Indexing of Reading Paths for a Structured Information Retrieval on the Web

Mathias Géry

Indexing of Reading Paths for a Structured Information Retrieval on the Web

Mathias Géry
Université de Lyon, UMR-5516, Saint-Étienne, France
Mathias.Gery@univ-st-etienne.fr

Abstract

In this paper, we present a hyperdocument model taking into account the essential aspects of information on the Web: content, composition (logical structure) and non-linear reading (hypertext structure). We have developed a Structured Information Retrieval System (SIRS) based on this model. Its phases of indexing and querying are based on a "reading paths" point of view of the Web: a Web site is considered as a set of potential reading paths, instead of a set of atomic and flat pages. We have developed an specific algorithm to index the reading paths. We present some experiments aiming at evaluating the interest of our indexing process of reading paths.

1 Introduction

Retrieving relevant information on the Web looks like *Finding the Needle in the Haystack*. The well-known search engines use criteria mainly related to the textual content. These systems are based on classical IR models[19]: the documents are considered as atomic and independent, as their physical HTML page aspect, without considering the relations linking them. In particular, the reading of a document is linear, whereas the main characteristic of a hypertext (like the Web) is to allow non-linear reading.

One of the most important source of information on the Web, except the textual content, is its structure. The Web is composed of structured documents (the HTML pages can be structured), and it has also hypertext characteristics (the HTML pages can be linked together). Several works have shown that it is possible to extract a hierarchical structure describing a Web site [9] [17], while others deal with macroscopic structure [3] [1]. The structure of the Web has to be considered during an IR process: the index should represent the semantic content of documents, including the structure. Especially, an IR model has to integrate links and their impact directly into the document model, instead of applying a simple re-ranking above a classical system.

In this paper, we present a Structured Information Retrieval System (SIRS) and its underlying document model, based on an informational unit suitable to the Web: the **Reading Path**. The outline of the paper is as follows: firstly, some related works using Web structure for IR are described in the section 2. Then, we present the theoretical principles of Reading Paths in the section 3. Our SIRS is based on a hyperdocument model that considers the three facets of Web structure for indexing: composition, reading paths and context. This model is presented in the section 4, together with the corresponding indexing process. Finally, we present some implementation facts and experiments in the section 5.

2 Structured IR on the Web

Most of the Web Search Engines use some information from the hypertext links in their ranking process, but they do not really integrate the links in the document model, which is always based on documents seen has atomic, flat and independent Web pages. However, several research directions have been proposed to improve these techniques. We distinguish four main approaches.

Some specifics techniques (e.g. [15]) propose to query the structure of documents (structured queries), mainly based on relational Databases, that is not suitable for the heterogeneous Web. Two other approaches use the "global link information" (i.e. using structure independently from the query) for the indexing phase of an IRS: we call them "propagation of information" and "propagation of popularity". The fourth approach uses the "local link information" (i.e. using structure considering the query). We call it "propagation of relevance". In the next sections, we give a few examples of these 3 last techniques.

2.1 Indexing: propagation of popularity

The popularity propagation aims at exploiting the link structure, considering that "A good page is a page referenced by many other good pages", typically with the calculation of a "prestige score" for each page of the collection. The simplest example is the "links voting", that counts the

number of links pointing to a page. A popular implementation is the *PageRank*, a prestige score calculated at indexing time, thus independently to the query [2]. The PageRank score is initialized to a given value for each page, and then this value is recursively propagated along links until a stable state is reached.

2.2 Indexing: propagation of information

This approach aims at considering links by propagating information along them in order to retrieve better the structured documents considering their sub-parts, but also in order to better retrieve the sub-parts considering their ancestors. For example, IOTA system propagates terms from sub-parts of a document to the top considering composition relation [7]. There are also many approaches aiming at retrieving some passages of a document, instead of the document itself: that is the "passage retrieval" problematic [18] [5]. Wilkinson proposes to take the documents sections into account to retrieve the whole documents, as well to consider the whole documents to retrieve the sections [20].

