Using natural language processing to construct large-scale hypertext systems

Will Fitzgerald and Christopher Wisdo
The Institute for the Learning Sciences, Northwestern University
Evanston, Illinois, USA
{will, hvizda}@ils.nwu.edu

Abstract

A reasoned approach to constructing hypertext and hypermedia systems requires a theory of how texts are connected (a theory of *text association*) and partial theories of what the text is describing (*domain theories*). However, systems built in this way, such as ASK Systems [Ferguson, et al. 1992], are difficult to build even when the domain theories and a theory of text association are in place. Natural language understanding technologies that take advantage of underlying semantic representations can assist hypertext builders create effective hypertext links. We describe how we are using Direct Memory Access Parsing, or DMAP [Martin 1989; Riesbeck and Martin 1986], as part of a project to deliver hypertext systems based on Compton's MultiMedia Encyclopedia.

Please note:

Should this paper be accepted, the first author (Will Fitzgerald) will attend the conference.

Workshop preferences:

Knowledge acquistion from natural language

Shareable and reusable ontologies

Please consider this paper for the graduate student award.

Submitted to the Eighth Knowledge Acquistion for Knowledge-Based Systems Workshop, Banff, Cananda, January 30-February 4, 1994.

Using natural language processing to construct large-scale hypertext systems[1]

Will Fitzgerald and Christopher Wisdo
The Institute for the Learning Sciences, Northwestern University
Evanston, Illinois, USA
{will, hvizda}@ils.nwu.edu

Abstract

A reasoned approach to constructing hypertext and hypermedia systems requires a theory of how texts are connected (a theory of *text association*) and partial theories of what the text is describing (*domain theories*). However, systems built in this way, such as ASK Systems [Ferguson, et al. 1992], are difficult to build even when the domain theories and a theory of text association are in place. Natural language understanding technologies that take advantage of underlying semantic representations can assist hypertext builders create effective hypertext links. We describe how we are using Direct Memory Access Parsing, or DMAP [Martin 1989; Riesbeck and Martin 1986], as part of a project to deliver hypertext systems based on Compton's MultiMedia Encyclopedia.

1. Organizing hypertexts for more intelligent use

What place do natural language processing techniques have in the creation of large-scale hypertext systems? In recent years, many collections of text have become available in electronic form, creating expectations that large amounts of information will be accessible to anyone with a computer and a CD-ROM drive or a high-speed modem. Public organizations such as the Gutenberg Project and the Oxford Text Archive and private corporations such as Encyclopedia Britannica are providing these texts. Yet, simply providing electronic texts in readable form is not enough. In parallel, there has been a great growth in interest in providing new ways to read these texts that go beyond the linear nature of written books. These systems of *hypertext* [Conklin 1987] offer the possibility of new ways of thinking about the texts we read. A number of vendors already offer first-generation hypermedia systems for browsing texts, such as Compton's MultiMedia Encyclopedia.

Hypertext systems are complex and difficult to build. A good *hypertext builder's workbench* would provide tools for automating their creation. In this paper, we will focus on how the parsing technology known as *direct memory access parsing (DMAP)* [Martin 1989; Riesbeck and Martin 1986], can assist in the creation of a hypertext system. Further, we discuss our hypertext builder's workbench for creating hypertext systems from on-line text. Before we discuss the role of parsing, however, we must begin by discussing our understanding of what constitutes a good hypertext or hypermedia system.

A common problem for users of hypermedia systems is getting 'lost in hyperspace.' A reader of a hypertext system navigates from one text to another, but it is usually not clear why the underlying links were built, where the reader is going, or whence the reader has come. Most hypermedia systems lack a transparent semantics for their links among texts—it is difficult to know why the hypertext creator is suggesting a link to another text [Spiro and

---

[1]This work was supported in part by the Defense Advanced Research Projects Agency, monitored by the Office of Naval Research under contracts N00014-91-J-4092 and N00014-90-J-4117. The Institute for the Learning Sciences was established in 1989 with the support of Andersen Consulting, part of the The Arthur Andersen Worldwide Organization. The Institute receives additional support from Ameritech and North West Water Group plc, Institute Partners, and from IBM.

Table 1
Conversational Associative Category (CAC) Links
(After [Bareiss and Osgood 1993])

Dimension	CAC Link	Typical Questions Raised
Refocusing	Context	Why does this matter? What's the big picture? What do I have to know to understand this?
	Specifics	Is there an example of this? Give me more detail about this.
Advice	Opportunities	Why do I want to know this? What value is it to me? What's interesting about this?
	Warnings	Are there any bad effects of this? How might I be wrong about this?
Causality	Previous Events and Causes	What happened before this? What caused this? Why is it true?
	Later Events and Results	What happened after this? What was the result of this?
Comparison	Alternatives	What's another way to do this? Are there any counterexamples? What other ways of looking at this are there?
	Analogies	What is this made up of? Is this anything else like this?

Jehng 1990]. Solving the navigation problem requires theories of how texts are connected (a theory of *text associations*) and what the text is describing (*domain theories*). We call hypertext or hypermedia organized by domain theories and constrained by a theory of associations a *knowledge-based hypertext*. We are using one family of knowledge-based hypertext systems, ASK systems [Ferguson, et al. 1992], in our research.

2. The Characteristics of ASK systems

There are three characteristics of ASK systems we would like to emphasize:

- ASK systems categorize links between texts—text associations—based on a theory of conversation, called conversational associative categories, or CAC links [Schank 1977].
- ASK systems use underlying representations—domain theories—that are sparse and tractable, leaving most of the intelligence of the system in the texts and their readers, not in the domain theory representation.
- As ASK systems become large, they require methods for automatically generating links between texts [Osgood and Bareiss 1993].

We now turn to a justification of these characteristics in the development of knowledge-based hypertext systems.

2.1 Text associations using CAC links

A coherent text provides a flow from point to point. This is all the more important in hypertext systems, because the path one reader takes may be different from any other reader. Without a theory to guide the creation of a link between one text and another, a hypertext becomes incoherent and difficult to navigate. The natural links between texts are based on the concepts that occur within the text. Schank [1977] describes eight types of associative links between points called *conversational associative categories*, or *CAC, links*. These CAC links can be thought of as links to other texts that answer questions raised in the text being read [Bareiss and Osgood 1993]. Table 1 lists a brief description of the CAC links and examples of the questions that typically fall within these categories.

