Preferential Infinitesimals for Information Retrieval

Maria Chowdhury, Alex Thomo, and William W. Wadge

Abstract In this paper, we propose a preference framework for information retrieval in which the user and the system administrator are enabled to express preference annotations on search keywords and document elements, respectively. Our framework is flexible and allows expressing preferences such as "*A* is infinitely more preferred than *B*," which we capture by using *hyperreal numbers*. Due to the widespread of XML as a standard for representing documents, we consider XML documents in this paper and propose a consistent preferential weighting scheme for nested document elements. We show how to naturally incorporate preferences on search keywords and document elements into an IR ranking process using the well-known TF-IDF ranking measure.

1 Introduction

In this paper, we propose a framework for preferential information retrieval by incorporating in the document ranking process preferences given by the user or the system administrator. Namely, in our proposed framework, the user has the option of weighting the search keywords, whereas the system administrator has the option of weighting structural elements of the documents. We address both facets of preferential weighting by using hyperreal numbers, which form a superset of the real numbers, and in our context, serve the purpose of specifying natural preferences of the form "*A* is infinitely more preferred than *B*."

Keyword Preferences. To illustrate preferences on keywords, suppose that a user wants to retrieve documents on research and techniques for "music-information-retrieval." Also, suppose that the user is a fan of Google technology. As such, this user would probably give to a search engine the keywords:

Department of Computer Science, University of Victoria, Canada,
e-mail: {mwchow,thomo,wwadge}@cs.uvic.ca

music-information-retrieval, google-search, google-ranking.

It is interesting to observe that if the user specifies these keywords in Google, then she gets a list of only *three*, low quality, pages. What happens is that the true, highly informative pages about "music-information-retrieval" are lost (or insignificantly ranked) in the quest of trying to serve the "google-search" and "google-ranking" keywords. Unfortunately, in Google and other search engines, the user cannot explicitly specify her real preferences among the specified keywords. In this example, what the user needs is a mechanism for saying that "music-information-retrieval" is of primary importance or *infinitely* more important than "google-search" and "google-ranking," and thus, an informative page about "music-information-retrieval" should be retrieved and highly ranked even if it does not relate to Google technologies.

Structural Preferences. The other facet of using preferential weights is for system administrators to annotate structural parts of the documents in a given corpus. In practice, most of the documents are structured, and often, certain parts of them are more important than others. While our proposed ideas can be applied on any corpus of structured documents, due to the wide spread of XML as a standard for representing documents, we consider in this paper XML documents which conform to a given schema (DTD). In the same spirit as for keyword preferences, we will use hyperreal weights to denote the importance of different elements in the schema and documents.

To illustrate preferences on structural parts of documents, suppose that we have a corpus of documents representing research papers, and a user is searching for a specific keyword. Now, suppose that the keyword occurs in the *title* element of one paper and in the *references* element of another paper. Intuitively, the paper having the keyword in the *title* should be ranked higher than the paper containing the keyword in the *references* element as the title of a paper usually bears more representative and concise information about the paper than the reference entries do. In fact, one could say that terms in the title (and abstract) are *infinitely* more important than terms in the references entries as the latter might be there completely incidental.

While weighting of certain parts of documents has been considered and advocated in the folklore (cf. [5, 8]), to the best of our knowledge there is no work dealing with inferring a consistent weighting scheme for nested XML elements based on the weights that a system administrator gives to DTD elements. As we explain in Section 4, there are tradeoffs to be considered and we present a solution that properly normalizes the element weights producing values which are consistent among sibling elements and never greater than the normalized weight of the parent element, thus respecting the XML hierarchy.

Contributions. Specifically, our contributions in this paper are as follows.

