Conceptual Information Retrieval using RUBRIC

Richard M. Tong
Lee A. Appelbaum, Victor N. Askman, James F. Cunningham

Advanced Decision Systems
201 San Antonio Circle, Suite 286
Mountain View, CA 94040.

1. INTRODUCTION

The problem of retrieving information from large full-text databases is unbiquitous and increasing in importance as low-cost optical storage media become available. In many cases, simple keyword-based retrieval systems have shown themselves inadequate for the task and a number of more or less sophisticated alternatives have been proposed. Of particular interest are those that derive from efforts in natural language understanding and which advocate a conceptually oriented approach (Schank et al., 1981; DeJong, 1982; Kolodner, 1983). These efforts emphasize semantically driven text parsing with the goal of understanding only so much of the text as is necessary to perform satisfactory retrieval.

The problem with any natural language approach is that the process is generally not well understood and is computationally very expensive. So in this paper we describe a more tractable approach that is based on our RUBRIC technology (Tong et al., 1986). We still give the idea of a retrieval concept a central place but rather than attempt to parse the text of the document we search for textual evidence (in the form of word patterns) that indicates that a concept is being discussed. The lack of certainty associated with this procedure is captured using a numerical uncertainty representation and is propagated through the reasoning using an appropriate calculus. Another important feature of the RUBRIC approach is that the knowledge required to describe retrieval concepts of interest is not pre-defined by the system developers, but rather is considered to be expert knowledge that the user will supply. Thus in RUBRIC there is the possibility for constructing highly customized knowledge structures that reflect the specific needs of individual users.

Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the ACM copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Association for Computing Machinery. To copy otherwise, or to republish, requires a fee and/or specific permission.

© 1987 ACM 089791-232-2/87/0006/0247—75¢

The paper is divided into three main sections. In the first we present some details of the knowledge representation used in RUBRIC. In the second we are concerned with the use of these knowledge structures in actual retrieval. Then finally, we present an example from our familiar domain of terrorism and show how these new extended structures are related to the structures presented in our earlier papers.

2. SEMANTIC STRUCTURES

The central idea behind our semantic structures approach is that a retrieval concept can be specified in terms of its constituent parts, which thereby form the definition to be used during retrieval. In RUBRIC the user specifies these components by defining "attributes." To illustrate, let us suppose that the user is interested in documents about meetings. Such events are composed of a number of elements of interest to the user, for example, the basic action (i.e., the act of meeting), the people involved, the topics of discussion, and the location of the meeting. The purpose of attributes, therefore, is to enable the user to define what it is about the main concept that is relevant for retrieval.

The ATTRIBUTE rule is used to capture the idea that concepts have components (or attributes), and that knowledge of these components may be used to help establish the presence of the concept itself. So for example if we take the attributes mentioned above, these would be defined in RUBRIC as:

(ATTRIBUTE **meetings** *action*)
(ATTRIBUTE **meetings** *actors*)
(ATTRIBUTE **meetings** *topic*)
(ATTRIBUTE **meetings** *location*)

where the syntax is LISP-like and we use emboldened text to indicate concepts and italicized text to indicate attribute names.

In conjunction with ATTRIBUTE rules we also need a mechanism for specifying the value of the attributes. In RUBRIC this is achieved by a rule of type DEFINES. So for example:

(DEFINES *location*
(*OR* **moscow washington vienna geneva**))

where **moscow, washington, vienna,** and **geneva** are the concept names for various locations of interest, and *OR* denotes logical disjunction.

Another important feature of the RUBRIC knowledge representation is that it allows us to express taxonomic relationships between the retrieval concepts of interest. So for example, if we are not only interested in **meetings** in general, but also in specific types of **meetings**, then we want to be able to express this in RUBRIC. Accordingly, we provide two rule types for expressing taxonomic relationships.

The first is the SUBSET rule. This rule type allows us to express the relationship between a sub-set of a set and the set itself. For example:

(SUBSET **meetings diplomatic-meetings**)

where again the syntax here is LISP-like, and the rule expresses the idea that **diplomatic-meetings** are a subset of all **meetings**.