For example, consider the following two texts about Kenya, which come from Compton's MultiMedia Encyclopedia [Compton's 1991; titles ours]:

Kenya's population problem. In the early 1980's it was estimated that Kenya's population was increasing at the rate of about 4 percent a year. This growth rate, one of the world's highest, greatly increases the people's demand for land, housing, food, jobs, education, medical care, and other services. These conditions place a severe strain on the economy of Kenya, a country whose resources are extremely limited.

Kenyan Dependence on Foreign Oil. One reason that Kenya has remained heavily dependent on agriculture is its lack of fuel resources, such as petroleum. Totally reliant on foreign countries for oil, Kenya's manufacturing industries have developed slowly.

These texts are not contiguous in the original source. A typical hypertext system might have a link between these two texts; perhaps clicking on the word 'resources' in the first paragraph would bring the reader to the second. That both paragraphs discuss the resources in Kenya isn't enough to provide coherence. As Figure 1 indicates, though, these texts can be made coherent by answering a question in the latter paragraph that is raised in the former.

There are several questions that are raised in the paragraph on Kenya's population problem. We can group these according to the CAC links. These questions include:

- **Specifics**: How many people are there in Kenya? Which resources in Kenya are limited?
- **Opportunities**: What is being done to improve the situation in Kenya?
- **Previous Events and Causes**: Why is the population growing so quickly?
- **Later Events and Results**: How does the population problem affect the people in Kenya?
- **Analogies**: What other problems does Kenya face?

We can now see how the texts can be linked in a coherent way; the second paragraph answers a question raised in the first: Which resources in Kenya are limited? Figure 1 shows this linkage. Although these two texts are not contiguous in the original article, linking them

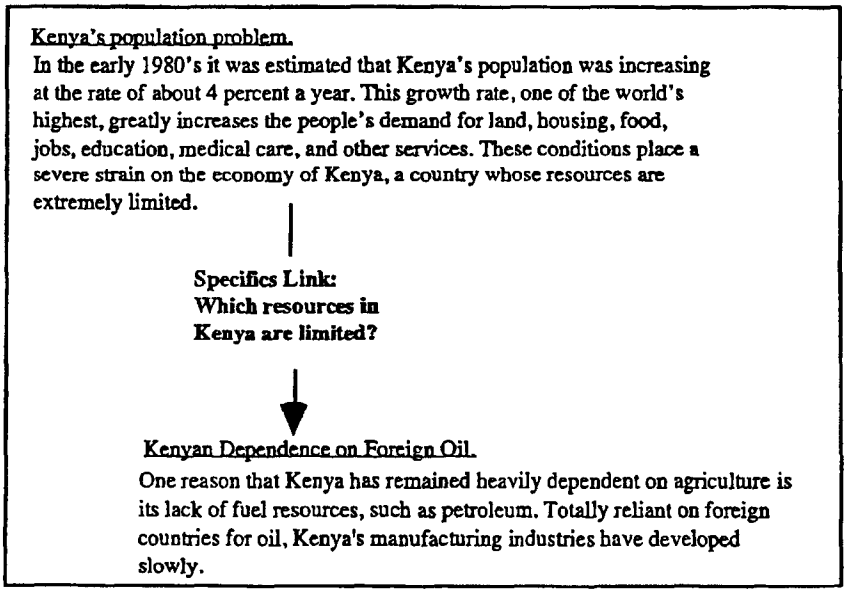

Figure 1: Coherently linked text about Kenya

through the specifying question lends them a coherence.

2.2 AskWrite provides a technology for delivering ASK systems

Linking texts by categorizing the questions they raise and the questions they answer provides the theoretical underpinnings for developing ASK systems [Ferguson, et al. 1992]. AskWrite [Schank and Osgood 1993] is one system for developing and delivering ASK Systems. Figure 2 shows an AskWrite screen for viewing the text about Kenya's population and the questions it raises. Clicking on one of the questions raised brings the user of the system to a text that answers the question.