On the Web, many search engine propagate terms from context (link anchors[1]) to a given page considering that "*anchors often provide more accurate descriptions of web pages than the pages themselves*" [2]. The anchors terms are added to the index of the referenced page.

2.3 Querying: propagation of relevance

Like the propagation of popularity, this approach aims at calculating a "prestige score", but only for a subset of pages that have been pre-selected considering the query. Frisse has proposed such a technique to calculate a prestige score similar to the PageRank, but initializing the prestige value with the relevance of each page, instead of the same value for each page [8]. A more popular example is the algorithm HITS, that calculates two "prestige scores" at query time: the *Hubs and Authorities*, assuming that "A good Hub points to many good Authorities, and a good Authority is pointed to by many good Hubs" [14].

2.4 Discussion

Popular search engines *seem* to give some good results on the Web. However, the scientific evaluation of these techniques are quite disappointing [12] [11]. These poor results are caused by the "triple-bag" problem: the Web is considered a bag-of-words, a bag-of-nodes and a bag-of-links. Most of these systems are based on a Web model simplified to a directed graph with HTML pages as nodes and hyperlinks as edges. Very few methods try to analyze what does mean a link regarding information, and how to consider it for IR. Gurrin proposes to distinguish functional and structural links, arguing that structural links are not useful for connectivity analysis [10]. Chakrabarti proposes a uniform fine-grained model representing pages as trees, and an associated fine-grained distillation algorithm giving better results in a fine-grained context than popularity propagation [6].

We conclude that we need to use a model of the Web that is more structured than the uniform "triple-bag", in order to really improve IR using links-based techniques. This model should describe the information as it is has been thought by the authors, and index it as it is understood by the readers.

[1] An anchor is a fragment of text, on which a user can click in order to activate a hypertext link.

3 Information reading and understanding

We consider three aspects of information reading on the Web: document structure, hypertext structure and context. Two of them concern the reading of a "document" (navigation inside a "document"), and are based on a tree structure (structured documents) and a reading path structure (hyperdocuments). The third one is not developed in this paper: it deals with the browsing outside documents, and with the concept of "context".

The Web is composed of structured documents, usually read in a linear way: introduction, then first section, etc., until conclusion. Web sites can be seen as describing a tree structure: so, the document model should be based on a hierarchical structure. For example, a chapter composed by 3 sections has to be represented by tree leaves and the semantic point of view of the composition has to be integrated.

On the other hand, the Web is also a hypertext: it is possible to link any part of a Web site to any other part of the Web. That allows the author to define reading paths: on each node, the reader chooses between one or more possibilities to navigate. For example, "*Afternoon, a story*" [13] is known as the first *HyperFiction*. This hypertext novel is built with the aim at forbidding linear reading. On each node, the reader chooses between one or more possibilities, which depend on his previous choices. Such reading paths were named "*trails*" by the inventor of hypertext, Vannevar Bush in "*As we may think*": "*It is exactly as though the physical items had been gathered together to form a new book. It is more than this, for any item can be joined into numerous trails*" [4]. Moreover, the same textual fragment may be understood differently, depending on the previous reading fragments. Thus, each reader and each reading build a new significance, a new comprehension of the story. The Web as hypertext has to be indexed considering several reading paths, in order to represent significance as close as possible to the semantic that the reader will extract himself while reading.

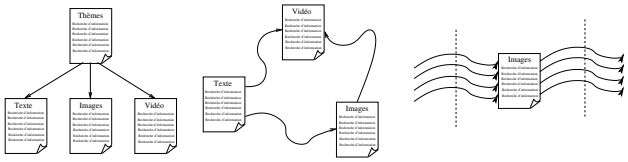
4 Hyperdocument modelling and indexing

Our hyperdocument model is based on the essential concepts: content, composition, linear or non-linear reading and context. We focus in this paper on the information description inside a (hyper-)document. The indexing process takes into account the hierarchical structure as well as the reading structure (i.e. the nodes ordering while reading).