Similarly, the INSTANCE rule allows us to express the relationship between an element of a set and the set itself. For example:

(INSTANCE **diplomatic-meeting**
us-soviet-summit)

which expresses the idea that within the set of **diplomatic-meetings** we are specifically interested in any document about a **us-soviet-summit**.

Of course, this taxonomy can be made as complete as we wish simply by adding more rules. For example:

(INSTANCE **diplomatic-meetings**
sino-soviet-summit)
(SUBSET **meetings white-house-meetings**)
(INSTANCE **white-house-meetings**
press-conference)
(INSTANCE **white-house-meetings**
cabinet-meeting)

would extend the taxonomy to include other kinds of **diplomatic-meetings** and also create a sub-taxonomy of **white-house-meetings**.

Concept attributes and their associated values are inherited from parent concepts unless specified locally. So for example, the location attribute for **white-house-meetings** would be defined as:

(DEFINES **white-house-meetings**:*location*
white-house)

where the notation **concept**:*attribute-name* is used to distinguish variants of global attributes.

Figure 1 depicts this taxonomic and attribute information in graph form and shows our standard notation as well as some further details of this simple definition. (Solid arcs denote subset/instance links, dotted arcs denote attribute links, and attribute names are shown underlined rather than italicized.) In particular, notice that only the *action* attribute is specified at the **meetings** node, and that the primary distinguishing factor amongst the various meeting types is the specification of the *actors* values.

3. EVIDENTIAL STRUCTURES

In the preceding section we described how the semantic structure of documents of interest can be encoded in RUBRIC. We now describe how to specify the textual evidence required to determine whether the document under consideration matches the semantic structure. We begin our discussion with a description of the basic string matching operations that RUBRIC can perform, we then describe the two types of inferential rule used to propagate the uncertainty values, and then finally we describe a number of functions used to combine evidence from multiple sources.

3.1. Text Reference Expressions

We provide a text reference language (henceforth TRL) that is used to describe patterns of text that should be searched for in support of retrieval concepts. The language consists of a number of basic operators that can be applied to one or more text arguments.

The first group of operators are just RUBRIC's equivalents of the conventional logical operators. The operators *AND* and *OR* can take multiple arguments, the operator *NOT* takes a single argument. So for example:

(*OR* "KREMLIN" "MOSCOW" "RUSSIA")
(*AND* "REAGAN"
(*OR* "MOSCOW" "BEIJING"))
(*NOT* (*AND* "REAGAN" "MOSCOW"))

are all legal expression in the TRL where upper-case quoted text is the actual pattern to be matched in the body of the document.

The second group of operators are those we call distance operators. These take a pair of text arguments and return a value which represents the distance between them. The NEAR-W, NEAR-S and NEAR-P operators all return a value in the interval [0,1] which is a normalized measure of the distance in words, sentences or paragraphs between their arguments. So for example:

(NEAR-W "PRESIDENT" "REAGAN")

performs a test to see how close the words "PRESIDENT" and "REAGAN" are in the document. If they are adjacent then the value returned is 1.0. If one or both are absent then the value returned is 0.0.

The third main group of operators are those we call location operators. These take text arguments and return a value from the set {0,1}. The SENTENCE and PARAGRAPH operators test to see if their arguments occur within the same sentence or paragraph in the document; if they are then they return a value of 1.0, and a value of 0.0 otherwise. The PRECEDES operator takes two keyword arguments and tests whether one occurs before the other. The WITHIN operator takes a numerical argument followed by two keyword arguments and tests whether they are within the specified number of words of one another. The PHRASE operator takes multiple text arguments and tests whether the phrase defined by concatenating the keywords occurs within the document. Examples are:

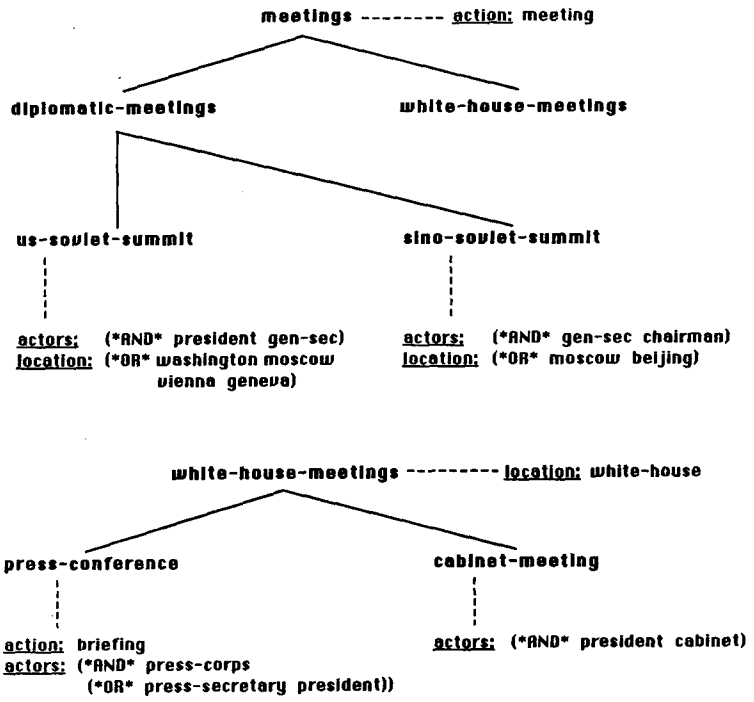

Figure 1 Semantic Structure for Meetings

(SENTENCE "GORBACHEV" "REAGAN")
(PARAGRAPH "GORBACHEV" "GHANDI")
(PRECEDES "SINO" "SOVIET")
(WITHIN 10 "GORBACHEV" "REYKJAVIK")
(PHRASE "STRATEGIC" "ARMS"
"LIMITATION" "TALKS")

Several of the operators in the TRL can also take concepts as arguments. For example, all the logical operators, *AND*, *OR* and *NOT*, can be used in this way. In addition, RUBRIC has two non-traditional operators, BEST-OF and WEIGHT-OF, which take concepts as arguments and which capture the idea that, in the first case, any one of the arguments would be appropriate so we might as well take the best, and in the second that the more arguments that are true the better. RUBRIC also has a number of experimental features for general purpose synonym handling and for concept proximity testing.

3.2. Inferential Rules

To provide the necessary links between the text expressions and the concept taxonomies, the RUBRIC knowledge representation includes two inferential rules. These are the principal carriers of the uncertainty information which is appended to the rule as an uncertainty value much as in any of the "MYCIN-like" rule-based systems. However, RUBRIC's models of uncertainty are considerably more sophisticated than that found in most systems. We can select from amongst a number of representations. These include standard probability and infinite-valued logics, various interval representations and even linguistic ones. We refer the reader to Tong and Shapiro (1985) and Tong and Appelbaum (1987) for some discussion of these details.

The EVIDENCE rule is used to link text reference expressions to concepts. It captures the notion that text expressions are used as direct evidence in determining the relevance of the document to the retrieval topic. So for example:

(EVIDENCE **moscow**
((*OR* "MOSCOW" "KREMLIN") 0.6))

where "MOSCOW" and "KREMLIN" are text strings, and the number 0.6 is the degree of relevance to be assigned to the concept **moscow** if either of the text strings "MOSCOW" or "KREMLIN" are found in the document.

The IMPLIES rule is used for indirect evidence. That is, it is used to link retrieval concepts that are not in any taxonomic relationship to one another. So for our example domain of meetings, we might have:

(IMPLIES **salt** (**us-soviet-summit** 0.7))

where **salt** is the concept name for the Strategic Arms Limitation Treaty. The number 0.7 is the degree of relevance we wish to assign to a document that describes a **us-soviet-summit**, when we are in fact interested in documents about **salt**.

3.3. Combining Functions

As described above, evidential structures are comprised of inferential rules and text reference expressions. In addition, though, we need functions that enable us to combine evidence from multiple sources. So if we have several EVIDENCE and/or IMPLIES rules for a particular concept we need to specify how each contributes to the overall weight of evidence assigned to that concept. In addition, and perhaps not so obviously, once the evidential weights have been propagated through to the semantic structures we also need to specify how attribute information is combined with subset/instance information.

The mechanism provided in RUBRIC for this combination is the COMBINE rule. Such rules have the following syntax:

(COMBINE **concept** *function-name*)

where **concept** is the concept for which we need to combine evidence, and *function-name* is the name of the actual combining function. The idea here is to allow the user to define arbitrary combining functions tailored to the special needs of the domain. We provide default combining functions, however, and in the case that the user does not specify a particular function, then one of these defaults is selected.

