Using Memory in Text Understanding

Michael Lebowitz

May, 1984

This paper appeared in the proceeding of ECAI-84. The research was supported in part by the Defense Advanced Research Projects Agency under contract N00039-82-C-0427

Using Memory in Text Understanding[1]

Michael Lebowitz
Department of Computer Science
Computer Science Building, Columbia University
New York, NY 10027 USA

This is a *long* paper to be considered in the area of *cognitive modelling*.

Abstract

Text processing undoubtedly takes place at many levels simultaneously. In this paper, we discuss how the access of detailed long-term memory can be used in low-level text processing in the context of a computer system, RESEARCHER, that reads, generalizes, and remembers information form patent abstracts. We show specific points where memory can be applied during text processing, rather than just suggesting general principles. In particular, we focus on how linguistically ambiguous structures can be resolved using memory (and only using memory). A computer example of RESEARCHER applying our memory application principles (with a simulated detailed memory) is presented.

[1]This research was supported in part by the Defense Advanced Research Projects Agency under contract N00039-82-C-0427.

Using Memory in Text Understanding[1]

Michael Lebowitz
Department of Computer Science
Computer Science Building, Columbia University
New York, NY 10027 USA

1 Introduction

It is clear that text processing proceeds at many levels simultaneously [Marslen-Wilson 75; Schank, et al. 80; Charniak 83]. Such processing presumably includes the access of detailed, long-term memory for the purpose of finding information relevant to a new text. It seems plausible that such high-level information should be useful in assisting low-level processing. We have suggested in our earlier work, as have others, that memory access might help determine resource allocation and in identifying the important parts of a text [Schank, et al. 80; Lebowitz 81]. However, these are rather imprecise ideas, and difficult to apply. For example, IPP [Lebowitz 80; Lebowitz 83a], a program that read, remembered and generalized from news stories about international terrorism, might know from accessing memory that the destination of a hijacking in a story that began, "A United 727 en route to Miami was commandeered ..." is likely to be Cuba. However, it was not clear how to use this information should the story continue, "to Havana." Due to the inherent redundancy of language, it is quite easy to process this information bottom up.[2]

It is our feeling that the best way to use specific information in memory for text understanding in the context of current systems is to identify specific points during processing where memory can be applied. In this paper, we will present suggestions for how memory can be used for low-level text processing in the context of RESEARCHER, a program that reads patent abstracts and builds up a generalized long-term memory [Lebowitz 83b]. We will illustrate this process with a version of RESEARCHER that simulates memory access by asking a user questions at key points during the processing of an abstract. Crucial to the development of RESEARCHER is that its text understanding process be robust enough to handle many patents without special preparation. We feel that memory application is necessary to achieve this ability.

[1]This research was supported in part by the Defense Advanced Research Projects Agency under contract N00039-82-C-0427.

[2]Of course, had no destination been mentioned, we would have used Cuba as a default.

Notice that we are talking about using *memory* for understanding, not just general semantic information about words or concepts. While such general information is crucial for our conceptually-based understanding methods, in order to resolve many understanding questions it will be necessary to look at very detailed information in memory -- in our case, how the various parts of the objects described in patent abstract are constructed and how their pieces relate to each other.

To illustrate why memory is needed, consider the following simple example:

EX1 - A read/write head touching a disc made of XXX

In EX1, there is no way to determine whether XXX is the material used for the read/write head or the disc without knowing something about the objects involved. Indeed, depending on the context (the type of object being described), we might come up with different analyses.

The main area in which we feel memory will be immediately useful in understanding (other than supplying default values) is in resolving linguistic ambiguities of the kind illustrated by EX1. While it is probably possible to use memory to resolve lexical ambiguity, we will be looking one structural level higher, as word ambiguity has not proven to be a major problem for RESEARCHER.

