Building a Lexicon from Machine-Readable Dictionaries for Improved Information Retrieval

By J. Terry Nutter, Edward A. Fox, and Martha W. Evens

TR 90-17

To appear in *Literary & Linquistic Computing*

BUILDING A LEXICON FROM MACHINE-READABLE DICTIONARIES FOR IMPROVED INFORMATION RETRIEVAL*

J. TERRY NUTTER AND EDWARD A. FOX
Department of Computer Science
Virginia Polytechnic Institute and State University
Blacksburg, VA 24061-0106

and

MARTHA W. EVENS
Computer Science Department
Illinois Institute of Technology
Chicago, IL 60616

ABSTRACT

Information retrieval systems have a tremendous potential for contributing to research in virtually all areas. To date, this potential has not been fully realized, largely because of problems with controlling retrieval. One way of viewing these problems is that retrieval systems use keywords as indices to retrieve texts, as opposed to understanding the words in requests. We describe a project for creating a lexicon from machine-readable dictionaries, which information retrieval systems can use to go beyond present indexing methods, bringing the actual performance of such systems closer to their potential.

INTRODUCTION

In about 1970, a teacher of one of the current authors, an optimistic and reasonably computer-sophisticated humanities researcher, put a query to an information retrieval (IR) system. About a week later, he came back to his office from class, to find his door blocked by a stack of computer accordion paper about two feet high: his answer. Not knowing what else to do with it, he scanned the listing for the three crucial references he knew were in the collection that the system was searching. They weren't listed.

There have been decades of hype about the advantages for almost any kind of research of storing large on-line full-text databases. Up to now, though, the hype has infrequently been matched by direct advantages for people who are not computer experts, because it is so hard to get what the user wants out of the retrieval systems.

* This research was supported in part by a grant from NCR Corporation and by the National Science Foundation under grant IRI-8703580 to Virginia Polytechnic Institute and State University and under grant IRI-8704619 to Illinois Institute of Technology.

Things are improving, but there are still two complementary problems: the search tends to get too much, including lots of irrelevant references; and it tends to miss central references, even when the retrieval system "knows" about them.

IR systems get at documents using indices: either descriptors (key words) somebody chose or words that occur in the documents themselves. Retrieval matches the words in a request with the index words for the documents. Retrieval failures can stem from either of two sources: the documents may be indexed badly, or the retrieval system may do a bad job of matching the user's information need with the entries in document indices.

One aspect of the problem is that retrieval systems don't know much about words. The situation is not quite as bad is it would be if the retrieval systems used arbitrary indices (say numbers). They can often perform functions like stemming, so that if the user asks for information about "structural elements", the system can figure out that s/he may be interested in something that is indexed by "elements of the structure". But they can't figure out that users interested in sheep husbandry are probably interested in lamb raising.

The solution this research investigates is to provide IR systems with information about words -- meanings as well as grammatical features -- to help them relate the *concepts* in the query with the *concepts* in the texts they retrieve. Since users will continue to use words in queries (and not for instance some special "direct representation" of concepts), what this means in practice is that the system will know what words are related to the various senses of the words that appear in both the query and the documents. In particular, if a user asks about sheep husbandry, the system will know the following:

* "lamb" is related to "sheep" by representing the young of the species, so information about lambs is also information about sheep;
* "husbandry" is related to "raising" because both are forms of animal production, so information about animal raising is relevant to animal husbandry; and for that matter --
* "merino" is an immediate taxonomic subordinate of "sheep" (i.e. a kind of them), and breeding is also producing, so someone interested in sheep husbandry probably wants to know about merino breeding; and so on.

In addition, insofar as the system can use clusters of relevant words to identify the desired senses of at least some of them, the lexicon can help filter unwanted references as well as help find wanted ones that might otherwise be missed.

The following section describes the structure of relational lexicons. The third section gives a brief overview of the hierarchy of lexical relations under investigation at Virginia Polytechnic Institute and State University and in coordinated work at Illinois Institute of Technology. The fourth discusses the processes for extracting such lexicons from machine-readable dictionaries (MRDs), including techniques for identifying specific lexical relations from definition texts. The fifth section considers applications of such a lexicon, beginning with improving information retrieval, especially by non-experts, and going on to other applications and the potential benefits of using the same lexicon for several different applications. Finally, the last section looks briefly at the state of the art, at current limitations, and at some possible ways around them.

RELATIONAL LEXICONS

Lexical relations are specialized links which join concepts as expressed by or embodied in words. These links may represent semantic or syntactic relationships, and can be used to reflect not only major aspects of meaning, but also morphological relationships, implicatures, and so on. A wide study of lexical semantic relationships was launched in the U.S.S.R. in connection with development of the *Explanatory Combinatory Dictionary* (*ECD*) [Apresyan et al. 1969; Mel'cuk and Zholkovsky 1988]. Lexical relations were part of each entry in the unilingual Russian dictionary, and played a key role in the "meaning <=> text" model [Mel'cuk 1973; Mel'cuk 1988]. On the basis of investigations of the dictionaries and surveys of other work with lexical relations, the authors have identified a complex hierarchy of over one hundred lexical relations [Nutter 1989].

However, while theoretical work involving lexical relations has been going on for over twenty years, few machine-readable lexicons based on them have yet appeared. Evens investigated how such a lexicon might be prepared [Evens et al. 1985], and using the Linguistic String Parser [Sager 1981], she and Ahlswede developed a grammar for parsing adjective definitions in *Webster's Seventh*. The current joint project extends this work, using several dictionaries and a variety of methods to create a large, usable lexicon for use in IR systems. Some preliminary results appear in [Fox et al. 1988].

The fundamental concept behind relational lexicons is that as much as possible of the information in the lexicon is represented directly in terms of lexical relations among words (ideally among word senses, or occasionally, as in relations like "alternate spelling", between a head sense and a string). Classical work in the literature (e.g., [Apresyan et al. 1969]) concentrates heavily on semantic relations. The stress on semantics is important, since most traditional computational lexicons (e.g., for natural language processing) contain only syntactic information, with semantic relations represented separately. But the intent of this work is not to continue a tradition which partitions the two kinds of knowledge, simply shifting the term "lexicon" from the syntactic to the semantic representation. Instead, the work aims to combine the two into a single lexicon, with a unified representation. Hence this research talks about lexical relations, as a broad class which combines semantic, syntactic, morphological, and other kinds.