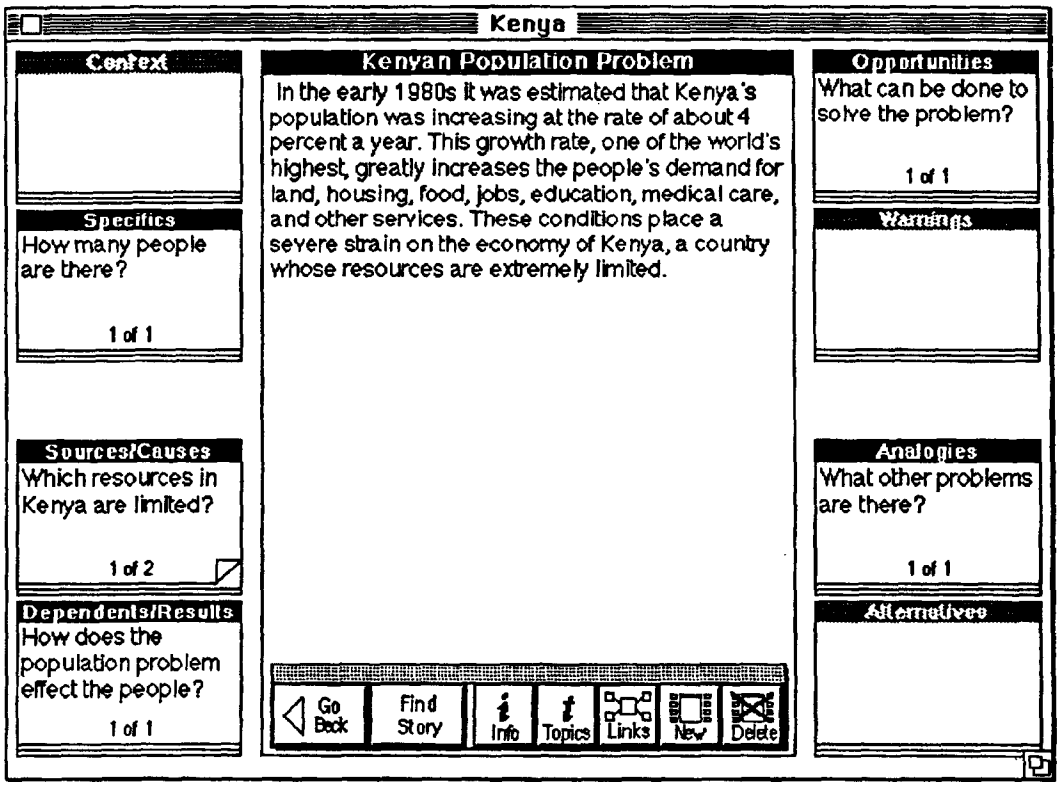

Figure 2: ASK System for Kenya

2.3 The partial representation of texts

We are using the text of Compton's MultiMedia Encyclopedia as our source. This encyclopedia has approximately 5000 articles, literally on every subject from A to Z. One approach to creating domain theories to build hypertext systems out of this encyclopedia would be to represent as much as possible of the underlying knowledge. Unfortunately, this is beyond the state of the art in knowledge representation—and we would like to develop practical tools today, without having to wait until large scale knowledge-based systems (exemplfied by the CYC project [Lenat and Guha 1990]) are available.

Instead, we prefer to use partial representations of the texts we have, leaving most of the intelligence of the system remaining in the texts and their readers, not in the domain theory representation. Our task is not to create representations of everything; our task is to create representations that assist in the creation of links among texts. The theory of CAC links

provides a minimalist representation for the presentation of texts in a coherent way; our domain theories will be only as elaborate as is needed to create these links. We turn to a discussion of the types of representations we will need to do this.

2.4 Computer-assisted indexing requires some domain-specific knowledge

As we have said, even our moderately-sized encyclopedia has some 5000 articles in it. Potentially, any article can be linked to any other article. In the worst case, linking an article requires checking every other article to determine whether a link can be made between them, a factorially difficult task. Working with subsections of articles makes the task even more difficult. The difficulty of the task makes the cross-linking prone to errors, especially the error of not creating a link when one is appropriate. Osgood and Bareiss [1993] call this the *indexer saturation problem*: the inability of a person to manage the task of linking up large numbers of texts. (Their experience shows the limit is about 100 stories).

What is wanted, then, is a way to automate the creation of links. We would like the person who is developing the knowledge-based hypertext, the *indexer*, to focus on each article, marking up, or *indexing*, each article in such a way that the cross-links are created automatically—or at least that good suggestions for cross-links are made automatically.

Let's consider the example of the two texts on Kenya referred to above. We can create a link between the two texts because we understand that Kenya is a country, countries have resources, and the first paragraph claims Kenya has a lack of resources. This leads us to ask what sorts of resources Kenya lacks. Noticing that the second paragraph claims Kenya has a lack of oil resources completes the link. First, we understand a great deal about the domains in question: what it's like to be a country, for instance, and what resources are. Second, we recognize specific instances of this knowledge in the text: Kenya lacks resources says the first paragraph; and Kenya lacks oil resources says the second. Third, we can generate a link between the two texts, because there is a rational connection between them: the second paragraph answers a specific question raised in the first.

3. Creating hypertext links requires four stages

In order to achieve the goal of computer-assisted linking, the hypertext builder's workbench also requires internal representations of the types of things that are referred to in the text. The process of linking text in our system is accomplished in four stages:

1. Domain Analysis, in which general partial representations are created,
2. Knowledge Recognition, in which the indexer examines the text and identifies references to specific concepts, aided by the system's natural language understanding system,
3. Link Generation, in which the workbench creates links between texts, and
4. Conceptual Proofreading, in which the indexer decides which of the links generated by the system should be kept, deleted or modified.

We now describe these stages in more detail.

3.1 Domain Analysis

The product of domain analysis is a hierarchy of conceptual representations important to understanding the domain as well as a set of interesting relationships that occur between these concepts. For example, the domain of geopolitical entities such as Kenya requires representations of such things as:

political entities: city, country, county, town, village, capital, district,...

forms of government: democracy, republic, monarchy, dictatorship,...

activities: engaging in war, engaging in trade, manufacturing goods,...

people groups: religious groups, political groups, racial groups, tribes,...

events: election, rebellion, war, battle, natural disaster,...

etc.

This domain analysis suggests what kinds of relations are likely to be found among domain concepts. For example:

CAPITAL-OF-COUNTRY :relation CAPITAL :object COUNTRY :identity CITY

the capital of a country is a city

GOVERNMENT-OF-COUNTRY :object COUNTRY :identity GOVERNMENT-TYPE

the government of a country coincides with some government type

LACK-OF-RESOURCES :object RESOURCES :scene COUNTRY

A country can lack resources

etc.

The combination of domain representations and the relations among them defines a virtual abstract hypertext system because each of the representations implies a set of conceptual links and thus possible hypertextual links.

For example the concept LACK-OF-RESOURCES is a potential answer to the following questions:

Specifics: "What can a COUNTRY lack?" (linking from a reference to COUNTRY)

Specifics: "What can lack RESOURCES?" (linking from a reference to RESOURCES)

3.2 Knowledge Recognition

The process of Knowledge Recognition takes representations from the domain analysis phase and instantiates them as they are found to be relevant to a piece of input text. Instantiation involves not only recognizing the applicability of a general conceptual representation, but also noting how the particular roles of the representation are filled. For example, in the text on population growth in Kenya, several general concepts are referenced.

> In the early 1980's it was estimated that Kenya's population was increasing at the rate of about 4 percent a year. This growth rate, one of the world's highest, greatly increases the people's demand for land, housing, food, jobs, education, medical care, and other services. These conditions place a severe strain on the economy of Kenya, a country whose resources are extremely limited.

The domain concepts referenced include:

General Concepts: ECONOMY, EDUCATION, FOOD, HOUSING, INCREASED-DEMAND, JOBS, MEDICAL-CARE, POPULATION, POPULATION-GROWTH, POPULATION-GROWTH-CAUSES-INCREASED-DEMAND, RESOURCE, LACK-OF-RESOURCES, etc.

