Conceptual Query Formulation and Retrieval *

Sanjiv K. Bhatia
Department of Mathematics & Computer Science
University of Missouri – St. Louis
St. Louis, MO 63121-4499

Jitender S. Deogun
Department of Computer Science & Engineering
University of Nebraska - Lincoln
Lincoln, NE 68588-0115

Vijay V. Raghavan
Center for Advanced Computer Studies
University of SW Louisiana
Lafayette, LA 70504.

Abstract

In this paper, we advance a technique to develop a user profile for information retrieval through knowledge acquisition techniques. The profile bridges the discrepancy between user-expressed keywords and system-recognizable index terms. The approach presented in this paper is based on the application of personal construct theory to determine a user's vocabulary and his/her view of different documents in a training set. The elicited knowledge is used to develop a model for each phrase/concept given by the user by employing machine learning techniques.

Our model correlates the concepts in a user's vocabulary to the index terms present in the documents in the training set. Computation of dependence between the user phrases also contributes in the development of the user profile and in creating a classification of documents. The resulting system is capable of automatically identifying the user concepts and query translation to index terms computed by the conventional indexing process. The system is evaluated by using the standard measures of precision and recall by comparing its performance against the performance of the SMART system for different queries.

Keywords: information retrieval, user profiles, knowledge acquisition, adaptive query translation

*This research is supported by the NSF grant IRI-8805875.

1 Introduction

In conventional information retrieval systems, a query is submitted as a set of keywords. These keywords may not be able to precisely capture the information requirements of a user. Moreover, any contextual information not specified by a user can not be taken into account during retrieval. The need for consideration of contextual information led to two major research directions. The first direction involved development of user profiles [19, 20] in which a user is matched with one of the existing generic profiles and retrieval is based on the selected profile. The second research direction involves construction of a conceptual query [21, 24]. A query is in the form of one or more *phrases* given by a user, to express his information requirements. In the existing systems, such as RUBRIC [21], this query is submitted as a set of production rules that map system-level keywords to a conceptual phrase. Such a query allows the system to infer the presence of a concept in a document. Employing these production rules, a user can specify a customized retrieval criterion that incorporates the syntactic, semantic, and contextual information in the query.

There are several problems with the use of production rule-based systems for conceptual query formulation from a user's perspective. These include the need for a user to learn system specific protocols to formulate a query, the inability of a user to adequately specify all the concepts and subconcepts used in the formulation of rules, the quantification of relationships between concepts and subconcepts, and the expectation for a user to be knowledgeable about the reasoning employed by the inference engine in the system [2, 24]. It is our hypothesis that these problems can be resolved in an efficient manner by employing the automated interview techniques from knowledge acquisition, for example personal construct theory. We have developed a system based on personal construct theory that allows a user to formulate a conceptual query. Like RUBRIC, the conceptual query is expressed by a user as a sequence of phrases to convey his information requirements to the system. However, our system already knows the keywords corresponding to each given phrase and therefore, the system-level query is developed automatically from those keywords.

In the existing approaches to conceptual query retrieval, the systems are passive in nature. They provide an inference engine to perform reasoning using the production rules given by a user and may allow a user some facilities in terms of editors that facilitate the development of production rules. These systems are more aptly classified as expert system shells rather than information retrieval systems. On the other hand, our system actively interviews a user to *learn* the concepts in the user's vocabulary and then, correlates this information to the system level keywords. Thus, the user is not responsible to learn the specific protocols for control knowledge and reasoning mechanisms.

The use of automated knowledge acquisition and machine learning techniques has several advantages. First, the user does not have to learn the implementation details of the system; instead the system itself attempts to learn user preferences. Second, the user is directly involved in query formulation and document clustering, eliminating any oversights and information loss that may be introduced by a system specialist. Document clusters are created in the system by keeping the documents, that are generally retrieved together, close to each other to facilitate retrieval. Finally, the system can keep track of any changes in user interests and adapt itself to reflect those changes.

The mapping between the linguistic concepts elicited from a user and the actual keywords that describe the documents forms one of the two important sources of knowledge for the rule-base for conceptual queries. The other knowledge source for the rule-base is the information on mutual relationship between the linguistic description of concepts from the user's viewpoint. Knowledge of mutual relationships between concepts enables a system to automatically provide the user with additional documents, if so desired.

In this paper, we present techniques to formulate a query using a concept given by the user. A keyword representation for each concept is developed and compared with the keyword representation of the documents in the bibliographic collection to perform retrieval. The query is translated to the system-level representation using the keyword representations for the concepts (shown by the dashed box in Figure 1). The techniques are experimentally validated and compared with the retrieval by the SMART system [8].

We have divided the knowledge acquisition and query formulation process into three steps:

1. Develop a keyword representation for documents.

2. Develop concepts from a training set of documents selected from the entire collection.

3. Learn the rules for conceptual retrieval.

The first step is realized by using the indexing techniques from the SMART system. The entire collection is indexed, using a stop list and word stemming, so that each document is represented as a set of keywords extracted from the document. The *stop list* is made up of commonly used words in the language that may not convey any information (such as articles and pronouns). *Stemming* reduces the equivalent forms of a word to an approximation of their *root* (such as reducing "computer," "computation," and "computing" to "comput").

In Section 2, we present the background information on knowledge acquisition by using personal construct theory. In Section 3, we present the techniques to develop concepts from a training set of documents. Section 4 presents the techniques to determine concept representation from elicited knowledge. In Section 5, we present the profile development process. Section 6 describes the experimental set up and results.

2 Knowledge Acquisition and Personal Construct Theory

Personal construct theory was pioneered by George Kelly in modern clinical psychology [17]. It is an elegant technique used by psychologists to gain insight into a person's behavior. The basis of personal construct theory is the hypothesis that every action of a person is motivated by his environment and background. This effectively implies that if a certain set of conditions prompt a person to perform an action at a certain time, the same set of conditions will result in the same action at any other time. In personal construct theory, no attempt is made to determine the stimuli for an action. It is the property of the stimulus, *from the point of view of the subject*, which gains more importance in triggering an action compared to the stimulus itself [17]. This point of view may or may not be explicit. If the point of view of a person is known, the actions of the person in a limited domain can be predicted with reasonable accuracy.

Kelly uses systematic methods to elicit knowledge from a patient. This makes personal construct theory a good choice for use in any knowledge acquisition system that extracts subconscious or default knowledge [16]. Personal construct theory can also be used to structure the extracted knowledge. This was the basic motivation behind its successful adaptation for knowledge elicitation by John Boose [4, 5, 6, 7]. He designed the Expertise Transfer System (ETS) to automatically interview an expert. ETS analyzes and constructs production rules from the elicited heuristic knowledge for adaptation into expert systems [5].

In a decision-making environment, the elements that have an influence on a person's actions are known as *entities*. An entity on its own does not contribute towards an action of a person but it is a property of the entity that does so. The significant properties of the entities, that have an influence on the actions and behavior of a person, are called the *constructs*. The constructs are characteristics that can be rated on a linear scale [16]. Constructs can be exemplified by *friendly-unfriendly, good-bad*, and *clear-hazy*.

Constructs are elicited by asking the subject to compare a few entities and enumerate a property to differentiate between those entities. Personal constructs provide a means to abstract the behavior of a person explicitly. An entity in a person's environment contributes towards his behavior in a positive or negative manner with respect to each construct. Two entities are considered to be similar if they have similar contributions towards all the constructs of the person. An entity may influence the actions of a person to some degree that can be quantified on a rating scale for the entity with respect to a construct.

The knowledge acquisition process makes a person enumerate the constructs through a structured interview using a training set. The entities in the training set are selected by the user. The user is interviewed to determine his opinion of the entities in the training set. Therefore, it is important that the user be well acquainted with all entities in the training set. Also, the training set should contain at least one example for each construct. This is required to have an adequate case library of constructs for the person [18]. The user should be allowed to add entities and constructs at any stage during the interview.

A person assigns the extent of relevance to constructs for different entities on a *rating scale*. The maximum value to be assigned indicates full confidence of the person in the relevance of the construct for the entity while the minimum value indicates that the construct is not relevant at all. The maximum and minimum values can also be interpreted in terms of the entity being *good* or *bad* in the viewpoint of the person. The maximum and minimum values are known as the *poles* of the rating scale. The rating of all the entities on all the constructs results in a rectangular matrix known as a *repertory grid* [16].

A repertory grid is a knowledge structure that is used to store the judgement of a user on the entities in the training set with respect to a set of constructs [13]. It is constructed by assigning a rating to all the entities for each construct. Formed like a rectangular matrix, a repertory grid can be analyzed to determine the relationship between different entities and constructs as perceived by a person. In the repertory grid, the entities form the columns, and the constructs, the rows.

In the repertory grid approach, the constructs are represented as a set of distinctions made with respect to a set of entities relevant to the problem domain. To construct a repertory grid, a person identifies the most important entities in his environment that are well-known to him. This ensures that the person being interviewed will be able to express an honest opinion of the extent to which various constructs are present in the entities. This selected set of entities is known as the *training set*.

The knowledge acquisition interview proceeds by asking the user to categorize a subset of the elements in the training set. This categorization is meant to force the user to bring forth and evaluate the difference between the elements in the subset. This segment of the interview also results in the user enumerating a construct that forms the basis for this categorization.

During the interview, the system randomly selects three entities from the training set and presents them to the interviewee. The person is asked to identify an important concept such that two entities in this triad are jointly relevant or non-relevant to the concept. This concept forms one construct. Next, the person is asked to assign a rating to all the entities in the training set with respect to the identified construct. This rating is assigned on a pre-established rating scale. The ratings of a construct, with respect to each entity in the training set, are collected as a row in the repertory grid.

The above interview segment is repeated with a different triad of entities, selected at random from the training set, and the constructs and ratings are elicited. The interview continues until the person is satisfied that there are no more constructs. The same rating scale is employed throughout the experiment. At any stage in the interview, the person can volunteer impromptu a new construct and rate the entire training set on this construct. The person is also at liberty to add new entities to enlarge the training set. In this case, the new entities are rated on all the concepts that have been previously elicited.

After the elicitation, the repertory grid is analyzed to identify the important structures and patterns in the expert's thought process. This analysis is meant to help the expert to easily recognize the natural associations between the elicited constructs. If the expert does not agree with the extent of associations between different concepts, he can alter the repertory grid to reflect the changes in associations [16].