Knowledge Representation

From the point of view of computer systems, a relational lexicon is a graph. Any given word (or word sense) is represented by a node, from which there are pointers to (nodes representing) all the related words and the relations that link them. For instance, suppose that the system has some information about sheep, including the information described in the introduction. In particular, it knows that "lamb" names the young of sheep, "ram" names the male, "ewe" names the female, and "merino" names a kind of sheep. It also knows that "young of", "male of", and "female of" are semantic markers [1], which in turn are a kind of semantic relation, and that "taxonomic subordinate" is also a semantic relation. Notice that there are two kinds of information here: information about relations between words (how "lamb" is related to "sheep"), which is derived from dictionaries, and information about the relations themselves ("young of" is a semantic marker, which is a semantic relation), which is derived from a theoretical analysis of lexical relations. Building the lexicon begins by hand building information about the relations; there are few enough of them, and the structure is

simple enough, that this is a feasible operation. Then relations derived from dictionaries are added.

Because there is information about relations, to be represented in the same network as the information about words, the relations themselves are not merely links, but "objects" (nodes) in the network. An instance of another type of node called a proposition node represents the information that "lamb" is the young of "sheep". This node has argument arcs (ordered, to tell the arguments apart) to the nodes for lamb and sheep, and a relation arc to the node representing the relation young-of. Another proposition node has a member arc to the node for young-of and a class arc to the node for semantic markers, indicating that young-of is a semantic marker.

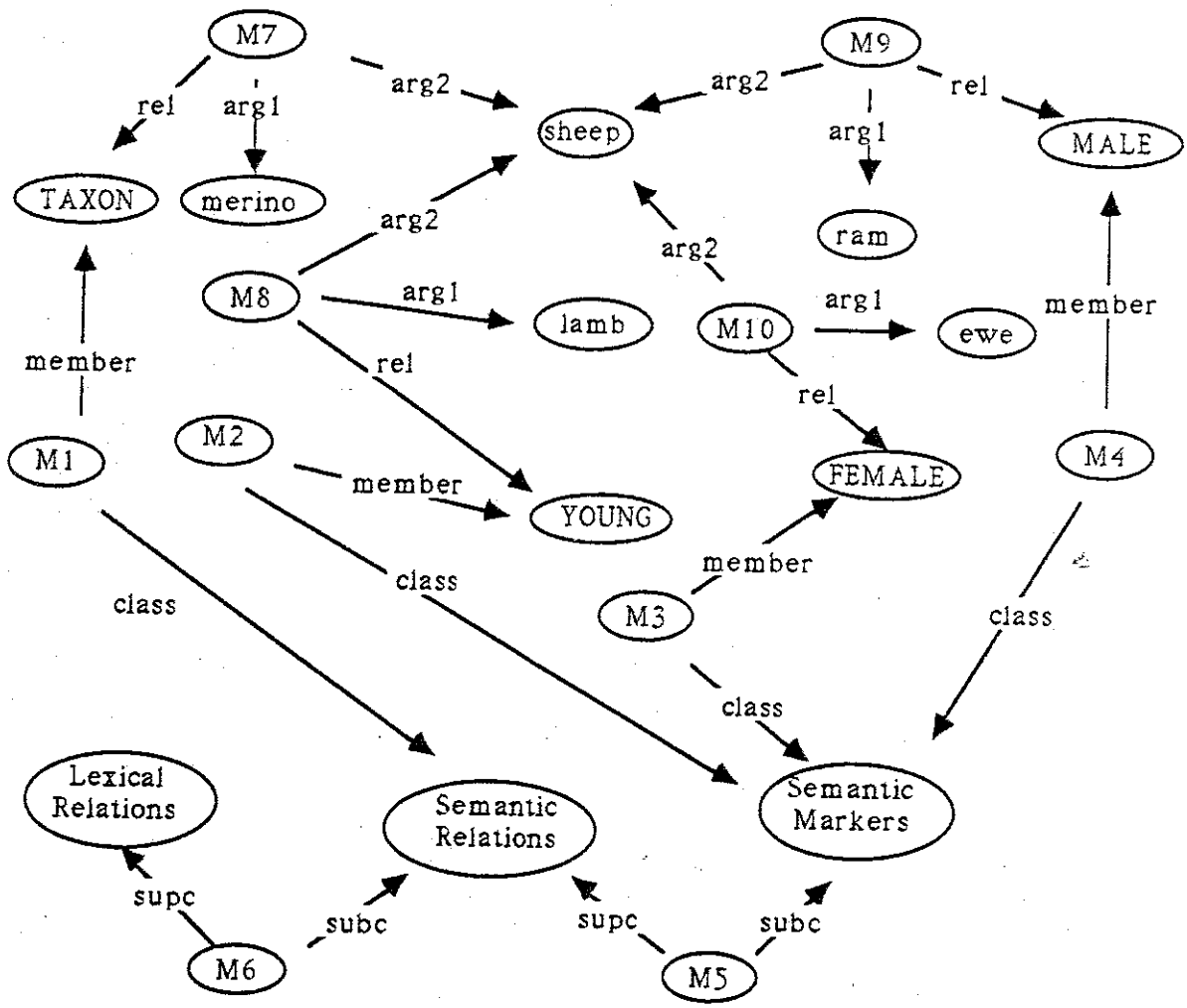

Figure 1. Representation for information about "sheep"

Figure 1 gives a network representation for the information above. Proposition nodes have labels beginning with M's. M1 represents the proposition that the taxonomic subordinate relation is a semantic relation. M2, M3, and M4 represent the information that young-of, female-of, and male-of are semantic markers. M5 and M6 use

subclass/superclass arcs to indicate that semantic markers are semantic relations, which in turn are lexical relations. M7 says that merino is a kind of sheep, and M8 through M10 tell us about lambs, rams, and ewes. The order in which the items appear (in the network diagram or in the computer) is conceptually irrelevant. Arcs are distinguished by their labels and the nodes on either end, nodes by their identifiers (labels). All arcs can be traversed in either direction; that is, for every arc, there is automatically an inverse arc, which points from the original arc's tail to its head.

