THESUS, A CLOSER VIEW ON WEB CONTENT MANAGEMENT ENHANCED WITH LINK SEMANTICS

Iraklis Varlamis[1], Michalis Vazirgiannis[1,2], Maria Halkidi[1], Benjamin Nguyen[2]

[1]Department of Informatics
Athens University of Economics and Business
Patision 76, Athens, 10434
GREECE
{varlamis,mvazirg, mhalk}@aueb.gr

[2]INRIA Futurs
4 rue J.Monod
Orsay Parc Club
91893 Orsay
FRANCE
Firstname.Lastname@inria.fr

"The greatest help we can give others is not to share our riches with them, but to discover theirs."

Louis Lavelle, *L'Erreur de Narcisse*, 1939

ABSTRACT

With the unstoppable growth of the WWW, the great success of Web Search Engines, such as Google and Alta-vista, users now turn to the Web whenever looking for information. However, many users are neophytes when it comes to computer science, yet they are often specialists of a certain domain. These users would like to add more *semantics* to guide their search through WWW material, whereas currently most search features are based on raw lexical content. We show in this article how the use of the incoming links of a page can be used efficiently to classify a page in a concise manner. This enhances the browsing and querying of web pages. In this article, we focus on the tools needed in order to manage the links, and their semantics. We further process these links using a hierarchy of concepts, akin to an ontology, and a thesaurus. This work is demonstrated by an prototype system, called THESUS, that organizes thematic web documents into semantic clusters. Our contributions in this article are the following: 1- a model and language to exploit link semantics information, 2- the THESUS prototype system, 3- its innovative aspects and algorithms, more specifically the novel similarity measure, between web documents applied to different clustering schemes (DB-Scan and COBWEB), 4- a thorough experimental evaluation proving the value of our approach.

Keywords: world wide web, link analysis and management, semantic web.

1. INTRODUCTION

The traditional process applied to web searching is to consider that the semantics of a web page can be computed solely from the contents of the page. Typical queries submitted to web search engines are short lists of keywords given by the user. There has been a significant research effort, mainly by Google [5], to find a way of ordering the results of such queries, in order to significantly decrease the time spent browsing irrelevant URLs. The *PageRank* algorithm [5] takes into account various other parameters, such as the link structure of the WWW, in order to give an 'importance' ranking to pages retrieved by a query. We believe, and we show in this article, that we can derive information about the semantics of a page, by analyzing the links that point to a given page. It is in this sense that WWW documents are different from a simple collection of documents. We insist on the fact that links convey simple yet robust information. Such information deserves to be taken into account when answering queries. Our objective is to construct thematic subsets of WWW documents called THESUs (<u>The</u>matic <u>Su</u>bset<u>s</u> of the WWW). The semantic proximity of the pages in a THESU is not only derived by the pages' content but by the semantics of the links pointing to this page, that we call **link semantics**.

Thus, the important feature of the WWW on which we capitalize is the following:

Link semantics enrich web pages' semantics and can be valuable when searching for similarities in the Web

The contributions of this paper are summarized in the following:

- The main contribution is a *model* and a *language* to manage the extraction of pages inlinks, and construct a semantic characterization of each page by enriching these keywords using an ontology.
- A clustering mechanism that exploits a similarity measure to organize a set of web documents into groups of similar pages. The similarity measure deals with weighted sets of terms in a hierarchy. The similarity between sets is not based on exact matching of terms in the two sets but on the combined similarity between all terms of the first set and all terms of the second set. Two

distance (similarity) based clustering algorithms have been adapted to the system requirements, to perform web document clustering.

- A fully implemented client/server system, called THESUS. Using the THESUS link management language, WWW pages are collected, based on an initial list of keywords. These pages are processed, short lists of more precise descriptive keywords are extracted (mainly from the incoming and outgoing links), and mapped to an ontology provided by the user. A clustering scheme is then applied to this enriched data, and the results are stored in a relational database. The THESUS client allows users to perform SQL queries over the stored data and enables searching for interesting documents or document sets based on connectivity and semantics.

We also demonstrate a set of services, which make use of the incoming or outgoing link semantics. They allow users:

- to provide a set of URLs and get the terms that frequently appear in their incoming or outgoing hyperlinks
- to provide two sets of URLs (A and B) and get the terms that frequently appear in the hyperlinks from URLs in A to URLs in B.
- to provide a set of URLs and an ontology and have the URLs characterized and clustered.

All these services are available at our website http://www.db-net.aueb.gr/thesus/.

1.1. Motivation for collective link use

Web content is not always self-descriptive (i.e. multimedia contents). Generally, when answering queries such as looking for a car advertisement, looking for a picture etc. there will be some added value if we also take into account what *others* think about the page. This extra information concerns the type or quality of the targeted page and is not always limited to the contents inside the <A HREF>…</A> tag. We claim that the semantics of a document U can be derived from the semantics of the incoming links that point to it. Assume a document V points to U, using the term "computer science" in the neighborhood of the hyperlink. This is a strong indication that U is characterized by the semantics the term "computer science" conveys. This indication becomes stronger when links from many different documents pointing to U bear the same semantics. This is in a way a collective semantics.

1.2. Motivation for ontology use

Web documents are usually characterized by extracted keywords and are assigned a grade of importance that takes into account link structures [5]. Finding the similarity between documents is based on exact matching between the keywords, and the terms of the users' query. For instance, a document d_1 characterized by the keyword list: d_1={snake, desert} would be judged irrelevant to a document d_2 characterized by the list: d_2={adder, Sahara} yet it is arguable that the two lists of keywords (and thus the documents) are in fact related, since an "adder" is a "snake" and the "Sahara" is a "desert", therefore d_2 deals with the same concepts as d_1, they are just more specialized. By replacing keywords with concepts in a hierarchy, a more flexible document matching process than binary matching can be achieved, handling both specializations and generalizations of senses.

In the current implementation the ontology terms are connected using only one type of relation (a term X "**is a** subclass of" term Y, which denotes that X is less general than Y). This hierarchy of concepts forms a taxonomy, a simplified form of an ontology. Replacing extracted keywords with semantics from an ontology does not aim to identify the objective semantics of hyperlinks but the subjective semantics in a certain domain of interest as those are perceived by the creators of the domain ontology. The main advantage of mapping extracted keywords to an ontology is that a large set of extracted keywords, is mapped to a smaller set of concepts, which are organized in a hierarchy and represent the domain of interest. Users' queries are keyword sets too that can be mapped to concepts. This mapping reduces the dimensionality of the problem of matching documents to queries and moreover allows approximate matching based on concepts instead of "exact keyword matching".