The rows and columns of the repertory grid constitute the complete operational definition of the entities in a person's universe of discourse. Therefore, it can be concluded that two constructs are *functionally identical* if they are assigned the same ratings on all the entities [16]. The analysis of a repertory grid to determine the extent of mutual relationship between constructs is presented in the next section.

3 Repertory Grid Analysis - A Probabilistic Approach

The repertory grid provides data to approximate the probability distribution of each construct with respect to the entities in the training set. The probability distribution can be used to determine the extent of relevance of a construct in the description of an entity. The raw data in the repertory grid can also be analyzed to quantify and display the extent of mutual relationship between different constructs.

A number of researchers have proposed different methods for the analysis of repertory grids. The most notable of them include: a technique based on the logic of confirmation analysis [14], multidimensional scaling [15], and cluster analysis technique [22]. In this section, we describe a technique to improve the analysis stage of a repertory grid by constructing a *dependence tree* by analyzing feature dependencies [9]. The resulting dependence tree has a number of advantages over the earlier proposals and is particularly suitable for the domain of information retrieval [26].

3.1 Extraction of Relationships Between Constructs

The repertory grid contains information about the criteria employed by a user in making decisions or performing some actions. However, the information in a repertory grid is obscured by detail and therefore, it is not easy to directly see the extent of relationship between different constructs. This excessive information motivates the need for a compact structure, *e.g.* a maximum weight dependence tree, that can readily display the relationship between constructs. In this section, we present a technique to reduce the repertory grid to a maximum weight dependence tree that displays the mutual relationship between constructs.

Let c_{j_1} and c_{j_2} represent two constructs, corresponding to the rows j_1 and j_2, respectively, in the repertory grid. Let $P(c_j) = u$ be the *a priori* probability of the construct c_j being assigned the rating u and $P(c_{j_1} = u, c_{j_2} = v)$ be the joint probability of the construct c_{j_1} being assigned a rating u when the construct c_{j_2} is being assigned a rating v. The extent of mutual dependence between the two constructs c_{j_1} and c_{j_2} is measured by I, the *expected mutual information measure* (EMIM) [9, 25], given by

$$I(c_{j_1}, c_{j_2}) = \sum_{u,v} P(c_{j_1} = u, c_{j_2} = v) \log \frac{P(c_{j_1} = u, c_{j_2} = v)}{P(c_{j_1} = u) \cdot P(c_{j_2} = v)} \tag{1}$$

where u and v vary between the two poles of the rating scale used to elicit the repertory grid. It may be noted that $I(c_{j_1}, c_{j_2}) = I(c_{j_2}, c_{j_1})$ implying that the EMIM is symmetric [25]. Moreover, it can be easily seen that when c_{j_1} and c_{j_2} are independent, $P(c_{j_1}) \cdot P(c_{j_2}) = P(c_{j_1}, c_{j_2})$ resulting in $I(c_{j_1}, c_{j_2}) = 0$.

The data in a repertory grid can be used to calculate the EMIM between every pair of constructs using Expression 1. Since the *emim* is symmetric, this calculation yields a triangular matrix, called a *similarity matrix* [12] that displays the extent of mutual relationship between all pairs of constructs. In graph-theoretic terms, a similarity matrix corresponds to a complete undirected weighted graph without any reflexive edges. Since our objective is to identify the most important relationships in the constructs given by the user, a maximum weight dependence tree is constructed from the complete graph.

3.2 Dependence Tree Representation of Construct Relationships

From the complete graph, a dependence tree can be identified such that the total weight of the arcs included in the dependence tree is optimal in some well-defined sense. Such a tree is referred to as the *maximum weight dependence tree*. It brings out the most significant relationships between constructs as perceived by the user. It directly displays the dependence of a construct upon a few strongly related constructs as elicited from the user; the dependence of remaining constructs can be approximated through a chain of intermediate constructs in the dependence tree.

Let $\mathcal{T}$ be the set of all possible spanning trees that can be derived from the graph representation of the similarity matrix. A maximum weight dependence tree is a spanning tree T such that for all trees $T' \in \mathcal{T}$,

$$\sum_{j=1}^{m} I_T(c_j, c_{N(j)}) \geq \sum_{j=1}^{m} I_{T'}(c_j, c_{N(j)}) \tag{2}$$

where m is the number of nodes (constructs) in the spanning tree and $N(\cdot)$ is a function mapping the construct c_j onto its neighbor in the maximum weight dependence tree under consideration. The maximum weight dependence tree can be easily computed from the complete dependence graph by modifying one of the minimum spanning tree algorithms, for example Kruskal's algorithm.

To compute the maximum weight dependence tree from the complete dependence graph, the arcs of the graph are ordered by decreasing weights. The tree is constructed by selecting arcs from this ordered list one at a time and adding the selected arc to the tree if it does not form a cycle with the arcs already selected. If a cycle results, this arc is rejected and the next in the list is considered. This procedure produces a unique solution if the arc weights are all different. If several weights are equal, multiple solutions are possible; however, all these solutions have the same maximum weight.

The maximum weight dependence tree provides the best possible approximation to the probability distribution of pairwise mutual dependence between constructs. Let X be a random variable representing the entities. X indicates the value assigned to different nodes in the dependence tree in a certain order, for example the level-order. The selected order describes a *valid permutation* corresponding to the variable X. Let $P_T(X)$ be the probability distribution of the random variable formulated on the basis of dependence tree T and $P(X)$ be the actual distribution of X, X being the random variable representing the entities. Chow and Liu [9] have shown that $P_T(X)$ is an optimal approximation to $P(X)$ if and only if T has maximum possible weight.

The extent of mutual dependence between two nodes c_{j_1} and c_{j_2} is given by the normalized distance between them. The *normalized distance* between the nodes c_{j_1} and c_{j_2} is the total distance between the two nodes along the arcs of the spanning tree divided by the number of arcs between the two nodes. The conditional probability of degree of dependence between the constructs c_{j_1} and c_{j_2} can be determined by tracing the path between the nodes c_{j_1} and c_{j_2} on the maximum weight dependence tree.

The maximum weight dependence tree captures the dependencies needed to obtain an approximation of the distribution of the random variable X, representing the choice of an entity by the user. The distribution of random variable X is quantified by

$$P_T(X) = \prod_{j=1}^{m} P(c_j|c_{N(j)}) \qquad\qquad 0 \leq N(j) < j \tag{3}$$

where $\pi = (j_1, j_2, \ldots, j_m)$ is a permutation of the integers $1, 2, \ldots m$, and $N(\cdot)$ is a function mapping j onto its neighbor in the maximum weight dependence tree. It is assumed that $j(1) = 0$ implying that $P(c_{j_1}|c_{j_0}) = P(c_{j_1})$. It may, however, be noted that the choice of π depends on the structure of the dependence tree. All dependence trees are equivalent for a given problem. Therefore, all permutations are equivalent in the sense that as long as π is chosen appropriately based on the dependence tree, the result should not be affected.

3.3 Binary vs. Multivalued Grids

A *binary repertory grid* describes the relevance of a construct to an entity in terms of true or false. Therefore, a binary grid may not be as rich in information as the repertory grid arrived with a multivalued rating scale. A maximum weight dependence tree based on a binary grid may be substantially different from the one generated from a multivalued repertory grid with values on a uniformly distributed rating scale.

A binary grid is, however, easy to elicit as a user has to simply indicate the absence or presence of a construct in an entity, without contemplating the degree of its presence. This makes a binary grid useful in terms of simplicity of elicitation and computation. The existence of only two values in the grid to indicate the relevance of a construct with respect to an entity greatly reduces the computation time for EMIM (Expression 1).

Multivalued repertory grids contain more information about the relevance of a construct with respect to an entity than a simple true or false observation. A sparser binary repertory grid loses a large amount of probabilistic relationship information. Hence, the correlations between concepts may not be as conclusive as the ones computed by multivalued grids.

A binary grid can also be constructed from a multilevel grid by using a threshold τ such that all the ratings belows τ are changed to 0 while all the ratings above and equal to τ are changed to 1. Using r_{ij} as the rating of jth construct with respect to ith entity in the multivalued grid, the corresponding quantity in the binary grid, denoted by r'_{ij} is given by

$$r'_{ij} = \begin{cases} 0 & \text{if } r_{ij} < \tau \\ 1 & \text{if } r_{ij} \geq \tau \end{cases} \tag{4}$$

Binary grids are useful in dividing the entities into two mutually exclusive classes – relevant and non-relevant – with respect to each construct.

4 Development of Keyword Representation for Concepts

The concepts in a user's vocabulary that are relevant to express the user's information needs, and that may be employed by a user in expressing a query, are determined by interviewing the user. These concepts must be correlated with the keyword patterns that allow the system to evaluate the relevance of a document with respect to the individual concepts. This correlation can be developed in the form of a mapping from the concepts in the user's vocabulary to the keywords that occur in the documents in the training set.

The training set is selected by the user from the given collection. It consists of a set of documents that are well known to the user. Typically, we expect the training set to contain between 20 to 30 documents, with at least one example for each concept that is to be used by the user during query.

In existing rule-based systems, such as RUBRIC, the user must specify the production rules to describe the retrieval criterion [21]. Our hypothesis is that the query formulation should not require the knowledge of production rules and thus, require only the modification of user-specific information. With an independent control structure to represent search strategies, the user preferences can be learned and used as static inference structures to achieve the goal in a satisfactory manner. In this section, we discuss the development of a mapping from user concepts to system keywords.

The user vocabulary is elicited as phrases through the interview using a training set. To develop a system vocabulary, a keyword representation for each document is extracted. The extracted part of each word is called a *term*. The indexing process also associates a weight of importance *(term weight)* with each term. This term weight is directly proportional to the number of times the term occurs in the documents *(term frequency)* and inversely proportional to the number of documents in which the term occurs *(inverse document frequency)* [25].

The terms extracted from the documents by the SMART system may contain a number of words that cannot be classified as typical *noise words* but contribute little in developing the representation of concepts. Although these words tend to have a pattern of high frequency, they are not useful in describing the contents of a document, and therefore, may be ignored. Examples of such words include "system," "paper," "basic," "problem," and "knowledge." These words can be identified by their low weights in the set of index terms, and can be removed through a filtering process. The filtering results in the selection of only those terms that have a weight greater than a prespecified threshold θ.

