RESEARCHER: AN EXPERIMENTAL INTELLIGENT INFORMATION SYSTEM*

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

Abstract

The development of very powerful intelligent information systems requires the use of many techniques best derived by studying human understanding methods. RESEARCHER is a system that reads, remembers, generalizes from, and answers questions about complex technical texts, patent abstracts in particular. In this paper we discuss three current areas of research involving RESEARCHER - the generalization of hierarchically structured representations; the use of long-term memory in text processing, specifically in resolving ambiguity; and the tailoring of answers to questions to the level of expertise of different users.

1 Introduction

In [Lebowitz 83a] we described the first stages of development of RESEARCHER, a prototype intelligent information system. RESEARCHER is intended to accept natural language input, patent abstracts in particular, and 1) understand the text, 2) add the acquired information to a long term memory, generalizing as it does so, and 3) answer questions from its memory. In this paper, we will present an overview of the new areas that we are using RESEARCHER to study. In each of these areas we apply techniques derived from cognitive modelling approaches to language and learning. We are using people as our model to help us achieve better performance on very hard tasks (which, in return, gives us insight into the human processing methods).

2 Generalizing hierarchies

The patent abstracts that we have been looking at describe the physical structure of complex objects. Since such objects are most naturally represented as hierarchies of parts, our learning research has addressed the generalization of hierarchically structured descriptions. EX1 is part of a typical patent abstract.

> EX1 - P81; U. S. Patent #4306258; Higashiyama Nobor et al.
>
> A magnetic head supporting mechanism equipped with a magnetic head positioning carriage of a interchangeable double side type flexible disc drive apparatus comprising a carriage having a pair of arms which is rotated in detachable to a double side type flexible disc and arms ...

*This research was supported in part by the Defense Advanced Research Projects Agency under contract N00039-84-C-0165 Many people have contributed to RESEARCHER. In particular, the work on generalizing hierarchies has largely been conducted by Kenneth Wasserman and the work on question answering by Cecile Paris, co-ad vised by Professor Kathleen McKeown.

Our representation of patent abstracts such as this one includes three dasses of information: 1) *a parts hierarchy,* that illustrates the components of each part; 2) *intcrpart relations,* physical and functional relations between various components; and 3) *properties* of the objects. We have concentrated on the parts hierarchy and physical relations [Wasserman and Lebowitz 83]. We are currently working on classification schemes for functional relations and object properties (such as size and composition).

Full understanding requires that we integrate new representations with existing knowledge in memory. RESEARCHER has as one of its goals the incremental generalization of hierarchical descriptions of objects such as EXI by finding similar examples in memory, comparing them with the new example, and abstracting out the similarities.

Generalizing hierarchical representations presents a number of difficult problems. Typical problems are: deciding how the components in the objects being compared correspond; dealing with differing levels of description of objects; and structuring memory so that maximally efficient inheritance of the sort used in semantic networks and frame systems (see [Barr et al. 82]) can be achieved automatically. In this paper, we will only give examples of how the generalization process works, and refer the reader to [Wasserman 85] for more details.

We can break generalization into two phases — 1) when a new example is presented, deciding what other objects to compare it to (since RESEARCHER is not given examples designed to teach a specific concept), and 2) the comparison process itself, which abstracts out similarities.

We will look at the comparison process first, as it is involved in the search process. EX2 and EX3 are two simplified disc drive patents.

> EX2 - A disc drive comprising an enclosure surrounding the disc drive, said disc drive including a spinning assembly, a disc and a readwrite head, said spinning assembly including a spindle connected to a motor, said enclosure comprising a cover on top of a support member.
>
> EX3 - A disc drive comprising an enclosure surrounding the disc drive, said disc drive including a spinning assembly, a magnetic assembly and a readwrite head, said spinning assembly including a spindle connected to a motor, said magnetic assembly comprising a disc, said enclosure comprising a cover on top of a support member.