This network is of course greatly simplified. For instance, it completely ignores the sense/subsense hierarchy. The ram that is a male sheep is a different sense from the one that bashes in doors; in a full network, these have different nodes, with different labels, which have paths (directly or through subsense and supersense arcs) to the nodes actually indexed on (labeled by; accessed through) the strings which are the words' spellings. Similarly all the syntactic information has been eliminated, for simplicity's sake, as well as "upward" information about sheep (that they are mammals, and so on). The representations for these are similar in spirit, although of course the details are different. In general, the representation of a non-symmetric lexical relation between two terms -- e.g., young-of(lamb, sheep) -- consists of a proposition node, with an *arg1* arc to the first argument, an *arg2* arc to the second argument, and a *rel* arc to the relation node. Symmetric relations (various versions of synonymy and antonymy, for instance) use unnumbered *argument* arcs, to make symmetry automatic. The overall representation is a semantic network, based on the SNePS formalism described by Stuart Shapiro [Shapiro 1979; Shapiro and Rapaport 1987], although we cannot use the SNePS software for full-scale implementation, for reasons discussed in the final section.

Path-based Operations

The key to using the relational lexicon, then, lies in finding useful and efficient ways to navigate through these graphs. The basic concept involved is that of a *path*. In the simplest cases, a path is a labeled arc or a defined pattern of labeled arcs. For instance, "a *class* arc, or a *class* arc followed by any number of pairs of arcs consisting of first an *inverse subclass* arc and then a *superclass* arc" defines a path which exists between any node and any other node representing a class to which the original node belongs. Paths can be concatenated (*this* followed by *that*), "and-ed" (a path that satisfies *this* and *that*), "or-ed" (one that satisfies any of the following), complemented (a path that does *not* have this), repeated (any number of these; one or more of these; etc.) or any combination of these operations.

In more complex cases, the path may involve restrictions which must be met. Consider a path that begins with a node representing a word sense and finds all explicit synonyms for that sense and in addition all explicit synonyms of explicit synonyms. That is, if "rapid" is given as a synonym for "fast", "speedy" is given as a synonym for "rapid", and "quick" is given as a synonym for "speedy" but not for "rapid" or "fast", then starting with "fast", we want to find "rapid" (explicit synonym) and "speedy" (explicit synonym of an explicit synonym) but not "quick" (because it is three steps away instead of one or two). Then the following rules describe the desired path:

1. every (other) target of an *argument* arc from a node which has an *argument* arc to this node and a rel arc to SYNONYM; and

2. every node which can be reached by the path in rule 1 from a node obtained from this one by rule 1. (This is applied *only* to nodes which were obtained by rule 1 in the first pass.)

The path defined here, then, gives synonyms (one "step" away) and synonyms of synonyms (two "steps" away), but nothing any more distant. In this example, the restriction on the path is a condition which a node must satisfy in order for the path to make use of it. This is one of two basic kinds of restrictions. The other kind of restriction allows a path of one kind provided that there is no path of another (separately defined) kind from its source to its target. Figure 2 below gives a brief summary of some ways to form paths.

Operator	Description
Converse	A path Q may be defined as the reverse of a path P (for any x and y, if there is a path P from x to y, then there is a path Q from y to x).
Composition	A path Q may be defined as the result of composing a sequence of possibly different paths. That is, if P_1 is a path from x_1 to x_2, P_2 is a path from x_2 to x_3, ..., and P_n is a path from x_n to x_{n+1}, then the path through all the P's from x_1 to x_{n+1} is a path Q.
Self-composition	A path Q may be defined as repetitions of a path P. Special cases: composition zero or more times (Kleene star), one or more times (Kleene +), up to n times, exactly n times.
Logical Not	A path Q may be defined to exist between x and y provided that there is not a path P between them.
Logical And	A path Q may be defined to exist between x and y provided that for some set $P = (P_1, ..., P_n)$ of paths, there is a path P_1 between x and y and ... and a path P_n between x and y.
Logical Or	A path Q may be defined to exist between x and y provided that for some set P of paths, there is at least one path P_i belonging to P between x and y.
Restriction	A path Q may be defined to exist between x and y provided that a path P exists between x and y and that x or y satisfies some further condition. Includes domain restriction (x belongs to a specified set), range restriction (the same for y), and restrictions such as the need to have a particular kind of path from x or y to some other node z).

Figure 2. Path definition operators

The network representation represents a few relations among words directly, and many, many more indirectly. For instance, "being a semantically marked related word for a word" could be viewed as a relation between words. This relation holds between "sheep" and "lamb" (because "lamb" is related to "sheep" by "young-of", which is a semantic marker relation), and similarly between "sheep" and "ram" or between "sheep" and "ewe". Another indirectly represented relation is the one that holds between words which are different semantically marked words relative to the same base term. This holds between "lamb", "ewe", and "ram"; and so on. The importance of paths is that they allow us to define indirectly represented relations, so that the system can retrieve on them as well as on the directly represented ones. Essentially, any relation which can be represented -- however indirectly -- by the formalism can be defined as a path. Path-based retrieval is efficient, because (so long as it is anchored at a starting point) it is just a matter of following pointers; there is no general pattern analysis involved. The operations of path definition and path-based retrieval are thus keys to making the lexicon work efficiently.

THE LEXICAL RELATION HIERARCHY

The example in the previous section demonstrates two points about the current approach to lexical relations that distinguish it from others. First, and most importantly, the relations are not all on a par. They form a hierarchy, which may be very important to understanding information which they convey, and which the current work represents directly. That is, lexical relations are not grouped only to make it easier to read tables made up of lots and lots of them. This grouping is part of the theory which they represent. Further, the hierarchy is not a simple two-tiered hierarchy. It has already been extended to five levels, and will very likely become deeper as investigations progress.