As a result of the indexing and filtering process, let the representation of a document d_i in the training set be the set of terms $\{t_{i_1}, t_{i_2}, t_{i_3}, \ldots\}$. This representation is employed to develop the representation of a concept c_j by correlating the documents in the training set to their relevance judgement with respect to the concept c_j as indicated by the user in the repertory grid.

The relevance information in the repertory grid is used to impose a partition on the documents in the training set with respect to each concept. A document d_i is considered relevant to a concept c_j if its rating r_{ij} is greater than or equal to a prespecified threshold τ; otherwise it is considered non-relevant. The threshold τ can be empirically adjusted. The set terms in the relevant documents with respect to concept c_j is known as δ_j while the set of terms in the non-relevant documents is called δ'_j. Since a document is represented by the index terms, we can represent δ_j and δ'_j as

$$\begin{array}{rcl} \delta_j & = & \bigcup\{t_{i_k} \in d_i \mid r_{ij} \geq \tau\} \\ \delta'_j & = & \bigcup\{t_{i_k} \in d_i \mid r_{ij} < \tau\} \end{array} \tag{5}$$

The partition imposed on the documents by the concept c_j and threshold τ, resulting in the set of index terms δ_j and δ'_j, is used to develop the representation for the concept c_j.

The sets of index terms of relevant and non-relevant documents, δ_j and δ'_j respectively, are employed to develop the representation C_j for the concept c_j as well as the representation C'_j for the complement concept. The *complement concept* provides the representation of a typical non-relevant document with respect to the concept. The set C_j consists of only those terms that occur in some relevant document $d_i \in \delta_j$ but that do not occur in any non-relevant document $d'_i \in \delta'_j$. Similarly, the set C'_j contains only those terms that do not occur in any document that is relevant to the concept c_j. Therefore, we have

$$\begin{array}{rcl} C_j & = & \delta_j - \delta'_j \\ C'_j & = & \delta'_j - \delta_j \end{array} \tag{6}$$

C_j and C'_j facilitate the formulation of a rule to determine if a document d_i is relevant to a conceptual query based on c_j. The relevance is determined by the extent of match between the document representation d_i and the concept representation (C_j, C'_j), using a suitable coefficient of match, *e.g.*, cosine coefficient. Let ρ_{ij} indicate the extent of relevance of any document d_i (not necessarily from the training set) to the concept c_j. Then, we have

$$\rho_{ij} = \frac{|C_j \cap d_i|}{\sqrt{|C_j|} \times \sqrt{|d_i|}} - \frac{|C'_j \cap d_i|}{\sqrt{|C'_j|} \times \sqrt{|d_i|}} \tag{7}$$

The user can now present a query in the form of concept(s). A query could be either a single-concept query or a multiple-concept query. Multiple-concept queries and query expansion are discussed in detail in Section 5. A single concept query can be easily handled by the system using the concept representation and Expression 7 to evaluate the relevance of each document in the collection with respect to the concept. The extent of relevance is used to rank all the documents in the collection in the decreasing order of relevance. These documents are then presented to the user. If a user desires only a certain number of documents, the presentation of retrieved documents can be terminated after giving those documents that have been judged by the system to be most relevant to the concept.

5 Development of User Profiles

A user profile provides flexibility and adaptability to an information retrieval system. The flexibility is achieved by the knowledge of a user's vocabulary and his interests that might be helpful in the retrieval of additional documents. Such knowledge allows the system to interpret a user query in accordance with the meaning implied by the user. Hence, a user profile helps in customizing the retrieval process according to individual user requirements [3]. The user profile is based on the relationship between different concepts given by the user that have been structured in the form of a maximum weight dependence tree. This relationship is used to expand a query by adding phrases that may be relevant to the current query. It may be noted that ultimately, an expanded query becomes a multiple-concept query, and is treated similarly.

The profile of a user is developed in two separate phases. In the first phase, the system develops the profile at a *higher level* in the form of a maximum weight dependence tree as described in Section 3.2. This phase establishes the vocabulary of the user in the form of concepts and the mutual relationship between these concepts. In the second phase, the system constructs a mapping, known as *phrase mapping*, for each conceptual phrase given by the user by developing a correspondence between user phrases and a set of index terms as described in Section 4. Here, we develop the use of mutual relationships between concepts in the maximum weight dependence tree to select additional concepts for query expansion.

The membership of a term t_i in the index term representation (C_j, C'_j) of a concept is used to develop the rule (Expression 7) that describes the contents of a typical document relevant to the concept. Combining the results of this membership of terms in the representation for the concept with the membership of the terms in the representation of adjacent concepts in the maximum weight dependence tree (Expression 2) allows for the assignment of probabilistic relevance weights to the documents and is used to control the number of documents in the retrieved set.

Each concept c_j forms a node in the maximum weight dependence tree. The rules for query expansion can be developed by extending each node (concept) in by a selective addition of adjacent nodes. These adjacent nodes contain the descriptions $(C_{N(j)}, C'_{N(j)})$ – the representation of concepts that are adjacent to or neighbors of the node c_j. The extent of relevance of all documents to the neighboring concept is propagated to the node that was originally given as the query by using the uncertainty calculus as described below. The maximum value of the match of a document to the concept at the node is assigned as the extent of relevance. The documents are ordered by decreasing relevance in the retrieved set.

For the propagation of extent of relevance with respect to the new query, the uncertainty calculus described in [21, 23] is employed. According to the uncertainty calculus, the propagation of uncertainty is explained by considering Figure 2. In Figure 2, the conclusion is represented by γ, the antecedents are represented by α and β, the weight of the arc is given by w, and the extent of relevance (or certainty factor) for antecedents α and β is given by ρ_α and ρ_β, respectively.

In Figure 2a, the relevance is propagated to the conclusion γ by a product of the extent of relevance of the antecedent and the weight of the arc w between the antecedent and the conclusion. Figures 2b and 2c indicate the propagation of relevance in an OR-node and an AND-node, respectively. In an OR-node, the conclusion γ is true if either of α or β is true, the extent of relevance of γ is given by the maximum of certainty factors due to α or β. In an AND-node, the conclusion γ is true if *both* of α and β are true, the certainty factor of γ is given by the minimum of certainty factors due to α and β.

$$\gamma \Leftarrow \alpha, \quad \rho_\gamma = w \times \rho_\alpha \tag{8}$$

$$\gamma \Leftarrow \alpha \vee \beta, \quad \rho_\gamma = \max(w_1 \times \rho_\alpha, w_2 \times \rho_\beta) \tag{9}$$

$$\gamma \Leftarrow \alpha \wedge \beta, \quad \rho_\gamma = \min(w_1 \times \rho_\alpha, w_2 \times \rho_\beta) \tag{10}$$

The query is specified in terms of some concept(s) as described in Section 4 using some phrase(s) whose representation is known to the system. The specified concept(s) are used to impose a directional structure on the maximum weight dependence tree by interpreting the node corresponding to the specified concept as the root of the tree.

If the user specifies more than one concept in the query, a *virtual root* is created for the result. The concepts in the query are treated as children of this virtual root and are correlated by using the uncertainty calculus (Expressions 8-10). The virtual root is joined to its children (the specified concepts) by arcs with a weight of 1.0. The root can be in the form of an OR-node (Figure 2b), or an AND-node (Figure 2c), or a combination of both. To determine the result of the query, a separate query is performed for each concept and the virtual root is treated as the goal for the complete query.

Starting with the specified concept as the root in the maximum weight dependence tree, a best-first search is performed to determine the documents relevant to the concept. The most promising set of documents for retrieval is the one in which the representation for each document contains terms that are also part of the representation of the equivalence class given by $(C_{\mathrm{root}}, C'_{\mathrm{root}})$ using Expression 7. Next, the system selects a node from the neighborhood of the root that has the maximum value of EMIM with respect to the root. The relevance of the documents selected at this node is propagated to the root by using Expression 8. The result is combined with the previously retrieved set by using the maximum relevance value for each document after propagation. The next node for the expansion of the query is selected from the neighborhood of the nodes selected so far such that the new node has the maximum value of EMIM with respect to the root using Expressions 3 and 8.

6 Design of Experiments

Several experiments were conducted to elicit the opinion of users on the relevance of a sample of documents through a knowledge acquisition program. The knowledge acquisition program has been developed and implemented in `C` using principles of personal construct theory. The program automatically interviews a user to determine the relevance of documents in a training set from the user's viewpoint.

The initial collection used in the experiment consists of 158 documents. It is in the `refer` format – a standard format to organize bibliographies under `Unix`. The entries in the bibliographic database consist of a complete bibliographic reference in separate fields in addition to an abstract for each entry. The abstracts in the bibliography are the same as the ones that appear as a part of the document. In some cases, where such an abstract was not available, the abstract was selected from a standard review source, *e.g.*, *Computer and Control Abstracts*. A typical entry from this bibliography is presented in Figure 3.

The bibliography was developed as a part of the evaluation of this work. The evaluation itself was based on the computation of precision at various points of recall. *Precision* is defined to be the ratio of the number of relevant documents retrieved to the number of documents retrieved. *Recall* is defined as the ratio of the number of relevant documents retrieved to the number of relevant documents in the collection. It is essential for evaluation that each document in the bibliography be known to the user. This requirement is imposed only for the evaluation of results and is not a constraint in an operational system. Since the results of a query are not to be evaluated to measure the performance of the system in an operational environment, the knowledge of each document is not necessary in such a case. As the work progressed, the size of the bibliography increased and later experiments used a bibliography of 250 documents.

Personal construct theory requires that the user be familiar with all the entities in the training set. Therefore, the user is presented with a few documents in the collection and asked to select a training set such that the documents in the training set represent a complete gamut of his interests and information requirements[1]. This is necessary because the system must have at least one example of a relevant document for each concept that is employed by the user to indicate his information requirements. The training set is created by the system as a separate bibliographic database in the `refer` format complete with the abstracts.

In the experiment presented here, the training set was made of 24 documents. In this test case, the user looked at approximately 120 documents to select the training set. However, in larger collections, we do not expect the user to view more than 100–150 documents to select the training set. Selection of a training set from within the total collection is a standard practice in the systems based on personal construct theory [16]. In our system, this selection is made as follows. The system presents a document to the user and asks whether the document is well-known to him and whether it should be included in the training set. This is repeated with many documents till the user is satisfied with the size of the training set, 24 documents in this case. In a large system, this approach has obvious flaws. Therefore, it is proposed that the clustering of documents [1, 10] be exploited to restrict the documents presented to a user for training set selection. Such a system can present a user with different classes of documents and asks him to select some classes.

