Discovering the Context of WWW Pages to Improve the Effectiveness of Local Search Engines

Fernando Aguiar and Michel Beigbeder

Ecole Nationale Supérieure des Mines de Saint-Etienne, Departement RIM,
158 cours Fauriel, 42023 Saint-Etienne France
e-mail: {aguiar, mbeig}@emse.fr

Abstract This work proposes a method of searching for information in hypertext systems representing WWW sites. The method is based on the creation of a 2-level index. The first level of the index is related to information located only inside the nodes. The second level of the index relates to information which is not restricted to one node but encompasses a set of related nodes. The second level is based on the context hierarchy which is a hierarchical organization of the main themes dealt with by the information contained in the site and gives a notion of context to the pages. This notion permits a new operator named *context:* to be added to the query language allowing the user to better express his information need.

1 Introduction

One of the main reasons for the low effectiveness of actual search engines is probably related to the absence of the notion of context for the information expressed in HTML pages. The polysemy of terms leads to inaccurate indexing. The automatic detection of the context of a text is a problem that has been studied for a long-time by co-operations between linguistics and computer science researchers [Sci99]. In hypertext systems the problem is aggravated since the context of a page is not necessarily included in the content of the page, instead it is often situated in the content of *neighboring* pages. In fact, when the author of a site defines a graph representing HTML pages and the links between them, he supposes that when the reader accesses one page he has already browsed through other pages which gave him the knowledge necessary to understand the content of the current page.

Indexing methods applied by traditional search engines generate flat indexes for HTML pages. The index of a page is basically composed of a bag of terms without relationships between them, i.e., all the terms belong to the same conceptual level[1]. Moreover, the index of a page is solely based on the content of the page itself. In consequence of these two facts, the concept of context of the information contained in a HTML page does not exist in flat indexes. In this work we try to discover the context of the pages local to WWW sites and, by taking it into account in the page index, we hope to increase the effectiveness of search engines specialized to WWW sites. A 2-level index (context and subject levels) is proposed to index the

[1] Thus, the term *flat index*.

pages, and to make use of this additional functionality, the user is provided with a new query operator named the *context operator*. In the remainder of this paper we explain our approach to index and search a WWW site. Section 2 presents related works. In section 3, we describe the page indexing method which is based on the 2-level index. Section 4 presents the matching function adapted to the 2-level index. Finally we conclude.

2 Related Work

In this section we review related works from three different points of view: the type of functionality put forward by making use of clustering techniques; the assumption that an HTML page should not be considered as a fully-fledged document with regard to an IR system; and the use of hypertext structure evidence by search engines to better index/retrieve relevant information.

2.1 Taking Advantage of Clustering Techniques in IRSs

One of the first applications of clustering techniques in IRs consists in the organization of the file in clusters of documents. Such an organization is aimed at improving the efficiency of the system. Instead of calculating the similarity between the query and all documents in the file, incoming queries are compared first with cluster centroids and after to a reduced number of documents. By considering the cluster hypothesis, *"closely associated documents tend to be relevant to the same queries"*[vR79], it was suggested that file clustering could also improve the effectiveness of the system. Our proposition also aims at improving the effectiveness of the system by the use of clustering techniques, however, instead of clustering documents, pieces of documents are clustered in order to compose an amount of information that is self contained and thus can be considered as a document.

Clustering techniques have been used in the design of extra functionalities in IRSs as a means of helping the user to better visualize, browse and filter the results of a query. For instance, the system proposed in [ZE99] helps the user to browse the results by clustering them. Clustering techniques have also been applied to group terms instead of documents. Until a few months ago, the Altavista search engine (http://www.altavista.com) proposed the *refine* functionality which consisted in proposing clusters of terms to the user to refine his query. Our proposition differs from the aforementioned works in the sense that clustering techniques are not used in the design of extra-facilities for a basic information retrieval system. In our work, clustering techniques are involved in the indexing phase which is part and parcel of whatever information retrieval system.

2.2 HTML Pages not Considered as Fully-Fledged Documents

Hypertext systems representing WWW sites are particular in the sense that the information contained in the pages is very often not self-contained and thus insufficient for indexing them. [DB99] defines Virtual Web Documents, VWDs, as a set of HTML pages. It differs from our work in two aspects. First, the grouping of semantically related pages is manually made and relies on the author of the pages. In our work, the pages are automatically grouped. Second, the user is aware of the existence of a graph of VWDs superimposed on the graph of pages. In contrast, the notion of "document" does not exist in our system. Grouping is solely used as a means to discover lacking/complementary information from pages in order to better index and retrieve them.