1. We propose using hyperreal numbers (see [6, 7]) to capture both "quantitative" and "qualitative" user preferences on search keywords. The set of hyperreal numbers includes the real numbers which can be used for expressing "quantitative" preferences such as, say "*A* is twice more preferred than *B*," as well as *infinites-*

imal numbers, which can be used to express "qualitative" preferences such as, say "*A* is infinitely more preferred than *B*." We argue that without such qualitative preferences there is no guarantee that an IR system would not override user preferences in favor of other measures that the system might use.

2. We extend the ideas of using hyperreal numbers to annotating XML (DTD) schemas. This allows system administrators to preferentially weight structural elements in XML documents of a given corpus. We present a normalization method which produces consistent preferential weights for the elements of any XML document that complies to an annotated DTD schema.
3. We adapt the well-known TF-IDF ranking in IR systems to take into consideration the preferential weights that the search keywords and XML elements can have. Our extensions are based on symbolic computations which can be effectively computed on expressions containing hyperreal numbers.
4. We present (in the appendix) illustrative practical examples which demonstrate the usefulness of our proposed preference framework. Namely, we use a full collection of speeches from the Shakespeare plays, and a diverse XML collection from INEX ([13]). In both these collections, we observed a clear advantage of our preferential ranking over the ranking produced by the classical TF-IDF method. We believe that these results encourage incorporating both quantitative and (especially) qualitative preferences into other ranking methods as well.

Organization. The rest of the paper is organized as follows. In Section 2, we give an overview of hyperreal numbers and their properties. In Section 3, we present hyperreal preferences for annotating search keywords. In Section 4, we propose annotated DTDs for XML documents and address two problems for consistent weighting of document elements. In Section 5, we show how to extend the TF-IDF ranking scheme to take into consideration the hyperreal weights present in the search keywords and document elements. In Appendix, we present experimental results.

2 Hyperreal Numbers

Hyperreal numbers were introduced in calculus to capture "infinitesimal" quantities which are infinitely small and yet not equal to zero. Formally, a number ε is said to be *infinitely small* or *infinitesimal* (cf. [6, 7]) iff $-a < \varepsilon < a$ for every positive *real* number a. Hyperreal numbers contain all the real numbers and also all the infinitesimal numbers. There are principles (or axioms) for hyperreal numbers (cf. [7]) of which we mention:

Extension Principle.

1. The real numbers form a subset of the hyperreal numbers, and the order relation $x < y$ for the real numbers is a subset of the order relation for the hyperreal numbers.
2. There exists a hyperreal number that is greater than zero but less than every positive real number.

3. For every real function f, we are given a corresponding hyperreal function f^* which is called the *natural extension* of f.

Transfer Principle. Every real statement that holds for one or more particular real functions holds for the hyperreal natural extensions of these functions.

In short, the Extension Principle gives the *hyperreal* numbers and the Transfer Principle enables carrying out computation on them. The Extension Principle says that there does exist an infinitesimal number, for example ε. Other examples of hyperreals numbers, created using ε, are: ε^3, $100\varepsilon^2 + 51\varepsilon$, $\varepsilon/300$.

For $a,b,r,s \in \mathbb{R}^+$ and $r < s$, we have $a\varepsilon^r < b\varepsilon^s$, regardless of the relationship between a and b.

If $a\varepsilon^r$ and $b\varepsilon^s$ are used for example to denote two preference weights, then $a\varepsilon^r$ is “infinitely better” than $b\varepsilon^s$ even though a might be much bigger than b, i.e. coefficients a and b are insignificant when the powers of ε are different. On the other hand, when comparing two preferential weights of the same power, as for example $a\varepsilon^r$ and $b\varepsilon^r$, the magnitudes of coefficients a and b become important. Namely, $a\varepsilon^r \leq b\varepsilon^r$ ($a\varepsilon^r > b\varepsilon^r$) iff $a \leq b$ ($a > b$).

3 Keyword Preferences

We propose a framework where the user can preferentially annotate the keywords by *hyperreal numbers*.