Second, which follows from the first point, the relations are not just a shorthand for giving information about words. They are also themselves objects of knowledge, which the network also directly represents. This knowledge of course begins with the hierarchical information about the relations, but it does not have to stop there. For example, the network can directly represent such information about relations as domain dependence or independence, and in the case of domain dependence, which domain the relation normally reflects. Many domains involve relations among words which are not usually represented in the language at large. For instance, a medical dictionary could be expected to reflect such relations as "symptom of", "counteragent to", and the like. Ultimately, a realistic lexicon for information retrieval may need many such relations, and may also want to know the domain(s) in which the relation applies.

If the hierarchy of relations were viewed as static, designers might exploit information of all these kinds without ever expressing it in the lexicon. The knowledge would then be "compiled into" the procedures used in retrieval; primarily, into path conditions, which would then refer throughout to specific relations, rather than a restriction, for example, that the relation must belong to a certain class. There are at least three good reasons not to do this.

First, even if the relational hierarchy were static, it would not follow that the paths of interest always are. There are huge numbers of possible path definitions, only some of which will ever be useful. Which paths aid effective query expansion for

information retrieval, for instance, is an open research topic. There are some preliminary results [Fox 1980; Evens et al. 1985; Wang et al. 1985; Lesk 1987; Jensen and Binot 1988], but substantial further experimentation will be needed before drawing any very strong conclusions. Since the lexicon will be used to run the experiments, the conclusions (in the form of evaluations of paths) cannot be hard-wired in. Further, there will probably be no *final* conclusions for a very long time to come. It is therefore desirable to maximize flexibility.

Second, even if everything could be hard-wired, it is not clear that one would want to. In any knowledge representation, there are trade offs between making information explicit (cost in space) and making it implicit (cost in time). Following paths will be about equally complex on either plan. Formulating path definitions is relatively straightforward, given the information about the hierarchy. For instance, it is easy to write definitions for the paths that would give us all terms that are semantic markers of a given term, because the system directly represents which relations are semantic markers. Without that information in the network, designers would have to list all the relations in the program. The path definition immediately becomes far more complex (lots of "or's"), and writing it becomes far more error prone (what if one is forgotten?).

Third, current knowledge is not complete. In fact, it probably cannot be: from the viewpoint of this work, the class of lexical relations is open. It follows that at some point it may be desirable to add new relations to the lexicon, because they were found in a new dictionary, because they appeared in some non-dictionary source, because researchers finally learned how to get them out of an old dictionary, or perhaps for other reasons. If hierarchical information appears in the network, then to add (say) a new semantic marker involves adding the information that it is a semantic marker, as well as the new propositions about what it marks. If the path which selects semantic markers looks for a path from the relation to the node for semantic marker, it will now find the markers in this new class "for free". But if the path is defined by listing semantic marker relations, the designers would have to go in and redefine it. They would *also* have to know which paths they will now have to go in and change. After substantial time has passed, there is no reason to believe that any of the paths would do what they were supposed to do any more.

Hence the current studies imply that the hierarchical organization of lexical relations is central not only to their theoretical understanding but also to the structure of the lexicon. The hierarchy itself is large: the authors have identified over a hundred relations, in a structure five levels deep. The appendix gives a stripped down outline of part of the hierarchy; a more detailed description can be found in [Nutter 1989].

EXTRACTING RELATIONAL LEXICONS FROM MRDS

Many (but not all) of the lexical relations so far identified can be extracted automatically from dictionary definitions. The process of identifying relations in definitions has several distinct phases. The current work has not attempted to find a single processing technique, which after one pass will give a list of all the relations in the dictionary. Instead, it involves a process of successive refinement, using successively deeper and deeper techniques until an adequate level of analysis emerges.

Surface Analysis

The first pass uses simple text-processing methods to retrieve relations which are in some sense right on top. Trivially, the dictionary format provides extractable part-of-speech information. A sense/subsense hierarchy for individual words, alternative spellings, and the like can also be established. At a slightly less trivial level, synonyms are often explicitly flagged. Depending on the dictionary, very simple techniques may suffice to identify synonyms (though not necessarily the degree of synonymity; we can reflect this degree of imprecision by asserting the relation at an intermediate point in the relational hierarchy which dominates all the various synonymy relations, rather than selecting one over another). The same goes for antonyms. Single-word definitions, far more common than one might expect, also yield readily to very primitive analysis techniques.

Deeper Analysis by Defining Formulae

The surface techniques described above do get the process off the ground, but at that point it is still flying pretty low. The next step involves dealing with the content of the definitions. Deep understanding of arbitrary texts lies far beyond the state of the art, certainly now and for decades to come. The saving grace of this undertaking is that dictionary definitions are no more arbitrary in their language than in their structure. On the contrary, dictionaries contain many formalized constructs, expressions (like "of or pertaining to") never seen outside their covers, whose purpose is to flag specific lexical relations. This phase of analysis, then, has two parts: identifying such defining formulae, and then using the defining formulae to locate lexical relations in definitions.

It would be lovely to go to (or even make) a comprehensive list of all defining formulae, which could then be used in processing all dictionaries. Unfortunately, while most dictionaries "speak" related languages, they don't use exactly the same one. Each dictionary's defining formulae must be identified, and associated with lexical relations. Fortunately, this need not be done entirely by hand.

This first step is to form KWIC indexes for the definition texts, looking for words and phrases that are very common. The definitions in which they occur are then examined to see whether these words and phrases are (or are part of) defining formulae and which relations they indicate, to group multiple formulae which indicate the same relation, and so on. This effort is interspersed with a certain amount of random sampling, together with a certain amount of guided sampling (to see how they handle "male-of", look up "ram", "stallion", "bull", etc.; and so on). The process is tedious, but feasible and not particularly hard. To date such analyses have been performed on the such documents as *Collins English Dictionary* [Fox et al. 1986] and *Webster's Seventh* [Ahlswede 1988; Ahlswede and Evens 1988a].