THESUS[1] is a system that enhances the semantic organization of a set of pages and guides the

[1] THESUS is the name of the system and the language. A THESU is a thematic subset, and the plural is THESUs. We refer indifferently to a document, a page or as we will see later, a node, in order to define the basic entities stored in our system.

user within it. It is envisioned as a personal service that will assist the user in the creation and querying of rather compact high quality thematic collections of pages. The concepts developed here are expandable to the full WWW; we bore scalability in mind during the creation and the implementation of THESUS. In order to give a real feeling of what sort of results can be achieved by the system, we also compare it to some popular search engines, even though it is NOT a search engine.

1.3. Paper Organization

The organization of the paper is the following: In section 2 we refer to related work. Section 3 provides the THESUS information model and introduces the notion of "link semantics". Section 4 contains the definitions of the THESUS language operators divided into three main categories: crawling, semantics extraction, and link analysis operators. The architecture of the system follows in section 5. In section 6, we give some examples and results of our system on real data, proving it returns good results. The final section contains conclusions and discusses further work.

Appendices contain the following information: Appendix A summarizes the definitions of data types and operators of the THESUS Model, while appendix B provide the algorithms of the main operators in pseudo-code.

2. RELATED WORK

This section refers to related work mainly from the areas of link information analysis and querying of the WWW. Additionally since we capitalize on web documents semantics and clustering based on semantics, we will present similar efforts in these areas. On the general THESUS topic, we refer to [17, 26] which focus more precisely on the impact of the similarity measure on the clustering schemes of THESUS, and the subsequent experimentation. Readers familiar with these articles may want to skip section 5.4.

2.1. Hyperlink information and semantics

The main issue addressed in this paper is that of managing keywords that are extracted from the links of web pages. This is not a novel idea, since Phelps et al [28] introduce the idea of "robust" hyperlinks that contain "descriptive" information on the target document which can be limited to five words, and some experiments have already been run, such as [7] where an "anchor window", the text neighboring the hyperlink, is defined to be 50 characters. In [32] a structure for hyperlink information with rich semantics emanating from a domain specific conceptual hierarchy is proposed. They also introduce a system to increase the web searching capabilities by attaching information to a document, concerning its concept and its hyperlinks semantics.

2.2. Web querying

WebSQL [3] is a language for extracting information from the web. It models the web as a relational database composed of two (virtual) relations, Document and Anchor, and provides an expandable set of Java classes, for querying the two virtual relations. WebSQL queries serve either information retrieval or web maintenance tasks. Google's [14] advanced search allows inclusion and exclusion of terms that appear in the title or URL of a page. The ranking algorithm, PageRank [5] gives higher rank to most important pages, but does not group pages of similar topic. However, the algorithm exploits information of incoming links to enhance content information of pages. WebWatcher [2] detects the presence of certain keywords inside the hyperlink and neighboring text and ranks the links according to the user's interests. The set of keywords is produced using a training set but is limited to a few hundred terms. ARC system [7] uses the hyperlink neighboring text to describe the contents of the pointed page and to enhance the hub and authority weight on a certain topic. This kind of ranking is called connectivity-based ranking [19].

2.3. Link analysis and Web document clustering

Web document clustering is usually based on connectivity between documents [8] (web structure) and not on conveyed semantics. Web content mining techniques mainly perform text mining on the whole document whilst ignoring web structure. It would be very useful to consider both hyperlinks of pages and their contents when clustering large collections of web documents. Zamir and Etzioni in [35] propose a suffix tree-clustering (STC) algorithm that uses phrases shared between documents to create the clusters. Haveliwala et al. [20] propose a clustering approach in which the similarity is based on words found in the documents along with their occurrence frequencies and the term matching is exact. Some works relate to the importance of links in promoting semantics in a hypermedia network.

Kleinberg [22] states: "The link structure of a hypermedia environment can be a rich source of information about the content of the environment (...) But for the problem of searching in hyperlinked environments such as the World Wide Web, it is clear from the prevalent techniques that the information inherent in the links has yet to be fully exploited."

There is a consensus that clustering techniques should be applied to the results of a query rather than on the whole web space, in order to discover groups of relevant documents. An interesting server that uses hyperlink's structure to group interconnected results is Kartoo [23]. Vivísimo [32] proposes a clustering approach for web document organization, making use of the short descriptions returned by other search engines. Northern Light [25] classifies all the documents of the entire collection into pre-defined subjects and, at query time, selects the subjects that best match the search results.

Our working hypothesis in THESUS is that web page characterization is more valid when external information is used instead of the page's contents. As a result, hyperlink information can refer to the semantics of the target as they are provided by the source of the hyperlink. Hopefully, external authors act independently of the page's author and their opinions can be added to the ranking of the page, yet our system could also be able to detect 'malicious' links by finding that they have no term in the ontology that sets them close to the ones that appear in other links for instance, and a scheme could be devised to filter them out. In our system, we rely on the use of an ontology and WordNet, in order to compute similarities between words. For more details on ontologies you can refer to [15] and for information on WordNet [33],[1].

3. THESUS' MODEL

In this section, we provide an informal introduction of the THESUS model that motivates the formal specifications of the basic data types that will be detailed in section 4.

We model the WWW using the following concepts :

- *Pages,* or more precisely URIs, that provide a unique way of accessing some information on the web. A page can be empty of textual content (for instance a picture) yet have other pages pointing to it, with meaningful semantics. Henceforth, we will interchangeably use the terms "page", "node" and "document" for representing WWW pages.

- *Links,* uniquely defined by source and target nodes. We believe that it is the links that carry the semantics of the web page they point to. Therefore, we are interested in the union of the semantics of all the links pointing from a given page to another one. The exact location of link anchors in the two pages is also interesting to consider.

3.1. Link semantics

Assume two pages S (source) and T (target) and the set of links $\{l_i\}$ that point from S to T. It is quite common for authors of a page to link it to another page by using a small set of keywords to describe the target page. These keywords usually appear in the hyperlink source tag, i.e., *<A HREF=…>a page on computer science</A>*, or in a short area around the hyperlink, i.e.,*you will find <A HREF=…>here</A> information on computer science*. We call this set of keywords $\{k_j\}$ **link keywords**. Semantics is the study of meaning in language, and refers to the concepts to which extracted keywords map to given an ontology and a mapping mechanism from keywords to concepts. Given an ontology, which is the representation of the domain of interest and is the bearer of semantics in our system and by using the WordNet thesaurus, we are able to map extracted keywords to ontology concepts, thus converting link keywords to what we call **link semantics**.