These general concepts are instantiated by recognizing relationships among concepts and specifying the attribute values of the general concepts. Specific concepts in the above text include:

Specific Concepts: KENYA, KENYA'S-ECONOMY, POPULATION-GROWTH-IN-KENYA, POPULATION-GROWTH-IN-KENYA-CAUSES-INCREASED-DEMAND-FOR-JOBS-IN-KENYA, KENYA-LACKS-OIL-RESOURCES, etc.[2]

The combination of a specific concept and a pointer to the text forms an *index* for the text, out of which links between texts can be built. Parsing techniques—discussed in section 4—are used to recognize the constituents of these particular concepts and to assemble the conceptual structures.

3.3 Link Generation

Once a domain concept is recognized in a portion of text, it is compared to other references to the concept from other portions of text. A reference to a general concept differs from other references to the general concept with regard to the concepts that fill its attribute slots.

There are three possible results of a comparison between two concepts:

- The concepts are equivalent
- A more general concept subsumes a more specific concept
- The concepts are mismatched.

For example, consider the generalized concept: FISHING-INDUSTRY. A text that references this concept and specifies no contextual attributes asserts relations about the fishing industry in general. Another text can reference the concept and specify the attribute of `:scene` to discuss the fishing industry in Africa. The first reference subsumes the second because the value of the attribute `:scene` is less specific. A third reference to the concept might specify the `:scene` as North America and thus would be subsumed by the first and mismatch the second.

Generalized linking strategies can sometimes create links between concepts that do not exactly match. So for example concept references with mismatched `:identity` attributes can often be linked as *Alternatives* while concepts with mismatched `:scene` attributes can be linked as *Analogies*.

In addition to these syntactically derived links the system suggests links derived from the domain analysis. Thus the general concept LACK-OF-RESOURCES with the implicit links:

Specifics: "What can a COUNTRY lack?" (linking from a reference to COUNTRY)

Specifics: "What can lack RESOURCES?" (linking from a reference to RESOURCES)

---

[2]We use these longish, hyphenated names as pointers to the underlying representations. KENYA-LACKS-OIL-RESOURCES, for example, is the structured representation:

KENYA-LACKS-OIL-RESOURCES `:object` OIL-RESOURCES `:scene` KENYA

where OIL-RESOURCES and KENYA are themselves pointers to structured representations, and KENYA-LACKS-OIL-RESOURCES is a specific instance of LACK-OF-RESOURCES which is a kind of LIMITED-RESOURCE, etc. See Figure 3.

is realized as a concrete link when the system finds text that discusses KENYA-LACKS-OIL-RESOURCES.

3.4 Conceptual Proofreading

The human indexer remains in control of the indexing process; the indexer filters and supplements the suggestions made by the system for a particular portion of text. Suggested links can be deleted, new links made, and the type of CAC link suggested (source, analogy, opportunity, etc.) can be changed.

This four step process implies that the indexer makes two passes through the text base: first to identify the specific concepts referenced in the texts; then, to filter and supplement the links suggested by the computer.

4. Parsing-assisted knowledge recognition

The task of creating indices for text belongs to the human indexer; however, we want to provide as many tools to automate this process as possible. The task is one of associating text with a conceptual representation. This task is that of *parsing* or natural language understanding (See Figure 3).

Given the task of associating text to a conceptual representation, what desiderata are there for a parser? Here are three:

- The parser should *intimately integrate* with the conceptual representation.
- The parser should ignore information in the text that is irrelevant to the goals of the system using the parser.
- It should be easy to add new ways to associate text to conceptual representations.

We will justify these desiderata, exploring how the parsing technology Direct Memory Access Parsing [Martin 1989; Riesbeck and Martin 1986] meets them.

4.1 Parsers should be intimately integrated with representations

The goal for the use a parser is to connect some text to some conceptual representation; to meet this goal, the parser should be intimately integrated with the representational system. By *intimately integrated* we mean that the parser should map as directly as possible to the representation. The grammar of the parser should be in the same language used to describe the internal representations. The parser should have access to the same memory search and inferential capabilities of the internal representations. There shouldn't be special representations and processing for the parser unless this is shown to be necessary.

If the parser can piggy-back on the inferential capabilities and representational power of the conceptual memory, we simplify the addition and maintenance of a parser module. Furthermore, having achieved this piggy-backing, we have some hope that changes to the conceptual memory will not negatively affect the process of parsing. Rather, we can improve the coverage of our parser by simply expanding the scope of representations.

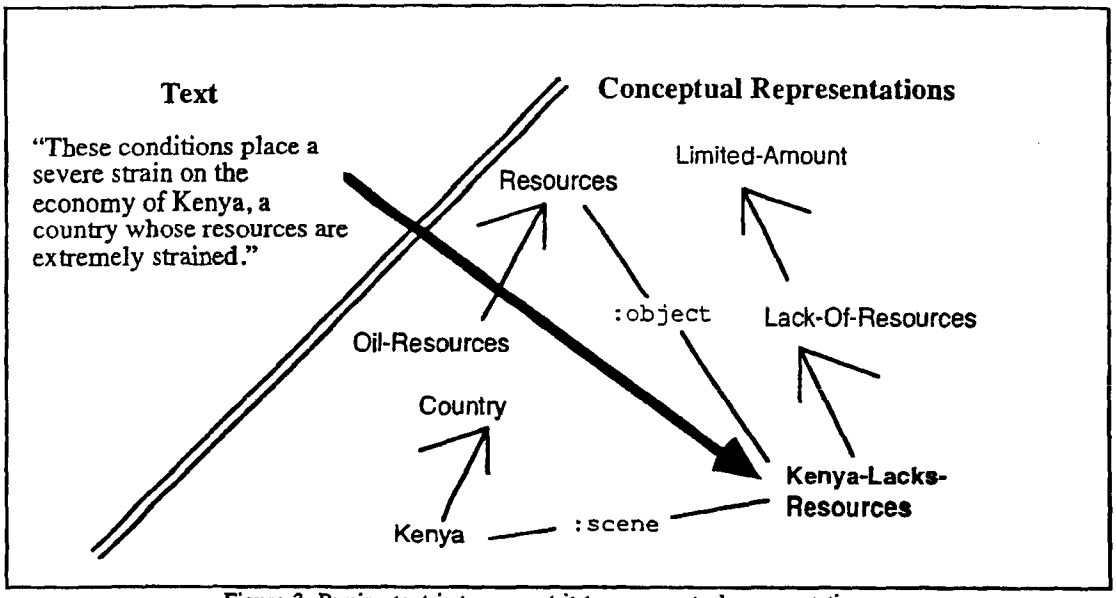

Figure 3. Parsing text is to connect it to a conceptual representation

4.2 Parsers should ignore irrelevant information

On the other hand, we would like our parsers to ignore irrelevant information. Most parsing systems have their origins in syntactic theories of language analysis [e.g., Marcus 1980; Winograd 1983; Woods 1970]. The goal of many of these systems is the syntactic analysis itself: a deeper understanding of natural language syntax. This goal has often been taken over as a precondition for parsing to more semantic representations. But it is not obvious why this should be so. Although some syntactic knowledge will be required by our parsers, the syntactic knowledge we add should be in the service of parsing to the conceptual memory, not an end it itself. Therefore, we seek a parsing technology that allows us to represent just the right amount of linguistic knowledge to get the job done.