[Dyr98] and [MT99] present systems developed with the same motivation, i.e., concepts can span several HTML pages. Such concepts are called multi-page concepts and are represented by concept paths. Concept paths are merged to compose a concept graph which is built on the top of the original graph of HTML pages. In the same way of the paths existing in the context hierarchy (see Sec. 4), concept paths are supposed to refine or narrow the concept space. Concept paths are discovered by analyzing both the directory structure embedded in the URLs of the HTML pages and the links between the pages. Our system does not take into account the directory structure to infer specialization/generalization relationships between pages. In addition to the links between the pages, it considers the content of the pages to analyze possible relations between the pages. Second, in our system, a single link between two pages does not establish a relation between them. It is rather link structure patterns that are analyzed to discover implicit relations between pages.

2.3 Taking Hypertext Structure into Account in WWW Search Engines

The exploitation of links structure has already been widely studied in the context of well-built hypertext systems where information contained in the nodes is self-explained and the links are typed and express well defined relationships[Fri88]. The particularity of the WWW relies on the fact that concepts often span pages and the links are not typed. The Google search engine [BP98] is the first commercial engine to intensively make use of links structure. The ranking of the pages is largely influenced by the popularity of the pages, i.e., how frequently pages are pointed by other pages. The Clever project [CDG$^+$99] also takes into account incoming and outgoing links to determinate the final rank of the pages. Furthermore, Google uses link anchor text to index the pages pointed to. It is assumed that the anchor text often describes more accurately the content of the page pointed to than the content of the page itself. In the same way, the work of Marchiori [Mar98] makes use of links to propagate metadata[2] information. All these works differ from ours in the sense that they consider that information contained in the pages is self contained. The structure is used as a means to find a better description of the information contained in one page somewhere else than in the content of the page itself. Indeed, the structure is not used to complete the information contained in the page.

[2] Metadata, literally "information on information", is available in HTML <meta> tags in the heading of some HTML pages. The content of <meta> tags is supposed to briefly describe the semantic content of the page.

3 Indexing WWW Sites

As commented above, information contained in HTML pages is very poor in terms of completeness. Linguistic regularities and behavior patterns within the WWW seem to be the fruit of a simple language evolution rule : *"maximum communication with minimal effort"* [Ami97]. In hypertext systems, the neighborhood of a node is often considered as being its context [Man97]. On the WWW, there is a high diversity of link types. One can distinguish not only structural links which establish a contextual relationship between pages (e.g., *People*, *Research*, *Academic*, *International Relations*, outward links from an university homepage), but also navigational links (e.g., *Previous*, *Next*, *Homepage*,...), reference links (e.g., outward links from *My Hotlinks* pages), etc. Therefore, the context of the content of a given page shall not be considered as the merger of the content of all pages pointing to it.

Two approaches are possible to identify the structural organization of a site. On the one hand one can assume the hypothesis that structural links exist but are mixed up with links of other types. In this situation, a method to sort out the links is necessary. On the other hand one can hypothesize that structural links do not necessarily exist and methods based (i) on statistical analysis of term distribution and (ii) on linkage analysis must be developed to reveal the structural organization of a site. In this work the latter hypothesis is assumed. The method of discovering the context of pages we put forward here is based on clustering techniques that explore the graph underlying a WWW site in order to build what we call *the context hierarchy*. The context hierarchy reveals the structure of the site by making the context(s) of its pages explicit. The context of a page may be interpreted as an additional information that complements the page's content. Both the page's context(s) and the page's content are used to index one page.

3.1 Building the Context Hierarchy

The context hierarchy construction process consists merely of alternating the following procedures: (i) the clustering of one graph and (ii) the application of an abstraction function to the clusters which produces a new graph with the same format as that of the graph clustered in the (i). These procedures are alternated until either the clustering generates only one cluster or a predefined maximum number of iterations is reached. Each iteration adds a level to the context hierarchy. The first level of the hierarchy is generated from the clustering of the graph underlying the pages of the site. The clustering procedure aims at grouping together nodes that complement each other. In the following we define the complementarity function used in our model.