3.2. Page characterization

As mentioned before, page T is effectively characterized by the semantics of its incoming links. Therefore the set of keywords (and the corresponding set of concepts) that is derived from $\{l_i, \{ k_j\}\}$ semantically defines the classification of page T seen from page S. In other words the union of the sets of keywords $\{k_j\}$ is the keyword characterization that T assigns to S. This keyword characterization is the basis for producing the semantic properties that T assigns to S. Taking into account all the links to page T from different source pages $\{S_i\}$ we get an aggregate characterization, a collective view of how the set of pages $\{S_i\}$ characterizes T. In the case of a set of target pages $\{T_i\}$ the result of processing the links from $\{S_i\}$ to $\{T_i\}$ will be the collective characterization that $\{S_i\}$ assigns to $\{T_i\}$.

3.3. THESUS Model datatypes

In this section we define the entities that constitute the nucleus of the THESUS language design. Our reference space is the WWW, perceived as a collection of documents (`w_docs`) connected with links (`w_links`). For a thematic area we assume a subset of the documents and links of the WWW (`docs` and `links` respectively) form a thematic subset (THESU). For completeness we assume the data types `URL` (the unique identifier of a page in the WWW context), `keyword` (a string of characters characterizing a page) and `ontology_term` (a concept from the domain ontology). We also consider sets of `URL`, `keyword` and `ontology_term` as types in our system. We will now define the fundamental entities necessary for THESUs' creation and manipulation.

3.3.1. THESUS Model entity definitions

Definition 1: A *collection of documents*, `docs` is a set of vectors of the form `(URL, {keyword}, {ontology_term}, other info)`, where a. `URL` is the identifier of the document as it appears on the WWW, b. `{keyword}` is a set of keywords characterizing the document, c. `{ontology_term}` is the set of corresponding terms of the ontology that semantically describe the document, and have been constructed from the set of keywords, and d. `other info` are additional informative facts that can be assigned to the document, such as the content of the page perceived as a string of characters (text) or the last date of change of the document (modification date). The documents in such a collection constitute a THESU.

Definition 2: A *collection of links*, `links`, is a set of vectors: `(URLS, URLT, {keyword})` where: `URLS` is the URL of the document from which the link emanates, `URLT` is the URL of the document to which the link points, `{keyword}` is a set of keywords characterizing the document and `{ontology_term}` is a set of terms from the ontology that semantically define the document.

3.3.2. WWW Entities

In order to construct `docs` we need to extract information from pages of the web. The following collections, `w_docs` and `w_links` are used during construction. In a nutshell, these entities are our vision of the WWW as a virtual entity. We are able to materialize portions of it by combining web services, such as search engines and web crawling techniques.

Definition 3: The *WWW collection of documents*, `w_docs`, is a set of vectors of the form `(URL,text)`, where a. `URL` is the identifier of the document as it appears on the WWW, b. `text` is content perceived as a string of characters.

Definition 4: The *links in the WWW*, `w_links`, is a set of vectors of the form `(URLS, URLT)`, where a. `URLS` is the URL of the document from which the link emanates, b. `URLT` is the URL of the document to which link points. Here we assume that all the links between two pages are aggregated in one vector in the `w_links` entity.

4. THESUS language

In this section we introduce THESUS language that enables thematic selection of WWW subsets and subsequent enrichment by extracting semantics from the links pointing to the pages of the subset. The THESUS language defines operators that combine connectivity features and related semantics, yielding meaningful results that are otherwise not obtainable from existing search mechanisms. The model's datatypes, the language's operators along with necessary extensions and auxiliary functions are summarized in APPENDIX A.

When designing a language or a set of predicates we bear in mind the following issues:

- minimality, we are searching for a minimal set of operators, which have the least possible overlap of semantics, are simple in their design and can be combined in constructing richer operations. The operators are grouped in distinct categories (crawling, link semantics extraction, link analysis) aimed at different tasks. Nonetheless, there is some overlap between some of them since certain operators could be expressed in terms of the others.
- expressiveness, the set of operators should be as high level as possible, easily understandable by the potential users and with clear semantics. The operators are semantically rich and are directly applicable to the domain considered (i.e., traversal of the WWW graph and semantics extraction).
- closure, the sets of types and operators on them are closed. This implies that any simple or complex operator returns a type that is already defined in our system.

Let us note that the elements of type `ontology_term` are used as a specialization of the `keyword` type in some of the operators defined later on. In order to avoid repetition, in the definition of the language operators, we use only the `keyword` type. However, operators based on the semantics of the extracted keywords can be defined in a similar way. The transformation from one to the other is done by calling the function getSemantics({keyword}) (see section 5.3).

In this section we define the basic operators as introduced above and define additional operators on top of them, which convey richer knowledge about a set of WWW pages. The following list of operators is neither complete nor exhaustive, but it gives an indication of the potential of the fundamental set of THESUS operators in terms of finding rich semantics in the WWW. These higher level operators can be all expressed based on the previously introduced ones.

4.1. Crawling operators

In order to create and characterize THESUs it is necessary to be able to traverse the WWW graph in order to a. find the target pages of a page's outgoing links and b. to find the source pages of a page's incoming links. This crawling function can be carried out in several levels in the WWW graph starting from some node under consideration. We introduce the following "crawling" operators:

Definition 5: `fetch(URL)`. Given a URL, it returns either its textual content or null in case the link is broken.

Definition 6: **`groupMatch({URLexpr})`**. Given a set of URL expressions, `{URLexpr}`, the operator returns a set of URLs that match any of the expressions. The expression is a partial URL string and all returned URLs should begin with the URL expression.

Example For instance the operation `groupMatch({http://www.nasa.gov/)` returns all the pages under http://www.nasa.gov/ .

Definition 7: **`crawl({URL},N)`**. Given a set of URLs `({URL})`, and an integer N the operator returns a set of URLs, consisting of the pages that are being pointed to from each page in `{URL}` in each level of the WWW graph until level N. Positive values of N imply that we crawl the WWW in a "forward" manner by following the outgoing links of

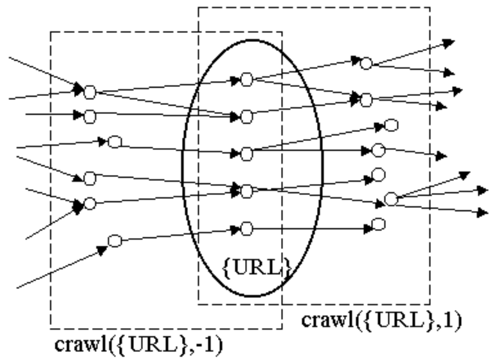


Figure 1 Crawl operator

pages in {URL}, whereas negative values of N denote "backward" crawling through the incoming links of pages in {URL}. The example in figure 1, shows that for N=1 the crawl operator returns the documents in {URL} and all the documents pointed by them, whereas for N=-1 it returns the documents in {URL} and all the documents that point to them.