Within evidential structures, the default combining function is a disjunction of the evidential weights; that is, an *OR* of the evidences. Thus if any of the paths contributes a weight of 1.0, then the concept receives a weight of one; if all the paths contribute a zero weight, then the concept receives a zero weight. Intermediate conditions result in an intermediate weight for the concept. A typical functional form is:

$$v(\mathbf{c}) = \min\left[\, 1.0, \sum_i v(\mathbf{c}_i) \,\right]$$

where $v(\mathbf{c})$ denotes the resulting weight for the concept and $v(\mathbf{c}_i)$ denotes the weights from the individual IMPLIES/EVIDENCE rules.

Within semantic structures, two default combining functions are provided. For the SUBSET/INSTANCE rules we again use an *OR* function, although not the same one used for evidential combination. So we have:

$$v(\mathbf{c}) = \max\left[\, v(\mathbf{s}_i) \,\right]$$

where $v(\mathbf{s}_i)$ denotes the contribution from the individual SUBSET/INSTANCE rules. Then to combine the weights from attributes we use a WEIGHT-OF operator. That is:

$$v(\mathbf{c}) = \frac{1}{N} \sum_{i=1}^{N} v(\mathbf{a}_i)$$

where $v(\mathbf{a}_i)$ denotes the contribution from the individual ATTRIBUTE rules.

One level of customization of combining functions can be done by specifying whether attributes are either necessary, sufficient, or auxiliary in determining the value of their associated concept. So for example we might modify our attribute definitions for **us-soviet-summit** to be:

(ATTRIBUTE **us-soviet-summit** ((*NEC* *actors*) 0.6))
(ATTRIBUTE **us-soviet-summit** (*AUX* *location*))
(ATTRIBUTE **us-soviet-summit** (*AUX* *action*))

which expresses the idea that in determining the relevance of a document to a request for information about a **us-soviet-summit** it is necessary that the value of the *actors* attribute is at least 0.6 and that if it is then we can also take into account the values of the auxiliary attributes *location* and *action*. Functionally, this might look like:

if $v(actors) < 0.6$ then
  $v(\textbf{us-soviet-summit}) = 0.0$
otherwise
  $v(\textbf{us-soviet-summit}) =$
    $\min\left[\, 1.0, v(actors) + v(location) + v(action) \,\right]$

Much of our current research is directed towards a proper understanding of these evidence combination issues. We are especially interested in the internal form of the semantic structures and how these dictate the functional forms required to make them operational.

4. TERRORISM REVISITED

For illustration, let us assume that the RUBRIC user is interested in scanning the Reuters news wire for articles about terrorist events, and is sufficiently familiar with the topic and document type that a comprehensive description can be constructed. This example serves two purposes. First, it shows how a complex retrieval topic can be described in RUBRIC, and second it shows how we can interpret our earlier descriptions as approximations to this more complete structure.

We begin by recognizing that terrorist events are simply a subset of events in general. Further, events have defining attributes which we can easily specify; that is, events have actions (the central activity), actors (those performing the activity), targets (the person or thing against which the activity is directed), objects (the object or mechanism by which the activity is carried out) , and effects (the results of the activity). In Figure 2 we show the top-level semantic structure for a **terrorist-event**. Notice that the only attribute specified at this level is that of *actor*, and that we define four subsets of interest, namely **killing-event**, **bombing-event**, **encounter-event**, and **takeover-event**. Notice also that these sub-events are neither mutually exclusive nor exhaustive, they merely represent those distinctions we deem important in performing retrieval.