The Complementarity Function The nodes of the graph have two components: a content component and a structure component. The complementarity between two nodes of the graph is a function of the simlarity between the content components and the similarity between the structure components of the nodes. At the moment, the two similarity factors are summed to obtain the final complementarity between two nodes.

$$Compl(D_i, D_j) = Sim_{cont}(D_i, D_j) + Sim_{struc}(D_i, D_j)) \quad \text{(I)}$$

The content component is represented with the help of the vector model [SM83]. According to this model, each document is represented by a vector of term weights in a space of ρ dimensions where ρ corresponds to the number of single terms. A term weight reflects the significance of the term to the document. $Sim_{cont}(D_i, D_j)$ is calculated with the help of the cosine measure[vR79]. This measure calculates the cosine of the angle between two vectors in the ρ dimension space:

$$Sim_{cont}(D_i, D_j) = \frac{\sum_{k=1}^{\rho} w_{ik} \cdot w_{jk}}{\sqrt{\sum_{k=1}^{\rho} w_{ik}^2 \cdot \sum_{k=1}^{\rho} w_{jk}^2}} \quad \text{(II)}$$

where w_{ik} is the weight of the term T_k in the document D_i.

Regarding the term weight calculation a $tf^3 \times idf^4$ like formula is used [Sav97].

$$w_{ik} = \alpha \cdot ntf_{ik} \cdot nidf_k \text{ where } ntf_{ik} = \frac{tf_{ik}}{max_z\ tf_{iz}},\ nidf_k = \frac{idf_k}{\log(n)},\ idf_k = \log\left(\frac{n}{df_k}\right) \quad \text{(III)}$$

where n is the number of documents; tf_{ik} denotes the number of occurrences of the term T_k in the document D_i; df_k is the number of documents in which the term T_k occurs and α corresponds to the importance of the logical elements in which the terms occurs. Higher weights are associated to terms occurring in the specific HTML elements: <title>, <meta> and <h*>[5]. If term T_k occurs at least once inside one of those elements α is set to $1,5$. Otherwise, α defaults to 1.

The structural similarity between two nodes D_i and D_j in the graph is directly proportional (i) to the number of common ancestors (S_{ij}^{anc}) (Eq. IV), (ii) to the number of common descendants (S_{ij}^{des}) (Eq. V), (iii) to the number of independent paths linking them (Con_{ij}) (Eq. VI) and (iv) to the inverse of the lengths of the shortest paths between them (S_{ij}^{spl}) (Eq. VII).

Regarding the ancestors and descendants of a node, we hypothesize that the more two nodes have ancestors and descendants in common, the more they are likely to be semantically related. Node D_a is an ancestor (*descendant*) of node D_b if there is in the graph a path leading from D_a (D_b) to D_b (D_a). The equations are the same as those used in [Wei96].

$$S_{ij}^{anc} = \sum_{x \,\in\, Anc} \frac{1}{2^{(spl_{ix}^{\bar{j}} + spl_{jx}^{\bar{i}})}} \quad \text{(IV)} \qquad S_{ij}^{des} = \sum_{x \,\in\, Desc} \frac{1}{2^{(spl_{ix}^{\bar{j}} + spl_{jx}^{\bar{i}})}} \quad \text{(V)}$$

where Anc and $Desc$ are respectively the sets of common ancestors and descendants of D_i and D_j; spl_{ij} is the length of a shortest path between D_i and D_j and

[3] Term document frequency.
[4] Inverse document frequency.
[5] It stands for the elements <h1>, <h2>, <h3>, <h4>, <h5>, <h6>.

$spl_{ij}^{\overline{k}}$ is the length of a shortest path between D_i and D_j not traversing D_k.

In Eq. VI the number of independent paths between two nodes D_i and $D_j$is considered as an indication of semantic relationship. We hypothesize that when paths exist in both directions, i.e., from D_i to D_j and from D_j to D_i, the relationship is stronger than if paths exist only in one direction. Con_{ij} (Eq. VI) indicates how strong two nodes are connected and is a function of the number of paths connecting D_i and D_j and their direction. Con_{ij} is maximum when the number of independent paths in one direction is equal to the number of paths in the opposite direction. The vector $\vec{V_i}$ represents this situation. Con_{ij} is then calculated by the cosine of the angle between the vector $\vec{V_{sp}} = (S_{ij}^{Indep}, S_{ji}^{Indep})$ and the vector $\vec{V_i} = (1,1)$. S_{ij}^{Indep} denotes the number of independent paths leading from D_i to D_j.