4.3 It should be easy to add new linguistic knowledge

The point of adding parsing technology to an hypertext builder's workbench is to make the job of the human indexer easier. If adding the linguistic knowledge to a system is more difficult to do than creating the indices for a text directly, then it makes no sense to do so. Therefore, it must be easy to add new linguistic knowledge.

4.4 DMAP does the right thing

Direct Memory Access Parsing (DMAP) meets these desiderata. DMAP was designed for intimate integration with a conceptual representation. DMAP works by attaching its linguistic knowledge, in the form of *phrasal patterns*[3] directly to representations in conceptual memory. For example, the simple phrasal pattern "Kenya" points directly to the internal

[3]The details of the DMAP data structures and algorithms can be found in Appendix A.

representation of KENYA. Whenever "Kenya" is seen in the text, the internal representation is referenced.[4]

DMAP also uses the representational language of the conceptual memory to express its linguistic knowledge. For example, the phrasal pattern "`:identity` is the capital of `:object`" can be attached to CAPITAL-OF-COUNTRY[5]. To DMAP, this is a prediction that the `:identity` (which is some CITY) of CAPITAL-OF-COUNTRY will be seen, then the words "is the capital of," followed by the `:object` of CAPITAL-OF-COUNTRY (some COUNTRY). The exact details as to how this works are described in the Appendix; the important point is that DMAP is using the same language for its grammar that the conceptual memory uses for its representations.

Further, DMAP uses the same memory search and inferential capabilities for its own processing as are used in the conceptual memory. For example, processing "Nairobi is the capital of Kenya" causes a memory search for a CAPITAL-OF-COUNTRY which has an `:identity` of NAIROBI and an `:object` of KENYA. (Again, the details are in the Appendix). There is nothing unusual about this search through the abstraction hierarchy; it is a typical memory process.

DMAP provides the flexibility to represent as information for the parser just what is needed to create the connection to conceptual memory. For our needs, for example, information about Subject/Verb agreement (to take a simple case) is not needed; DMAP doesn't require us to represent it. Finally, it is easy for the indexer (or the original system builders) to associate new phrasal patterns to representations, for the patterns can be directly connected to the underlying representations. For example, the phrasal pattern "`:identity` is the capital of `:object`" can be attached directly to CAPITAL-OF-COUNTRY.

DMAP provides the parsing technology needed to suggest indices to the indexer. We now move to provide an example of indexing in action.

5. An extended example

In the following section we will step through the processes involved in creating the **Specifics** link between the following two sections of text from Compton's MultiMedia Encyclopedia which we have labelled Kenya and Nairobi. In this discussion we will focus on the Domain Analysis and Knowledge Recognition Phases. The following two sections of text are disjoint in the original source, but we would like to link them in our knowledge-based hypertext by the **Specifics** link "What is the capital of Kenya?"

> Kenya. A republic of Africa, Kenya is located on the equator on the continent's east coast. The country is well known for its scenic beauty and varied wildlife. Although only about 20 percent of the land is suitable for cultivation, the majority of Kenyans are farmers who produce crops mainly for their own needs. Coffee and tea, grown for export on large plantations and on small farms, together with tourism are Kenya's most important sources of foreign exchange—money used to buy foreign goods. The nation imports all

---

[4]What it means to 'reference' a representation is dependent on the goals of the system using DMAP. This is discussed further in the Appendix, and an example is given in the next section.

[5]As a reminder, the internal representation for CAPITAL-OF-COUNTRY is

CAPITAL-OF-COUNTRY
:relation CAPITAL :object COUNTRY :identity CITY.

of its petroleum and most manufactured products. Kenya is a poor country by comparison with industrialized countries such as those of Western Europe.

Nairobi. Nairobi, the capital city of Kenya, is located on the railway line at the junction between the lowlands and the highlands. More than 60 percent of Kenya's salaried workers live in the city, which dominates the nation's economy. It is an important commercial center and many foreign firms base their east African operations there. Most government employees also work in Nairobi.

5.1 Domain Analysis

The two sections of text we wish to join together by a **Specifics** link are linked by the concepts that are referenced in the text. The relevant concepts are Kenya and Nairobi as the capital of Kenya. Recognition of these concepts depends on the ability of the system to recognize the constituents KENYA, NAIROBI, CAPITAL, and CAPITAL-OF-COUNTRY. During the Domain Analysis phase these concepts are added to memory and related to their appropriate abstractions; e.g. KENYA isa COUNTRY and NAIROBI isa CITY.

After laying out the important concepts of the domain, DMAP phrasal patterns are associated with each concept so that DMAP can recognize instances of these important concepts in each of the text segments. For example, we can associate the following representations and phrasal patterns:

KENYA —» "Kenya"

NAIROBI —» "Nairobi"

CAPITAL —» "capital"

The last step in domain analysis is to note which interesting relations are expected among the domain concepts. In the case of our example link we predict that texts may reference a specific instance of the concept CAPITAL-OF-COUNTRY, represented as:

```
CAPITAL-OF-COUNTRY
      :relation CAPITAL
      :object   COUNTRY
      :identity CITY
```

We attach DMAP phrasal patterns to allow the system to recognize this relation in the texts:

```
CAPITAL-OF-COUNTRY —» ":object's capital is :identity"
                      ":object's capital city is :identity"
                      "::identity is the capital of :object"
                      ":identity, the capital city of :object"
```

5.2 Knowledge Recognition

In the knowledge recognition phase the indexer makes a first pass through the text material that is to be linked. The DMAP algorithm is used to identify important concepts and relations referenced in the text. These concepts are made available to the indexer who can incorporate

them as indices if appropriate. When indexing the story Nairobi DMAP reads "Kenya" and recognizes a potential reference to KENYA and reads "Nairobi" and recognizes a potential reference to NAIROBI. Furthermore, DMAP reads "Nairobi, the capital city of Kenya" and recognizes the relation CAPITAL-OF-COUNTRY where the `:object` is KENYA and the `:identity` is NAIROBI.