The idea of applying memory (or at least semantic information) to resolve ambiguity is not a new one (e.g., [Riesbeck and Schank 76, Small 80]). However, what we will do here is show exactly how detailed memory can be applied to resolve ambiguity, rather than just stating a general principle. We will begin by outlining the understanding techniques used in RESEARCHER, indicating where memory application might be useful, and then look at specific memory application techniques.

2 Basic RESEARCHER understanding techniques

In this section, we will show how RESEARCHER processes patent abstracts by using only very simple syntactic rules to identify "pieces" of the ultimate representation and then "putting the pieces together". EX2 shows a patent abstract typical of the sort read by RESEARCHER. We are concerned primarily with abstracts that describe the physical structures of objects. The goal of the text interpretation phase of RESEARCHER is to build up descriptions of objects, including the physical relations between various sub-parts of the objects, using a canonical, frame-based representation scheme [Wasserman and Lebowitz 83].

EX2 - P41; U.S. Patent Abstract #4323939

> A hard fixed head disc drive assembly having a rotating record disc with a transducer cooperating with the surface of the disc. The transducer is mounted on a carriage which has three spaced, grooved bearings, two of which are received by a fixed cylindrical track, the third bearing engages a spring-loaded cylindrical track which urges said first two bearings against said fixed track, whereby the carriage is centered on said tracks for movement therealong radially of said disc surface.

There are several important points to notice about EX2 for text processing purposes. First of all, in traditional terms, the syntax of the abstract is very strange; e.g., the first "sentence" has no main verb. Furthermore, very different syntactic structures can function quite similarly. For example, the phrases "a transducer cooperating with the surface of the disk" and "the third bearing engages a spring-loaded cylindrical track" describe very similar physical relations, but use different linguistic structures. While preliminary identification of the syntactic structure might aid in the building of a conceptual representation, patent abstracts seem like an ideal domain to test strongly semantic-based methods that build conceptual representations directly from text.

EX3 shows EX2 segmented in a manner that motivates RESEARCHER's text processing techniques. We see that this text, and most other patent abstracts that provide physical descriptions, can be broken into segments of two types -- those that describe physical objects, which we refer to as *memettes*, shown in italics in EX3, and those that relate various memettes to each other. The memette-describing segments are usually (though not always) simple noun phrases, but the relational segments are of many different forms, including verbs and prepositions. The key point is the functionality of the relational segments is largely independent of their syntactic form, so we can process them solely on the function they serve.

EX3 - (*A hard fixed head disc drive assembly*) (having) (*a rotating record disc*) (with) (*a transducer*) (cooperating with) (*the surface*) (of) (*the disc*). (*The transducer*) (is mounted on) (*a carriage*) (which has) (*three spaced, grooved bearings*), (*two*) (of which) (are received by) (*a fixed cylindrical track*), (*the third bearing*) (engages) (*a spring-loaded cylindrical track*) (which urges) (*said first two bearings*) (against) (*said fixed track*), (whereby) (*the carriage*) (is centered on) (*said tracks*) (for movement therealong radially of) (*said disc surface*).

The analysis shown in EX3 leads directly to RESEARCHER's text interpretation methods. These methods are based on the memory-based understanding techniques designed for IPP [Lebowitz 83a]. The RESEARCHER interpretation phase consists largely of two sub-phases -- memette identification and memette relation, or "identifying the pieces" and "putting the pieces together". Processing involves a top-down goal of recognizing conceptual structures integrated with simple, bottom-up syntactic techniques.

Since patents are not focused on events, as are the news stories IPP processed, the action-based methods of IPP (or other conceptual understanding systems, e.g., [Wilks 73; Riesbeck and Schank 76; Birnbaum and Selfridge 81]) must be modified in a manner consistent with the analysis shown in EX3. RESEARCHER does careful processing of noun phrases to identify memettes, modifications to memettes, and reference to previous mentions of memettes. This processing is integrated with the application of relational words to create relations among memettes.