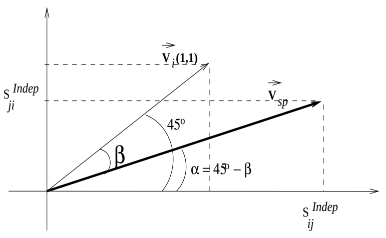


Figure1. $\boldsymbol{V_i}$ and $\boldsymbol{V_{sp}}$ vectors.

$$Con_{ij} = \cos\beta = \cos(45-\alpha) = \frac{\sqrt{2}}{2} \cdot \left(\frac{S_{ij}^{Indep} + S_{ji}^{Indep}}{\sqrt{(S_{ij}^{Indep})^2 + (S_{ji}^{Indep})^2}} \right) \quad \text{(VI)}$$

Finally, we hypothesize that if there is at least one path connecting two nodes, the shorter the path, the stronger the semantic relationship between the nodes. Often, the length of the shortest path leading from D_i to D_j is very different from the length of the shortest path leading from the D_j to D_i. We have observed that this situation is very common in hypertexts representing WWW sites and is often provoked by links to home pages or the like, i.e., links that exist only to "facilitate" browsing and do not necessarily express semantic relationships between nodes. That is the reason why the length of a possible path connecting the nodes in the opposite direction is also taken into consideration in the formula of S_{ij}^{spl} (Eq. VII).

$$S_{ij}^{spl} = \frac{1}{2^{\left(\frac{spl_{ij}+spl_{ji}}{2}\right)}} = S_{ji}^{spl} \quad \text{(VII)}$$

The four components above are normalized to lie between $[0,1]$ and $Sim_{struc}(D_i, D_j)$ is calculated by Eq. VIII with β, γ, δ and λ set to 1. In coming experiments, we expect to vary the weights of the different indicators in order to study the different degrees of influence they may have in $Sim_{struc}(D_i, D_j)$ and, consequently, in the effectiveness of the system.

$$Sim_{struc}(D_i, D_j) = \beta \cdot S_{ij}^{anc} + \gamma \cdot S_{ij}^{des} + \delta \cdot Con_{ij} + \lambda \cdot S_{ij}^{spl} \quad \text{(VIII)}$$

The Clustering Step The calculation of the pairwise complementarities between the documents permits the $n \times n$ *document-document* matrix *M* to be generated. From (i) the matrix *M* and (ii) a complementarity threshold, $thres_{MM'}$ (calculated as the average complementarity between all pairs of documents), a second $n \times n$ matrix *M'*, the *document relationship matrix,* is generated. M'_{ij} is set to 1 if $M_{ij} \geq thres_{MM'}$, otherwise it is set to 0.

We have implemented two O(n^2) clustering methods based on our complementarity measure. They are graph-theoretic methods and none of them are hierarchic, i.e., they do not generate a hierarchy of clusters. The first one is the connected component method. This method partitions the nodes of the graph generating non-overlapping clusters. A connected component of one graph is a set of nodes such that each node is connected to at least one other node of the set and the set is maximal with respect to this property. The second method used is the star method. It generates overlapping clusters. The star method firstly generates for each node a cluster containing all the other nodes linked to it. Finally, clusters that are contained in larger clusters are eliminated. The idea underlying the generation of overlapping clusters is that one node in a hypertext may be interpreted in different contexts according to the path followed by the reader before reaching the node. With the help of experimental results, we intend to verify if the use of overlapping clusters tends to improve the effectiveness of the system. From now on, consider that the clustering method used generates overlapping clusters.

Tracking Between Hierarchy Levels Before applying the abstraction function to the clusters, a relationship is established between the nodes composing one cluster and the cluster itself. Each node in level $j + 1$ keeps a track of the importance that nodes in level j had in the composition of the cluster that underlies it. A node situated on the level $j + 1$ of the context hierarchy is generated by the application of the abstraction function to a cluster composed of nodes situated in the level j (Fig. 2).