Thus, our hyperdocument model considers the two points of view of a Web site: the structured documents SD_i (hierarchical structure), and the hyperdocuments HD_i related to the reading paths $Path_j$ (reading structure).

A Web site is based on a *hierarchical structure* (composition relation), but contains also a *reading structure* (reading relation), and is in a *context* (reference relation). We consider two of these three types of relations (composition, reading and reference, cf. figure 1), especially their impact on Web information building and comprehension while reading.

Figure 1. Links types.



4.1 Atomic documents

In our model, based on the Vector Space Model (VSM) [19], the atomic document unit is a Web page or a fragment of a Web page (e.g. a paragraph): a_i is represented by a vector of weighted terms: $\vec{a_i} = (w_{i1}, w_{i2} \ldots w_{ij} \ldots w_{in})$. Classical weighting functions have been used (*tf.idf* variants).

4.2 Structured documents

Our document model is based on a hierarchical structure with various granularity levels. A structured document SD_i is composed by n fragments $a_1, a_2, \ldots, a_n$, linked together by a composition relation. The indexing of a SD_i propagates information along the hierarchical structure (cf. section 2.2). It considers the content of each non-leaf node as an aggregation of its children's contents, and their indexes are made recursively using children's indexes:

$$w_{ij} = \frac{\sum_{a_k \in child.(SD_i)} w_{kj}}{\sum_{a_k \in child.(SD_i)} Size(a_k)} \quad (1)$$

The indexes are propagated from the leaves to the root, building a vector $\vec{sd_i}$. This indexing process corresponds to a *linear reading* of a *structured document* SD_i.

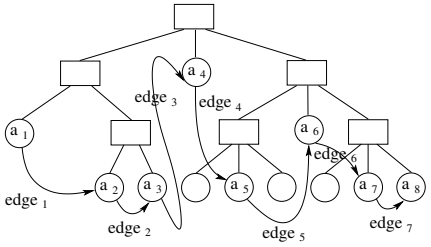
4.3 Reading paths for IR

A Web site is also represented by a hyperdocument HD_i, modeled as a directed graph with atomic documents as nodes and reading relations as edges. A set of reading paths (sequences of atomic documents) is defined on the atomic documents of HD_i. A unique reading path exists for each SD_i (linear reading), while several reading paths can exist for each HD_i, defining several readings potentialities. The indexing process for a hyperdocument corresponds to a *non-linear reading* (in fact, a set of potential linear reading paths) of a *hyperdocument*. We hypothesize that all the potential readings of a HD_i can be represented by a set of *reading paths* $path_j$.

- HD: hd = (docs = $\{a_i\}$, paths = $\{path_j\}$)
- Paths: $path_j$ = { $edges_k$ = (a_{src}, a_{dest}) }

The figure 2 shows an example of a hyperdocument with a reading path defined on the atomic documents.

Figure 2. Reading path example.



4.4 Reading Paths indexing

The semantic extraction of a HD_i aims at indexing each reading path by a vector $\vec{path_j}$, considering that the atomic documents are ordered in the reading path. Thus, a single Web site can be indexed by many different indexes, depending on the atomic documents appearing in the reading path, but also depending on the order in which they appear.

Our algorithm is based on several principles of thematic progress in a text, integrating the *reading memory* in order to simulate a human reading:

Hypothesis 1: reading memory: the reading of a a_i depends on the previous $a_1, a_2, \ldots a_{i-1}$ that were read.

Hypothesis 2: principle of accumulation: information that is read at the beginning of the reading path is more important, considering that it is reused afterward as reading memory.

Hypothesis 3: a semantic breakdown in a reading path shows a narrative discontinuity and implies a loss of reading memory.