As human understanders, we can easily see that patents EX2 and EX3 describe similar objects. However, to begin

to generalize the similarities, RESEARCHER must decide how the parts of the representations correspond – for example, that the enclosure in EX2 corresponds to the enclosure in EX3, and not to the spinning assembly, the magnetic assembly or the readwrite head, which are all parts of the disc drive in EX3. Here this is relatively easy, as the enclosures are identical, but we must be able to identify less perfect matches. RESEARCHER does this with a numerical scoring algorithm, similar to the one in [Winston 80).

Figure 1 shows RESEARCHER'S generalization of these objects, taken from [Wasserman 85]. When RESEARCHER makes correspondences of the sort mentioned above, one problem arises in dealing with the discs. The disc in EX2 is described as part of the disc drive, while in EX3 the disc is part of a magnetic assembly which is part of the disc drive. To make the representations match, RESEARCHER must insert a "null" part, which may or may not actually exist in any given object. The two input representations are stored as variants of the generalized object, recording only how they differ from it (basically, in this case, how the null object is resolved; in real examples there would usually be more differences).

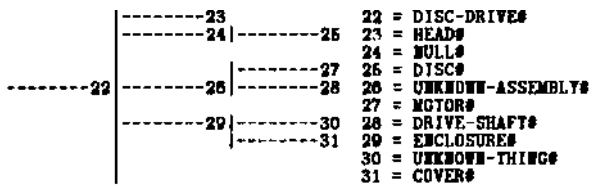

Figure 1: Generalization of EX2 and EX3

Even with just two hierarchical descriptions to compare, the matching process involves a number of problems in determining how the components of the hierarchies correspond. One such problem is the need to insert levels in a hierarchy to obtain a good match, as described above. (While the insertion of a null level by itself decreases the goodness of a match, it may increase the value of lower level matches.) The problem is that there are an exponentially large number of places where null levels can be inserted, each requiring a complex recursive match to test. We have used RESEARCHER to experiment with a variety of different algorithms for deciding where null levels should be inserted for optimal matching, concentrating on ones that only try the most obvious places near the top of the hierarchy.

Since the examples given RESEARCHER are not expressly designed for learning specific concepts (as they would be for a system being taught concepts), the program must decide which examples to compare for the purpose of generalization. This is done using a generalization-based memory of the sort in [Lebowitz 83b]. A hierarchy of concepts is created in memory (a hierarchy of hierarchies, in this case) that organizes specific examples.

In using its generalization-based memory, RESEARCHER takes each new example and searches down the tree for the example or generalized concept most similar to it. This process involves matching generalized concept-, with the new example in much the same way as EX2 and EX3 were matched. We begin by matching the new example with each of the children of the generalization tree's root RESEARCHER selects the best match and looks at that node's children. As long as one of the children produces a better match than the parent node, RESEARCHER continues down the tree. Eventually, it either reaches a leaf (an instance already in memory) or a maximally good generalization (i.e., all of the subordinate nodes contain factors that decrease the quality of the match).

Once the most similar previous example or existing generalization is found, RESEARCHER "factors out" similarities between these representations, and, if need be, creates a new generalization node. In any case, the new example is stored by recording how it differs from the generalizations in memory. This is an optimally space-efficient method of storage, which also captures significant generalizations about the objects in the domain.

The current implementation of RESEARCHER'S generalization scheme works quite well on modest-sized examples. In addition to disc drive patents, a modified version of the program (CORPORATE-RESEARCHER [Wasserman 85]) has been tested on hierarchical descriptions of corporate organizations.

3 Text processing using memory

Since intelligent information systems such as RESEARCHER have available many examples in memory, it seems natural to make use of this information for text processing (beyond identifying lexical items). Patent abstracts, despite being written in legalese are, like the rest of natural language, quite ambiguous. We can use memory to help resolve many ambiguities.

We feel that the best way to use detailed memory information during text understanding in the context of current systems is to identify specific tasks where a piece of information from memory will be useful. More general methods, such as using memory to determine the interesting aspects of a text to focus processing, we leave for the future. We have identified a set of "questions" that arise during text processing that can most easily be answered (and often can *only* be answered) by accessing long-term memory.

It is important to keep in mind that we are proposing using *memory* for understanding, as opposed to general semantic information about words or concepts. While such general information is crucial for our conceptually-based understanding methods, in order to resolve many ambiguities it will be necessary to look at very detailed information in memory – in our case, how the objects described in patent abstracts are constructed and how their pieces relate to each other. The use of the information base also reduces the need to initially hand-code information for RESEARCHER.