[1]In some cases, the user may not explicitly remember to include documents that may be relevant in the description of his viewpoint. In such a case, the user is allowed to add more documents at the end of the interview

The system then presents a few documents to the user from each class to determine the training set.

The knowledge elicitation interview begins after the training set is established. The system selects three documents from the training set and presents them to the user. The user distinguishes between these documents such that at least two of the documents address a common topic. In a case where two documents do not address a common topic, the user can distinguish a topic that is addressed in one document but not in the other two. This topic forms one concept (construct) in the user's vocabulary that is employed to formulate queries at a later stage. After the concept has been established, the user is asked to assign a grade of relevance on a predetermined rating scale to all the documents in the training set. The rating scale employed for our experiments ranged from 1 to 5 with a rating of 1 indicating that the document is not at all relevant to the concept and a rating of 5 indicating that the document is relevant to the concept.

The selection of rating scale was based on practical considerations. A binary rating scale would require a user to give a blanket judgement about a document being relevant with respect to a concept. A rating scale of 1-to-3 would be only marginally better than the binary rating scale. These two rating scales have less information compared to the 5-point scale selected and, therefore, can not accurately predict the extent of relationship between different concepts for the purpose of developing user profiles. On the other hand, a rating scale larger than 5-point can stress the user in contemplating the extent of relevance of a document with respect to a concept due to a larger number of choices.

After selection of the training set and explaining the use of the selected rating scale to the user, the knowledge elicitation interview is started. During any stage of the knowledge elicitation interview, the user can request the system to display the abstract of a textual item under review. This option allows the user to better judge the relevance of a document to a concept.

The elicitation interview is repeated with a different set of three randomly selected documents from the training set to elicit a different concept and ratings are assigned to all documents with respect to these concepts as well. At any stage during the interview, the user is free to volunteer some concepts on his own and ratings of all documents on those concepts are elicited as well.

The complete set of ratings elicited from the user is collected in a repertory grid. An example of such a grid is presented in Tables 1-2. This grid has been elicited using a training set of 24 documents from the overall collection. During the interview, 21 different concepts were elicited from the user to describe his interests in the documents in the training set. These concepts are presented in Table 3.

The repertory grid is the basic knowledge structure in the system. Thus, it can change with a change in a user's patterns of thinking. New concepts can be added into the repertory grid to reflect the changed viewpoint of user's model of thinking. The user has to assign a rating to each entity in the training set with respect to the new concepts to reflect the new thinking patterns. Similarly, if there are not enough entities in the training set, a user can expand the training set by adding new entities and assigning a rating to those entities with respect to the previously elicited concepts.

6.1 Repertory Grid Analysis

The unprocessed knowledge in a repertory grid can be analyzed by different techniques to extract significant relationships. The analyzed knowledge can be used to develop rules that can solve a problem, *e.g.*, the assignment of a document to a cluster, from a user's viewpoint.

In the probabilistic method, the repertory grid is analyzed using the expected mutual information measure (Expression 1) to determine the pairwise dependence of each possible pair of constructs. The resulting similarity matrix is presented in Tables 4-5. The similarity matrix is symmetric along the main diagonal. It corresponds to a complete undirected weighted graph, with the EMIM $I(c_{j_1}, c_{j_2})$ being the weight of the edge between the constructs c_{j_1} and c_{j_2}. Therefore, it can be used to identify a maximum weight dependence tree by applying Kruskal's algorithm. The edges chosen by Kruskal's algorithm are presented in Table 6. The resulting dependence tree is presented in Figure 4.

The use of the maximal weight dependence tree to determine the distribution of a random variable X is demonstrated as follows. Let d be any arbitrarily selected combination of values representing the relevance of concepts to a document; for example

$$d = (3, 4, 4, 5, 4, 5, 2, 2, 2, 1, 5, 4, 3, 2, 1, 1, 5, 2, 4, 3, 2)$$

This is to be interpreted as "Document's relevance to concept X_1 is 3, to concept X_2 is 4, and so on." Based on the maximum weight dependence tree in Figure 4 and the labels in Table 3, a valid permutation is as shown in Table 7. Each $p_{j'}$ in Table 7 corresponds to a concept c_j.

The values in d are ordered appropriately, with respect to a concept X_6, resulting in d' given by

$$d' = (5, 2, 1, 5, 4, 4, 5, 3, 2, 2, 2, 1, 3, 3, 2, 3, 1, 4, 2, 5, 4)$$

Now, the probability $P(X = d)$ can be approximated by using the repertory grid (Tables 1-2) as follows.

$$\begin{aligned}
&P(X = d) = \\
&\quad P(X_6 = 5) \cdot P(X_7 = 2|X_6 = 5) \cdot P(X_{15} = 1|X_7 = 2) \cdot P(X_4 = 5|X_{15} = 1) \cdot \\
&\quad P(X_{19} = 4|X_4 = 5) \cdot P(X_2 = 4|X_{15} = 1) \cdot P(X_{11} = 5|X_{15} = 1) \cdot P(X_1 = 3|X_{11} = 5) \cdot \\
&\quad P(X_{21} = 2|X_1 = 3) \cdot P(X_{14} = 2|X_{21} = 2) \cdot P(X_8 = 2|X_{14} = 2) \cdot P(X_{10} = 1|X_8 = 2) \cdot \\
&\quad P(X_3 = 3|X_8 = 2) \cdot P(X_{13} = 3|X_{21} = 2) \cdot P(X_{18} = 2|X_{10} = 1) \cdot P(X_{20} = 3|X_{10} = 1) \cdot \\
&\quad P(X_{16} = 1|X_3 = 3) \cdot P(X_{12} = 4|X_{13} = 3) \cdot P(X_9 = 2|X_{13} = 3) \cdot P(X_{17} = 5|X_{18} = 2) \cdot \\
&\quad P(X_5 = 4|X_9 = 2)
\end{aligned}$$

Finally, the binary grid computed from the multivalued grid of Tables 1-2 is presented in Tables 8-9.

6.2 Concept-based Query

To test the effectiveness of our approach in query formulation, several experiments were conducted using our knowledge-based approach as well as the SMART system. The document representation for the entire collection was developed by using the same indexing scheme – removal of common words by stop list, stemming of words, and weight assignment using term frequency weights as well as the inverse document frequency weights (TF×IDF weights). The index terms extracted from the bibliographic entry in Figure 3 as a result of this indexing are presented in Figure 5.

In our system, the repertory grid (Tables 1-2) elicited from the user was utilized to construct the representation for each concept elicited from the user. The documents that were ranked by the user as $\geq \tau$ ($\tau = 3$) in the repertory grid were deemed to be relevant with respect to the concepts. Selection of $\tau = 3$ is based on the following reasoning. Documents that have a ranking of 5 are obviously judged by the user to be relevant; documents with a ranking of 4 are also considered to be relevant though not to the same extent as the ranking 5; and the user is not very certain about the relevance of documents with a ranking of 3 but the user is fairly certain that these documents should not be ranked to be non-relevant with respect to the concept.

The output from the SMART system indexing process is in the form of a record of five fields with respect to each keyword. The five fields, in the order of output, are:

1. *Document-id.* An integer to represent the document in the order it was received for indexing.
2. *Concept-id.* An integer assigned to be zero in the current indexing.
3. *Term-id.* An integer corresponding to the keyword.
4. *Term weight.* A real number to indicate the TF×IDF weight of the keyword.
5. *Keyword.* The stemmed keyword.

To extract the index terms with respect to each document in the collection from the output of the SMART system, the output is sorted in ascending order using document-id as the key. A simple program uses the sorted records to create a `Prolog` predicate with respect to each document that consists of the index term extracted from the document along with its weight for the document. An example of such a predicate for the document in Figure 3 is provided in Figure 6. The weighted representation for each document is used to develop the representation (C_j, C'_j) for each concept by using the equivalence classes of documents (δ_j, δ'_j) with respect to the concept. The equivalence classes are selected based on the information in the repertory grid (Tables 1-2). In the construction of the equivalence classes δ_j and δ'_j with respect to a concept c_j (Expression 5), the value of θ is selected as 2.0. This value is empirically determined through various experiments [11]. The representation for equivalence classes with respect to each concept is developed through a `Prolog` program. A part of the representation for concept c_9 (programming tutors) is shown in Figure 7.

After the system has developed the index term representation for each concept, a query is submitted in the form of a concept. This concept is selected from the set of concepts elicited from the user during the knowledge acquisition interview (Table 3).

The system develops a keyword representation $(C_j,\ C'_j)$ corresponding to each concept c_j that can be submitted as the query. This representation (a list of keywords) is matched against the keyword representation of each document in the collection, ignoring the weight of the index term. The relevance ρ_{ij} of each document d_i with respect to the query (concept c_j) is calculated using Expression 7. The documents are presented to the user in the reverse order of their relevance to the query. Presentation of documents to the user can be terminated after enough documents have been given to the user to satisfy his/her information needs.

In the retrieval through the SMART system, the results were extremely poor when we submitted just the concept as a query. To be fair in comparison, we submitted the set of index terms from the documents, that were relevant to the concept, as the query. This is achieved by referring to the same repertory grid (Tables 1-2) that is utilized for the experiment with our approach. This also obviates the need for relevance feedback as the system receives the query in the form of keywords that are extracted from relevant documents only. The extent of relevance is quantified by using the cosine matching coefficient [25] given by

$$\text{Cosine coefficient} = \frac{|X \cap Y|}{\sqrt{|X|} \times \sqrt{|Y|}} \tag{11}$$

The query is formulated by combining the text of the relevant documents. The SMART system compares the query against the documents in the collection employing the cosine coefficient and ranks the documents in the collection in decreasing order of their relevance to the concept. The documents are then presented to the user in the decreasing order of their ranking with respect to the query.

The results from both the systems are evaluated by comparing the two outputs against the preassigned relevance values to the documents by the same user in the entire collection. The relevance values are assigned individually with respect to each concept by the same user who was interviewed to develop the concepts[2]. During this assignment, the user examines each document with respect to a concept and assigns it to be relevant or non-relevant to the concept. Using this preassigned relevance assignment, a precision curve at various points of recall is plotted for different conceptual queries using both our system as well as the SMART system.