In Figure 3 we show the next level of semantic structure for a **killing-event**. At this level we specify the *action* and *effect* attributes, and then further distinguish between a **shooting-event** and an **assassination-event**. Notice that what separates these instances is the specification of local attributes. So what distinguishes an **assassination-event**

Figure 2 Top-Level Semantics for Terrorist Events

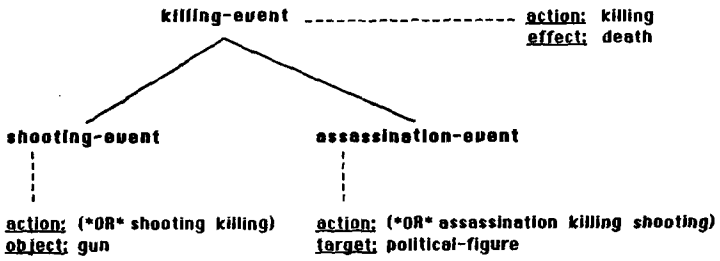

Figure 3 Semantics for Killing Events

from the general class of **killing-event** is the fact that the *target* is a **political-figure**. Notice however, that the **assassination-event** shares *action* attribute properties with both its parent and its sibling.

Examples of the supporting evidential structure for this semantic structure are shown in Figure 4 where we indicate typical EVIDENCE rules for the attribute values. Notice that we can have multiple rules for a concept and that the antecedent expressions for rules can be as complex as we wish. Of course, these rules are for illustration only and in practice we would expect to have both more rules and more complex structures. Nonetheless, these rules can be used for retrieval giving acceptable results, demonstrating that it is not necessary to have a complete vocabulary for the domain in order to do retrieval.

We do not have space here to develop the remainder of the structure, but it consists of definitions for **bombing-event**, **encounter-event**, **takeover-event** and **terrorist-actor**. Once it is in place, however, it is important to recognize that it can serve multiple purposes. It is primarily used for retrieval, of course, but can also be used to aid further knowledge acquisition and to provide semantically oriented explanations of retrieval. The kind of query that can be executed within this framework is itself very varied. For example, in addition to asking about keywords and concepts, or simple combinations of concepts and keywords, the user can ask about concepts with subset/instance constraints, about the subset/instance by itself, about particular attributes, or about concepts with modification to the preferences over attributes.

Finally some comments are in order on the relationship between the retrieval structure presented in this paper and other structures for terrorism that we have reported in earlier publications (for example, Tong et al., 1985). The main point for our discussion here is that whilst the structure we have developed (and partially described in Figures 2 through 4) may be thought of as the "true" definition of terrorist events, it is possible to generate less precise descriptions from it. We could construct many possible approximations, but an obvious one is to suppress the ATTRIBUTE, SUBSET and INSTANCE rules leaving only the EVIDENCE rules and some additional IMPLIES rules. For example the semantic structure for **killing-event** in Figure 3 might be replaced with the following evidential structure:

(IMPLIES **killing-event**
  ((*OR* **shooting-event assassination-event**) 1.0))
(IMPLIES **shooting-event**
  ((WEIGHT-OF
    (*OR* **shooting slaying**) **gun death**) 0.6))
(IMPLIES **assassination-event**
  ((WEIGHT-OF (*OR* **assassination killing shooting**)
    political-figure death) 0.7))

Of course, by doing this we loose the semantic information, but do retain the evidential structure. If this is a good approximation (i.e., the ratings given to documents are "close"

```
(EVIDENCE killing ((*OR* "KILL" "KILLING") 0.1))
(EVIDENCE killing ("MURDER" 0.4))

(EVIDENCE shooting ("SHOOT" 0.2))

(EVIDENCE assassination ((*OR* "ASSASSINATE" "ASSASSINATIONS") 0.5))

(EVIDENCE death ((*OR* "DEAD" "DEATH") 0.5))

(EVIDENCE gun ((*OR* "HANDGUN" "GUN" "RIFLE") 0.2))

(EVIDENCE political-figure ("POLITICIAN" 0.8))
(EVIDENCE political-figure ((*OR* "KING" "PRESIDENT" "AMBASSADOR"
                              (PHRASE "PRIME" "MINISTER")
                              (PHRASE "MEMBER" "PARLIAMENT")) 1.0))
```

Figure 4 Evidential Structure for Killing Events

to those given by the full retrieval structure) then we might be content to use it directly. In this sense it is a partially compiled version of the original retrieval structure. However by removing the semantics we make it difficult to extend or modify our definition of a **killing-event** should we wish to do so in the future. Notice that the first rule here is defining the subset/instance relationship between the various events, so we could actually write just one rule that would give similar results. That is:

```
(IMPLIES killing-event
  ((*OR*
   (WEIGHT-OF (*OR* shooting slaying) gun death)
   (WEIGHT-OF (*OR* assassination killing shooting)
    political-figure death)) 0.65))
```

Clearly then, this process is one of successively removing semantic information and replacing it with evidential proxies. Part of our current research effort is to understand the implications of these approximations in a variety of retrieval situations.