We propose a *reading path extraction algorithm* (cf. figure 3), extracting an index $\vec{path}_p$ from a reading path $path_p$ = { $edges_k$ = (a_k, a_{k+1}, $\beta_{break-k}$), $k \in [1..n-1]$ }. The vector of reading memory $\vec{mem}$ represents the information gain from the first nodes, that the reader keeps in mind and uses to understand the following nodes. We use also a vector of reading accumulator $\vec{read}$, that represents the whole collected information, and a coefficient of semantic breakdown β_{break} between the source and the destination node, expressing the narrative discontinuity during the reading. Finally, the vector $\vec{local}$ is used to store the information collected successively on each node.

Figure 3. Reading path indexing algorithm.

(a) Initialisation: first node a_1 :
(a.1) Reset reading memory: $\vec{mem}_1 = \vec{0}$
(a.2) Read a_1: $\vec{read}_1 = \alpha.\vec{mem}_1 + (1-\alpha).\vec{a_1}$
(b) Reading: $\forall edges_j = (a_j, a_{j+1}, \beta_j) \in edges_k$:
(b.1) Update reading memory:
$$\vec{mem}_j = \beta_j.(\vec{mem}_j + \vec{a_j})$$
(b.2) Local information:
$$\vec{local}_{j+1} = \alpha.\vec{mem}_{j+1} + (1-\alpha).\vec{a_{j+1}}$$
(b.3) Accumulation:
$$\vec{read}_{j+1} = \gamma.\vec{read}_j + \alpha.\vec{local}_{j+1}$$
(c) Activating the last edge : $edge_n = (a_n, a_{n+1}, \beta_n)$
(c.1) Result: return $\vec{read}_{n+1}$

The first step (a) initializes the reading memory $\vec{mem}$ to null and the information $\vec{read}$ collected from the first node. The second step (b) updates the reading memory considering the previous node (b.1), combines this memory with the node vector in order to calculate the *local* information for this node (b.2), and finally adds this information to the accumulator (b.3). In case of a semantic break, the reading memory is set to null. Finally, the last step returns the index of the whole reading path.

The reading memory and the reading path indexing can be expressed as follows:

$$\begin{array}{l} \vec{mem}_1 = \vec{0} \\ \vec{mem}_{j+1} = \beta_j * \{\vec{mem}_j + \vec{a_j}\} \\ \vec{mem}_n = \sum_{i=1}^{n-1} \left\{ \vec{a_j} * \pi_{k=1}^{j-1} \beta_k \right\} \end{array} \quad (2)$$

$$\vec{read}_1 = (1-\alpha) * \vec{a_1}$$

$$\vec{read}_{j+1} = \left\{ \begin{array}{c} \alpha * \vec{mem}_{j+1} \\ +(1-\alpha) * \vec{a_{j+1}} \end{array} \right\} + \gamma * \vec{read}_j \quad (3)$$

$$\vec{read}_n = \left\{ \begin{array}{c} \alpha * \left\{ \sum_{i=1}^{n} (\gamma^{n-i} * \vec{mem}_i) \right\} \\ +(1-\alpha) * \left\{ \sum_{i=1}^{n} (\gamma^{n-i} * \vec{a_i}) \right\} \end{array} \right\}$$

The parameter α aims at giving less or more importance to the reading memory against the local content, while the parameter γ aims at giving more importance to the beginning of the reading path (if $\gamma > 1$) or to the end (if $\gamma < 1$). If $\gamma = 1$ and $\alpha = 0$, then each node is considered equally, without considering the ordering, and the final index $\vec{read}$ is the average of all the atomic vectors a_j. In case of frequent semantic breakdowns ($\forall k, \beta_k \approx 1$), or if $\alpha = 0$, then the reading memory is not used.

5 Experiments

It is not possible to build a SIRS above an existing Web search engine, because the document model is structured. We have developed a complete SIRS and a set of tools to analyze, index and query the collected corpora (spider, HTML analyzer, links typing module, querying module, end-user interface).