In broad terms, the structure of our processing is similar to the cascaded ATN methodology [Woods 80; Bobrow and Webber 80], where syntactic grammars frequently hand off syntactic components to a semantic analyzer that builds semantic structures and eliminates impossible constructs. However, we use only a small number of different syntactic constructs, eliminating the need for a formal syntactic grammar by focusing on the roles of words in the conceptual representation. Furthermore, while the cascaded ATN methodology views the understanding process as a syntactic processor giving what it finds to the semantic analyzer, we look on the process as being primarily a conceptual analysis that requests linguistic information when needed (much as in [DeJong 79]).

The noun phrase recognition process involves the same "save and skip" strategy described in [Lebowitz 83a]. Using a one-word look-ahead process, RESEARCHER saves noun phrase words in a stack until the head noun is found. Then the words in the stack are popped off and used to modify the memette indicated by the head noun. Analyzing how the parts of noun groups interrelate is one place where information from memory will be needed. For example, in the first noun phrase of EX2, "A hard fixed head disc drive assembly", there is no way of knowing whether "hard" modifies "head", "disc", "disc drive" or "assembly" without using information about the structure of disc drives. We will discuss this further in the next section.

The final aspect to "finding the pieces" involves checking for previous reference in the text. Here we take advantage of some of the arcane nature of patent abstracts. A very strict formalism is used to identify previous references, involving the word "said" and repetition of identifying modifiers. Without such formalism, the process would be very complicated, as abstracts frequently refer to many very similar objects. As it is, we can use a fairly simple, procedural reference process

The second major sub-phase to RESEARCHER text processing involves putting together the pieces identified. This process occurs as soon as the objects involved are found. By and large, there are two different kinds of relations found that tie objects together -- assembly/component relations and physical (or functional) relations between memettes. The basic RESEARCHER strategy for each is the same -- maintain information from the relational segments of the text in short term memory and then, when the following memette is identified, determine how the appropriate pieces relate to each other. This process, which is largely independent of the form of the relational text segments, immediately builds up a conceptual representation for later use.

The process of relating memettes to each other involves a number of ambiguous situations that we will discuss in the next section. We initially used a set of focus heuristics including some related to [Grosz 77; Sidner 79] and others based on the various relations involved. However, we believe that this is only part of the solution (perhaps a small part), and must be extensively augmented with the kind of memory access described in this paper.

The point here is not that it is impossible to develop heuristics to handle any *specific* problem. We have done so for a number of cases in RESEARCHER. However, these rules get increasingly clumsy as we add them. An approach that accesses long-term memory will give us a more general method.

3 Using memory to resolve ambiguity

In this section we will show specifically how memory can be used to resolve certain classes of ambiguity. We will describe this process in terms of the understanding process "asking questions" of memory. The idea then becomes to specify exactly what questions should be asked and when. The questions will ask which of several possible physical structures is more plausible. Our discussion will be divided the same way as the processing -- identifying "pieces" (processing noun groups), and connecting the pieces. We are not entirely concerned here with the details of the cases we have identified, but also with illustrating the level at which we believe memory should be applied to parsing.

Noun phrases in English, phrases that describe objects, have very complex structure (see [Gershman 77] for a conceptual analysis approach to noun phrase processing). In our earlier work, we have concentrated on properly delineating such phrases. However, RESEARCHER's emphasis on object descriptions requires more attention to the internal structure of noun phrases, so that we can identify how the pieces of such phrases fit together. Memory application is crucial in doing this. EX4 shows the first case requiring memory.

EX4

Form: modifier object-word1 object-word2

Example: A metal drive cover ...

Question: Does the modifier (metal) apply better to object-word1 (drive) or object-word2 (cover)?

Noun-noun constructions in English introduce a multitude of problems. EX4 illustrates one such problem -- a modifier preceding such a combination can modify either of the objects mentioned. In EX4, either the drive or the cover could be made of metal. In more complex situations, i.e., more nouns or a series of modifiers, syntax can reduce the possible targets of a modifier, but only by asking memory which is more plausible, in the context of the object being described, can the right choice be made.