Using hyperreal annotations is essential for reasoning in terms of “infinitely more important,” which is crucially needed in a scenario with numerous documents. This is because preference specification using only real numbers suffers from the possibility of producing senseless results as those preferences can get easily absorbed by other measures used by search engines. For instance, continuing the example given in the Introduction,

music-information-retrieval, google-search, google-ranking,

suppose that the user, dismayed of the poor result from Google, containing only three low quality pages, changes the query into[1]

music-information-retrieval OR google-search OR google-ranking.

It is interesting to observe that if the user specifies this (modified) query in Google, then what she gets is a list of *many* web-pages (documents)! These pages are ranked by their Google-computed importance which is by far biased toward

[1] This second query style corresponds more closely than the first to what is known in the folklore as the popular “free text query:” a query in which the terms of the query are typed freeform into the search interface (cf. [5, 8]).

general pages about “google-search” and “google-ranking” rather than “music-information-retrieval.” The true pages about “music-information-retrieval” are simply buried under tons of other pages about “google-search” and “google-ranking” that are highly ranked, but contain “music-information-retrieval” either incidentally or not at all. Unfortunately, in Google and other search engines, the user cannot explicitly specify her real preferences among the specified keywords. In this example, what the user needs is a mechanism for saying that “music-information-retrieval” is of primary importance or infinitely more important than “google-search” and “google-ranking.”

But, let us suppose for a moment that Google would allow users to specify preferences expressed by real numbers. Now, imagine the user who is trying to convey that her “first and foremost” preference is for documents on “music-information-retrieval” rather than general documents about Google technology. For this, the user specifies that *music-information-retrieval* is 100 times more important than *google-search*. After all, “100 times more important” seems quite convincing in colloquial talking! However, what would happen if, according to the score computed by the search engine, general documents about *google-search* were in fact 1000 times more important than documents about *music-information-retrieval*? If the user preference levels were used to simply boost the computed document score by the same factor, then still, documents about *google-search* would be ranked higher than documents about *music-information-retrieval*. What the user would experience in this case is an “indifferent” search engine with respect to her preferences.

The solution we propose is to use hyperreal numbers for expressing preferential weights. In order to always have an effective comparison of documents with respect to a user query, we will fix an infinitesimal number, say ε, and build expressions on it. By the Extension Principle, such a number does exist. Now, we give the following definition.

Definition 1. An *annotated free text query* is simply a set of keywords (terms) with preference weights which are polynomials of ε.

For all our practical purposes it suffices to consider only polynomials with coefficients in $\mathbb{R}^+$. For example, $3+2\varepsilon+4\varepsilon^2$.

By making this restriction we are able to perform symbolic (algorithmic) computations on expressions using ε. All such expressions translate into operations on polynomials with real coefficients for which efficient algorithms are known (we will namely need to perform polynomial additions, multiplications and divisions[2]).

Let us illustrate our annotated queries by continuing the above example. The user can now give

$$\textit{music-information-retrieval, google-search} : \varepsilon\textit{, google-ranking} : \varepsilon^2$$

[2] The division is performed by first factoring the highest power of ε. For example, $(6+3\varepsilon+3\varepsilon^2)/(4+2\varepsilon+3\varepsilon^2)$ is first transformed into $(6\varepsilon^{-2}+4\varepsilon^{-1}+3)/(3\varepsilon^{-2}+2\varepsilon^{-1}+4)$, and then we perform the division as we would do for $(6x^2+4x+3)/(3x^2+2x+4)$. Observe that, as ε is infinitely small, ε^{-1} is *infinitely big*.

to express that she wants to find documents on Music Information Retrieval and she is interested in the Google technology for retrieving and ranking music. However, by leaving intact the *music-information-retrieval* and annotating *google-search* by ε and *google-ranking* by ε^2, the user makes her intention explicit that a document on *music-information-retrieval* is infinitely more important than any document on simply *google-search* or *google-ranking*. Furthermore, in accord with the above user expression, documents on *music-information-retrieval* and/or *google-search* are infinitely more important than documents on simply *google-ranking*. Of course, among documents on Music Information Retrieval, those which are relevant to Google search and Google ranking are more important.