The average performance of the two systems for a typical query is presented in Figure 8 in the form of a plot to show the precision values at different levels of recall. It is observed that the solid curve, representing the performance of our system, is consistently higher than the dashed curve that represents the precision values using the SMART system. As the user wants to retrieve more documents (better recall), the performance degrades in both the systems but overall, our system performs significantly better than the SMART system. The average performance is approximately 44% better than the retrieval by the SMART system. If P_s indicates the precision at recall point R in the SMART system and P_n indicates the precision at recall point R in our system, the difference in performance was calculated by

$$\frac{P_n - P_s}{P_s}$$

Figure 9 presents a case when our system performs significantly better than the SMART system. It is observed that the performance degrades in both the systems as the recall is improved. However, the precision value of retrieval by the SMART system drops suddenly at a low level of recall, from a near perfect

[2]The concept gets translated into an expanded query.

precision value of 1.00 at a recall point of 0.22 to a precision value of 0.56 at a recall level of 0.28. On the other hand, the most significant drop in precision values in the retrieval by our system occurs as the recall level moves from 0.72 (precision value 0.81) to 0.78 (precision value 0.64). In this case, the performance averaged over different points of recall improved by approximately 90% with the SMART system retrieval as the base.

The worst case performance is exemplified by Figure 10 where our system performs almost as well as the SMART system. The performance of the two systems in this case is fairly close to each other and at some levels of high recall, the SMART system performs slightly better than our system. Overall, the performance improved by approximately 14% in this case at various points of recall.

The average performance of the two systems over 14 different queries is presented in Figure 11. It is observed that on an average, our system performs significantly better than the conceptual query simulated on the SMART system. While examining these results, it must be noted that the results from the SMART systems are not those of conventional retrieval that has been performed by presenting a keyword but rather a conceptual query is formulated and presented to the system. On an average, our system improved the performance by approximately 32% compared to the SMART system.

6.3 Observations on the Experiments

For the purpose of knowledge elicitation, a user has to select a training set and assign the relevance judgement to each document in the training set with respect to each concept. A typical interview session for knowledge acquisition using 24 documents and resulting in 21 concepts lasts about two hours. If the user's interests are limited to a narrow field, this interview session can be concluded in less time. Also, a user can stretch the interview over several sessions as long as the user is consistent in his interpretations. If there is a change in interests of a user, the repertory grid can be edited and the profile modified accordingly with minimal effort on part of the user.

The multivalued repertory grid is used to develop the maximum weight dependence tree. This is because the multivalued grid captures the finer distinctions in the user's viewpoint. However, the relevance of the document is judged on the basis of binary grid by using the threshold τ to construct a binary grid from the multivalued grid.

The biggest hurdle faced in the evaluation process is to obtain and document the judgement of individual user over the entire collection. To perform this evaluation, the user has to assign a judgement of relevance or non-relevance to each document in the collection with respect to each query. This judgement is then used to evaluate the results from a query from the SMART system as well as our system. The assignment of relevance judgement to each document in the collection, with respect to each query, makes the entire process extremely tedious. It also limits the selection of the bibliographic collection to the subject's personal bibliography for the experiment. For the purpose of evaluation, the user must be familiar with each document in the collection thus eliminating the possibility of using a standardized collection.

7 Discussion

In this paper, we have presented a method to elicit the viewpoint of a person on a training set of entities that influence his/her behavior. The techniques described facilitate the development of systems that capture the user viewpoint for retrieval and classification of objects. The process is illustrated through knowledge elicitation from a person in the domain of information retrieval. The analysis of the repertory grid may bring forth the patterns and associations in a person's thought processes. The information elicited is analyzed by a probabilistic method that is particularly suitable to information retrieval.

The maximum weight dependence tree provides a simple and concise overall view of the information elicited during the knowledge acquisition interview. It clearly shows the relationships between different constructs as perceived by a person. It should be noted that the maximum weight dependence tree as derived from the repertory grid is not unique, however, all these solutions have the same maximum weight.

It is observed from the precision-recall curve (Figure 11) that there is a marked improvement in retrieval at different recall points. At low recall, both the systems have a near perfect precision. However, our approach converges to maximal recall with a higher degree of precision compared to the SMART system, as evidenced by our experiments and shown in Figures 8–11.

In the experiments, we have strived to keep the conditions as close as possible in both the approaches. The indexing technique is the same (TF×IDF weights) in both the experiments. Similarly, the conceptual query in the SMART system is formulated by employing the same set of documents as the ones used in the training set.

It is important that the conceptual query be formulated by submitting one of the concepts that are known to the system. Therefore, it is imperative that the user give all the concepts that can be used in the query and include at least one example document relevant to each concept in the training set. However, it is desirable that the user include more than one example document with respect to each concept. The knowledge acquisition part of the system relies on full cooperation of the user for the selection of documents in the training set and for the elicitation of concepts in the user vocabulary. If the user formulates a query that does not convey to the system one of the known concepts, the system is unable to create a representation for the concept and hence, the retrieval is not satisfactory.

A concept can be added to the known vocabulary by the expansion of the repertory grid in either direction – either by the addition of new documents in the training set or by iteratively adding a new concept and providing information about the relevant documents in the existing training set with respect to that concept. The system performs the analysis on the revised repertory grid and constructs new representations for C_j and C'_j with respect to all the concepts.

A good user interface for the described retrieval environment can be provided by a menu-driven system. The menu-driven system minimizes the chances of errors like spelling mistakes and the use of synonyms in the formulation of queries. Moreover, a user can readily see a word that was used during the training phase to convey the interpretation. The system also can have an option to modify the repertory grid by addition of documents in the training set or the addition of concepts.

References

[1] S. K. Bhatia, J. S. Deogun, and V. V. Raghavan. Assignment of Term Descriptors to Clusters. In *Proceedings of the 1990 Symposium on Applied Computing*, pages 181–185, Fayetteville, AR, April 1990. IEEE Computer Society Press.

[2] S. K. Bhatia, J. S. Deogun, and V. V. Raghavan. Automatic Rule-Base Generation for User-Oriented Information Retrieval. In *ISMIS '90: Proceedings of the Fifth International Symposium on Methodologies for Intelligent Systems*, pages 118–125, Knoxville, TN, October 1990. North-Holland.

[3] S. K. Bhatia, J. S. Deogun, and V. V. Raghavan. User Profiles for Information Retrieval. In Z. W. Ras and M. Zemankova, editors, *Methodologies for Intelligent Systems: 6th International Symposium, ISMIS '91*, pages 102–111, Charlotte, NC, October 1991. Springer-Verlag. Lecture Notes in Artificial Intelligence # 542.

[4] J. H. Boose. A Knowledge Acquisition Program for Expert System Based on Personal Construct Theory. *International Journal of Man-Machine Studies*, 23:495–525, 1985.

[5] J. H. Boose. Personal Construct Theory and the Transfer of Human Expertise. In T. O'Shea, editor, *Advances in Artificial Intelligence*, pages 51–60. Elsevier-Science Publishers B. V. (North-Holland), 1985.

[6] J. H. Boose. *Expertise Transfer for Expert System Design*. Elsevier-Science Publishers, New York, 1986.

[7] J. H. Boose. A Survey of Knowledge Acquisition Techniques and Tools. *Knowledge Acquisition*, 1(1):3–37, March 1989.

[8] C. Buckley. Implementation of the SMART Information Retrieval System. Technical Report TR 85-686, Cornell University, Department of Computer Science, Ithaca, NY, May 1985.

[9] C. K. Chow and C. N. Liu. Approximating Discrete Probability Distributions with Dependence Trees. *IEEE Transactions on Information Theory*, IT-14:462–467, 1968.

[10] J. S. Deogun, S. K. Bhatia, and V. V. Raghavan. Automatic Cluster Assignments for Documents. In *Proceedings of the Seventh IEEE Conference on Artificial Intelligence Applications*, Miami Beach, FL, February 1991.

[11] J. S. Deogun and V. V. Raghavan. UNL/USL: MUC-3 Test Results and Analysis. In B. Sundheim, editor, *Proceedings of the Third Message Understanding Conference*, San Diego, CA, June 1991.

[12] J. S. Deogun, V. V. Raghavan, and S. K. Bhatia. A Theoretical Basis for the Automatic Extraction of Relationships from Expert-Provided Data. In *ISMIS '89: Proceedings of the Fourth International Symposium on Methodologies for Intelligent Systems: Poster Session*, pages 123–131, Charlotte, NC, October 1989.

[13] K. M. Ford and P. J. Chang. An Approach to Automated Knowledge Acquisition Founded on Personal Construct Theory. In M. Fishman, editor, *Advances in Artificial Intelligence*. JAI Press, Greenwich, CT, 1989.

[14] K. M. Ford and F. E. Petry. The Production of Expert System Rules from Repertory Grid Data Based on A Logic of Confirmation. In *Seventh International Congress on Personal Construct Psychology*, Memphis, TN, 1987.

[15] J. G. Gammack. Different Techniques and Different Aspects on Declarative Knowledge. In A. L. Kidd, editor, *Knowledge Acquisition for Expert Systems: A Practical Handbook*, pages 137–163. Plenum Press, New York, NY, 1987.

[16] A. Hart. *Knowledge Acquisition for Expert Systems*. McGraw-Hill, New York, NY, 1986.

[17] G. A. Kelly. *The Psychology of Personal Constructs*. Norton Publishers, New York, NY, 1955.

[18] Y. Kodratoff, M. Manago, and J. Blythe. Generalization and Noise. In B. R. Gaines and J. H. Boose, editors, *Knowledge Acquisition for Knowledge-Based Systems (vol. 1)*, pages 301–324. Academic Press, San Diego, CA, 1988.

[19] A. J. Kok and A. M. Botman. Retrieval Based on User Behavior. In *Proceedings of the Eleventh International Conference on Research and Development in Information Retrieval*, pages 343–357, Grenoble, France, June 1988.

[20] R. R. Korfhage and H. Chavarria-Garza. Retrieval Improvement by the Interaction of Queries and User Profiles. In *COMPSAC 82: IEEE Computer Society's Sixth International Computer Software & Applications Conference*, pages 470–475, Chicago, IL, November 1982.

[21] B. P. McCune, R. M. Tong, J. S. Dean, and D. G. Shapiro. RUBRIC: A System for Rule-based Information Retrieval. *IEEE Transactions on Software Engineering*, SE-11(9):939–945, September 1985.