The importance a node D_x belonging to a level j has in the composition of a cluster C_y of level $j + 1$ is calculated by the similarity between the content components of the two entities, i.e., $Sim_{cont}(D_x, C_y)$ (Eq. II). We call this value the tightness of the link between the node D_x and the node D_y which is generated by the application of the abstraction function to C_y (Fig. 3). The vector which represents the cluster C_y in the similarity calculation is the centroid of the cluster.

The Abstraction Function The abstraction function derives a new node from each cluster generated by the clustering iteration. With the help of information concerning the graph that has been clustered, the abstraction function links the new nodes with directed edges generating the graph to be used in the next clustering iteration. The new nodes and their links to the nodes of the lower level are inserted in the hierarchy adding a new level to it. In the following, we describe how the content and structure components of a new node of level $j+1$ are generated from a cluster of nodes of level j.

The generation of the content component of the node proceeds as follows. First, the cluster centroid is calculated as the mean vector for all the vectors composing the cluster. Then the centroid vector is thresholded. A null weight is assigned to the vector components representing terms that do not have "a good distribution" in the vectors used to calculate the centroid. In our model, a good distribution of a term T_i implies a large mean and a small variance of the value of the component related to T_i, i.e., w_i, in the vectors belonging to the set. The thresholding algorithm starts by associating the null weight to the components of the centroid vector whose values do not exceed the *vector-median threshold*. An intermediate vector is generated. The *vector-median threshold* is defined as the median value of the components of the vector. Then, the variance of the remaining terms is calculated. The terms whose variance does not exceed a previously defined maximum variance, σ_{max} are maintained, the others are eliminated, i.e., they have their values set to the null weight. The resulting vector is associated to the content component of the new node.

The structure component of a node is composed of directed edges to other nodes of the same level. Suppose there are two nodes D_1 and D_2, both located in the level $j+1$ and derived from clusters C_1 and C_2 composed of nodes located in the level j. There is one edge between D_1 and D_2 if there is one edge leading from any node of C_1 to any node of C_2. Several edges leading from an arbitrary node of C_1 to a same node of C_2 generate a single edge between D_1 and D_2. Finally, new edges maintain the orientation of the edges that originated them. It is important to highlight the fact that edges within levels of the hierarchy are only used during the construction of the hierarchy. In effect, once the level $j+1$ is built, the edges between the nodes of the level j can be discarded.

3.2 Indexing the Pages

As we mentioned before, the hierarchy of contexts is used to supplement the information contained in the pages so that during indexing, any information supposed to be related to the page (interior or exterior to the content of the page) is taken into account. The index of the pages is thus composed of two levels. In the first level, the pages are analyzed individually and their textual content is indexed. First, HTML comments and tags are eliminated from the content of the pages. Then, a stemming algorithm is applied. Finally, weights are associated to the terms (Eq. III) and the inverted file is built.

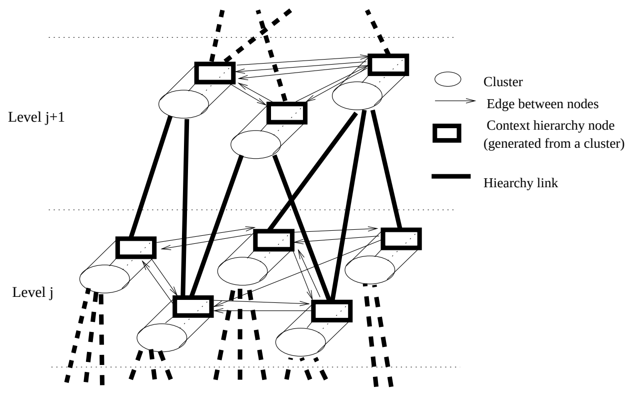


Figure2. Construction of the context hierarchy.