We note that our framework also allows the user to specify "soft" preference levels. For example, suppose that the user changes her mind and prefers to have both *google-search* and *google-ranking* in the same "hard" preference level as determined by the power of infinitesimal ε. However, she still prefers, say "twice more," *google-search* over *google-ranking*. In this case, the user gives

$$\textit{music-information-retrieval, google-search} : 2\varepsilon, \textit{google-ranking} : \varepsilon.$$

4 Preferentially Annotated XML Schemas

In this section, we consider the problem of weighting the structural elements of documents in a corpus with the purpose of influencing an information retrieval system to take into account the importance of different elements during the process of document ranking. Due to the wide spread of XML as a standard for representing documents, we consider in this paper XML documents which conform to a given schema (DTD). In the same spirit as in the previous section, we will use hyperreal weights to denote the importance of different elements in the schema and documents.

While the idea of weighting the document elements is old and by now part of the folklore (cf. [8]), to the best of our knowledge, there is no work that systematically studies the problem of weighting XML elements. The problem becomes challenging when elements can possibly be nested inside other elements which can be weighted as well, and one wants to achieve a consistent weight normalization reflecting the true preferences of a system administrator. Another challenging problem, as we explain in Subsection 4.4, is determining the right mapping of weights from the elements of a DTD schema into the elements of XML documents.

4.1 Hyperreal Weights

In our framework, the system administrator is enabled to set the importance of various XML elements/sections in a DTD schema. For example, she can specify that the *keywords* elements of documents in an XML corpus, with "research activities" as the main theme, is more important than than a section, say on *related work*. Intuitively, an occurrence of a search term in the *keywords* section is way more important than an occurrence in the *related work* section as the occurrence in the latter might be completely incidental or only loosely related to the main thrust of the document.

Thus, in our framework, we allow the annotation of XML elements by weights being, as in the previous section, polynomials of a (fixed) infinitesimal ε.

4.2 DTDs

Let Σ be the (finite) tag alphabet of a given XML collection, i.e. each tag is an element of Σ. Then, a DTD D is a pair (d,r) where d is a function mapping Σ-symbols to regular expressions on Σ and r is the root symbol (cf. [2]).

A *valid* XML document complying to a DTD $D = (d,s)$ can be viewed as a tree, whose root is labeled by r and every node labeled, say by a, has a sequence of children whose label concatenation, say $bc\ldots x$, is in $L(d(a))$.

A simple example of a DTD defining the structure of some XML research documents is the following:

$$\begin{aligned}\text{paper} &\rightarrow \text{preamble body}\\ \text{preamble} &\rightarrow \text{title author}^{+} \text{ abstract keywords}\\ \text{body} &\rightarrow \text{introduction section}^{*} \text{ related-work? references}\end{aligned}$$

where '$+$' implies "one or more," '$*$' implies "zero or more" and '?' implies "zero or one" occurrences of an element.

In essence, a DTD D is an extended context-free grammar, and a valid XML document with respect to D is a parse tree for D.

4.3 Annotated DTDs

To illustrate annotated DTDs, let us suppose that the system administrator wants to express that in the *body* element, the *introduction* is twice more important than a *section*, and both are infinitely more important than *related-work* and *references*, with the latter being infinitely less important than the former, we would annotate the rule for *body* as follows:

$$\text{body} \rightarrow (\text{introduction} : 2)\ (\text{section} : 1)^{*}\ (\text{related-work} : \varepsilon)?\ (\text{references} : \varepsilon^{2}).$$