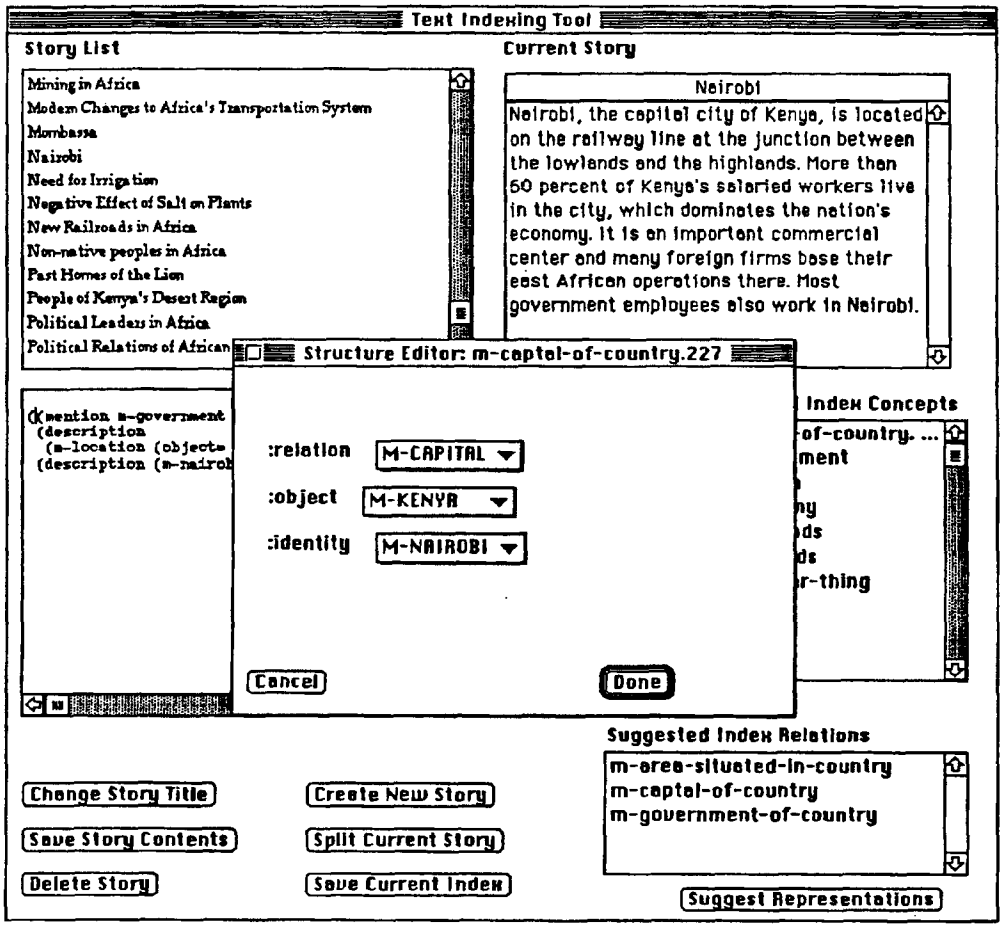

Figure 4. Text Indexing Tool of the Knowledge-based Hypertext Workbench

The indexer chooses this relation as an appropriate index to the story and thus labels it as an appropriate answer to the question "What is the capital of Kenya?" Similarly when indexing the text of the story titled Kenya the concept KENYA is offered as a potential index and becomes the basis for asking the above question. After the knowledge recognition phase the two stories are respectively indexed in part by these concepts:

Nairobi: Kenya's Population Problem:

CAPITAL-OF-COUNTRY
:relation CAPITAL
:object KENYA
:identity NAIROBI

KENYA
:isa COUNTRY

It is important to note that even if DMAP had not been able to recognize directly an instance of the relation CAPITAL-OF-COUNTRY the relation would still have been suggested to the indexer by following backpointers from the concepts recognized (KENYA, CAPITAL, and NAIROBI) and their abstractions to the relation and noting that all of the constituent elements are referenced by the text. The indexer could then confirm this suggestion by selecting it from the set of hypothesized relations. Figure 4 shows the Text Indexing tool of our knowledge-based hypertext builder's workbench suggesting indices for the Nairobi text.

What remains is to link the two texts by using these concepts.

5.3 Link Generation & Conceptual Proofreading

The link generation phase consists of the comparison of sibling concepts and the realization of abstract links defined in the domain analysis phase. The example link above is generated as an instantiation of the abstract link between a text describing a country and text identifying the capital of the country. This link is suggested to the indexer who can either ignore it or accept it and potentially edit the question text.

6. Summary: Computer assisted indexing through natural language understanding

The amount of information available in electronic form is increasing rapidly. Much of this information will come to us as electronic editions of works previously available in print. We want to take this old wine and put it into new wineskins: to remake what it means to read a text, to look something up in an encyclopedia. To automate some of the task of preparing and making use of this information we will have to give our computers some of the knowledge we already have.

We have presented a scheme for using natural language understanding techniques, along with domain knowledge and conversational categories, to automate the cross-linking of encyclopedic text. We have sought to demonstrate the feasibility of this scheme through explication and an extended example[6]. Next steps include further work on more strategies for using the domain knowledge representations for cross-linking and refining our underlying representation language. From there, of course, lie the tasks of creating representations and creating the cross-links. Our goal is to create 'on-demand' ASK systems on any topic represented in our system.