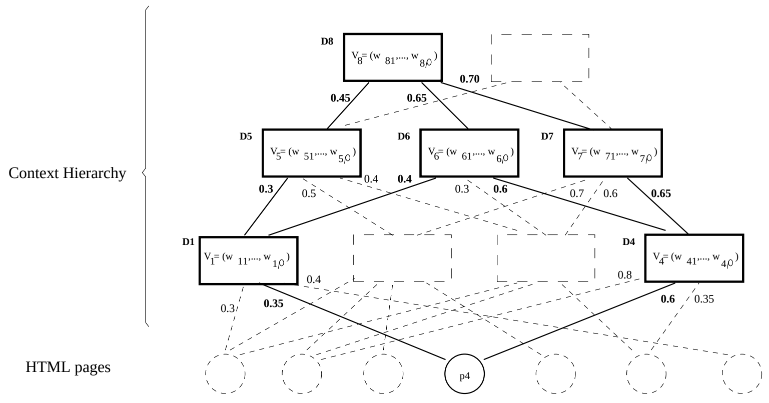
The second level of the index relates to the contexts of the pages, i.e., the nodes of the context hierarchy. In this index level, the content component of the nodes is indexed. A second inverted file is built. Moreover, the inverted file is associated to a table in which the structure of the context hierarchy, i.e., the parent/child relationships as well as their tightness are coded.

4 The Matching Function

The functionality put forward in our system is the operator *context:* which permits the user to differentiate the context of his query, $q_{context}$ from the subject of it, $q_{subject}$. The matching function is made up of three steps. First, the entries corresponding to the terms defined in the $q_{subject}$ are retrieved from the inverted index built from the content of the pages. The matching function used in both the first and second steps is that described in Sec. 3.1.1 (Eq. II). A ranked list of pages is then obtained. The number of retrieved pages is limited by a previously defined value. In the second step, the system retrieves from the context hierarchy index the entries corresponding to the query context. The nodes contained in these entries are thus identified as those bound to the contexts asked for by the user. A ranked list of contexts is then obtained. As in the first step, the number of retrieved contexts is also limited by a previously defined value.

At this moment the search engine has at its disposal two responses: a ranked list of pages and a ranked list of contexts. The third and final step will combine these two responses. It remains for the search engine to verify if the pages retrieved in the first response are bound to the contexts retrieved in the second response. The strength of this binding will be measured by the importance of each page in the construction of the nodes representing the contexts.

In the context hierarchy, nodes of level $j+1$ only know the importance of nodes of level j in their composition. The importance of nodes located in deeper levels is not known. Since the pages are located on the bottom level, a strategy for the propagation of the pages' importance must be set up. Consider that page p_4 and node D_8 have been retrieved in the first and second steps respectively and the importance of p_4 in the composition of D_8 must be calculated (Fig. 3). First, the paths between these two nodes in the context hierarchy are found. For example, the path $ch_1 = p_4 \rightarrow D_1 \rightarrow D_5 \rightarrow D_8$ leads from p_4 to D_8. Given a path $ch = n_1 \rightarrow \cdots \rightarrow$



Context Hierarchy
HTML pages
D8
$V_8 = (w_{81}, \ldots, w_{8\rho})$
D5
$V_5 = (w_{51}, \ldots, w_{5\rho})$
D6
$V_6 = (w_{61}, \ldots, w_{6\rho})$
D7
$V_7 = (w_{71}, \ldots, w_{7\rho})$
D1
$V_1 = (w_{11}, \ldots, w_{1\rho})$
D4
$V_4 = (w_{41}, \ldots, w_{4\rho})$
0.45
0.65
0.70
0.3
0.5
0.4
0.4
0.3
0.6
0.7
0.6
0.65
0.4
0.8
0.3
0.35
0.6
0.35
p4

Figure3. Paths between a page and a context.

$n_j \rightarrow \cdots \rightarrow n_k$, the importance of n_1 in the composition of n_k following a path ch, $Imp(n_1, n_k, ch)$, is calculated by the product of the weights of the links constituting the path. This calculation is done for all paths leading from node p_4 to node D_8. $Imp(p_4, D_8)$ is finally calculated as:

$$Imp(p_4, D_8) = \max(Imp(p_4, D_8, ch_i)) \quad \text{(IX)} \quad \text{for} \quad i = 1, 2, \cdots, m$$

where m is the number of paths between the two nodes. $Imp(p, D)$ is calculated for all pairs (p_j, D_i) where p_j is a page retrieved in the first response and D_i is a node retrieved in the second response. To calculate the final relevance of a page p_x to a query q, the system has at its disposal three values: the relevance of p_x to the query subject, the relevance of the nodes in the context hierarchy to the query context and the importance of p_x in the composition of the contexts retrieved, i.e., $Imp(p_x, D)$. The final relevance of p_x is directly proportional to all the three values. At the moment, (Eq. X) is the formula we have implemented.