Further annotations, expressing for example that the *preamble* element is three times more important than the *body* element, and in the *preamble*, the *keywords* element is 5 times more important than *title* and 10 times more important than the rest, would lead to having the following annotated DTD:

$$\begin{aligned}\text{paper} &\rightarrow (\text{preamble} : 3)\ (\text{body} : 1)\\ \text{preamble} &\rightarrow (\text{title} : 2)\ (\text{author} : 1)^{+}\ (\text{abstract} : 1)\ (\text{keywords} : 10)\\ \text{body} &\rightarrow (\text{introduction} : 2)\ (\text{section} : 1)^{*}\ (\text{related-work} : \varepsilon)?\ (\text{references} : \varepsilon^{2}).\end{aligned}$$

Since an annotated element can be nested inside other elements, which can be annotated as well, the natural question that now arises is: How to compute the actual weight of an element in a DTD? One might be tempted to think that the actual weight of an element should obtained by multiplying its (annotation) weight by the weights of all its ancestors. However by doing that, we could get strange results as for example a possibly increasing importance weight as we go deep down in the XML element hierarchy.

What we want here is “an element to never be more important than its parent.” For this, we propose normalizing the importance weights assigned to DTD elements. There are two ways for doing this. Either divide the weights of a rule by the sum of the rule’s weights, or divide them by the maximum weight of the rule. In the first way, the weight of the parent will be divided among the children. On the other hand, in the second way, the weight of the most important child will be equal to the weight of the parent.

The drawback of the first approach is that the more children there are, the lesser their weight is. Thus, we opt for the second way of weight normalization as it better corresponds to the intuition that nesting in XML documents is for adding structure to text rather than hierarchically dividing the importance of elements.

For example, in the above DTD, for the children of *preamble*, we normalize dividing by the greatest weight of the rule, which is 10. Normalizing in this way the weights of all the rules, we get

$$\begin{aligned}\text{paper} &\rightarrow (\text{preamble} : 1)\ (\text{body} : 1/3)\\ \text{preamble} &\rightarrow (\text{title} : 1/5)\ (\text{author} : 1/10)^{+}\ (\text{abstract} : 1/10)\ (\text{keywords} : 1)\\ \text{body} &\rightarrow (\text{introduction} : 1)\ (\text{section} : 1/2)^{*}\ (\text{related-work} : \varepsilon/2)?\ (\text{references} : \varepsilon^{2}/2).\end{aligned}$$

After such normalization, for determining the actual weight of an element, we multiply its DTD weight by the weights of all its ancestors. For example, the weight of a *section* element is $(1/3) \cdot (1/2)$.

As mentioned earlier, under this weighting scheme, the most important child of a parent has the same importance as the parent itself. Thus, for instance, element *introduction* has the same importance $(1/3)$ as its parent *body*. Note that the weight normalization can of course be automatically done by the system, while we annotate using numbers that are more comfortable to write.

4.4 Weighting Elements of XML Documents

In the previous section, we described how to compute the weight of an element in a DTD. However, the weight of an element in an XML document depends not only on the DTD, but also on the particular structure of the document. This is because the same element might occur differently nested in different valid XML documents. For example, if we had an additional rule, section $\rightarrow$ (title : 1) (text : $1/2$), in our annotated DTD, then, given a valid XML document, the weight of a *title* element depends on the particular nesting of this element. Namely, if the nesting is

$$\langle\text{paper}\rangle\langle\text{preamble}\rangle\langle\text{title}\rangle\ldots\langle/\text{title}\rangle\ldots\langle/\text{preamble}\rangle\ldots\langle/\text{paper}\rangle$$

then the normalized weight of the *title* element is $1/5$. On the other hand, if the nesting is

$$\langle\text{paper}\rangle\ldots\langle\text{body}\rangle\langle\text{section}\rangle\langle\text{title}\rangle\ldots\langle/\text{title}\rangle\ldots\langle/\text{section}\rangle\ldots\langle/\text{body}\rangle\langle/\text{paper}\rangle$$

then the normalized weight of the *title* element is $(1/3)\cdot(1/2)\cdot 1 = 1/6$.