Notice carefully that memory, not just general semantic properties, are needed here. While for most disc drives it is more likely that the cover, rather than the disc, is metal, the contrary could be the case in another device. We might even discover the abstract refers to a special class of disc drives with metal discs and plastic covers

EX5 illustrates a somewhat similar noun group problem.

EX5

Form: object-word1 object-word2 object-word3

Example: A disc-drive transducer wire ...

Question: Is object-word3 (wire) "related to" (one being a part of the other) object-word1 (disc-drive) or object-word2 (transducer)?

When multiple nouns appear in succession, it is not always easy to tell how they group together. In EX5, there us no way to tell whether the wire is a (direct) part of the disc drive or the transducer (or, more precisely, which it is functionally connected to). Again, an appeal to long-term memory of similar devices is the way to resolve this problem.

The "putting together the pieces" phase of RESEARCHER processing also involves ambiguities that must be resolved with memory. EX6 shows what happens when more than one relational word must be processed.

EX6

Form: object-word1 relation-word1 object-word2 relation-word2 object-word3

Example: A transducer on top of a disc supported by a rod ...

Question: Does relation-word2 (supported by) connect object-word3 (rod) with object-word1 (transducer) or object-word2 (disc)?

In some sense, the problem shown in EX6 involves noun group problems, in that it deals with prepositional phrase attachment. However, in our scheme of understanding, it falls into a somewhat different category. As described in the previous section, relating various objects together is a separate part of processing. This allows us to handle many other structural manifestations of this problem with the same mechanism. The mechanism here is to query memory about which of the possible objects most appropriately takes part in the relation. In EX6, we need to appeal to memory to determine whether the rod is more likely supporting the transducer or the disc, either of which is syntactically appropriate.

EX7 illustrate a similar problem, this time with "part indicators" (words that introduce a list of a part's subparts).

EX7

Form: object-word1 part-indicator1 object-word2 part-indicator2 object-word3

Example: A disc drive including a disc comprising a metal plate (and)

Question: Is object-word3 (metal plate) a part of object-word1 (disc drive) or object-word2 (disc)?

In patent abstracts there are frequently descriptions of parts and then of the subparts. This often creates considerable ambiguity of the sort shown in EX7, which is structurally similar to EX6. We handle this case separately as the part/component relations are represented differently from other relations, since they are so crucial, and are expressed slightly differently in the abstracts. Once again, the only way to determine the correct analysis, in this case whether the metal plate is part of the disc drive or the disc, is to query memory, quite possibly looking at descriptions of specific objects or classes of objects.

The final class of structural ambiguity we deal with involves a combination of physical relations and part/component indicators, as shown in EX8.

EX8

Form: object-word1 relation-word object-word2 part-of-indicator object-word3
(There are several related configurations.)

Example: A disc on a spindle for a disc drive ...

Question: Is object-word1 (disc) or object-word2 (spindle) a part of object-word3 (disc drive)? (Directly a part, as both are parts indirectly.)

EX8 is used as an illustration of the many ways part and relational indicators can interact. As always, the simple syntactic processing outlined in Section 2 limits the possibilities, but memory is required to evaluate the different choices, in this case whether the disc drive has as a part the spindle or the disc (or which is a direct functional part).

3.1 Integrating memory access with text processing

The disambiguation questions described in the previous section fit in easily with the overall RESEARCHER text processing algorithm. RESEARCHER's "save and skip" noun group strategy naturally accommodates memory-based disambiguation. As RESEARCHER is processing the items it has maintained in its short-term memory stack, it keeps track of the objects identified by the head noun and the most recent noun. Then, if there is more than one distinct object described in the noun group, as it continues to work back through the stack, memory can be queried to determine which object new words modify or relate to. Without performing the memory query, it is necessary to employ a set of complex and rather unsatisfying heuristics to determine how the parts of noun groups fit together

Memory access fits in equally well in the "putting together the pieces" phase of RESEARCHER understanding. RESEARCHER maintains short-term memory buffers with the most recent object described and the last "base object" (usually identified by the head noun of a noun phrase being modified by a series of prepositional phrases, although there are other possibilities). Then, if these two objects are different, and a new relation (physical or part/component) is being established, memory can be queried for the more plausible of the two objects with which to relate the new object. As with noun group processing, it is possible to develop heuristics that handle most cases, but they are complex and do not seem to be the right way to go for robust understanding.