EX4 illustrates two kinds of ambiguities that arise in patent abstracts.

> EX4 - A disc head supporting a spindle made of magnetic material.

The first ambiguity in EX4 involves "disc head". Although not syntactically ambiguous, an understanding system such as RESEARCHER must determine the conceptual relationship between the nouns. The phrase "made of magnetic material" is ambiguous in that we do

not know whether it refers to the head or the spindle Both of these ambiguities can only be resolved by looking at memory. In fact, it would be easy to construct scenarios where different states of memory would cause this example to be understood differently (e.g., whether we knew about. magnetic heads or magnetic spindles).

RESEARCHER makes use of relatively simple, but. heavily memory-based, techniques for handling ambiguities of the sort in EX4. Its conceptual analysis type text processing algorithm (described in [Lebowitz 83a; Lebowitz 84]), involves identifying object descriptions (usually noun groups) and connecting them with various relational words (usually prepositions – patent abstracts are quite short of verbs) which indicate the various physical, functional and assembly-component relations mentioned in Section 2 Within this processing algorithm, we have identified places where ambiguity can be identified and memory queried for resolution. Memory is asked which of two possible physical constructions is more likely or what relation is likely to occur between two objects. Questions in both dasses are answered by looking for examples of the possible configurations that already exist in memory.

Figure 2 lists some of the questions that RESEARCHER can currently ask memory for purposes of disambiguation. They primarily involve prepositional phrase attachment and noun groups with multiple nouns.* Our analyses of these ambiguities shares much with the linguistic work of [Levi 78] and the application of this work to AI in [Finin 82]. However, our method of resolving the ambiguities — the use of a dynamic, long-term memory - is rather different.

Form: object-word 1 object-word2
Example: An actuator housing ...
Question: What is the relation between object-word 1 and object-word2?

Form: modifier object-word 1 object-word2
Example: A metal drive cover ...
Question: Does the modifier better apply to object-word 1 or object-word2?

Form: object-word 1 relation-word 1 object-word2 relation-word2 object-word3
Example: A coating; on a disc touching a spindle ...
Question: Does relation-word2 connect object-word3 with object-word 1 or object-word2?

Figure 2: Some disambiguation questions

The search for possible examples that answer a given question is a relatively simple one. RESEARCHER uses its dynamically created device hierarchies to look for possible constructions and relations. It begins its search with general object descriptions and searches through more specific descriptions until a relevant example is found. If several possible constructions (or relations) are found, the one associated with the most general description is used, as that represents RESEARCHER'S most widely applicable information. RESEARCHER'S memory search disambiguation process is described in more detail in [Lebowitz 84].

*The word types used in Figure 2 are functional, rather than syntactic. However, object words are usually nouns and relation words are usually prepositions, although not always in either case.

Our disambiguation methodology bears resemblance to that of [Small 80] and [Hirst 83], except, crucially, it relies on information from a detailed, dynamic memory. Our algorithm does have the side-effect of making understanding subjective, in the sense of [Abelson 73; Carbonell 81], since new examples will be interpreted to correspond to old ones, but we view this as inevitable if we wish to achieve robust understanding.

As an illustration of RESEARCHER'S use of memory in text processing, we will show how it processes part of a real patent abstract, EX1, seen earlier.

Although it may not be immediately obvious, the beginning of EXI is extremely ambiguous. (It may not be obvious because people are so good at resolving ambiguity.) The internal structures of the various noun phrases and the determination of what is a part of what could all be resolved in several ways. Without any information in memory, RESEARCHER would have to rely on general heuristics which might or might not work, and would, in any case, be quite ad hoc. Instead, we will provide RESEARCHER with a few (admittedly somewhat artificial) examples that it can use Specifically, we will give it the following descriptions:

An apparatus with a support mechanism.

An interchangeable double sided floppy disc within a drive which has a magnetic head.

Having given RESEARCHER examples of support mechanisms and double sided floppy disc drives, we let it read EXI (Figure 3).