In general, in order to derive the correct weight of an element in an XML document, we need to first build the element tree of the document. This will be a parse tree for the context-free grammar corresponding to the DTD. For each node a of this tree with children $bc\ldots x$, there is a unique rule $a \rightarrow r$ in the DTD such that word $bc\ldots x$ is in $L(r)$.

Naturally, we want to assign weights to a's children b, c, $\ldots$, x based on the weights in annotated expression r. Thus, the question becomes how to map the weights assigned to the symbols of r to the symbols of word $bc\ldots x$.

Since $b, c, \ldots, x$ occur in r, this might seem as a straightforward matter. However, there is subtlety here arising from the possibility of ambiguity in the regular expression. For example, suppose the (annotated) expression r is $(b:1+c:1)^*(b:2)(b:3)^*$, and element a has three children labeled by b. Surely, bbb is in $L(r)$, but what label should we assign to each of b's? There are three different ways of assigning weights to these b's: $(b:1)(b:1)(b:2)$, $(b:1)(b:2)(b:3)$, and $(b:2)(b:3)(b:3)$.

However, according to the SGML standard (cf. [3]), the only allowed regular expressions in the DTD rules are those for which we can uniquely determine the correspondence between the symbols of an input word and the symbols of the regular expression.

These expressions are called "1-unambiguous" in [3].

For such an expression r, given a word $bc\ldots x$ in $L(r)$, there is a unique mapping of word symbols b, c, $\ldots$, x to expression symbols. Thus, when r is annotated with symbol weights, we can uniquely determine the weights for each of the b, c, $\ldots$, x word symbols.

Based on all the above, we can state the following theorem.

Theorem 1. *If T is a valid XML tree with respect to an annotated DTD D, then based on the weight annotations of D, there is a unique weight assignment to each node of T.*

Now, given an XML document, since there is unique path from the root of an XML document to a particular element, we have that

Corollary 1. *Each element of a valid XML document is assigned a unique weight.*

The unique weight of an element is obtained by multiplying its local node weight with the weights of the ancestor nodes on the unique path connecting the element with the document root.[3]

5 Preferential Term Weighting and Document Scoring

Formally, let V (vocabulary) be the set of distinctive terms in a collection C of documents. Denote by m and n the cardinalities of V and C respectively. Let t_i be term in V and d_j a document in C. Suppose that t_i occurs f_{ij} times in d_j. Then, the normalized term frequency of t_i in d_j is

$$tf_{ij} = \frac{f_{ij}}{\mathrm{m}ax\{f_{1j},\ldots,f_{mj}\}},$$

where the maximum is in fact computed over the terms that appear in document d_j.

Considering now XML documents whose elements are weighted based on annotated DTDs, we have that *not* all occurrences of a term "are created equal." For instance, continuing the example in Section 4, an occurrence of a term t_i in the *keywords* element of a document is 5 times more important than an occurrence (of t_i) in the *title*, and infinitely more important than an occurrence in the *related-work* element.

Hence, we refine the TF measure to take the importance of XML elements into account. When an XML document conforms to an annotated DTD, each element e_k will be accordingly weighted, say by w_k.

Suppose that term t_i occurs f_{ijk} times in element e_k of document d_j. Now, we define the normalized term frequency of t_i in d_j as

$$tf_{ij} = \frac{\sum_k w_k f_{ijk}}{\mathrm{m}ax\{\sum_k w_k f_{1jk},\ldots,\sum_k w_k f_{mjk}\}}.$$

The other popular measure used in Information Retrieval is the *inverse document frequency* (IDF) which is used jointly with the TF measure. IDF is based on the fraction of documents which contain a query term. The intuition behind IDF is that a query term that occurs in numerous documents is not a good discriminator, or does not bear to much information, and thus, should should be given a smaller weight than other terms occurring in few documents. The weighting scheme known as TF*IDF, which multiplies the TF measure by the IDF measure, has proved to be a powerful heuristic for document ranking, making it the most popular weighting scheme in Information Retrieval (cf. [10, 5, 8]).