4.2. Links' Semantics operators

Here we propose a set of auxiliary functions to be used in the definition of this category of operators. In order to extract the semantics of links we need to be able to locate the position of the source of the link (i.e. <A HREF='…'>) in a page and subsequently process its neighborhood in order to extract keywords. This is performed using a set of auxiliary functions.

Function 1: `getpos(text, URLT)`: Given the text of a page, `text`, and the string corresponding to the target URL of the link, `URLT`, the function returns the set of positions -{pos}- in the `text`, where the string of URLT occurs.

Function 2: `process(w_docs.TEXT, pos, 100)`: It is called for each occurrence of URLT, processes the hyperlink's neighboring area (i.e. 100 characters) and returns a set of keywords -{keyword}- extracted from the area that represent the keyword characterization of the specific link.

Function 3: `getSemantics({keyword})`: We use this function, that returns a set of terms from the ontology -`{ontology_terms}`. The function maps the extracted set of keywords to the closest concepts and is detailed in section 5.3.

THESUS' language works not only with specific keywords contained in the pages, but also with ontology terms, and thus truly represent 'semantics' as an understanding of what a page deals with.

Definition 8: **`linkKeywords (URLS, URLT)`**. Given two pages identified by URLs, `URLS and URLT`, the operator returns the union of keywords that are extracted from all the links emanating from `URLS` targeting to `URLT`.

Definition 9: **`groupKeywords ({URLS}, {URLT})`**. Given two sets of URLs, `{URLS}` and `{URLT}`, the operator returns the keywords extracted from all the links that point from any page in `{URLS}` to any page in `{URLT}`. The sets of URLs {URLS} or {URLT} in groupKeywords may contain a single URL, or can be empty. When {URLS} is empty, operator `groupKeywords` examines every incoming link to pages in {URLT}. This information indicates how pages in {URLT} are described; in other terms "what the world thinks" for pages in {URLT}. When {URLT} is empty, the operator examines every outgoing link of pages in {URLS}. The extracted information indicates how pages in {URLS} describe the pages they point to, in other words, how pages in {URLS} "perceive the world".

Definition 10: `thematicCrawl({URL},{keyword},N)`. The operator combines the definitions 7 and 8. Given a set of URLs `({URL})`, a set of keywords `({keyword})` and an integer N the operator returns a set of URLs. This set of URLs consists of the pages that are being pointed to from each page in `{URL}` in each level of the WWW graph until level N, having at least one keyword from the `{keyword}` set in the link keywords.

Example: In the case of thematic`Crawl({U1},{keyword1,keyword2}, 2),` starting from page `U1` the operator returns all the URLs that are targeted by links starting from `U1`, containing either keyword1 or keyword2 and that are reachable after at most 2 hops.

The operator is called thematicCrawl since when it is used with keywords that fall in the same thematic area it collects URLs that have a high probability to be on similar subjects. The operator can also defined as thematic`Crawl ({URL}, {ontology_term}, N)`, where only the links with specific semantics (i.e., terms of the ontology or similar words) are followed.

4.2.1. Advanced link semantics operators

The definition of the `groupKeywords` operator assumes that the set of keywords returned, is the union of the keywords that appear in each link without counting keyword occurrences. However if we take into account keyword occurrences, the modified `groupKeywords` definition conveys three different meanings based on the way keyword occurrences are aggregated:

In the definitions that follow, we use three auxiliary functions:

Function 4: getkeys(WKEYS), which takes a set (`WKEYS`) of keywords and number of occurrences pairs `(KEY,TIMES)` - as an input and returns the set of keywords {`KEY`}

Function 5: getTimes(WKEYS,K), which takes a set of `(KEY,TIMES)` pairs and a keyword `K` as an input and returns the number of occurrences (`TIMES`) of keyword `K`

Function 6: update(WKEYS,K,N), which takes a set of (KEY,TIMES) pairs, a keyword K and an integer N as an input and increases the TIMES value of K by N (updates pair (K,T) of WKEYS to (K,T+N)).

Definition 11: **weightedGroupKeywords ({URLS}, {URLT})**. Given two sets of URLs, {URLS} and {URLT}, the operator returns a set of pairs {(KEY, TIMES)}, where KEY represents a keyword that appears in the links that point from any page in {URLS} to any page in {URLT} and TIMES the number of occurrences of keyword KEY.

Definition 12: **weightedTargetKeywords ({URLS}, {URLT})**. Given two sets of URLs, {URLS} and {URLT}, the operator returns a set of pairs {(KEY,TIMES)}, where KEY represents a keyword that appears in the links that point from any page in {URLS} to any page in {URLT}and TIMES the number of target pages characterized by keyword KEY.

Definition 13: **weightedSourceKeywords ({URLS}, {URLT})**. Given two sets of URLs, {URLS} and {URLT}, the operator returns a set of pairs {(KEY,TIMES)}, where KEY represents a keyword that appears in the links that point from any page in {URLS} to any page in {URLT}and TIMES the number of source pages that use keyword KEY in their links to {URLT}.

The examples in section 6.2.2 illustrate the different semantics of the various implementations of groupKeywords.

4.3. Link Analysis operators

Apart from the core set of fundamental operators, we considered defining specific operators that are of high added value. Moreover, potential users of the system can easily define other operators in terms of the fundamental ones. We extend the definitions of hub and authorities [22] and those of co-citations [29] and couplings [21]. We exploit and enrich the above concepts with semantics extracted from the links. We introduce the following concepts:

- *Thematic Hubs*: We enhance the semantics of a hub by considering the semantics of the outgoing links. It is obvious that a hub with many outgoing links of similar semantics is thematically focused and potentially more interesting. Kleinberg in [22] provides a recursive definition of hubs and authorities, however, for simplicity we use the one step definition.

- *Thematic Authorities*: We enhance the semantics of an authority by considering the semantics of the incoming links. For instance if an authority node is pointed to by links of type "databases" it is very likely that this is a page that contains reference material on databases.
- *Thematic Co-citations and Couplings*. If nodes B and C contain links to node A then A is "co-cited" by B and C. Moreover if the links have similar semantics then the semantics of A are strengthened and we have an indication that B and C have some similarity (thematic co-citation). On the other hand, if document C points both to documents A and B and the two links bear similar semantics, we have an indication that A and B are similar (thematic coupling).

5. THESUS SYSTEM ARCHITECTURE

This section presents the architecture of the THESUS system. The system components, their input and output, their operation and the innovative aspects they introduce follow.