[22] M. L. G. Shaw and B. R. Gaines. An Interactive Knowledge-Elicitation Technique Using Personal Construct Technology. In A. L. Kidd, editor, *Knowledge Acquisition for Expert Systems: A Practical Handbook*, pages 109–136. Plenum Press, New York, NY, 1987.

[23] R. M. Tong, V. N. Askman, J. F. Cunningham, and C. J. Tollander. RUBRIC: An Environment for Full Text Information Retrieval. In *Proceedings of the Eighth International ACM Conference on R&D in Information Retrieval*, Montreal, Canada, June 1985.

[24] R. M. Tong and D. G. Shapiro. Experimental Investigations of Uncertainty in A Rule-Based System for Information Retrieval. *International Journal of Man-Machine Studies*, 22:265–282, 1985.

[25] C. J. van Rijsbergen. *Information Retrieval.* Butterworth Publishers, Boston, MA, 2nd edition, 1981.

[26] C. J. van Rijsbergen, D. J. Harper, and M. F. Porter. The Selection of Good Search Terms. *Information Processing and Management*, 17:77–91, 1981.

	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9	d_{10}	d_{11}	d_{12}
c_1	4	5	4	5	2	2	5	5	5	5	5	5
c_2	3	2	3	4	4	5	2	3	5	5	2	5
c_3	1	1	1	1	1	1	1	1	1	1	1	1
c_4	2	2	2	1	1	1	3	5	1	1	5	1
c_5	1	1	1	1	1	1	1	1	1	1	1	1
c_6	1	1	1	3	1	1	1	1	3	1	1	1
c_7	2	1	2	4	5	4	1	2	3	2	1	2
c_8	1	1	1	4	1	2	2	2	4	2	1	1
c_9	1	1	1	1	1	1	1	1	1	1	1	1
c_{10}	1	4	1	4	1	1	4	2	3	2	1	1
c_{11}	5	3	5	3	2	2	3	2	3	3	2	2
c_{12}	1	1	1	1	1	1	1	1	1	1	1	1
c_{13}	1	1	1	1	1	1	3	1	1	1	1	1
c_{14}	2	3	2	2	1	1	5	2	3	3	2	1
c_{15}	2	3	2	4	5	5	2	1	4	4	1	5
c_{16}	1	1	1	1	1	1	1	1	1	1	1	1
c_{17}	1	5	1	5	2	2	4	2	5	4	2	1
c_{18}	2	5	2	5	1	1	4	3	5	4	1	1
c_{19}	1	1	1	1	1	1	1	5	1	1	5	1
c_{20}	1	3	1	2	1	1	3	5	2	1	5	1
c_{21}	1	2	1	1	1	1	3	2	2	2	2	1

Table 1: Repertory grid of the expert's evaluation of documents 1-12

	d_{13}	d_{14}	d_{15}	d_{16}	d_{17}	d_{18}	d_{19}	d_{20}	d_{21}	d_{22}	d_{23}	d_{24}
c_1	3	5	1	1	1	1	1	1	3	1	1	1
c_2	1	5	1	3	3	2	2	3	5	1	1	1
c_3	1	1	1	5	5	5	5	5	2	1	1	1
c_4	5	1	5	5	5	2	5	3	2	5	4	5
c_5	1	1	5	1	1	1	1	1	1	5	5	5
c_6	1	1	1	1	1	1	1	1	5	1	1	1
c_7	1	2	1	1	1	2	1	1	3	1	1	1
c_8	1	3	2	5	5	4	5	5	2	1	2	1
c_9	1	1	5	1	1	1	1	1	1	3	5	5
c_{10}	1	4	3	5	5	4	5	4	3	2	3	2
c_{11}	1	3	1	1	1	1	1	1	2	1	1	1
c_{12}	1	1	5	1	1	1	1	2	1	5	5	5
c_{13}	1	1	5	1	1	1	1	1	1	4	5	5
c_{14}	1	3	5	4	4	3	4	4	1	1	5	2
c_{15}	1	4	1	1	1	2	1	1	4	1	1	1
c_{16}	1	1	1	5	5	5	5	5	2	1	1	1
c_{17}	1	5	4	5	5	4	5	5	2	1	5	3
c_{18}	1	4	5	5	5	4	5	5	2	3	5	4
c_{19}	4	1	5	5	5	4	5	2	1	5	4	4
c_{20}	2	2	1	4	4	1	4	2	2	2	1	1
c_{21}	1	2	4	5	4	3	4	5	2	3	4	4

Table 2: Repertory grid of the expert's evaluation of documents 13-24

c_1	information retrieval	c_2	classification
c_3	personal construct theory	c_4	expert systems
c_5	intelligent tutoring systems	c_6	rough sets
c_7	graph theory	c_8	knowledge acquisition
c_9	programming tutors	c_{10}	machine learning
c_{11}	hypertext	c_{12}	education training
c_{13}	student models	c_{14}	user profile
c_{15}	cluster analysis	c_{16}	repertory grid
c_{17}	adaptive systems	c_{18}	user-oriented systems
c_{19}	rule-based systems	c_{20}	fuzzy logic
c_{21}	psychology		

Table 3: Constructs elicited from the user

	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	c_{10}	c_{11}
c_1	-	0.62	0.47	0.63	0.28	0.26	0.64	0.42	0.28	0.57	0.97
c_2		-	0.33	0.71	0.41	0.25	0.68	0.46	0.41	0.49	0.59
c_3			-	0.26	0.10	0.20	0.25	0.49	0.10	0.44	0.36
c_4				-	0.27	0.24	0.63	0.43	0.28	0.53	0.59
c_5					-	0.05	0.21	0.13	0.45	0.31	0.25
c_6						-	0.37	0.30	0.05	0.23	0.24
c_7							-	0.41	0.21	0.44	0.52
c_8								-	0.17	0.79	0.48
c_9									-	0.34	0.25
c_{10}										-	0.55
c_{11}											-
c_{12}											
c_{13}											
c_{14}											
c_{15}											
c_{16}											
c_{17}											
c_{18}											
c_{19}											
c_{20}											
c_{21}											

Table 4: Expected Mutual Information Measures (Constructs 1-11)

	c_{12}	c_{13}	c_{14}	c_{15}	c_{16}	c_{17}	c_{18}	c_{19}	c_{20}	c_{21}
c_1	0.35	0.30	0.72	0.75	0.47	0.54	0.64	0.55	0.56	0.79
c_2	0.46	0.47	0.70	0.80	0.33	0.37	0.39	0.58	0.49	0.65
c_3	0.13	0.12	0.46	0.23	0.68	0.28	0.31	0.29	0.37	0.37
c_4	0.38	0.40	0.57	0.92	0.26	0.26	0.52	0.78	0.52	0.57
c_5	0.45	0.45	0.19	0.23	0.10	0.16	0.14	0.24	0.14	0.31
c_6	0.06	0.06	0.16	0.30	0.20	0.20	0.24	0.16	0.25	0.15
c_7	0.26	0.26	0.54	0.74	0.25	0.40	0.50	0.43	0.45	0.59
c_8	0.18	0.24	0.83	0.47	0.49	0.74	0.54	0.34	0.53	0.54
c_9	0.45	0.55	0.28	0.23	0.10	0.24	0.23	0.27	0.22	0.41
c_{10}	0.35	0.39	0.74	0.60	0.44	0.69	0.93	0.41	0.70	0.75
c_{11}	0.32	0.29	0.70	0.83	0.36	0.69	0.68	0.59	0.51	0.68
c_{12}	-	0.45	0.26	0.29	0.13	0.19	0.20	0.39	0.19	0.43
c_{13}		-	0.41	0.29	0.12	0.35	0.32	0.28	0.33	0.53
c_{14}			-	0.64	0.46	0.77	0.72	0.37	0.61	0.78
c_{15}				-	0.23	0.46	0.66	0.65	0.66	0.76
c_{16}					-	0.28	0.31	0.29	0.37	0.37
c_{17}						-	0.89	0.23	0.56	0.65
c_{18}							-	0.43	0.46	0.66
c_{19}								-	0.51	0.53
c_{20}									-	0.57
c_{21}										-

Table 5: Expected Mutual Information Measures (Constructs 12-21)

Concept c_{j_1}		Concept c_{j_2}		EMIM
c_1	information retrieval	c_{11}	hypertext	0.97
c_{10}	machine learning	c_{18}	user-oriented systems	0.93
c_4	expert systems	c_{15}	cluster analysis	0.92
c_{17}	adaptive systems	c_{18}	user-oriented systems	0.89
c_8	knowledge acquisition	c_{14}	user profile	0.83
c_{11}	hypertext	c_{15}	cluster analysis	0.82
c_1	information retrieval	c_{21}	psychology	0.79
c_2	classification	c_{15}	cluster analysis	0.79
c_8	knowledge acquisition	c_{10}	machine learning	0.79
c_4	expert systems	c_{19}	rule-based systems	0.78
c_{14}	user profile	c_{21}	psychology	0.78
c_7	graph theory	c_{15}	cluster analysis	0.74
c_{10}	machine learning	c_{20}	fuzzy logic	0.70
c_3	personal construct theory	c_{16}	repertory grid	0.68
c_9	programming tutors	c_{13}	student models	0.55
c_{13}	student models	c_{21}	psychology	0.53
c_3	personal construct theory	c_8	knowledge acquisition	0.49
c_5	intelligent tutoring systems	c_9	programming tutors	0.49
c_{12}	education training	c_{13}	student models	0.45
c_6	rough sets	c_7	graph theory	0.37

Table 6: Constructs with maximal information measure

p_1	$=$	c_6	rough sets
p_2	$=$	c_7	graph theory
p_3	$=$	c_{15}	cluster analysis
p_4	$=$	c_4	expert systems
p_5	$=$	c_{19}	rule-based systems
p_6	$=$	c_2	classification
p_7	$=$	c_{11}	hypertext
p_8	$=$	c_1	information retrieval
p_9	$=$	c_{21}	psychology
p_{10}	$=$	c_{14}	user profile
p_{11}	$=$	c_8	knowledge acquisition
p_{12}	$=$	c_{10}	machine learning
p_{13}	$=$	c_3	personal construct theory
p_{14}	$=$	c_{13}	student models
p_{15}	$=$	c_{18}	user-oriented systems
p_{16}	$=$	c_{20}	fuzzy logic
p_{17}	$=$	c_{16}	repertory grid
p_{18}	$=$	c_{12}	education training
p_{19}	$=$	c_9	programming tutors
p_{20}	$=$	c_{17}	adaptive systems
p_{21}	$=$	c_5	intelligent tutoring systems