Formally, suppose that term t_i occurs n_i times in a collection of n elements. Then, the *inverse document frequency* of t_i is defined to be

[3] All weights are considered being normalized.

$$idf_i = \log \frac{n}{n_i}.$$

IDF has a natural explanation from an information theoretic point of view. If we consider a term t_i as a "message" and $p_i = \frac{n_i}{n}$ as the probability of receiving message t_i, then, in Shannon's information theory [9], the information that the message carries is quantified by

$$I_i = -\log p_i,$$

which coincides with the IDF measure. The connection is clear; terms occurring in too many documents do not carry too much information for "discriminating" documents ([1]). On the other hand, terms that occur in few documents carry more information and hence have more discriminative power.

In XML Information Retrieval, considering each XML element that contains text as a mini-document, we can compute multiple IDF scores for a given term. Note that here, we restrict ourselves to *textual* elements only, i.e. those elements that contain terms. For instance, in the above example, *introduction* is a textual element, while *body* is not.

Depending on the importance weight of each textual element, the IDF scores should be appropriately weighted. Intuitively, in the above example, the IDF score of a term with respect to the *related-work* elements is infinitely less important than the IDF score of the term with respect to say *introduction* elements.

Formally, let E be the set of textual element-weight pairs (e_h, w_h) extracted from XML document collection C. This set is finite because C is finite, and for each element in an XML document, there is a unique weight assigned to it (see Corollary 1).

For a textual element-weight pair (e_h, w_h), let n_h be the total number of such elements in the XML documents in collection C. Suppose that a term t_i occurs in n_{hi} of these e_h elements (of weight w_h). Then, we define the IDF of t_i with respect to these elements as

$$idf_{hi} = \log \frac{n_h}{n_{hi}}.$$

Next, we define the IDF score of a term t_i with respect to the whole document collection as

$$idf_i = \frac{\sum_h w_h \cdot idf_{hi}}{\sum_h w_h}.$$

This is the weighted average of IDF scores computed for each textual element-weight pair (e_h, w_h).

Finally, the TF*IDF weighting scheme combines the term frequency and inverse document frequency, producing a composite weight for each term in each document. Namely, the TF*IDF weighting scheme assigns to term t_i a weight in document d_j given by

$$tfidf_{ij} = tf_{ij} \times idf_i.$$

In the vector space model, every document is represented by a vector of weights which are the TF*IDF scores of the terms in the document. For the other terms in vocabulary V that do not occur in a document, we have a weight of zero.

Similarly, a query q can be represented as a vector of weights with non-zero weights for the terms appearing in the query. The weights are exactly those hyperreal numbers specified by the user multiplied by the IDF scores of the terms.

Now, we want to rank the documents by computing their similarly score with respect to a query q. The most popular similarity measure is the *cosine similarity*, which for a document d_j with weight vector $\mathbf{w}_j$ and a query q with weight vector $\mathbf{w}_q$ is

$$cosine(\mathbf{w}_j, \mathbf{w}_q) = \frac{\langle \mathbf{w}_j, \mathbf{w}_q \rangle}{||\mathbf{w}_j|| \times ||\mathbf{w}_q||} = \frac{\sum_{i=1}^{m} w_{ij} \times w_{iq}}{\sqrt{\sum_{i=1}^{m} w_{ij}^2} \times \sqrt{\sum_{i=1}^{m} w_{iq}^2}},$$

where m is the cardinality of vocabulary V.

The above formula naturally combines the query preference weights, XML element weights, and Information Retrieval measures. Note that, we can in fact rank documents using instead the square of the cosine similarity. Thus, we only need to compare fractions of polynomial expressions based on the (fixed) infinitesimal ε. As such, these expressions allow for an algorithmic (symbolic) comparison procedure for ranking XML documents.