These defining formulae are now used to reexamine the definitions. Once again, the strategy has several layers. For example, identifying defining formulae makes it possible to recognize a new set of "virtually one word" definitions, that is, definitions which contain only one word (disregarding articles) over and above a recognized defining formula. That is, once "of or pertaining to" has been identified as a defining formula for the "related adjective" relation, the definition of "solar" as "of or pertaining to the sun" is reduced effectively to one word plus an identified relation. Such definitions are obviously easy to deal with, and require no deep parsing or understanding.

There remain definitions which require more sophisticated analysis, and some which simply go beyond what can be analyzed fully. Progress reports on our experience with *Webster's Seventh* and *Collins English Dictionary* can be found in [Fox et al. 1988] and [Ahlswede and Evens 1988b].

INTEGRATING THE LEXICON INTO LARGER SYSTEMS

Immediate Advantages

The kind of relational lexicon described above offers IR systems several immediate advantages. One possibility, not under current investigation, is enhanced indexing. Substantial work has been done on disambiguating words in large text databases using their immediate (two to three surrounding word) context (see e.g. [Choueka et al. 1983] and [Choueka and Lusignan 1985]). Suppose all the terms in the text base were disambiguated using these techniques, and then a relational lexicon were used to index automatically on an expanded term set, including words that do not occur in the document, but are related to relevant senses of those that do. Such an approach might give some substantial advantages immediately. No work is currently scheduled in this direction, but it is an interesting avenue, which may be explored eventually. This approach addresses the potential problem of bad indexing, mentioned in the first section.

But IR system designers may not want to store very large term sets, and may not be able to analyze the entire corpus for disambiguation. Without disambiguation, expanded indexing would create garbage indexes (expanded from wrong senses of words in the text), thus aggravating the problem of retrieving documents the user doesn't want. Happily, there is a second direction to work from, and this is where the authors are concentrating our efforts.

Recall that the second problem involved poor matching between the document indices and the terms in the query. As an alternative to expanding the indices, the query can be expanded. This process can be *semi*automated. That is, given a query, the system can show the user a list of words which can be derived from its terms, and ask the user which of these the user is interested in. This amounts to getting the user to perform disambiguation for the system. Showing the user all the possible expansions on the first pass might overwhelm the user (and the screen!) with too much information. Fortunately, there is no need to do that, so long as the initial selections include enough information to let the system disambiguate to a useful level (if the user said "ram", we might want the initial choices to include "sheep" and "battering"; we could use the one selected to identify the relevant sense of "ram"). Then, with major terms disambiguated, the system could come back with a "full" expansion, and give the user a last chance to perform additions or deletions, or to try again. The aim is to include enough related terms *for the actual query* to find documents that use terms the user didn't think of, without capturing documents that use terms related to the wrong senses of the terms the user did think of. Obviously, this could be combined with the expanded indexing approach described above. In that case, with senses identified within reason on both sides -- document and query -- it may also be possible to filter documents indexed by the same terms as the user's request, but used in entirely different senses.

This second approach, expanding the query, presupposes that the system would know how to expand a term if it knew what senses of the terms in the query the user had in mind. This is not necessarily so. It is clear that morphologically related words

would be useful, and that getting them is not as simple as stemming algorithms make it seem. For instance, "neatness" is a legal morphological variant of "neat" when it means "orderly" (a neat room), but not when it means "undiluted" (a neat drink). If the user includes "neat" in a query, a straightforward stemming algorithm will either always or never match "neatness", thus systematically erring for one of the senses. Hence including morphological relations and expanding on them involves an improvement over stemming algorithms, and is almost surely desirable. What else?

Using synonyms is a glaringly obvious idea. But there are hidden pitfalls, even given disambiguation. Essentially, there are very few true synonyms. Most word pairs identified as synonyms in dictionaries in fact have at least subtly different meanings. So while synonymy ought *prima facie* to be transitive -- if x is a synonym for y and y is a synonym for z, then x should be a synonym for z -- in fact it isn't. We have anecdotal evidence [2] that someone has measured the average length of the chain through identified synonyms from a given word to its antonym in one of the Collins dictionaries, and it came to seven. So obviously it is a bad idea to look seven synonyms away expecting to find synonyms, at least in that dictionary. How far should the system look?

It is fairly obvious that the system should look *down* the taxonomic hierarchy at least a little; people interested in cats are probably interested in Persian cats. But how far does this generalize? Are people interested in mammals really interested in frost point Siamese cats? At what point does the system decide that the expanded terms indicate that texts involving them but not their superordinates are getting too specialized for the user's interests? Conversely, it seems obvious that going *up* the taxonomic hierarchy is often not appropriate. People who want information on frost point Siamese certainly don't want information on animals in general, or even just on mammals. But is it equally obvious that they don't want to hear about other kinds of Siamese, or Siamese in general, without considering point color?

We mentioned earlier that there is some preliminary evidence that expanding on lexical relations helps in information retrieval (see [Evens et al. 1985], [Fox 1980], [Jensen and Binot 1988], and [Lesk 1987]). But to date, there is little evidence on *which* relations are most useful, or *how far* to pursue expansion. There is good reason to believe that the only way to find out is to run experiments and see. With the lexicon described above, including the ability to define a wide variety of paths dynamically, such studies can be run effectively, and several are planned for the relatively near future.

Putting the Lexicon to Multiple Uses

Relational lexicons are not restricted to uses in IR, whether on the text or the query side. They also have applications in natural language understanding and generation systems, and potential for extension into multilingual contexts, including machine translation and computer-aided language instruction, although the multilingual potential is far more conjectural at the moment than the single-language applications.

Given the current state of computational linguistics, it is already an intriguing idea to have a single lexicon that can be used by two programs, let alone by programs with fundamentally different areas of application. A separate project at Virginia Polytechnic Institute and State University in connected text generation, as yet in the early stages, plans to use a scaled down version of the lexicon actually implemented on the SNePS software mentioned before to aid in natural language generation. For a preliminary report on some aspects of the project, especially regarding significant uses of the taxonomic hierarchy which go well beyond the traditional feature inheritance

through IS-A, see [Cline and Nutter 1989].

