods, Ron M. Kaplan, and Bonnie Nash-Webber, "The Lunar Sciences Natural Language Information System: Final Report," BBN, Cambridge, MA, June 1, 1972. See also William A. Woods, "Progress in Natural Language Understanding – An Application to Lunar Geology," *AFIPS Conference Proceedings*, Vol. 42, pp. 441–450, Montvale, New Jersey: AFIPS Press, 1973. [185]
15. See http://www.ils.albany.edu/IQA06/Files/Bill_Woods_IQA06.pdf. [186]
16. William A. Woods, "Progress in Natural Language Understanding – An Application to Lunar Geology," *AFIPS Conference Proceedings*, Vol. 42, pp. 441–450, Montvale, New Jersey: AFIPS Press, 1973. [186]
17. Noam Chomsky, *Syntactic Structures,* 's-Gravenhage: Mouton & Co., 1957. [187]
18. James Thorne, Paul Bratley, and Hamish Dewar, "The Syntactic Analysis of English by Machine," in D. Michie (ed.), *Machine Intelligence 3*, pp. 281–309, New York: American Elsevier Publishing Co., 1968; Hamish Dewar, Paul Bratley, and James Thorne, "A Program for the Syntactic Analysis of English Sentences," *Communications of the ACM*, Vol. 12, No. 8, pp. 476–479, August 1969. [187]
19. Daniel Bobrow and Bruce Fraser, "An Augmented State Transition Network Analysis Procedure," *Proceedings of the International Joint Conferenece on Artificial Intelligence*, pp. 557–567, Washington, DC, 1969. [187]
20. William A. Woods, "Augmented Transition Networks for Natural Language Analysis," Report CS-1, Aiken Computation Laboratory, Harvard University, Cambridge, MA, December 1969; William A. Woods, "Transition Network Grammars for Natural Language Analysis," *Communications of the ACM*, Vol. 13, No. 10, pp. 591–606, 1970. The ACM article has been reprinted in Yoh-Han Pao and George W. Ernest (eds.), *Tutorial: Context-Directed Pattern Recognition and Machine Intelligence Techniques for Information Processing*, Silver Spring, MD: IEEE Computer Society Press, 1982, and in Barbara

Grosz, Karen Sparck Jones, and Bonnie Webber (eds.), *Readings in Natural Language Processing*, San Mateo, CA: Morgan Kaufmann, 1986. [188]

21. William A. Woods, *op. cit.*, pp. 602ff. [188]
22. Daniel G. Bobrow *et al.*, “GUS, A Frame-Driven Dialog System,” *Artificial Intelligence*, Vol. 8, pp. 155–173, 1977. [188]
23. Ronald Kaplan, “A General Syntactic Processor,” in R. Rustin (ed.), *Natural Language Processing*, New York: Algorithmics Press, 1973. [190]