3.2 Can memory answer the questions?

We have shown here how certain ambiguities in language can be resolved with the answers to specified questions. Obviously, we must believe that our system can automatically answer these questions. The questions we are concerned with take the form of asking which of two memettes can more reasonably be modified in a certain way or relate to another memette, in the context of the device being described. Phrased another way, the needed process is to determine which of two partial descriptions of a device is more plausible.

Indeed, we do believe that these questions will be answerable by RESEARCHER. At the moment, since the memory and search mechanisms are still under development, we have tested the disambiguation process by allowing a user to answer the key questions. However, since the process of finding an object in memory that fits a specified partial description is crucial to generalization and question answering, the disambiguation process will need no information not needed for other purposes.

Our basic approach to memory in RESEARCHER is to store objects in terms of prototypes automatically generalized from earlier patent abstracts [Lebowitz 83b, Lebowitz 83c; Lebowitz 83d]. This approach was successful in answering similar sorts of questions for IPP, and other Generalization-Based Memory systems. It allows us to take partial descriptions of objects and determine whether they are represented by objects or generalized objects in memory. We do not plan, at least at first, to answer questions of this sort that require more complex inferencing from information in memory, instead concentrating on having as complete as possible a set of examples in memory.

3.3 An example

We will complete this discussion of RESEARCHER's use of memory in text understanding by showing how the program processes the first sentence of the patent abstract we looked at in Section 2, using a version of the program that queries a user at the points where memory access is needed. Figure 1 shows how the first noun group is processed. Queries to memory are preceded by ">>>" and each response follows an asterisk.

```
Patent: P41

(A HARD FIXED HEAD DISC DRIVE ASSEMBLY HAVING A ROTATING RECORD DISC WITH A
TRANSDUCER COOPERATING WITH THE SURFACE OF THE DISC *PERIOD* THE TRANSDUCER
IS MOUNTED ON A CARRIAGE WHICH HAS THREE SPACED *COMMA* GROOVED BEARINGS
*COMMA* TWO OF WHICH ARE RECEIVED BY A FIXED CYLINDRICAL TRACK *COMMA* THE
THIRD BEARING ENGAGES A SPRING-LOADED CYLINDRICAL TRACK WHICH URGES SAID
FIRST TWO BEARINGS AGAINST SAID FIXED TRACK *COMMA* WHEREBY THE CARRIAGE IS
CENTERED ON SAID TRACKS FOR MOVEMENT THEREALONG RADIALLY OF SAID DISC
SURFACE *STOP*)

Processing:

A             : New instance word -- skip
HARD          : Memette modifier; save and skip
FIXED         : Memette modifier; save and skip
HEAD          : Memette within NP; save and skip
DISC DRIVE    : Phrase
-> DISC-DRIVE : Memette within NP; save and skip
ASSEMBLY      : Memette word -- memette UNKNOWN-ASSEMBLY#
New UNKNOWN-ASSEMBLY# instance (&MEM0)
New DISC-DRIVE# instance (&MEM1)
Assuming &MEM1 (DISC-DRIVE#) is part of &MEM0 (UNKNOWN-ASSEMBLY# -- 'ASSEMBLY')
New HEAD# instance (&MEM2)

>>> Select memette to merge with &MEM2 (HEAD#) from &MEM0 ('ASSEMBLY')
&MEM1 (DISC-DRIVE#) * &mem1

Assuming &MEM2 (HEAD#) is part of &MEM1 (DISC-DRIVE#)

>>> Select memette modified by MOBILITY/NONE from &MEM0 ('ASSEMBLY')
&MEM2 (HEAD#) * &mem2

Augmenting &MEM2 (HEAD#) with feature: MOBILITY = NONE

>>> Select memette modified by TEXTURE/HARD from &MEM0 ('ASSEMBLY')
&MEM2 (HEAD#) *&mem2

Augmenting &MEM2 (HEAD#) with feature: TEXTURE = HARD
```