Creating and presenting effective knowledge-based hypertexts out of the large, loosely organized text of an encyclopedia requires knowledge. Representing some of this knowledge in our computer systems will help us create links between texts. A representation scheme can also be leveraged by providing an integrated natural language understanding system. This parsing technology can suggest the indices out of which we can build cross-links among texts. The combination we have described—ASK system technology, automatic generation of hypertext links from partial representations, and Direct Memory Access Parsing—will provide

---

[6]The current state of our project: We have done the domain analysis for geopolitical entities. We have built a Text Indexing tool which creates indices with the help of DMAP, and used it for the texts associated with Kenya—some thirty subtexts in all. We also have a Text Linking tool which suggests appropriate CAC-based links among the texts.

powerful new ways of using and interacting with the constantly expanding amounts of electronically available information.

Acknowledgments

The authors wish to thank Ray Bareiss, Chip Cleary and Chris Riesbeck for helpful comments on earlier drafts of this paper.

Appendix: The DMAP algorithm

As we have indicated, DMAP requires some elaborated conceptual representation, such as a frame-based memory or a database of propositions and means for searching through those representations, which we will call *concepts*. DMAP requires some basic abilities for memory search. Two of these are finding all of the 'parents' or superclasses of a concept stored in memory and finding the value of an attribute for a concept (either directly or through inheritance from the concept's parents).

DMAP is an *index-based* algorithm. For each concept that we wish to reference, we attach one or more indices. Each of these indices will be a *phrasal pattern*, which is a (totally ordered) sequence of items. After all of the items in the phrasal pattern have been referenced the concept (the *base concept* for that phrasal pattern) has been referenced.

In the following discussion, we will put phrasal patterns in double quotation marks. References to an attribute of a concept will be prefixed with a colon and printed in a `typewriter` font. References to concepts we will put in SMALL-CAPS. For example, we might attach the phrasal pattern "`:identity` is the capital of `:object`" to the concept CAPITAL-OF-COUNTRY. This phrasal pattern contains six items, in which `:identity` and `:object` refer the *identity* and *object* attributes of the concept CAPITAL-OF-COUNTRY, and the other words refer to themselves.

For example, we might attach the phrasal pattern "United States" to the system's internal conceptual representation of the United States. When the system has referenced the words 'United' and 'States,' then it references the conceptual UNITED–STATES. Words reference themselves, so processing 'United' is to reference the word itself.

DMAP is *prediction-based*. That is, DMAP makes predictions about what it expects to see. For example the system might see the word 'United,' and hence a possible reference to UNITED–STATES, UNITED-KINGDOM, or UNITED-ARAB-EMIRATES. Having seen 'United,' DMAP then predicts it could see 'States,' 'Kingdom' or 'Arab.' The prediction must also carry along with it where the prediction started in the text stream, because there may be other predictions looking for a reference to the prediction's base concept but at a different point in the text stream[7]. Predictions are stored in a table, which is keyed on the first item in the prediction's phrasal pattern and the starting point. We can distinguish between *anytime predictions*, which are predictions that might fire at any time, and *dynamic predictions*, which DMAP creates dynamically out of anytime predictions and other dynamic predictions. This process of creating new predictions from old we call *advancing* the old prediction.

---

[7]Consider, for example, recognizing an instance of a KISSING-EVENT, with attributes `:actor` and `:object`, with a selectional restriction that both the actor and object be HUMAN. If the concept sequence is "`:actor` kissed `:object`" we want different predictions to be advanced during the parsing of "Milton Friedman kissed Julia Roberts," even though MILTON-FRIEDMAN and JULIA-ROBERTS are both instances of HUMAN. The difference is where a predicted reference will occur.

Table 2
Example Prediction Table, having read "United"

Key		Prediction			
Item	Start	Base Concept	Start	Next	Sequence to be seen
		Anytime Predictions			
United	ANY-TIME	UNITED-STATES	.....	.....	"United States"
		UNITED-KINGDOM	.....	.....	"United Kingdom"
		UNITED-ARAB-EMIRATES	.....	.....	"United Arab Emirates"
U	ANY-TIME	UNITED-STATES	.....	.....	"U S A"
		UNITED-KINGDOM	.....	.....	"U K"
		UNITED-ARAB-EMIRATES	.....	.....	"U A E"
Great	ANY-TIME	UNITED-KINGDOM	.....	.....	"Great Britain"
London	ANY-TIME	LONDON	.....	.....	"London"
Abu	ANY-TIME	ABU-DHABI	.....	.....	"Abu Dhabi"
CITY	ANY-TIME	CAPITAL-OF-COUNTRY	.....	.....	":identity is the capital of :object"
		Dynamic Predictions			
States	0	UNITED-STATES	0	1	"States"
Kingdom	0	UNITED-KINGDOM	0	1	"Kingdom"
Arab	0	UNITED-ARAB-EMIRATES	0	1	"Arab Emirates"

For example, there is an anytime prediction for 'United' that will advance a prediction for the base concept UNITED–STATES. (We set the start position for anytime predictions to ANY-TIME to indicate they are predicted at any time). Consider a conceptual memory containing three concepts (among others), UNITED–STATES, UNITED-KINGDOM, and UNITED-ARAB-EMIRATES. UNITED-STATES has two phrasal patterns associated with it: "United States" and "U S A." UNITED-KINGDOM and UNITED-ARAB-EMIRATES, similarly, have phrasal patterns associated with them. Table 2 shows the state of the prediction table after reading 'United.'

We have said that words reference themselves; we have not said what it means to reference an attribute specifier in a phrasal pattern. For example, we described attaching the phrasal pattern "`:identity` is the capital of `:object`" to the concept CAPITAL-OF-COUNTRY. Recognizing an attribute in a phrasal pattern means recognizing the *value* of that attribute. The *identity* of a CAPITAL-OF-COUNTRY must be a city; finding the target of `:identity` means recognizing a reference to some CITY. DMAP would create an anytime prediction for a CITY. Having seen a reference to a city, then DMAP would create a dynamic prediction for 'is.' In this we see the need for one of the memory searching basics we described earlier, that is, the need to find the value of attribute references for concepts. Table 3 shows the state of the prediction table after reading "Abu Dhabi is".

One question we have left unanswered is what it means to *reference* a concept. One reason we have left it unanswered is that it will depend a great deal on the functional requirements of the system we are creating. It may be enough just to print a message saying we have referenced something; it may invoke complicated additional memory processing of the item. DMAP provides a way of defining functional *call-backs* that are run when a concept is referenced; these call-backs may be inherited from the concept's parents. In the application we are considering in this paper, DMAP will assist the *knowledge recognition task* we discussed earlier. That is, it suggests references to specific knowledge representations referenced in a text.