The most important difficulty encountered is the lack of explicit structure on the Web. Especially, the links are not typed, and the SIRS has to analyze the links in order to extract the structures (hierarchical, reading and reference). The idea is not to type the links very precisely, but rather to extract the link's function on the reader's point of view. Our typing module uses simple heuristics on the links syntax, to determine if a link is a hierarchical, a reading or a reference link. These heuristics are based on the hierarchical structure of the underlying Web server filesystem and consider some frequent structure patterns of Web sites. Also, links that have no semantic significance (for example the organizational links "back to the top") are eliminated.

In order to evaluate the quality of IRS, the classical test collections have been developed on the basis of atomics, flats and independents documents. A document is judged as relevant considering its content only, without taking into account its structure nor its neighborhood. Other collections as the one proposed by INEX initiative, focuses on structure but only on logical structure of documents.

Thus we have built our own structured test collection, based on a new definition of a structured relevance. That is a huge work to create manually a structured test collection: to evaluate the relevance of each page, the judges have to consider complex informational units instead of single doc-

uments. Thus we have built a new collection automatically from an existing one.

5.1 A structured test collection

We have modified the classical French test collection "OFIL" (from the Amaryllis competition), in order to take into account the reading paths. The OFIL collection contains 11'000 documents (about 30 Mb) from the French newspaper "Le Monde". We have fragmented the existing documents in atomic documents of similar length [16], to rebuild the existing reading paths.

The new collection contains 86'000 atomic documents, 11'000 structured documents and 11'000 reading paths. On average, each document is composed by 7.87 atomic documents that have a size of a paragraph (360 characters on average). We define sim_{sd},the average similarity measure in a structured document, and sim_{rp}, the average similarity measure along a reading path:

$$sim_{sd}(sd) = \frac{\sum_{(i,j)\in[1..size_{sd}]^2}(sim_{vec}(a_i, a_j))}{size(sd)} \quad (4)$$

$$sim_{rp}(path) = \sum_{edge=(a_i \to a_j)\in path}(sim_{vec}(a_i, a_j)) \quad (5)$$

Those measure are based on sim_{vec}, the similarity measure between two atomic documents, calculated by the cosine measure. It is interesting to see that $sim_{sd}(sd)$ is lower (10.33 on average) than $sim_{rp}(path)$ (14.14 on average). That means that it exists a coherence in the reading paths building, as their atomic documents are more similar compared 2 by 2 in that order than when they are compared 2 by 2 in the whole structured documents.

5.2 Reading paths building

The reading paths are those defined by the author: the nodes are ordered as they appear in the initial document. In addition to these 11'000 initial reading paths (*Initial order*), we have built three other kinds of other reading paths, in order to compare the author's reading path with other virtual or random reading paths: *Random order*, *LowerSim* (minimizing the average similarity between each atomic document along the reading path), and *HigherSim* (maximizing this similarity).

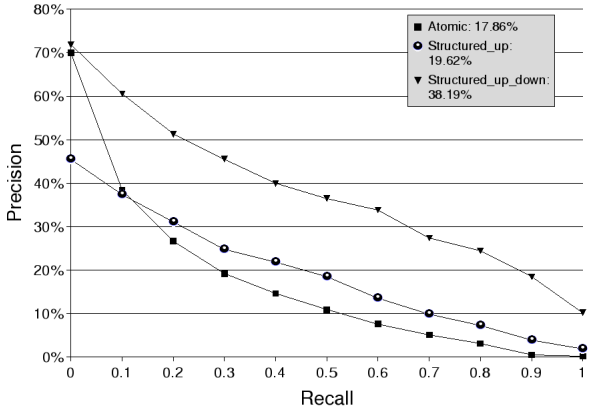
5.3 Evaluation of a structured indexing

We have evaluated three strategies to index and query the atomic documents, called "$atomic$", "$struct_{up}$" and "$struct_{updown}$". The first one is a classical "atomic indexing": each atom is indexed independently, using a *tf.idf* weighting scheme. It uses stemming, stop-words, and an optimized weighting functions for the documents and for the queries.