5. SUMMARY

In this paper we have presented an extension and re-interpretation of the retrieval techniques used in RUBRIC. We now see that the approach has strong connections with natural language processing. Indeed, we might argue that at the semantic level, RUBRIC can represent exactly the same knowledge encoded by script-based methods. Where RUBRIC differs is that it performs no syntactical analysis of text, but instead simply searches for word patterns which it takes to be partial evidence for the presence of the retrieval concept. Further, this text searching is completely prespecified in the evidential structures. Thus text for RUBRIC is the evidential foundation from which to draw conclusions about the presence of the topic specified by the query.

Of course, by ignoring the syntactical features of the text RUBRIC is sometimes not able to discern shades of meaning in text, and generally does not achieve the same level of understanding as natural language processors. This is exactly the trade-off we might expect. RUBRIC is computationally efficient and can deal with fragmentary and ungrammatical material, at the cost of introducing some uncertainty into the interpretation of the text. We believe that in a large number of important practical cases, the level of understanding that a system such as RUBRIC can provide is consistent with the needs of users. Thus in many situations the users themselves are the final analyzers of the text and so only require that the system determine whether a document has the main semantic features of interest.

The current version of RUBRIC is implemented in CommonLisp and C on a Sun Microsystems SUN-3 Workstation, and makes use of an ADS proprietary object-oriented programming environment (Cation, 1986; Cation, 1987). Within RUBRIC there are a number of modules that provide the user with facilities for creating and editing rules, for browsing existing rule-bases, for performing various kinds of sensitivity analysis, and for performing various kinds of retrieval analysis. Additional tools can be added easily as necessary, and the systems can be made to work with a variety of database management systems. The user interface is highly interactive and makes extensive use of graphics.

The space limitations imposed by the proceedings format prevent us from giving any detailed performance results, so during the conference proper we will present some actual

retrievals using the current system. In particular, we will show how performance changes as we use coarser and coarser approximations to the base retrieval structure. We will also give more discussion of combining functions and their impact on performance.

REFERENCES

Cation, M.K. (1986). SOPE: A Systems Oriented Programming Environment. *AI 1986, Artificial Intelligence and Advanced Computer Conference and Exhibition*, Long Beach, CA.

Cation, M.K. (1987). *KIT Reference Manual (Version 3)*. Advanced Decision Systems, Mountain View, CA.

DeJong, G. (1982). An Overview of the FRUMP System. In *Strategies for Natural Language Understanding*, W.G. Lehnert, M.H. Ringle (eds), Lawrence Erlbaum Assoc., Hillsdale, NJ.

Kolodner, J.L. (1983). Indexing and Retrieval Strategies for Natural Language Fact Retrieval. *ACM Trans. Database Systems*, 8(3):434-464.

Schank, R.C., J.L. Kolodner, G. DeJong (1981). Conceptual Information Retrieval. In *Information Retrieval Research*, R.N. Oddy et al. (eds), Butterworths, London.

Tong, R.M., V.N. Askman, J.F. Cunningham, C.J. Tollander (1985). RUBRIC: An Environment for Full Text Information Retrieval. *Proc. 8th Int. ACM SIGIR Conf. on R&D in Information Retrieval*, Montreal, June.

Tong, R.M., D.G. Shapiro (1985). Experimental Investigations of Uncertainty in a Rule-Based System for Information Retrieval. *Int. J. Man-Machine Studies*, 22:265-282.

Tong, R.M., L.A. Appelbaum, V.N. Askman, J.F. Cunningham (1986). RUBRIC III: An Object-Oriented Expert System for Information Retrieval. *Proc. 2nd Annual IEEE Symposium on Expert Systems in Government*, McLean, VA, October.

Tong, R.M., L.A. Appelbaum (1987). Experiments with Interval Valued Uncertainty. In *Uncertainty in Artificial Intelligence*, J.F. Lemmer, L.N. Kanal (eds), North-Holland.