Finally, the query can be a complete document in its own. Such queries are of the type: Find all the documents which are similar to a given document. We derive weights for the elements of the query document in exactly the same manner as described in Section 4. The vector of weights for the query document is computed as for any other document in the collection. Then, this vector is compared against the vectors of the documents in the collection by computing the cosine similarity as described above.

6 Experiments

We have implemented a system incorporating our proposed framework and compared its ranking effectiveness with that of a system that ranks using the classical TF-IDF measure. The main research question we address is:

Does our preferential IR improve users' search experience compared to a traditional IR?

Through experiments we provide practical evidence that our preferential IR does indeed perform better than a traditional IR.

As described in the previous sections, we annotated XML schema elements and search keywords in order to mark their importance in ranking the documents. We designed our experiments for both document retrieval and element retrieval. We used the following corpora as test-beds.

Corpus I On-line Internet Shakespeare Edition of the English Department ([12]), University of Victoria for element retrieval. This corpus consists of all the Shakespeare plays in XML format. The elements of interest are the speeches which total more than 33,000. For this corpus we consider all the speeches to be of the same importance, and thus, only search keyword preferences are in fact relevant for this corpus in influencing the ranking process.

Corpus II An INEX (INitiative for the Evaluation of XML retrieval) (cf. [13]) corpus. INEX is a collaborative initiative that provides reference collections (corpora). For evaluating our method, we have chosen a collection named "*topic-collection*" with numerous XML documents of moderate size. The topics of documents vary from *climate change* to *space exploration*. We preferentially annotated the DTD of this collection and gave many preferentially annotated search queries.

Due to space constraints, we do not show our results here, but we point the reader to the full version of this paper [4].

References

1. Aizawa N. A. An Information-Theoretic Perspective of TF-IDF measures. *Inf. Process. Manage.* 39(1): 45–65, 2003.
2. Bex J. G., F. Neven, T. Schwentick and K. Tuyls. Inference of Concise DTDs from XML Data. *Proc. VLDB '06*, pp. 115–126.
3. Bruggemann-Klein A. and D. Wood. One-Unambiguous Regular Languages. *Inf. Comput.* 140(2): 229–253, 1998.
4. Chowdhury M., A. Thomo, and W. Wadge. Preferential Infinitesimals for Information Retrieval. *Full version:* http://webhome.cs.uvic.ca/∼thomo/papers/aiai09.pdf
5. Liu B. *Web Data Mining: Exploring Hyperlinks, Contents and Usage Data.* Springer, Berlin Heidelberg, 2007.
6. Keisler H. J. *Elementary Calculus: An Approach Using Infinitesimals.* On-line Edition: http://www.math.wisc.edu/∼keisler/keislercalc1.pdf 2002.
7. Keisler H. J. *Foundations of Infinitesimal Calculus.* On-line Edition: http://www.math.wisc.edu/∼keisler/foundations.pdf 2007.
8. Manning D. C, P. Raghavan and H. Schutze *Introduction to Information Retrieval.* Cambridge University Press. 2008.
9. Shannon C. E. A Mathematical Theory of Communication. *The Bell System Technical Journal* 27: 379-423, 1948.
10. Robertson S. Understanding Inverse Document Frequency: On theoretical arguments for IDF. *J. of Documentation* 60: 503–520, 2004.
11. Rondogiannis P., and W. W. Wadge. Minimum Model Semantics for Logic Programs with Negation-as-Failure. *ACM Trans. Comput. Log.* 6 (2): 441–467, 2005.
12. On-line Internet Shakespeare Edition. English Department, University of Victoria. http://internetshakespeare.uvic.ca/index.html
13. Malik S., A. Trotman, M. Lalmas, N. Fuhr. Overview of INEX 2006. *Proc. 5th Workshop of the INitiative for the Evaluation of XML Retrieval*, pp 1-11, 2007.