Table 7: A permutation for concepts

	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9	d_{10}	d_{11}	d_{12}
c_1	1	1	1	1	0	0	1	1	1	1	1	1
c_2	1	0	1	1	1	1	0	1	1	1	0	1
c_3	0	0	0	0	0	0	0	0	0	0	0	0
c_4	0	0	0	0	0	0	1	1	0	0	1	0
c_5	0	0	0	0	0	0	0	0	0	0	0	0
c_6	0	0	0	1	0	0	0	0	1	0	0	0
c_7	0	0	0	1	1	1	0	0	1	0	0	0
c_8	0	0	0	1	0	0	0	0	1	0	0	0
c_9	0	0	0	0	0	0	0	0	0	0	0	0
c_{10}	0	1	0	1	0	0	1	0	1	0	0	0
c_{11}	1	1	1	1	0	0	1	0	1	1	0	0
c_{12}	0	0	0	0	0	0	0	0	0	0	0	0
c_{13}	0	0	0	0	0	0	1	0	0	0	0	0
c_{14}	0	1	0	0	0	0	1	0	1	1	0	0
c_{15}	0	1	0	1	1	1	0	0	1	1	0	1
c_{16}	0	0	0	0	0	0	0	0	0	0	0	0
c_{17}	0	1	0	1	0	0	1	0	1	1	0	0
c_{18}	0	1	0	1	0	0	1	1	1	1	0	0
c_{19}	0	0	0	0	0	0	0	1	0	0	1	0
c_{20}	0	1	0	0	0	0	1	1	0	0	1	0
c_{21}	0	0	0	0	0	0	1	0	0	0	0	0

Table 8: Binary repertory grid of the expert's evaluation of documents 1-12

	d_{13}	d_{14}	d_{15}	d_{16}	d_{17}	d_{18}	d_{19}	d_{20}	d_{21}	d_{22}	d_{23}	d_{24}
c_1	1	1	0	0	0	0	0	0	1	0	0	0
c_2	0	1	0	1	1	0	0	1	1	0	0	0
c_3	0	0	0	1	1	1	1	1	0	0	0	0
c_4	1	0	1	1	1	0	1	1	0	1	1	1
c_5	0	0	1	0	0	0	0	0	0	1	1	1
c_6	0	0	0	0	0	0	0	0	1	0	0	0
c_7	0	0	0	0	0	0	0	0	1	0	0	0
c_8	0	1	0	1	1	1	1	1	0	0	0	0
c_9	0	0	1	0	0	0	0	0	0	1	1	1
c_{10}	0	1	1	1	1	1	1	1	1	0	1	0
c_{11}	0	1	0	0	0	0	0	0	0	0	0	0
c_{12}	0	0	1	0	0	0	0	0	0	1	1	1
c_{13}	0	0	1	0	0	0	0	0	0	1	1	1
c_{14}	0	1	1	1	1	1	1	1	0	0	1	0
c_{15}	0	1	0	0	0	0	0	0	1	0	0	0
c_{16}	0	0	0	1	1	1	1	1	0	0	0	0
c_{17}	0	1	1	1	1	1	1	1	0	0	1	1
c_{18}	0	1	1	1	1	1	1	1	0	1	1	1
c_{19}	1	0	1	1	1	1	1	0	0	1	1	1
c_{20}	0	0	0	1	1	0	1	0	0	0	0	0
c_{21}	0	0	1	1	1	1	1	1	0	1	1	1

Table 9: Binary repertory grid of the expert's evaluation of documents 13-24

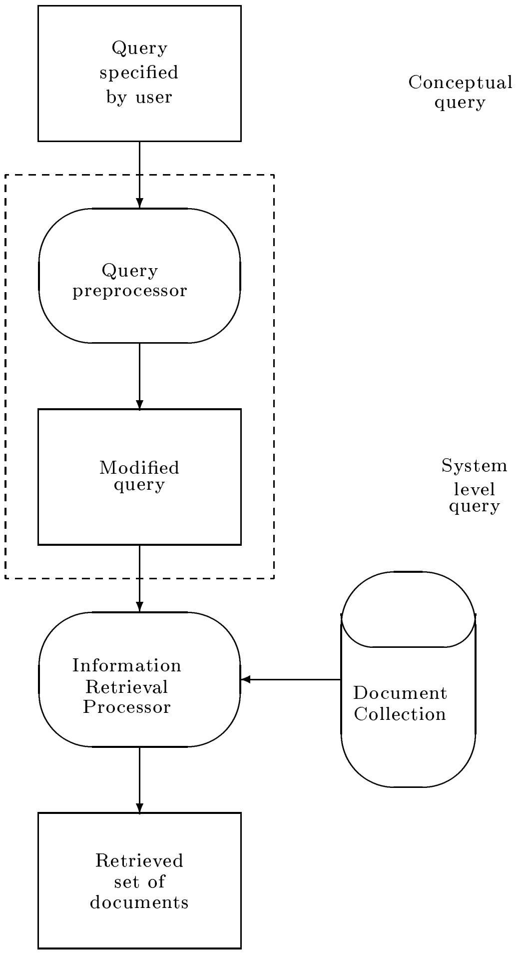

Figure 1: Information retrieval process with conceptual query translation

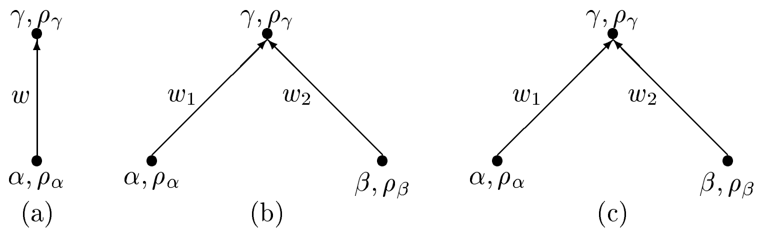

Figure 2: Propagation of uncertainty

```
%A B. P. McCune
%A R. M. Tong
%A J. S. Dean
%A D. G. Shapiro
%T RUBRIC: A System for Rule-Based Information Retrieval
%J IEEE Transactions on Software Engineering
%V SE-11
%N 9
%D September 1985
%P 939-945
%O 6 references
%K information retrieval, rule-based systems, RUBRIC
%X A research prototype software system for conceptual information retrieval has been
developed.  The goal of the system, called RUBRIC, is to provide more automated and
relevant access to unformatted textual databases.  The approach is to use production
rules from artificial intelligence to define a hierarchy of retrieval subtopics, with
fuzzy context expressions and specific word phrases at the bottom.  RUBRIC allows the
definition of detailed queries starting at a conceptual level, partial matching of a
query and a document, selection of only the highest ranked documents for presentation
to the user, and detailed explanation of how and why a particular document was selected.
Initial experiments indicate that a RUBRIC rule set better matches human retrieval
judgement than a standard Boolean keyword expression, given equal amount of effort in
defining each.  The techniques presented may be useful in stand-alone retrieval systems,
front-ends to existing systems, or real-time document filtering and routing.
```