The THESUS system is fully implemented. The system's components include (see Figure 2.):

- The "information acquisition" module, that gathers a set of URLs that appear relevant to the topic under consideration.
- The "information extraction" module that extracts keywords within an expanded hypertext area around a hyperlink.
- The "information enhancement" module that enhances extracted hyperlink information with semantic information by mapping extracted keyword sets to sets of concepts in an ontology.
- The "clustering module" that partitions the set of URLs into semantically coherent subsets based either on the extracted keywords or on the respective ontology concepts. Similarity between pages is computed using a novel similarity measure for sets of ontology terms. The measure is based on the distance of terms in the ontology hierarchy, instead of exact term matching.
- The "query engine" that: enables searching in the collection. It takes advantage of the clusters that are closer to the user's query and ranks the results accordingly.

Information concerning the documents is stored in a relational database. Knowledge from the WordNet 1.6 database [33] and the ontology is also exploited. The system has been evaluated with experimental sets of the order of 10^4 web documents, a reasonable size for a thematic web repository.

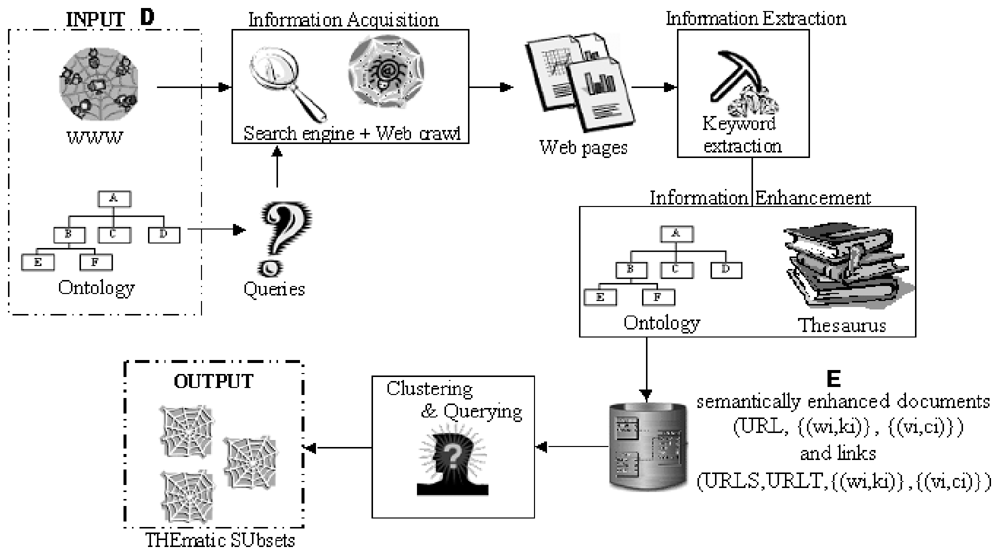


Figure 2. THESUS Functional Architecture

It is reasonable that the acquisition and processing of a large corpus of web documents is a resource demanding task. As a consequence the first three modules of the system are mainly used as off-line tools for building a semantically enriched repository of web documents and the clustering module assists in organizing this repository into meaningful subsets. The query module operates as an end-user tool that allows for browsing and searching the enriched information repository. See section 6 for details on running times for the various tasks.

The crawling system was developed as a multithreaded Java application. Information extracted from web pages and related semantics are stored in a relational DBMS (Microsoft-SQL Server®). The access to the database is performed using a JDBC native driver. The client application remotely connects to the database server using JDBC and allows users to perform the THESUS language operations on the collected data. The web services are implemented as Java classes.

5.1. Information acquisition

We assume an initial set of documents $D = \{d_i\}$, that are related to a certain thematic domain of interest, and an ontology O on the same domain (i.e., Arts, Music, Technology, etc). The crawler generates the core set of URLs depending on the existence of an initial set of documents D:

a) If D is given, the crawling starts from this set without using any web search engine services. Expansion is performed as multiple iterations of the "root-set generation" algorithm of HITS

[22], limited to those links that display on-topic keywords in an expanded window of hypertext. In opposition to the full focused crawling [6], this crawling is totally unsupervised and is not based on exact matching between found keywords and ontology terms. Expansion follows hyperlinks that are "*characterized*" by at least one concept in the ontology (the anchor-text area around and inside the hyperlink contains a keyword that is mapped to the specific concept in the ontology). This process stops after a certain number of crawling levels have been completed or after all documents suggested for crawling in a step have already been collected. Due to the weak connectivity of the WWW documents (only 28% of the web is strongly connected according to [4]) the crawling process for a small ontology (i.e. less than 20 concepts) and a small initial set stops in a few steps, whereas, for an ontology with a few hundreds concepts and a larger initial set stops after many crawling steps.

b) If D is not given, we consider no prior knowledge on the specific domain, so we use a web search engine to locate documents that possibly relate to the domain. The only input is the ontology O describing the user's interests formed as a hierarchy of concepts. The system generates the paths from the root to all the nodes in the ontology, queries a web search engine for all the generated paths (node$_1$ AND node$_2$ AND … AND node$_K$) and collects the top answers returned for each query. If the ontology contains N concepts, then N paths and respectively N queries are generated. If the search engine returns a maximum of M answers the total number of URLs in D will be no more than MxN. Experimentation shows that the results are usually highly overlapping (in terms of URLs) so the final number of documents in D after removing duplicates is significantly less than MxN.

5.2. Information extraction

In order to enrich the information on the core set of URLs, we crawl one level backwards and collect the incoming links information. The crawl({URL},-1} operator requires the use of a web service that gives the top N incoming links of a page. The documents are processed as follows to enhance their classification quality. The first step is to process the incoming links to D and extract keywords from the respective source pages. Calling the weightedGroupKeywords operator for the documents {b} that point to each document d_i in the set we get a set of pairs of keyword/times of occurrences $\{(k_{di}, t_{kdi})\}$ that characterize d_i.

In section 6 we experimentally verify the potential of the approach.

The issue of extracting keywords from the links is important in the context of this paper. Taking into account the works of [28], [7], in THESUS, keywords are extracted from i) the link anchor (i.e. the string between the <a> and </a> tags) and ii) from two text strings (each of 100 characters long), one preceding the starting link tag and another following the ending link tag. The "window" is trimmed whenever certain html tags appear, such as <a>, <li>, <tr>, <td> etc, because such tags usually signify the logical end of the hyperlink neighboring area. The window size has been chosen *after* experimentation so that the resulting mean number of keywords is approximately five [28]. The procedure is summarized in Figure 3.