Figure 1: RESEARCHER processing the first noun group of P41

In Figure 1 we can see how RESEARCHER first skips and save the words "hard fixed head disc drive" (treating the phrase "disc drive" as a single word) until the head noun, "assembly" is reached. It then works back through the noun group, establishing "disc drive" as part of the assembly. Then, when it processes "head", RESEARCHER must decide whether the head relates directly to the assembly or the disc drive, by making a memory check. Either object is plausible here, but we will assume the assembly is the correct parent part of the head. Then, the modifiers "fixed" and "hard" are processed. In each case, RESEARCHER must decide whether the assembly or the head is being modified. (The disc drive is eliminated by linguistic considerations.) Again, these questions can only be answered from memory, and we presumably conclude that the head is fixed (mobility = none, in our representation scheme) and hard.

Figure 2 shows the rest of the first sentence of the abstract being processed, illustrating additional disambiguation points.

```
HAVING          : Parts of &MEM0 (UNKNOWN-ASSEMBLY# -- 'ASSEMBLY') to follow
A               : New instance word -- skip
ROTATING        : Memette modifier; save and skip
RECORD          : Memette modifier; save and skip
DISC            : Memette word -- memette DISC#
New DISC# instance (&MEM3)
Augmenting &MEM3 (DISC#) with feature: DEV-PURPOSE = STORING
Augmenting &MEM3 (DISC#) with feature: DEV-PURPOSE = ROTATION
Assuming &MEM3 (DISC#) is part of &MEM0 (UNKNOWN-ASSEMBLY# -- 'ASSEMBLY')
WITH (WITH1)    : Parts of &MEM3 (DISC#) to follow
A               : New instance word -- skip
TRANSDUCER      : Memette word -- memette TRANSDUCER#
New TRANSDUCER# instance (&MEM4)

>>> Select assembly of &MEM4 (TRANSDUCER#) from &MEM3 (DISC#)
&MEM0 ('ASSEMBLY') * &mem0

Assuming &MEM4 (TRANSDUCER#) is part of &MEM0 (UNKNOWN-ASSEMBLY# -- 'ASSEMBLY')
COOPERATING WITH
                : Phrase
-> COOPERATING: Relation word -- save and skip
THE             : Antecedent word -- skip
SURFACE         : Memette word -- memette SURFACE#
New SURFACE# instance (&MEM5)

>>> Refine roles for &REL5 [R-ADJACENT-TO
SUBJECT &MEM4 (TRANSDUCER#) &MEM0 (UNKNOWN-ASSEMBLY# -- 'ASSEMBLY')
OBJECT &MEM5 (SURFACE#)
Enter memette for SUBJECT slot *] &mem4

Establishing R-ADJACENT-TO relation; SUBJECT: &MEM4 (TRANSDUCER#);
OBJECT: &MEM5 (SURFACE#) [&REL5]
OF              : Part of indicator
Assuming &MEM5 (SURFACE#) is part of the following
THE             : Antecedent word -- skip
DISC            : Memette word -- memette DISC#
Reference for DISC#: &MEM3

>>> Select component of &MEM3 (DISC#) from &MEM5 (SURFACE#)
&MEM0 ('ASSEMBLY') * &mem5

Assuming &MEM5 (SURFACE#) is part of &MEM3 (DISC#)
*PERIOD*        : Break word -- skip  end of sentence -- resetting part flag
```

Figure 2: RESEARCHER processing the rest of P41

RESEARCHER must determine whether the "transducer" is part of the "record disc" or the assembly and whether the "surface of the disc" is "cooperating with" (adjacent to) the assembly or the transducer. Each of these cases is ambiguous structurally, and only be disambiguated with memory access.