Having described something of the basics for the data structures for DMAP, we now turn to a description of the DMAP algorithms. We must stress that this is a basic version of the

Table 3
Example Prediction Table, having read "Abu Dhabi is"

Key		Prediction			
Item	Start	Base Concept	Start	Next	Sequence to be seen
		Anytime Predictions			
United	ANY-TIME	UNITED-STATES	.....	.....	"United States"
		UNITED-KINGDOM	.....	.....	"United Kingdom"
		UNITED-ARAB-EMIRATES	.....	.....	"United Arab Emirates"
U	ANY-TIME	UNITED-STATES	.....	.....	"U S A"
		UNITED-KINGDOM	.....	.....	"U K"
		UNITED-ARAB-EMIRATES	.....	.....	"U A E"
Great	ANY-TIME	UNITED-KINGDOM	.....	.....	"Great Britain"
London	ANY-TIME	LONDON	.....	.....	"London"
Abu	ANY-TIME	ABU-DHABI	.....	.....	"Abu Dhabi"
CITY	ANY-TIME	CAPITAL-OF-COUNTRY	.....	.....	":identity is the capital of :object"
		Dynamic Predictions			
Dhabi	0	ABU-DHABI	0	1	"Dhabi"
is	1	CAPITAL-OF-COUNTRY	0	2	"is the capital of :object"
the	2	CAPITAL-OF-COUNTRY	0	3	"the capital of :object"

algorithm, which might be complicated or made more efficient by additional means of memory search. We informally describe four procedures, **parse**, **reference**, **target-for**, and **advance-prediction**.

To **Reference** an item means to advance all predictions looking for an item, and run all call-back functions on the item. More accurately, to reference an item is to advance the predictions on the item *and the item's abstractions*. Because predictions need to know where the reference to the item started, we need to inform the reference function of the start position. Pseudo-code for **reference** looks like this:

```
procedure reference (item start end)
begin procedure
   for each abstraction in all the abstractions of item do
      for each prediction looking for abstraction do
         call advance-prediction(prediction, item, start, end)
      end for
      for each call-back for abstraction do
         call the call-back with item, start, end
      end for
   end for
end procedure
```

There are two cases to consider when **advancing** a prediction. If the sequence yet to be seen is empty, then we can reference the base concept of the prediction. Otherwise, we want to create a new dynamic prediction for the next target in the sequence starting at the next position.

```
procedure advance-prediction (prediction item start end)
begin procedure
```

```
if null(phrasal-pattern.prediction)
   then call reference(base.prediction, start.prediction,
                       end.prediction)
   else create a dynamic prediction with the:
      base.prediction,
      a sequence created with all but the first item in
      prediction.sequence,
      a start position one more than prediction.start
      and keyed on:
      call get-target with prediction.base and the first item
      in prediction.sequence
end procedure
```

To **get the target** of a concept and a item is trivial: if the item is a word, return the item; if the item is an attribute specifier, return the value of the attribute in the concept.

```
function get-target (concept item)
begin function
   case
      item is a word: return item
      item is a attribute specifier: return attribute value
      of item in concept
   end case
end function
```

All of this makes the **parse** routine trivial as well. To parse a sentence is just to reference each word in the sentence.

```
procedure parse (sentence)
set currentposition to 1
begin procedure
   for word in sentence do
      advance a position counter
      call reference (word, currentposition, currentposition)
      increment the currentposition
   end for
end procedure
```

The data structures and algorithms we have described are relatively simple to implement and efficient to use. By taking advantage of the search and inferential capabilities of the conceptual memory (which themselves are straightforward) we are able to create an elegant but powerful parser.

References

Bareiss, Ray and Richard Osgood. "Applying AI models to the design of exploratory hypermedia systems." *Hypertext '93*, 1993, to appear.

Comptons on "Kenya." Compton's MultiMedia Encyclopedia: Compton's Learning Company, Britannica Software, Inc. San Francisco, California, 1991.

Conklin, E. "Hypertext: An introduction and survey." *IEEE Computer* 2 (1987): 17-41.

Ferguson, William, Ray Bareiss, Lawrence Birnbaum and Richard Osgood. "ASK Systems: An approach to the realization of story-based teachers." *The Journal of the Learning Sciences* 1 (2 1992): 95–134.

Lenat, Douglas B. and R.V. Guha. *Building Large Knowledge-Based Systems: Representation and Inference in the CYC Project*. Reading, Massachusetts: Addison-Wesley, 1990.

Marcus, Mitchell P. *A Theory of Syntactic Recognition for Natual Language*. Cambridge, MA: The MIT Press, 1980.

Martin, Charles E. "Case-Based Parsing" and "Micro DMAP." *Inside Case-Based Reasoning*, ed. Christopher K. Riesbeck and Roger C. Schank. Hillsdale, N.J.: Lawrence Erlbaum Associates, 1989.

Osgood, Richard and Ray Bareiss. "Automated index generation for constructing large-scale conversational hypermedia systems." *Eleventh National Conference on Artificial Intelligence*, 1993, 309-314.

Riesbeck, Christopher K. and Charles E. Martin. "Direct Memory Access Parsing." *Experience, Memory and Reasoning*, ed. Janet Kolodner and Christopher K. Riesbeck. Hillsdale, N.J.: Lawrence Erlbaum Associates, 1986.

Schank, Roger C. "Rules and topics in conversation." *Cognitive Science* 1 (1977): 421-441.

Schank, Roger C. and Richard E. Osgood. *The communications story*. The Institute for the Learning Sciences, 1993. 37.

Spiro, Rand J. and Jihn-Chang Jehng. "Cognitive flexibility and hypertext: Theory and technology for the nonlinear and multidimensional traversal of complex subject matter." *Cognition, Education and Multimedia: Exploring Ideas in High Technology*, ed. Don Nix and Rand Spiro. Hillsdale, NJ: Lawrence Erlbaum Associates, 1990.

Winograd, Terry. *Language as a Cognitive Process*. Vol. 1: Syntax. Reading, MA: Addison-Wesley, 1983.

Woods, W. A. "Transition network grammars for natural language analysis." *Communications of the ACM* 13 (10 1970): 591-606.