$$R(p_x, Q) = R(p_x, q_{subject}) \cdot \max_{k=1,2,\cdots,n} (R(D_k, q_{context}) \cdot Imp(p_x, D_k)) \quad \text{(X)}$$

where n is the number of contexts retrieved in the second step of the matching function. Regarding the results output format, the current prototype furnishes a ranked list of pages with their relevance scores. It could be interesting if the system responds not only pages but also the contexts bound to the pages with the help of the context hierarchy. Indeed, this could be very helpful to the user as he would be able to disregard pages straightaway depending on their contexts. We intend to add this feature to the system in the next prototype.

5 Conclusion and Future Works

What we propose here is a simple text search engine specialized to WWW sites. It is based on the construction of a context hierarchy that permits us to build a 2-level structured index: the index of the pages; and the index of the contexts of the pages. The notion of context between pages permits also the system to introduce a 2-level structured query based on the context operator. This type of query permits the user to better describe his information need. The work reported in this paper is a work in progress. All modules have been implemented in C++. To evaluate our system we intend to compare it with other search engines specialized to WWW sites such as FreeFind (http://www.freefind.com), WWW Glimpse (http://glimpse.cs.arizona.edu/). With relation to the test collection to be used in the evaluation, a set of 30 topics — in the same format of those used in TREC[6] experiences — related to the information — approximately four thousand pages — contained in the agora21 site (http://www.agora21.org) is being defined. The different search engines will be installed in this site and their effectivenness will be compared with the help of traditional information retrieval evaluation measures, i.e., precision $\times$ recall figures.

References

[Ami97] E. Amitay. *Hypertext: the importance of being different*. MSc Dissertation. Centre for Cognitive Science. The University of Edinburgh, 1997.

[BP98] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. In *Proceedings of the Seventh International WWW Conference*. IW3C2, 1998.

[CDG$^+$99] S. Chakrabarti, B. Dom, D. Gibson, S. R. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins. Hypersearching the web. *Scientific American*, June 1999.

[DB99] B. L. Doan and M. Beigbeder. Virtual www documents : A concept to explicit the structure of www sites. In *Proceedings of the 21st Colloquim on Information Retrieval*. BCS-IRSG, 1999.

[Dyr98] C. E. Dyreson. A jumping spider: Restructuring the www graph to index concepts that span pages. In *Proceedings of the Seventh International WWW Conference (Workshop on Information Reuse)*. IW3C2, 1998.

[Fri88] M. E. Frisse. Searching for information in a hypertext medical handbook. *Communications of the ACM*, 31(7):880–886, 1988.

[Man97] U. Manber. Webglimpse - combining browsing and searching. In *In Usenix Technical Conference*, 1997.

[Mar98] M. Marchiori. The limits of web metadata and beyond. In *Proceedings of the Seventh International WWW Conference*. IW3C2, 1998.

[MT99] Y. Mizuuchi and K. Tajima. Finding context paths for web pages. In *Proceedings of the Ninth ACM Conference on Hypertext and Hypermedia*. ACM, 1999.

[6] *Text REtrieval Conference*. URL: *http://trec.nist.gov/*

[Sav97] J. Savoy. *Information Retrieval and Hypertext (M. Agosti and A. Smeaton)*, chapter 5 (Citation Schemes in Hypertext Information Retrieval. Kluwer Academic Publishers, 1997.

[Sci99] A. M. Sciullo. Formal context and morphological analysis. In *Proceedings of the CONTEXT'99 Conference. 2nd International and Interdisciplinary Conference on Modelling and Using Context Colloquium*, 1999.

[SM83] G. Salton and M.J. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill, New York, NY, 1983.

[vR79] C.J. van Rijsbergen. *Information Retrieval*. Butterworths, 2nd edition, 1979.

[Wei96] R. Weiss. Hypursuit: A hierarchical network search engine that exploits content-link hypertext clustering. In *Proceedings of Hypertext '96*. ACM, 1996.

[ZE99] O. Zamir and O. Etzioni. Grouper: A dynamic clustering interface to web search results. In *Proceedings of the Eighth International WWW Conference*. IW3C2, 1999.