Figure 3 shows the final representation constructed by RESEARCHER after reading all of P41. It consists of a set of identified memettes, indications of which memettes are parts of others, and a list of relations between memettes. The relations prefixed with R- are physical and those beginning with P- are functional (purposive). There is also a single "meta-relation" that indicates a causal connection between its component relations. This representation captures all the informati~~on~~ from P41 that is needed for the learning aspects of RESEARCHER. It was acquired using the memory-augmented "putting pieces together" strategy described in this paper.

```
Text Representation:
** ACTIVE INSTANCES **
&MEM0 (UNKNOWN-ASSEMBLY# -- 'ASSEMBLY')
    Components: &MEM1 &MEM3 &MEM4
&MEM1 (DISC-DRIVE#)
    Components: &MEM2
&MEM2 (HEAD#) [Mods: TEXTURE/HARD MOBILITY/NONE]
&MEM3 (DISC#) [Mods: DEV-PURPOSE/ROTATION DEV-PURPOSE/STORING]
    Components: &MEM5
&MEM4 (TRANSDUCER#)
&MEM5 (SURFACE#)
&MEM6 (CARRIAGE#)
    Components: &MEM7 &MEM9 &MEM11
&MEM7 (BEARING#) [Mods: NUMBER/3 DISTANCE/SEPARATE TEXTURE/INCISED]
    Components: &MEM8 &MEM10
&MEM8 (BEARING#) [Mods: NUMBER/2 ORDINAL/1]
&MEM9 (TRACK#) [Mods: MOBILITY/NONE SHAPE/CYLINDRICAL]
&MEM10 (BEARING#) [Mods: ORDINAL/3]
&MEM11 (TRACK#) [Mods: TENSION/SPRING SHAPE/CYLINDRICAL]

A list of relations:

            Subject:                   Relation:                Object:

[&REL5]   &MEM4   (TRANSDUCER#)   {R-ADJACENT-TO}      &MEM5   (SURFACE#)
[&REL6]   &MEM6   (CARRIAGE#)     {P-SUPPORTS}         &MEM4   (TRANSDUCER#)
[&REL7]   &MEM9   (TRACK#)        {P-RECEIVES}         &MEM8   (BEARING#)
[&REL8]   &MEM10  (BEARING#)      {P-ENGAGES}          &MEM11  (TRACK#)
[&REL9]   &MEM11  (TRACK#)        {P-IMPELS}           &MEM8   (BEARING#)
[&REL10]  &MEM8   (BEARING#)      {R-ADJACENT-TO}      &MEM9   (TRACK#)
[&REL11]  &MEM11  (TRACK#)        {R-SURROUNDED-BY}    &MEM6   (CARRIAGE#)
[&REL12]  &MEM11  (TRACK#)        {R-ALONG}            &MEM5   (SURFACE#)
                                  ORIENTATION/RADIAL

A list of meta-relations:

Subject:          Meta-rel:          Object:

&REL10            {M-CAUSES}         &REL11
```

Figure 3: RESEARCHER Representation of P41

4 Conclusion

As we have seen in this paper, memory application is an absolute necessity in understanding. However, it is crucial to delineate exactly how memory should be used, as illustrated here. In the RESEARCHER framework, simple syntactic rules, driven by generic memory structures ("semantics") limit the possible ways a representation can be constructed, and searching detailed memory resolves ambiguities. This allows each phase of the processing to be relatively simple, and lets the redundant nature of language help us obtain robust performance. While a different conceptual understanding scheme or a different domain would likely require different points of memory access, this same framework should still be appropriate.

To date, we have run RESEARCHER without memory access on about ~~50~~ patent abstracts as complex as P41, about 20 texts being fully processed with good accuracy. However, even at this stage of development the heuristics needed to avoid memory use are rather complex. Hence we view the addition of the memory access methods described in this paper as crucial to the further progress of RESEARCHER as a robust understander.