For example in the HTML fragment that follows, keywords are extracted from a hyperlink to the Music Library of the University of North Texas homepage:

```
<p><a href="http://www.lib.unc.edu/music/">University of North Carolina, Chapel Hill
- Music Library</a></p><p><a href="http://www.library.unt.edu/music/default.htm">
University of North Texas - Music Library</a> <br> &quot;...one of the largest
university music collections in the United States, has over 250,000 volumes of
books, periodicals,
```

The phrases extracted from the highlighted URL are:

- From the hyperlink area (dark color white text) the phrase: "University North Texas Music Library", after removing frequent keyword "of" and character "-".
- From the 100 characters preceding (gray area) the hyperlink: The area is trimmed where </a> appears, tags </p> and <p> are then removed, so no characters remain.

```
1)   Find the limits of the hyperlink area (start, end position)
2)   Find the limits of the pre-hyperlink area (start-100, start)
3)   Find the limits of the post-hyperlink area (end, end+100)
4)   Search for image tags in the inside hyperlink area.
    -Extract keywords from the alt attribute of image tags
    -Remove image tags
    -Remove punctuation, stopwords, small words
    -Extract keywords from the remaining text
5)   Search for special html tags (table row, table column, hyperlink, list item
        etc) in the pre-hyperlink or post-hyperlink area. Since such tags denote a
        change on the document structure, they should limit the hyperlink's
        neighboring area.
    -Remove complete html tags inside the area. For example <B>, <I>, <IMG….> etc.
    -Remove punctuation, stopwords, small words
    -Extract keywords from the remaining text
```

Figure 3 The procedure of keyword extraction from hyperlinks

- From the 100 characters following (gray area) the hyperlink: The last partially selected number is ignored, <br>, &quot; and punctuation marks are removed, stop-words are removed too. The remaining phrase is: "largest university music collection United States".

The set of (keyword, occurrences) pairs that are extracted for this target URL is {(university,2), (north,1), (texas,1), (music,2), (library,1), (largest,1), (collection,1), (united,1), (states,1)} }. The system will then try to map this set of keywords to relevant terms in the ontology. "music" and "library" for instance will probably be mapped as such, whereas "north" might not be mapped at all.

5.2.1. Amendments on keyword extraction process

Some of the incoming links of a page are the navigation links that usually provide abundant and possibly irrelevant information for the targeted page (i.e. "back", "next", "home" etc.). Additionally, intra-site hyperlinks (links that point to pages in the same host), may carry misleading information on purpose, and attempt to affect web-agents that extract information from hyperlinks. A common solution to the first problem is to ignore all the navigation links when performing link analysis. One could also argue that it is better to disregard intra-site links, since they also could provide biased information about pages. However, in our system, keywords extracted from such links do not affect the characterization of a web page. Keywords that are commonly used for navigation purposes (i.e. back, next etc) are included in the stop-words list and consequently ignored by the keyword extraction module. Nevertheless, keywords that are not included in the stop-words list but are irrelevant to the domain of interest (as it is expressed by the ontology) will not be mapped to a node in the ontology, while keeping links that are relevant.

The system also provides a solution for malicious hyperlinks, which are deliberately added in web pages to increase the access and ranking rates for other web pages. They usually point to pages in the same or affiliated web sites using certain keywords that do not really characterize their contents. The output of the keyword extraction module, for a web page, is a set of keywords associated with weights that represent the number of ***distinct*** links that point to the page using each keyword. This weakens the spamming effect that misleading pages create using the same keyword many times, and strengthens the characterizations provided by many different web pages.

5.3. Information enhancement

Assuming a domain specific ontology O representing the domain semantics. Each set of keywords $\{k_{di}\}$ is mapped to a set of terms $\{t_j\}$ of the ontology O, using a thesaurus, in our case WordNet. Based on the Wu & Palmer similarity measure [34] that measures similarity between single terms, we build our measure that gives a correct assumption of the similarity between sets of terms, and use it to find the closest set of ontology terms to the set of extracted keywords. The outcome of this process is that every document in the original set is now enriched with:

- Keywords and weights (indicating the occurrences of a keyword in the incoming links of a page)
- Terms of the ontology into which a document is classified and the respective weights.

We use the following notation to define these ***enhanced documents***:

Definition: an enhanced document is the triplet (Doc, K, C), where Doc is the document identifier (i.e. URL), K (resp.C) is the set of couples $\{(w_i,k_i)\}$ (resp. $\{(v_j,t_j)\}$) of weighted keywords (resp. weighted terms) that define the document (w_i (resp. v_j) is a real, k_i (resp. t_j) is a string). w_i and v_j are not necessarily the same if i=j.

A weight is a real from the interval [0;1]. A value of 1 indicates total relevance, 0 indicates no relevance. A value of 0.2 would indicate low relevance.

5.3.1. WordNet

WordNet is an online lexical reference system, in which English nouns, verbs, adjectives and adverbs are organized into synonym sets, each representing one underlying lexical concept. Different relations link the synonym sets, such as IS-A for verbs and nouns, IS-PART-OF for nouns etc. For a verb or noun WordNet provides a *synset*, a set of different senses that the verb or noun may have. Verbs and nouns senses, are organized in hierarchies forming a "forest" of 25 trees (for the IS-A relationship) of nouns and 15 of verbs. For adjectives and adverbs WordNet provides synonyms.

For each keyword in WordNet we can have a set of senses and in the case of nouns and verbs, a generalization path from each sense to the root sense of the hierarchy. For example the word "wind" has 8 verb senses and 8 noun senses. The first sense of "wind" as a noun gives the following path:

wind => *weather, weather condition, atmospheric condition* => *atmospheric phenomenon*
=> *physical phenomenon* => *natural phenomenon, nature* => *phenomenon*

5.3.2. Ontology

In order to abstract extracted keywords to the closest semantics, we need to refer to an ontology of terms that are relevant to our domain of interest. Any hierarchy of terms can be used, provided it can be modeled as a tree. Examples of ontologies that can be employed, are the ones suggested by The DAML Program [9] (e.g. ontology on music http://www.daml.org/ontologies/276). In order for our system to work, we need to have a mapping of each ontology term to a *synset* in WordNet. Normally all the senses of an ontology term are considered. However, semantics extraction can be improved if the irrelevant senses are ignored. For example, for an ontology on music, that contains the term "wind", it is preferable to select only those senses relating to music and ignore the senses relating to weather. This optional step requires the user intervention. The system presents all the possible senses given by WordNet and the user selects those that relate to the domain. Following the example on music, for the term "wind" we keep only 2 out of the 16 suggested paths:

wind instrument, wind => *musical instrument* => *instrument* => *device*
=> *instrumentality, instrumentation* => *artifact, artefact* => *object, physical object* => *entity, something*
wind, wind up => *tighten, fasten* => *change, alter*

5.3.3. Associating sets of keywords to concepts in the ontology

The information extraction module generates for each document d_i a set of keywords $\{k_i\}$ with a respective set of weights representing the number of documents that point to d_i using a particular keyword. So each document d_i is described by the following structure: d_i= (URL, $\{(k_j , n_j)\}$) where n_j represents the number of documents that point to d_i using keyword k_j. It is widely accepted that the keyword importance increases proportionally to n_j [5].