The second strategy "$struct_{up}$"[2] propagates the index from the leaves of the documents to the top, and propagates the relevance from the top to the leaves. A structured document is indexed with a linear combination of the index of its components (cf. section 4.2), and an atomic document is ranking as relevant if its father is relevant.

The third strategy "$struct_{updown}$" propagates also the index from leaves to top and the relevance from top to leaves. It propagates the "*df*" part of the weighting from the top to the leaves. The figure 4 shows the recall/precision curves for each strategy.

Figure 4. Recall/Precision: $atomic$, $struct_{up}$ and $struct_{updown}$.



The indexing is quite better when the algorithm propagates the indexes. These experiments show the interest of taking into account the hierarchical structure of a document, in order to retrieve parts of this document.

5.4 Evaluation of reading paths indexing

The evaluation of the reading paths indexing aims at comparing it against the structured indexing (using the strategy "$struct_{up}$", cf. section 5.3).

We have fixed α to 0 to cancel the effect of the reading memory, and we have evaluated the average precision for γ values in $[0..1.3]$. The figure 5 shows that the average precision without the principle of accumulation (i.e. $\gamma = 1$) is equal to 19.62%: that is the average precision shown in the figure 4 for the strategy "$struct_{up}$".

These results show an important increase in the average precision when the principle of accumulation is used (i.e. γ less than 1): 32.26% for the $Initial$ strategy against 19.62%

[2]This strategy corresponds to the reading paths indexing with the parameters α and γ set to respectively 0 and 1.

Figure 5. γ from 1 to 1.3, $\alpha = 0$.

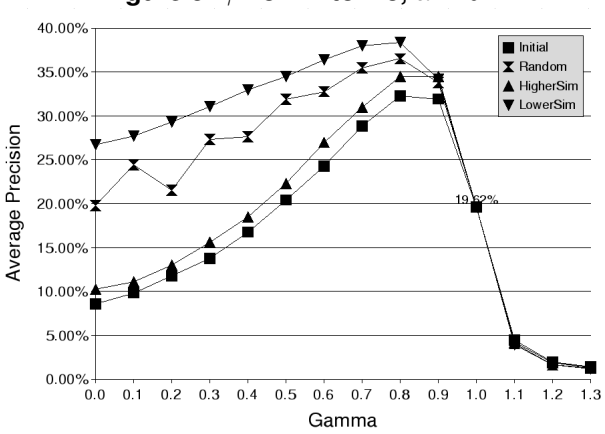
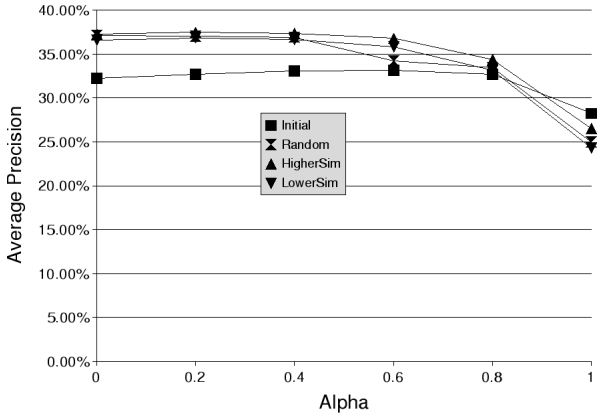


Figure 7. $\gamma = 0.8$, α from 0 to 1.



for the baseline (+64%). It is even better when more importance is given to the end of the reading paths. The figure 6 shows the best results for each strategy.

Figure 6. Best choices for γ.