A number of aspects of RESEARCHER'S text processing are shown in Figure 3. We will focus on its use of memory. Each memory access is indicated by " > > > " The first such use occurs when processing the initial noun group, "A magnetic head supporting mechanism". RESEARCHER uses a. "save and skip" strategy for noun groups - it saves words in a short term memory stack until the head noun is reached. Then it works backwards processing the stacked words. Here, it easily sets up two relations between the head and the mechanism, both from the word "supporting", indicating that the mechanism supports and is connected to the head.

Next RESEARCHER must process "magnetic". It is syntactically ambiguous here whether the modifier applies to the head or the mechanism. (To see the other case, consider, "a complicated head supporting mechanism".) So, RESEARCHER searches its memory for examples of magnetic heads or magnetic mechanisms. It finds the former, and appropriately resolves the ambiguity.

The processing of the next part of EX3, "equipped with a magnetic head positioning carriage" is relatively sedate. "Magnetic" is again ambiguous, but this time refers to an object already described in the patent. Memory again becomes important in processing "a interchangeable double side type flexible disc drive apparatus".

The first problem arises in determining the relation between the drive and the apparatus (remember, noun groups are processed, in effect, backwards). Here, since RESEARCHER has no examples in memory, it uses a

```
Running RESEARCHER at 6:03:10 PM

(A MAGNETIC HEAD SUPPORTING MECHANISM EQUIPPED WITH A
MAGNETIC HEAD POSITIONING CARRIAGE OF A INTERCHANGEABLE
DOUBLE SIDE TYPE FLEXIBLE DISC DRIVE APPARATUS COMPRISING ..

Processing:

A              : New instance word -- skip
MAGNETIC       : Memette modifier; save and skip
HEAD           : Memette within NP; save and skip
SUPPORTING     : Purpose word within NP; save and skip
MECHANISM      : NP word -- memette UNKNOWN-ASSEMBLY#

New UNKNOWN-ASSEMBLY# instance (&MEM6)
New HEAD# instance (&MEM7)
Establishing R-CONNECTED-TO; SUBJECT: &MEM6 ('MECHANISM');
  OBJECT: &MEM7 (HEAD#) [&REL2]
Establishing P-SUPPORTS; SUBJECT: &MEM6 ('MECHANISM');
  OBJECT: &MEM7 (HEAD#) [&REL3]
>>> Select memette modified by DEV-TYPE/MAGNETISM from
      &MEM6 ('MECHANISM') &MEM7 (HEAD#)
Augmenting &MEM7 (HEAD#) with feature: DEV-TYPE = MAGNETISM

EQUIPPED WITH  : Parts of &MEM6 to follow
A              : New instance word -- skip
MAGNETIC       : Memette modifier; save and skip
HEAD           : Memette within NP; save and skip
POSITIONING    : Purpose word within NP; save and skip
CARRIAGE       : NP word -- memette CARRIAGE#

New CARRIAGE# instance (&MEM8)
Establishing P-GUIDES; OBJECT: &MEM8 (CARRIAGE#);
  SUBJECT: &MEM7 (HEAD#) [&REL4]
Recognized instance of &MEM7 (HEAD#)
Assuming &MEM8 (CARRIAGE#) is part of &MEM6 ('MECHANISM')
OF (OF1)       : Part of indicator
Assuming &MEM8 or &MEM6 is part of the following

A              : New instance word -- skip
INTERCHANGEABLE : Memette modifier; save and skip
DOUBLE SIDE    : Memette modifier; save and skip
TYPE           : Skip (SKIP)
FLEXIBLE       : Memette modifier; save and skip
DISC           : Memette within NP; save and skip
DRIVE          : Memette within NP; save and skip
APPARATUS      : NP word -- memette UNKNOWN-ASSEMBLY#

New UNKNOWN-ASSEMBLY# instance (&MEM9)
>>> Looking for relation between DRIVE# and
      &MEM9 ('APPARATUS')
New DRIVE# instance (&MEM10)
Assuming &MEM10 (DRIVE#) is part of &MEM9 ('APPARATUS')
>>> Looking for relation between DISC# and one of
      &MEM9 ('APPARATUS') &MEM10 (DRIVE#)
New DISC# instance (&MEM11)
Establishing R-INSIDE-OF; SUBJECT: &MEM11 (DISC#);
  OBJECT: &MEM10 (DRIVE#) [&REL5]
>>> Select memette modified by RIGIDITY/NONE from
      &MEM9 ('APPARATUS') &MEM11 (DISC#)
Augmenting &MEM11 (DISC#) with feature: RIGIDITY = NONE
>>> Select memette modified by NUMBER-OF-SIDES/2 from
      &MEM9 ('APPARATUS') &MEM11 (DISC#)
Augmenting &MEM11 (DISC#) with feature: NUMBER-OF-SIDES = 2
>>> Select memette modified by DEV-PURPOSE/MANY from
      &MEM9 ('APPARATUS') &MEM11 (DISC#)
Augmenting &MEM11 (DISC#) with feature: DEV-PURPOSE = MANY
>>> Selecting comp for &MEM9 ('APPARATUS') from among
      &MEM8 (CARRIAGE#) &MEM6 ('MECHANISM')
Assuming &MEM6 ('MECHANISM') is part of &MEM9 ('APPARATUS')

COMPRISING     : Parts of &MEM9 or &MEM6 to follow
```