In the long run, these separate applications for the lexicon can be brought together. It is partly this (also a desire to design along principled, linguistic lines and a reluctance to keep reinventing the wheel every time a system needs to know about words) that motivates a design which will be reusable across systems. The IR system in the context of which the initial implementation of the lexicon and experimentation with it are taking place is a large knowledge-based system (CODER) designed to study how advanced representational and inferential techniques can be used to improve information retrieval [Fox and France 1987; Fox 1987]. CODER goes far beyond indexing documents and searching on queries. It is a large, distributed AI system, with subsystems for many different kinds of tasks which could contribute to an IR environment, and with a modularized design which allows cooperation among tasks and extension to new tasks. In this environment, it is anticipated that the lexicon will be used by the retrieval system for query expansion, but also eventually by a natural language interface, a document analysis system, and the like.

The Holy Grail, then, is not just bigger and better queries. We have in mind a single system, with one relational lexicon, which uses it to --

* *read* and *understand* texts,
* *formulate and store* representations of their *abstracts*;
* *talk* with users who want to retrieve texts;
* *understand* their requests;
* *expand* them as useful; and finally
* *explain* what it did and what it found.

Today, this system is still a distant goal. But it is worth noting that the architecture which would be needed to put the parts together exists now, and several of the parts exist in prototype form. Hence while the quest will be difficult, we do know which directions to start out in, and what the goal will look like when we find it.

CONCLUSIONS: THE STATE OF THE ART, LIMITATIONS, AND OUTLOOK

Up to now, this paper has looked at what has been done, and what can be done. Before closing, it seems appropriate to look at what has yet to be done, and what may not be possible to do, at least in any near future. The previous section broadened our scope to look at a wider challenge. Narrowing in from that utopian outlook to the original, more precise task of building the lexicon, three roadblocks present themselves.

1. Not everything can be gotten out of the dictionaries yet.
2. Realistic lexicons need more information that isn't in the dictionaries.
3. The sheer size of these systems starts posing serious problems.

Getting Everything out of the Dictionaries

Dictionaries use specialized language, but they also use general English. Complete understanding of all of a given dictionary's definitions essentially entails solving the full natural language understanding problem. This is so big and so hard that nobody is even working on it: lots of people are working on lots of corners, but the full problem is out of sight. It won't be solved in the next decades. So at least some content of the dictionaries will continue to elude us for a long time to come.

This problem is frustratingly acute when it comes to trying to understand usage examples. Usage examples are clearly one of the more useful (to humans) aspects of dictionaries. They tell a great deal about things like selection preferences (the verb "drink" can be used in many ways, but it prefers an animate subject and a liquid object) that are very useful to applications in natural language understanding and generation and that are difficult to find in other ways. But the usage examples are the one part of the definition most likely to use English in its fullest generality. How can a parsing system tease out the lexical information desired without having to solve the natural language understanding problem first?

Also frustrating is the problem of inconsistent formula use. Most traditional dictionaries, which form the pool of available MRDs, are compiled by people using little slips of paper. Under these circumstances, any regularity in definition forms is a triumph. Dictionaries clearly strive for standardized ways of expressing relations, but they sometimes miss. It is frustrating to fail to identify a perfectly familiar and analyzable relation because in this definition, the dictionary uses a rare variant defining formula which the concordance approach did not pick up.

Missing Lexical Information

Some kinds of lexical information simply are not in dictionaries. One example is the problem of proper names. It is arguable that some names should simply be taken as grammatically structured strings (yes, there is a given name/family name structure which can be derived, but the individual names are simple facts which may be completely individual and should simply be accepted). But common sense says that while this may be a reasonable approach to a name like "D'Veara Plavcan" [3], it is wrong to apply it to "Richard Nixon", or even "Michael White". Some entire names (like "Richard Nixon") should probably be recognized whole, and certainly common name elements (like "Michael" and "White") should be recognized separately. There are huge numbers of such names, and by-and-large, they don't occur in the dictionary. The same goes for names of places, buildings, and anything else that has a proper name.

There are other kinds of missing information: new words, for instance. They are relatively rare in what one might think of as "core English", but they crop up in technical sublanguages all the time. How can lexicons keep up-to-date, including words that have entered the language but may not yet appear in dictionaries?

Technical sublanguages present problems of their own. A philosopher trying to do retrieval on issues in analytic epistemology probably does not want to rely on either the dictionary definition of "analytic" or the dictionary definition of "epistemology", although both entries exist in most dictionaries. It is actually unclear whether the problem is worse for people like philosophers, for whom the terms appear in dictionaries but without a suitably technical definition, or for (say) lawyers, many of whose terms may not appear at all. Some fields have technical dictionaries, some of which are even available in machine-readable form. But that is by no means the rule. Suppose an on-line text database includes the full contents of a university library. In this case, no reasonable assumptions about the domains the base covers can limit the range of vocabulary that documents and queries will use. It now becomes clear that IR systems will need comprehensive technical vocabularies, and it is unclear where to get them. Research is also ongoing on how to identify and process definitions which occur in non-dictionary texts, but to date has achieved only limited success. So while processing dictionaries gives a start, it is only a start.

Problems of scale

Traditional computational lexicons sit in main memory, where they take up an amount of space that is negligible in terms of interfering with other system operations. This is because most traditional computational lexicons are toys. Thinking in terms of information retrieval, it is clear that toy lexicons will not do. It is possible to go through the text collection (assuming, which may be false, that it is fixed) and identify all the terms that occur in it. Restricting the lexicon to those terms might work if the system dealt only with the text base. But it must deal with queries. If users always knew exactly the terms that occurred in the documents and exactly how they are used to index the collection, there would be no need for improved retrieval. The whole problem is that users *haven't* generally read the documents they are trying to find, and if they have, they don't remember the words (as opposed to ideas) in them. Users speak English, and the system has to deal with that.