Strategy	γ	$AvgPrec_{11}$	Increase
Initial	0,8	32,26%	+ 64 %
Random	0,8	36,50%	+ 86 %
HigherSim	0,9	34,46 %	+ 75 %
LowerSim	0,8	38,34%	+ 95 %

We have also evaluated the effect of the reading memory. The γ parameter is fixed to its best value ($\gamma = 0.8$), and we have evaluated the average precision for α values in $[0..1]$. The figure 7 shows that the reading memory has a slight positive effect on the IR precision. In fact, for the $Initial$ strategy there is a slight increase (+ 3%) with $\alpha = 0.6$.

Finally, we have evaluated many combinations of γ and α. The effect of these parameters on the indexing process are interdependent, as one can see in the formula 3. That explains that the SIRS can give the best results with γ and α different than the best choices seen in the figure 5 and 7. Depending on the strategy used to build the reading paths, the combined use of the principle of accumulation and reading memory gives some the best results (cf. table 8).

6 Conclusion and future works

In this paper, we have presented an original point of view on the Web indexing using its structure. The semantic of a hyperdocument is extracted by considering various Web structures, and the indexing process takes them into account. We have focused on the information description inside a (hyper)document: the indexing process takes into account the hierarchical structure of the documents (as seen in section 2.2), as well as the reading structure (i.e. the nodes ordering while reading).

Figure 8. Best choices for combined γ and α.

Strategy	γ	α	$AvgPrec_{11}$	Increase
Initial	0,85	0,15	34,51%	+ 75%
Random	0,75	0,4	39,21%	+ 99 %
HigherSim	0,85	0,25	39,47 %	+ 101 %
LowerSim	0,85	0,2	38,89%	+ 98 %

The Web is considered as a set of potential reading paths in context, instead of a set of flat, atomic and independent HTML pages. It allows retrieving information according to:

- Granularity: from a paragraph to an entire Web site, allowing retrieving parts of SD which would not have been retrieved otherwise, because of their fragmentation in several HTML pages.
- Reading: Reading relations are considered, allowing finding the best reading path among all the potential ones proposed by the author. Moreover, it allows to retrieve a sub-set of pages from a HD which would not have been retrieved otherwise as a SD, because a "good reading path" can link few pages of a SD mixed with hundreds of other pages.

We have shown that our approach is feasible in the context of the Web, using a reasonably large corpus, and we have emphasized the major problems of such a SIRS. Especially, the links typing problematic is essential to rebuild structure from the heterogeneous Web. We have evaluated our SIRS using our own test collection, which has been built automatically, above an existing one, with the main objective to evaluate the reading paths indexing.

The evaluation of three strategies for indexing the structured documents, agrees the results from Wilkinson [20], showing that the best solution is to use information from both atomic documents and structured documents ($struct_{updown}$ strategy), in order to index the atomics documents.

We have also shown that it is interesting to take into account the nodes ordering, in order to index reading paths. Our results show that it is very useful to use both the principle of accumulation and the reading memory. It gives better results than a simple information propagation as seen in the section 5.3. However, we still have to investigate deeply on the impact of our approach. It gives good results, but it is surprising that the best results are those indexing the reading paths in a special order (sometimes random). Our algorithm is able to improve the SIR, but we have to study how to optimize it for the characteristics of the author's ordering.

The automatic building of structured test collection has many disadvantages. Especially, it is difficult to check if the new collection has the same characteristics than a real-world collection. The rebuilt reading paths are also limited to the initial order of the documents, except some artificial strategies, as presented in this paper. Our model thus integrates relations and allows finding reading paths, it is necessary to work on the notion of "reading path relevance", according to its granularity, its textual context, etc. Furthermore, a system should be evaluated in the case of a search for different granularities of hyperdocuments, and in the case of a focused or unfocused relevance. Such an evaluation of a system could be made according to 4 axes: precision, recall, granularity, and focus.