Given a set of keywords that characterize a document, an ontology and WordNet, the "information enhancement" module attempts to find a set of concepts that best match to this set. WordNet provides all the possible senses for each concept in the hierarchy and a path for each sense starting from an abstract concept (i.e. entity) and specializing down to the specific sense.

In order to find the closest term in our ontology for a keyword k, we compute the Wu & Palmer similarity between each sense of k and each of the corresponding terms of $\{t_j\}$ in the ontology. If maximum Wu & Palmer similarity is found between sense k_x of k and sense t_{jy} of ontology term t_j, we select the pair (k,t_j) and map keyword k to the ontology term t_j. This approach extensively tests all the senses of the ontology terms and of the extracted keywords, even those that are outside the scope of the ontology. Thus, it is slow and tends to provide wrong mappings.

The first step to improve mapping is to reject the irrelevant senses of the terms in the ontology, as described previously. In order to further improve mappings, it is essential to reduce the number of senses examined for each keyword k by removing senses that are completely irrelevant in the scope of the ontology. To do this, for each keyword k in $\{k_j\}$ we compute the similarity of each of its senses to the senses of all other keywords in $\{k_j\}$ and keep the set of senses that gets the highest similarity score. This set is characterized by high self-correlation and gives significantly better results.

If the keyword set contains *n* keywords, we first create all the possible sets of senses that contain one sense for each keyword (cardinality *n*). Since the Wu & Palmer similarity measure can be employed for pairs of senses, we compute the average Wu-Palmer similarity for all the different pairs of senses in each set. For example, for the words *guitar, flute* and *wind*, WordNet provides 1, 3 and 8 senses respectively. For all the 24 triplets of senses the average Wu-Palmer similarity for all pairs of senses ($\frac{n \cdot (n-1)}{2}$ combinations, 3 for each triplet) is computed. The keyword set (*guitar, flute, wind*) gives a score of 0.8 to the triplet of meanings (*guitar, flute/transverse flute, wind instrument/wind*) and less than 0.5 to any other combination. Similarly the set (*storm, cloud, wind*) gives a score of 0.8 to the triplet of meanings (*storm/violent storm, cloud, wind/air moving*) and lower scores to any other combination. These are indications that "*wind*" is closer to the sense of "*wind instrument*" when it appears in the set (*guitar, flute, wind*), whereas it is closer to the sense of "*wind as weather phenomenon*" in the set (*storm, cloud, wind*). As a result the first set will be mapped to the set of ontology terms (*string instrument, wind instrument, wind instrument*) using an ontology on music, whereas the second set will not be mapped at all to a set of terms related on music.

Since the most appropriate senses for each keyword have been selected, each keyword in $\{k_j\}$ is mapped to the closest ontology term as previous shown. As a result each k_j is mapped to a term t_i with a similarity s_j. It is expected that more than one keywords in the set of keywords $\{k_j\}$ are mapped to the same ontology term, or to no term at all, so the cardinality of $\{t_i\}$ is usually smaller than that of $\{k_j\}$. The weight assigned to each term t_i is the average similarity of all keywords k_i mapped to t_i, taking into account the respective weight n_j of each keyword k_j. The weight is computed using the formula:

$$r_i = \sum_{k_j \to t_i} (n_j \cdot s_j) \Big/ \sum_{k_j \to t_i} n_j \qquad \text{Equation 1}$$

So each document d_i is represented as (URL_i, $\{(t_i, r_i)\}$), where $r_i \in [0,1]$ since $s_j \in [0,1]$.

5.4. Clustering module

In this phase, the documents are fed into the clustering module. To be able to run the clustering algorithm, we need a similarity measure for documents. A lot of research has been conducted in this field (see [16]). One contribution of THESUS is the introduction of a novel similarity measure between sets of keywords.

5.4.1. Similarity measure

The similarity measure introduced in [17] is used for clustering the documents, and for answering queries. Both documents and users' queries are seen as weighted sets of terms and this measure is employed to evaluate the similarity between web documents or between a document and a query. As a result, queries may retrieve documents that do not contain a certain keyword but a synonym, hypernym or hyponym of it, and that are nonetheless very relevant.

Let us not forget that we aim at computing similarity between sets of *weighted* words (the ontology terms), and not simply words. There has been very little research on similarity measures between sets of elements (see [13],[11], [24]). The similarity measure employed in THESUS is a generalization of Wu & Palmer's [34] measure and is defined in the following. A detailed discussion on the properties of this measure can be found in [17].

Notations:

1. Let Ω represent the ontology (set of terms in a hierarchy).
2. We use **cursive capitals** $\mathcal{A},\mathcal{B}$ to represent sets of weighted words, such as: $\mathcal{A}=\{(w_i, k_i)\}$, and $\mathcal{B}=\{(v_i, h_i)\}$, with $k_i, h_i \in \Omega$ and $w_i, v_i \leq 1$.

We define: $\zeta(\mathcal{A},\mathcal{B}) = \frac{1}{2}\left[\left(\frac{1}{K}\sum_{i=1}^{|\mathcal{A}|} \max_{j\in[1,|\mathcal{B}|]}\left(\lambda_{i,j}\times S_{W\&P}\left(k_i,h_j\right)\right)\right)+\left(\frac{1}{H}\sum_{i=1}^{|\mathcal{B}|} \max_{j\in[1,|\mathcal{A}|]}\left(\mu_{i,j}\times S_{W\&P}\left(h_i,k_j\right)\right)\right)\right]$ Equation 2

Where $\lambda_{i,j} = \frac{w_i + v_j}{2\times \max\left(w_i, v_j\right)}$, $K = \sum_{i=1}^{|A|}\lambda_{i,x(i)}$ and $x(i) = x \mid \lambda_{i,x}\times S_{W\&P}\left(k_i,h_x\right) = \max_{j\in[1,|B|]}\left(\lambda_{i,x}\times S_{W\&P}\left(k_i,h_j\right)\right)$

Simply put, K is a normalizing factor (the sum of all the $\lambda_{i,j}$ used). $\mu_{i,j}$ and H are defined similarly.