Figure 3: RESEARCHER using memory

heuristic to assume that since "apparatus" describes a rather vague assembly, the drive is probably a part of it. When "disc" is reached, the problem is more complex, since RESEARCHER must determine both whether the disc is related to the drive or the apparatus, and what the relation is. Here, as always, memory is used. Since RESEARCHER does not have an example of a relation between a disc and an apparatus, but knows of an example of a disc being inside a drive, it assumes that the disc is inside the drive here.

When the modifiers, "flexible", "double side" (RESEARCHER has a phrasal lexicon) and "interchangeable" are processed, the program must attach them to either the apparatus or the disc. (The drive is ruled out by syntactic considerations.) The processing is similar to the first noun group, using memory to resolve the conflict. Note that the disambiguation search has a semantic basis, so that the "floppy" disc in memory resolves the ambiguity over "flexible".

Finally, RESEARCHED must decide whether the apparatus has as a part the carriage or the mechanism. (The "part of" relation is indicated by the word "of".) Once again, the routine is the same - search memory for examples and find one that resolves the ambiguity in favor of the mechanism. Different examples in memory would lead to a different resolution.

We have much left to do in our integration of text processing and memory. However, we feel our general approach is quite promising, as our work in building up memory has a positive synergistic effect on text processing robustness. The identification of specific questions to ask memory seems to be much more effective than looking for more general applications of memory to understanding.

4 Q/A in RESEARCHER

Once a substantial knowledge base has been built up by RESEARCHER, it is important that it can be queried intelligently. In [Lebowitz 83a; Paris 84] we described an early question answering module. Recently, our work has concentrated on how RESEARCHER might tailor its answers for individual users. There are many elements to such tailoring, the goal of the user, for example, but here we will concentrate on just one factor - the user's expertise. We have tried to determine the sorts of basic answering strategies that would be appropriate for expert and naive users of the system* Eventually, we will also look at how expertise affects other levels of processing (such as word choice) as well as other factors on answering.

In order to get an idea about the kinds of strategies that might be appropriate for various users, we have looked at texts that describe objects that are aimed at readers with different levels of expertise — several adult and junior encyclopedias. As described fully in [Paris 85], the strategies used in the adult and junior encyclopedias are quite different – the adult encyclopedias, presumably aimed at relative experts, tend to describe the part structure of objects, while the junior encyclopedias describe the processes that take place in the device. EX5 and EX6 show this distinction for descriptions of telephones.

> EX5 - The hand-sets introduced in 1947 consist of a receiver and a transmitter in a single housing available in black or colored plastic. The transmitter diaphragm is clamped rigidly at its edges to improve the high frequency response. The diaphragm is coupled to a doubly resonant system — a cavity and an air chamber - which

* Actually, user expertise falls into two areas — familiarity with the system and familiarity with the domain. We are concerned here with the latter.

broadens the response... (Collier's Encyclopedia, 1962)

As we can see, EX5, taken from an adult encyclopedia, describes a telephone by presenting its parts. The description continues in this vein. It is using a construction quite similar to the constituency schema that McKeown used in her question answering work (McKeown 82], providing an almost tree-like description of the parts of the object. This is in contrast with a description aimed at younger readers, EX6.