A very promising research problematic is the development of links-based IR methods (as seen in section 2), in the context of a reading paths based model. We think that considering the reading path as the information unit should gives a lot of advantages. It should be easier to propagate information, because of the more suitable information granularity. It should also be more efficient, because an information unit as a reading path makes more sense than the criticized notion of Web page.

References

[1] T. Bray. Measuring the Web. In *5th World Wide Web Conference (WWW'96)*, pages 994–1005, Paris, France, May 1996.

[2] S. Brin and L. Page. The anatomy of a large-scale Hypertextual Web Search Engine. In *7th World Wide Web Conference (WWW'98)*, Brisbane, Australia, April 1998.

[3] A. Broder, R. Kumar, F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A. Tomkins, and J. Wiener. Graph structure in the Web. In *9th World Wide Web Conference (WWW'00)*, Amsterdam, Netherlands, May 2000.

[4] V. Bush. As We May Think. *The Atlantic Monthly*, 176:101–108, July 1945.

[5] J. Callan. Passage-Level Evidence in Document Retrieval. In W. B. Croft and C. van Rijsbergen, editors, *17th ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 302 – 310, Dublin, Ireland, July 1994. Spring-Verlag.

[6] S. Chakrabarti. Integrating the Document Object Model with Hyperlinks for Enhanced Topic Distillation and Information Extraction. In *10th World Wide Web Conference (WWW'01)*, Hong-Kong, China, May 2001.

[7] Y. Chiaramella, P. Mulhem, and F. Fourel. A model for multimedia information retrieval, 1996.

[8] M. E. Frisse. Searching for Information in a Hypertext Medical Handbook. *Communications of the ACM*, 31:880–886, July 1988.

[9] M. Géry and J.-P. Chevallet. Toward a Structured Information Retrieval System on the Web: Automatic Structure Extraction of Web Pages. In *International Workshop on Web Dynamics (WebDyn'01)*, London, United Kingdom, January 2001.

[10] C. Gurrin and A. F. Smeaton. A Connectivity Analysis Approach to Increasing Precision in Retrieval from Hyperlinked Documents. In *8th Text REtrieval Conference (TREC'99)*, Gaithersburg, Maryland, USA, November 1999.

[11] D. Hawking and N. Craswell. Overview of the TREC-2001 Web Track. In *10th Text REtrieval Conference (TREC'01)*, pages 61–67, Gaithersburg, Maryland,USA, November 2001.

[12] Jacques Savoy et Yves Rasolofo. Report on the TREC-9 Experiment: Link-based Retrieval and Distributed Collections. In *9th Text REtrieval Conference*, Gaithersburg, Maryland, United States, November 2000.

[13] M. Joyce. *Afternoon, a story*. Eastgate Systems, Watertown, 1985.

[14] J. M. Kleinberg. Authoritative Sources in a Hyperlinked Environnement. *Journal of the ACM*, 46:604–632, September 1999.

[15] A. Mendelzon, G. A. Mihaila, and T. Milo. Querying the World Wide Web. *Journal of Digital Libraries*, 1:68–88, 1997.

[16] A. Moffat, R. Sacks-Davis, R. Wilkinson, and J. Zobel. Retrieval of partial documents. In *Text REtrieval Conference*, pages 181–190, 1993.

[17] P. Pirolli, J. Pitkow, and R. Rao. Silk from a Sow's ear : extracting usable structures from the Web. In *ACM Conference on Human Factors in Computing Systems (CHI'96)*, pages 118–125, Vancouver, Canada, April 1996.

[18] G. Salton, J. Allan, and C. Buckley. Approaches to Passage Retrieval in Full Text Information Systems. In *16th ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 49–58, 1993.

[19] G. Salton and M. J. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill, Janvier 1983.

[20] R. Wilkinson. Effective Retrieval of Structured Documents. In *17th ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'94)*, pages 311–317, Dublin, Ireland, July 1994.