It follows that lexicons have to be dictionary sized and larger. But the lexicon doesn't just hold strings. It stores structured entities relating ideas. The natural way to do that is to use a semantic network. But this approach runs into three problems.

First, the lexicon won't fit in main memory. It's much too big. It won't even fit in the addressable virtual memory space of many machines. That means that building it needs referencing abilities that simply are not found in existing AI software. Second, even if it would fit into the virtual space, there is no locality, and the machine will spend virtually all its time swapping. With highly connected structures of this kind, we can't afford to rely on, for example, the UNIX paging algorithms. Third, never mind, because the usual AI languages won't let programmers build what the lexicon needs. The current estimate of the total count of nodes in the lexicon network is in the neighborhood of 2^{30}. The system will barely have begun building the structure when it overflows the LISP or PROLOG hash table. Present AI languages just plain can't handle that many objects.

This means that building the lexicon entails building a separate network manager and database back-end, which is why our lexicon does not actually use the SNePS software. SNePS supports many desirable facilities, some of which we do not anticipate being able to reimplement soon, the most important being a style of node-based inference that is predicate-logic-like. Because that goes beyond current projects, the backend design enhances path-based inference to include some aspects (especially restrictions) which would normally be done in SNePS by combining node- and path-based reasoning. The resulting system, LEND (Large Extended Network Database), will use perfect hashing to construct its bases, and supports both semantic network and hypertext functions [Fox et al. 1989; France et al. 1989]. The LEND design is very near completion, and many of its modules have been implemented in prototype form.

Where We Are, and Where We're Going

The work reported here has identified thousands of lexical relations among terms, some of which have been load into parts of LEND. For prototype experiments, stripped down versions which are small enough to fit can be loaded into SNePS. A fully functional relational lexicon remains a goal, but now a rapidly approaching one. Once achieved, it will not be complete, for the reasons given above, but it will be extendible. In other words, all the limitations except the first are ordinary research issues, and researchers are currently working on them all. The state of the art can't do it now -- but in five years, who knows?

ENDNOTES

[1] Actually, they are a special kind of semantic markers, namely object property markers. For the purposes of this example, we will ignore this further level of the hierarchy. For an idea what semantic markers are, see the appendix, which lists some of the relations we consider in this class.

[2] Personal communication, Patrick Hanks, February 1989.

[3] No, we don't know anybody named D'Veara Plavcan. But the first author has known someone with the given name D'Veara and someone else with the family name Plavcan, so in a few generations there could be one....

REFERENCES

Ahlswede 1988
Ahlswede, T. E. Syntactic and Semantic Analysis of Definitions in a Machine-Readable Dictionary. Ph.D. Dissertation, Computer Science Department, Illinois Institute of Technology, Chicago, IL 60616, 1988.

Ahlswede and Evens 1988a
Ahlswede, T.E. and M.W. Evens. Generating a relational lexicon from a machine-readable dictionary. *International Journal of Lexicography* 1:3 (1988) 214-237.

Ahlswede and Evens 1988b
Ahlswede, T. E. and M. W. Evens. Parsing vs. Text Processing in the Analysis of Dictionary Definitions, *Proc. 26th Ann. Meeting of the Assoc. for Comp. Ling.*, Buffalo, June 1988, 217-224.

Apresyan et al. 1969
Apresyan, Y. D., I. A. Mel'cuk, and A .K. Zolkovsky. Semantics and Lexicography: Towards a New Type of Unilingual Dictionary. In *Studies in Syntax and Semantics*, ed. F. Kiefer, 1-33. Dordrecht — Holland: D. Reidel, 1969.

Choueka and Lusignan 1985
Choueka, Y. and S. Lusignan. Disambiguation by Short Contexts. *Computers and the Humanities* 19:3 (1985) 147-157.

Choueka et al. 1983
Choueka, Y., T. Klein, and E. Neuwitz. Automatic Retrieval of Frequent Idiomatic and Collocational Expressions in a Large Corpus, *ALLC Journal* 4 (1983) 34-38.

Cline and Nutter 1989
Cline, B. E. and J. T. Nutter. Implications of Natural Categories for Natural Language Generation. *Proceedings of the First Annual SNePS Workshop*, Buffalo, NY, November 13, 1989, 125-132.

Evens et al. 1985
Evens, M. W., J. Vandendorpe, and Y.-C. Wang. Lexical-Semantic Relations in Information Retrieval. In S. Williams, ed., *Humans and Machines: The Interface through Language*, Ablex (Norwood, N.J.) 1983, 73-100.

Fox 1980
Fox, E. A. Lexical Relations: Enhancing Effectivenes of Information Retrieval. *ACM SIGIR Forum* 15:3 (Winter 1980), 6-35.

Fox 1987
Fox, E. A. Development of the CODER System: A Testbed for Artificial Intelligence Methods in Information Retrieval, *Information Processing and Management* 23:4 (1987) 341-366.

Fox and France 1987
Fox, E. A. and R. K. France. Architecture of an Expert System for Composite Document Analysis, Representation, and Retrieval, *International Journal of Approximate Reasoning* 1 (1987) 151-175.

Fox et al. 1986
Fox, E. A., R. C. Wohlwend, P. R. Sheldon, Q. F. Chen, and R. K. France. Building the CODER Lexicon: The Collins English Dictionary and Its Adverb Definitions. Department of Computer Science Technical Report TR-86-23, Virginia Polytechnic Institute and State University, Blacksburg, VA 24061-0106, 1986.

Fox et al. 1988
Fox, E. A., J. T. Nutter, T. E. Ahlswede, M. W. Evens, and J. A. Markowitz. "Building a Large Thesaurus for Information Retrieval", *Proc. Second Conf. of Applied Natural Lang. Proc.*, Austin, February 1988, 101-108.

Fox et al. 1989
Fox, E. A. , L. S. Heath and Q. F. Chen. An O(n log n) Algorithm for Finding Minimal Perfect Hash Functions. Department of Computer Science Technical Report 89-10, Virginia Polytechnic Institute and State University, Blacksburg, VA 24061-0106, 1989.