EX6 - When one speaks into the transmitter of a modern telephone, these sound waves strike against an *aluminum disk or diaphragm* and cause it to vibrate back and forth in just the same way the molecules of air are vibrating... (Britannica Junior, 1963)

Here the description is process-oriented. It traces the process of transmitting sound, introducing part descriptions only when necessary. This is clearly a different presentation strategy, one that our study of texts indicates is much more widely used in texts aimed at less experienced readers. We feel that a process-oriented answer would be appropriate for RESEARCHER to use when dealing with a novice user not likely to know what various parts are used for.

We are currently in the early stages of implementing these two different strategies for describing the same object. We have implemented simple techniques for producing "expert" type responses using McKeown's constituency schema (although our low-level generation, even here, is quite basic). In addition to looking at the different generation strategies, we are also studying ways to determine the expertise of a user as well as mixed strategies that make use of elements of each generation technique.

5 Conclusion

We have described here three areas of investigation in the study of intelligent information systems focused around the program RESEARCHFER The generalization of hierarchical representations allows the system to learn about a wide range of complex objects and build up a rich memory. This memory is used extensively in text-processing, primarily for disambiguation, to achieve robust performance. Finally, awareness of the expertise level of a user will allow RESEARCHER to tailor it answers to each user. The sum of these three related areas of investigation should lead towards the development of powerful intelligent information systems.

References

[Abelson 73] Abelson, R. P. The structure of belief systems. In R. C. Schank and K. Colby, Ed., *Computer Models of Thought and language,* W. H. Freeman Co., San Francisco, 1973.

[Barr et al. 82] Barr, A., Cohen, P. R. and Feigenbaum, E. A., eds. *The Handbook of Artificial Intelligence, Volumes 1 - S.* William Kaufmann, Inc., Los Altos, California, 1982.

[Carbonell 81] Carbonell, J. G. *Subjective Understanding- Computer Models of Belief Systems.* UMI Research Press, Ann Arbor, Michigan, 1981.

[Finin 82] Finin, T. W. The interpretation of nominal compounds, in discourse. Technical Report MS-CIS-82-3, Moore School of Engineering, University of Pennsylvania, 1982.

[Hirst 83] Hirst, G. *Semantic interpretation against ambiguity.* Ph.D. Thesis, Department of Computer Science, Brown University, 1983.

[Lebowitz 83a] Lebowitz, M. RESEARCHER: An overview. Proceedings of the Third National Conference on Artificial Intelligence, Washington, DC, 1983, pp. 232 - 235.

[Lebowitz 83b] Lebowitz, M. Concept learning in a rich input domain. Proceedings of the 1983 International Machine Learning Workshop, Champaign-Urbana, Illinois, 1983, pp. 177 - 182. To appear in *Machine learning 11.*

[Lebowitz 84] Lebowitz, M. Using memory in text understanding. Proceedings of ECAI-84, Pisa, Italy, 1984.

[Levi 78] Levi, J. N. *The Syntax and Semantics of Complex Nominals.* McGraw Hill, New York, 1978.

[McKeown 82] McKeown, K. R. *Generating natural language text in response to questions about database structure.* Ph.D. Thesis, University of Pennsylvania, 1982.

[Paris 84] Paris, C. L. Determining the level of expertise, Proceedings of the First Annual Workshop on Theoretical Issues in Conceptual Information Processing, Atlanta, Georgia, 1984.

[Paris 85] Paris, C. L. Description strategies for naive and expert users. Proceedings of the 23rd Annual Meeting of the Association for Computational Linguistics, Chicago, 1985.

[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 85] Wasserman, K. *Unifying representation and generalization: Understanding hierarchically structured objects.* Ph.D. Thesis, Columbia University Department of Computer Science, 1985.

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

[Winston 80] Winston, P. H. "Learning and reasoning by analogy." *Communications of the ACM 28* (1980), 689 - 702.