Figure 3: A typical entry from bibliography

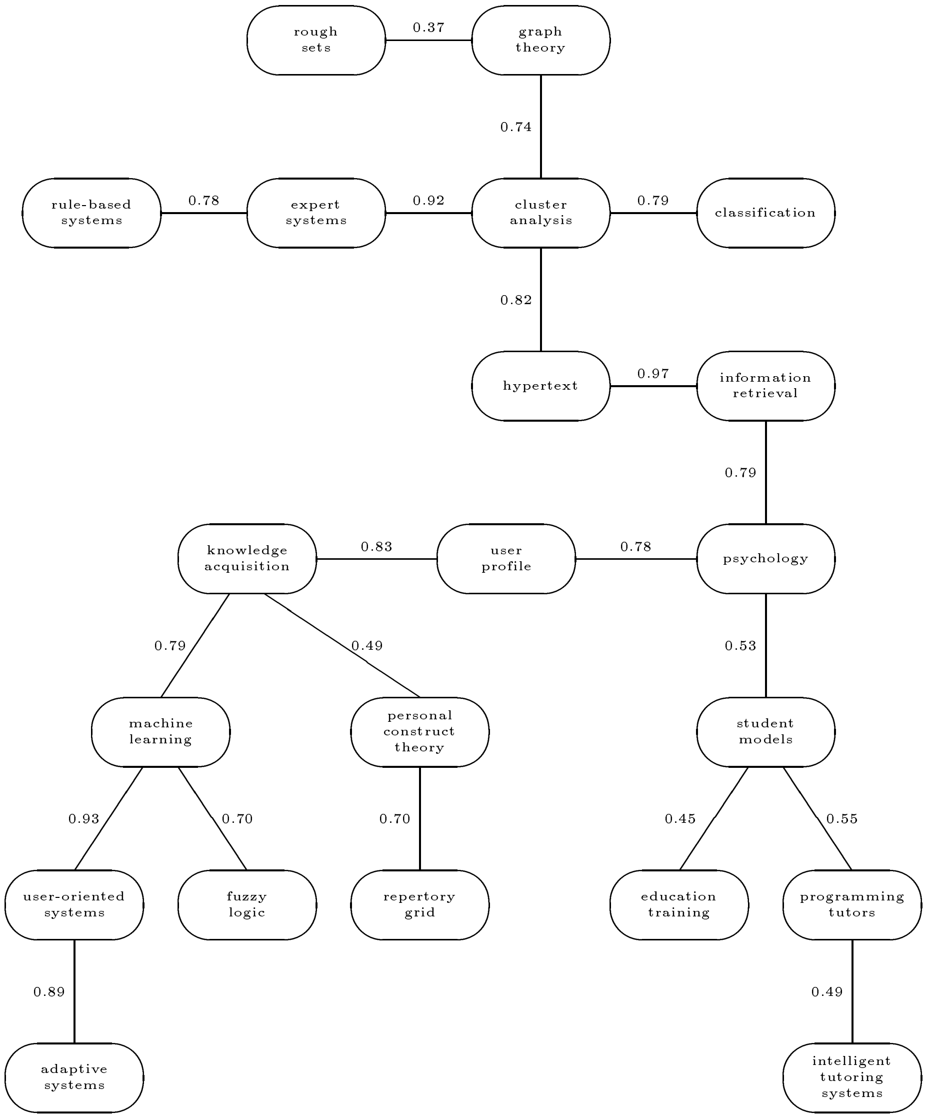

Figure 4: Maximum weight dependence tree

Weight	Keyword	Weight	Keyword	Weight	Keyword	Weight	Keyword
1.43119	part	2.58968	match	2.47738	equ	2.73079	high
1.66536	cal	3.16398	bottom	0.67098	develop	1.35758	provid
1.51465	prototyp	1.72493	acces	1.61099	artific	1.15227	set
1.51465	speci	1.62843	retrief	1.43119	hum	1.72493	level
3.02123	rubr	1.93313	def	1.92689	docu	1.86439	init
1.86439	que	2.29759	keyword	2.29759	softwar	2.47738	phras
1.04920	pres	1.81752	select	1.43119	autom	3.16398	stand
1.86439	exist	2.04418	effort	3.16398	start	1.26124	research
3.16398	filt	1.47153	qu	3.16398	amount	2.29759	fuzz
2.06986	expres	1.35758	produc	3.16398	subtop	0.99799	user
2.73079	front	2.47738	textu	1.38269	rl	2.73079	rout
1.72493	context	1.61099	databas	1.61099	relev	1.51465	tim
2.47738	rank	0.43757	system	2.47738	judg	0.84310	approach
2.29759	boolean	1.26124	intellig	1.72493	goal	1.99838	conceptu
2.47738	unformat	1.99838	detail	0.99799	inform	2.15813	end
2.29759	word	2.15813	exper	1.86439	real	2.47738	standard
1.47153	hierarch	2.47738	defin	0.84310	techniqu	2.47738	explan

Figure 5: Weighted index terms selected from a document

```
d104([(1.43119,part),(2.58968,match),(2.47738,equ),(2.73079,high),(1.66536,cal),
(3.16398,bottom),(0.67098,develop), ..., (0.84310,techniqu),(2.47738,explan)]).
```

Figure 6: Prolog predicate to show index terms and their weight

```
conceptplus9([instruc, shortcam, troubl, opinion, day, ..., geomes, tut, proof]).
conceptminus9([model, aris, method, knowledg, provid, ..., ibi, pos, issu]).
```

Figure 7: Prolog predicate to show concept c_9

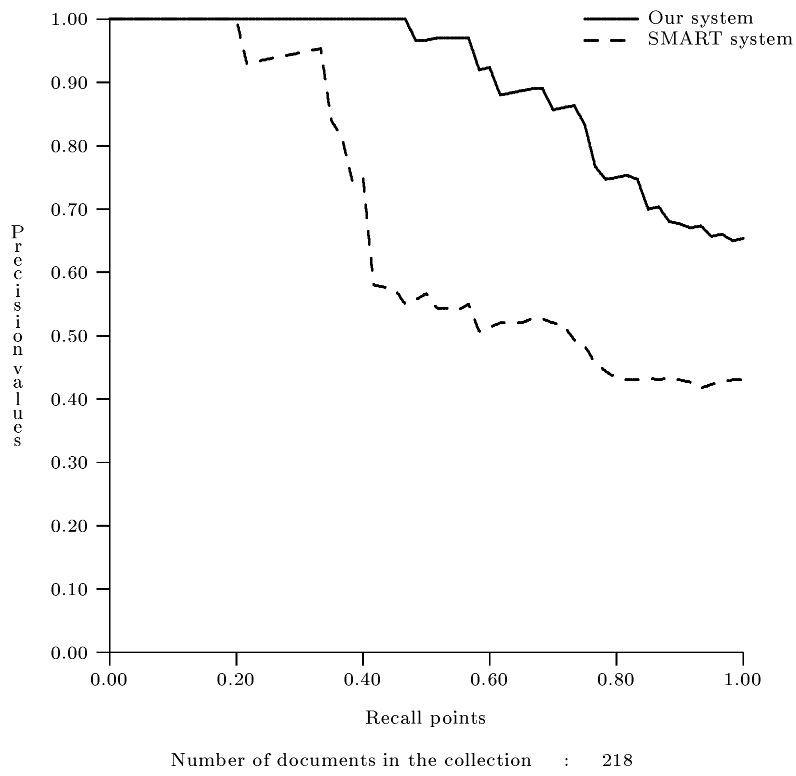

Figure 8: Average Case Performance: Precision vs. Recall (Query 1)

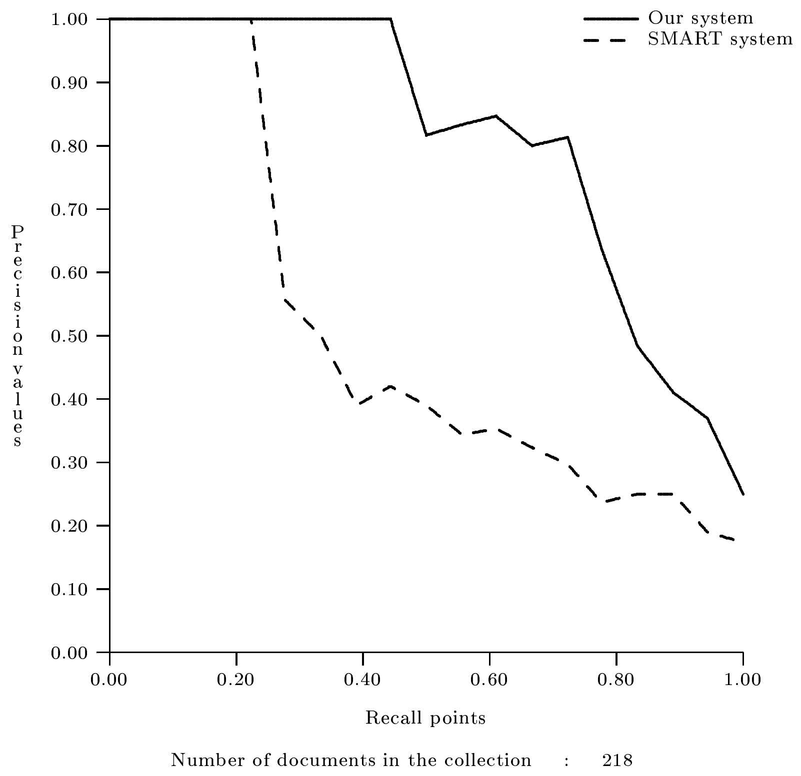

Figure 9: Best Case Performance: Precision vs. Recall (Query 5)

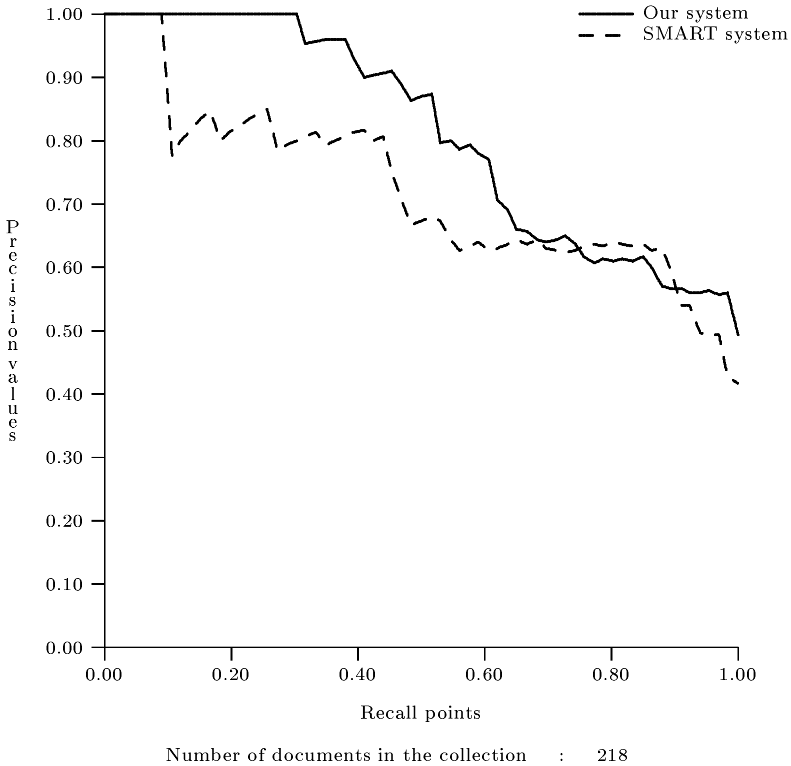

Figure 10: Worst Case Performance: Precision vs. Recall (Query 8)

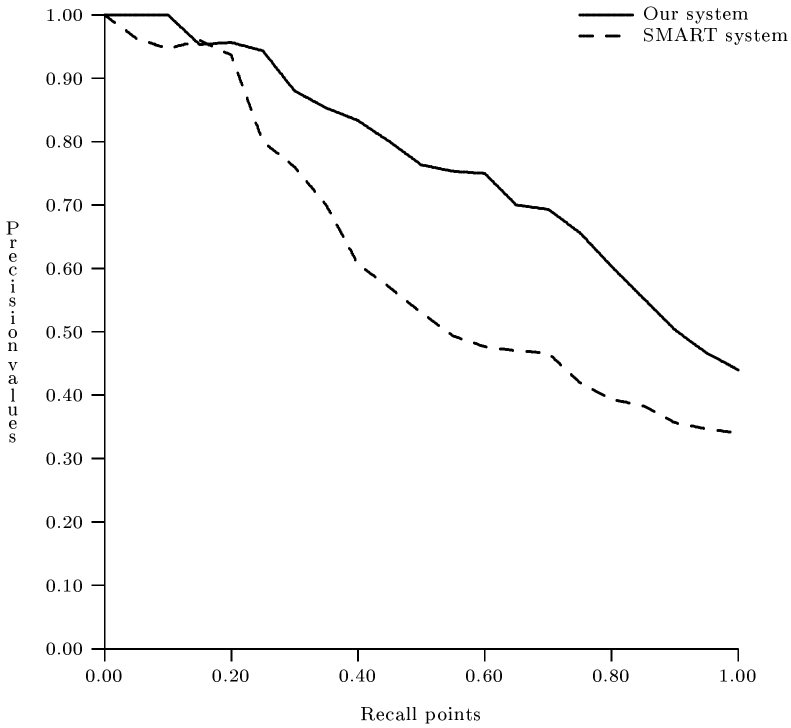

Figure 11: Precision-recall curve (Average performance over 14 queries)