References

[Birnbaum and Selfridge 81] Birnbaum, L. and Selfridge, M. Conceptual analysis of natural language. In R. C. Schank and C. K. Riesbeck, Ed., *Inside Computer Understanding*, Lawrence Erlbaum Associates, Hillsdale, New Jersey, 1981, pp. 318 - 353.

[Bobrow and Webber 80] Bobrow, R. J. and Webber, B. L. PSI-KLONE - Parsing and semantic interpretation in the BBN Natural Language Understanding System. Proceedings of the CSCSI/CSEIO Annual Conference, 1980.

[Charniak 83] Charniak, E. "Passing markers: A theory of contextual influence in language comprehension." *Cognitive Science* 7, 3 (1983), 171 - 190.

[DeJong 79] DeJong, G. F. "Prediction and substantiation: A new approach to natural language processing." *Cognitive Science 3* (1979), 251 - 273.

[Gershman 77] Gershman, A. V. Analyzing English noun groups for their conceptual content. Technical Report 110, Yale University Department of Computer Science, 1977.

[Grosz 77] Grosz, B. J. Representation and use of focus in a system for understanding dialogs. Proceedings of the Fifth International Joint Conference on Artificial Intelligence, International Joint Conference on Artificial Intelligence, Cambridge, MA, 1977.

[Lebowitz 80] Lebowitz, M. Generalization and memory in an integrated understanding system. Technical Report 186, Yale University Department of Computer Science, 1980. PhD Thesis.

[Lebowitz 81] Lebowitz, M. Cancelled due to lack of interest. Proceedings of the Seventh International Joint Conference on Artificial Intelligence, Vancouver, Canada, 1981.

[Lebowitz 83a] Lebowitz, M. "Memory-based parsing." *Artificial Intelligence 21*, 4 (1983), 363 - 404.

[Lebowitz 83b] Lebowitz, M. RESEARCHER: An overview. Proceedings of the Third National Conference on Artificial Intelligence, Washington, DC, 1983.

[Lebowitz 83c] Lebowitz, M. "Generalization from natural language text." *Cognitive Science 7*, 1 (1983), 1 - 40.

[Lebowitz 83d] Lebowitz, M. Concept learning in a rich input domain. Proceedings of the International Machine Learning Workshop, Champaign-Urbana, Illinois, 1983, pp. 177 - 182.

[Marslen-Wilson 75] Marslen-Wilson, W. D "Sentence perception as an interactive parallel process." *Science 189* (1975), 226 - 228.

[Riesbeck and Schank 76] Riesbeck, C. K. and Schank, R. C. Comprehension by computer: Expectation-based analysis of sentences in context. In W. J. M. Levelt and G. B. Flores d'Arcais, Ed., *Studies in the Perception of Language*, John Wiley and Sons, Chichester, England, 1976

[Schank, et al. 80] Schank, R. C., Lebowitz, M., and Birnbaum, L. "An integrated understander." *American Journal of Computational Linguistics 6*, 1 (1980), 13 - 30.

[Sidner 79] Sidner, C. L. *A Computational model of co-reference comprehension in English*. Ph.D. Thesis, Massachussets Institute of Technology, Cambridge, MA, 1979

[Small 80] Small, S. Word expert parsing: A theory of distributed word-based natural language understanding. Technical Report TR-954, University of Maryland, Department of Computer Science, 1980.

[Wasserman and Lebowitz 83] Wasserman, K. and Lebowitz, M. "Representing complex physical objects." *Cognition and Brain Theory 6*, 3 (1983), 333-352.

[Wilks 73] Wilks, Y. An artificial intelligence approach to machine translation. In R. C. Schank and K. Colby, Ed., *Computer Models of Thought and Language*, W. H. Freeman Co., San Francisco, 1973.

[Woods 80] Woods, W. A. "Cascaded ATN grammars." *American Journal of Computational Linguistics 6*, 1 (1980).