France et al. 1989
France, R. K., Q. F. Chen, and J. T. Nutter. LEND Design Specifications. Draft 1989.

Jensen and Binot 1988
Jensen, K. and J.-L. Binot. "Dictionary Text Entries as a Source of Knowledge for Syntactic and Other Disambiguations," *Proc. Second Conf. on Applied Natural Lang. Proc.*, Austin, February 1988, 152-159.

Lesk 1987
Lesk, M. Automatic Sense Disambiguation Using Machine Readable Dictionaries: How to tell a Pine Cone from an Ice Cream Cone." *Proc. ACM SIGIR Internat. Conf. of Research and Development in I. R.*, 1987, 24-26.

Mel'cuk 1973
Mel'cuk, I. A. Towards a Linguistic 'Meaning <=> Text' Model. In *Trends in Soviet Theoretical Linguistics*, ed. F. Kiefer, D. Reidel (Dordrecht), 1973, 33-57.

Mel'cuk 1988
Mel'cuk, I. A. *Dependency Syntax: Theory and Practice.* State University of New York Press (Albany) 1988.

Mel´cuk and Zholkovsky 1988
Mel´cuk, I.A. and A. Zholkovsky. The Explanatory Combinatorial Dictionary. In *Relational Models of the Lexicon*, ed. M.W. Evens, Cambridge University Press (Cambridge), 1988, 41-74.

Nutter 1989
Nutter, J. T. A Lexical Relation Hierarchy. Department of Computer Science Technical Report TR 89-6, Virginia Polytechnic Institute and State University, Blacksburg, VA 24061-0106, 1989.

Sager 1981
Sager, N. *Natural Language Information Processing*. Addison-Wesley, New York, 1981.

Shapiro 1979
Shapiro, S. C. The SNePS Semantic Network Processing System. In *Associative Networks: The Representation and Use of Knowledge by Computers*, N. V. Findler, ed., Academic Press (New York) 1979, 179-203.

Shapiro and Rapaport 1987
Shapiro, S. C. and W. J. Rapaport. SNePS Considered as a Fully Intensional Propositional Semantic Network. In *The Knowledge Frontier*, N. Cercone and G. McCalla, eds., Springer-Verlag (New York) 1987, 263-315.

Wang et al. 1985
Wang, Y.-C., J. Vandendorpe, and M.W. Evens. Relational thesauri in information retrieval. *J. ASIS* 36:1 (1985) 15-27.

APPENDIX: PARTIAL OUTLINE OF THE LEXICAL RELATIONS HIERARCHY

FUNDAMENTALLY SEMANTIC RELATIONS

TAXONOMIC CLASSIFICATION RELATIONS
- *Hierarchical Location*
  - Subclass/Superclass
  - Set membership
  - Hierarchical siblings
- *Sameness and Likeness*
  - Synonyms
  - Cross-language synonymy
  - Similarity/near synonymity
  - Specialized synonymy: idiomatic synonyms
  - Similarity + Difference
- *Opposites*
  - Undistinguished opposition
  - Logical opposites
  - Contrasting extremes
  - Complements
  - Inverse operations
  - Reversing operations

PARTS, WHOLES AND AGGREGATES
- Aggregate name
- Part-Whole
- Head-Organization

ORDERING AND MEASURING RELATIONS
- *Order*
  - Sequence
  - Alternate form
- *Measure*
  - Unit - Dimension
  - Intensifying verb
  - Reducing verb

SEMANTIC MARKERS
- *Object property markers*
  - Is-the-male-of
  - Is-the-female-of
  - Is-the-young-of
  - Material/object
  - Property/object
  - Object/designation
  - Use/Object
  - User/Object
- *Selectional markers*
  - Selects for human
  - Selects for male/female
  - Selects for animate/inanimate
  - Selects for abstract/concrete
- *Physical properties and relations*
  - General description
  - Characteristic sound
  - Relative spatial location

GENERIC TYPICAL CASE FRAME FILLERS
- *Act-Slot Relations*
  - All purpose slot relation
  - Act/actor
  - Act/object
  - Act/recipient
  - Act/product
  - Act/instrument
  - Act/location
- *Object-object relations*
  - Thing/container
  - Thing/producer
  - Thing/habitat
  - Agent/product
  - Situation-verb relations
    - Situation/slot-verb
    - Situation [subj] + verb
    - Verb + situation [obj]
- *Miscellaneous*
  - Act/manner
  - Agent/product
  - Agent/goal
  - Agent/material
  - All-purpose case frame

OTHER PARADIGMATIC RELATIONS
- *Causal Relations*
  - Situation/Cause (collocational)
  - State/Action
  - Affector/Affected
  - Activity/Outcome
- *Paradigmatic verb relations*
  - State/verb expressing state
  - State/verb to achieve state
  - State/copular verb used with state
  - Object/verb to make ready
  - Object/verb to destroy or remove
  - Object/verb to deteriorate
- *Dispositional relations*
  - Generic dispositional
  - Act/Disposition to act
  - Act/Disposition to be acted on

FUNDAMENTALLY MORPHOLOGICAL AND SYNTACTIC RELATIONS

MORPHOLOGICAL RELATIONS
- State/verb (nominalized verb/verb)
- Noun/related adjective
- Adjective/related adverb

INFLECTIONAL RELATIONS
- Past/inf
- Past participle/inf
- Plural/singular

ASPECTIVE RELATIONS
- Is-state
- Is-action
- Object/realization
- Event/initiation
- Event/maintaining
- Event/termination
- Perfective

OTHER SYNTACTIC RELATIONS
- SComp
- Reflex
- Recip

PROPOSITIONAL ATTITUDE RELATIONS

- Factive
- Implicative
- Only-if
- If
- Negative-if
- Negative-implicative
- Counter-factive
- Dull

TRULY MISCELLANEOUS RELATIONS

- Ownership
- Circularity
- Contingency