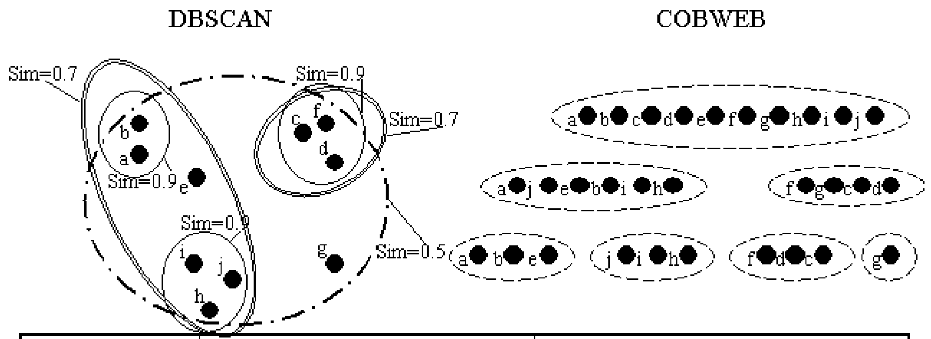
Intuitive explanation: the weight factors give less importance to terms that do not clearly describe the document. For instance if a document is described by a term with a low weight, we do not want this weight to play too great a role in the *commonality* and the *differences* with other terms describing other documents. The *commonality* is calculated by finding the terms in document B that are closest to those in document A, and the *differences* are calculated by taking the terms in B and finding those that they are close to in A. As far as the weights are concerned, $\lambda_{i,j}$ and $\mu_{i,j}$ reduce the overall impact of terms with low weights. We also note that regardless of the values of the weights w_i and v_j the maximal value for $\lambda_{i,j}$ (resp. $\mu_{i,j}$) is equal to 1 and is reached for $w_i = v_j$.

We refer to [26] for more details on the similarity measure its properties.

5.4.2. Web document Clustering algorithms

The problem is considerably different compared to the case of points in a metric space since there are no coordinates and ordering. We only have a similarity measure between the objects to be clustered, which are sets of (weighted) strings that correspond to concepts of an ontology. The first algorithm used in THESUS is a modification of the density-based algorithm DBSCAN [10]. Modification details and the algorithm itself are described in [17]. An incremental hierarchical algorithm, a variation of the COBWEB algorithm[12], has also been implemented.

The two algorithms operate differently and produce different final partitioning. In the modified version of DBSCAN each document D of a cluster must contain a *minimum number of neighboring documents* (MinDoc). Neighbors of D are the documents whose similarity to D is higher than or equal to a *minimum similarity* (MinSim) threshold. DBSCAN considers as "*noise*" the documents that are not density-connected [10] with other documents. DBSCAN produces a flat clustering scheme. By decreasing MinSim the size of clusters increases, several clusters are merged and fewer documents are considered as noise. By repeating DBSCAN in the same document set for decreasing MinSim values we produce a pseudo-hierarchy of clusters. On the other hand, COBWEB puts every document in the same cluster in the first level and in one of its children clusters in the next level. It merges or splits clusters aiming to increase the quality of the produced clusters. It produces a hierarchy of clusters, in which each level contains all the documents in the set but the number of clusters increases from the top to the bottom levels. The two clustering schemes are depicted in Figure 4.



Sim. threshold (DBSCAN only)	DBSCAN clusters			COBWEB clusters			
0.5	{a,b,c,d,e,f,g,h,i,j}			{a,b,c,d,e,f,g,h,i,j}			
0.7	{a,b,e,h,i,j}		{c,d,f}	{a,j,e,b,i,h}		{f,g,c,d,}	
0.9	{a,b}	{h,i,j}	{c,d,f}	{a,b,e}	{j,i,h}	{f,d,c}	{g}

Figure 4. Clustering hierarchies produced by THESUS

5.5. Query module

The clusters resulting from the clustering module can be exploited to answer user queries, in a more meaningful way. Let q={k_i} a query, where k_i are keywords defining the user's interest. The set {k_i} is mapped to a set of ontology terms {t_j}, thus converting q to a query with ontology terms q'={t_j}. Subsequently, the cluster(s) relevant to q' are identified by computing the similarity between q' and the clusters. Clusters whose similarity to q' surpass a threshold are considered relevant and the query. The query-processing module focuses within the relevant cluster(s) and computes the similarity of q' to the description of each document. Results are grouped by cluster and ranked within each cluster based on their similarity to q'.

In the experimental results that follow, the evaluation is focused on the "information extraction" and on the "clustering" modules. More specifically, hyperlink information extraction is evaluated for single pages and sets of pages employing the incoming and outgoing hyperlink semantics. Experiments on clustering, test the ability of the similarity measure and the clustering algorithm that we plug it into, to group together pages based on incoming links semantics.

6. TESTING AND EXPERIMENTS

In this section we present an extensive experimental evaluation of THESUS. The section begins with some performance indications on the system. Two sets of experiments follow: one investigating the value of incoming links semantics in providing characterizations for a page or a set of pages and one testing the capabilities of the system in clustering sets of URLs by comparing to the results of clustering done by humans.

In all the tests concerning web page characterization, we used unbiased 'blind' testers to evaluate the results of THESUS, and those of other systems involved. The testers have no knowledge of the system that provided the characterization, thus the results are not skewed in favor of our system. For the clustering test we used a predefined (and pre-classified by humans) collection of URLs.

A complementary set of experiments that test the ability of the system to characterize and consequently cluster large document sets can be found in [17]. In those experiments different information sources (content, inlinks, user provided summary) and different similarity measures are used in order to evaluate the performance of the THESUS clustering module.

6.1. System performance

The first three modules of the THESUS architecture (information acquisition, extraction and enhancement) take as input an ontology and optionally an initial set of documents and produce an extended set of documents and links, enriched with keywords, semantics and the respective weights. This procedure creates the information pool for the THESUS clustering and querying modules.

The duration of the crawling procedure is mainly affected by the network bandwidth and the architecture of the crawling system itself as well as by the number of documents collected. In a prototype implementation, we used an ontology on musical instruments (containing 52 terms, http://www.db-net.aueb.gr/thesus/onto/instrum.rdf), we started from 2 directory pages on music (those of DMOZ and Google directory) and crawled for 12 levels. The crawling was performed using a single computer (a Pentium III PC with 256MBytes RAM and an IDE hard drive), in a multithreaded approach, attempting to take advantage of the available network bandwidth (10MBps). The information acquisition took place simultaneously to the crawling phase. In less than a day, we collected hyperlink information for approximately 190000 documents. However, only 4000 of these documents have at least 2 incoming links, which proves the weak connectivity of the collected document set. We may increase incoming link information by using a web service that provides backward links for the documents in our collection (e.g. Google's backward link service,[14]).

The performance of the information enhancement module is affected by the size of the keyword set of each document and by the size of the ontology. As the number of keywords in the set increases the number of combinations of senses that should be considered for each document increases. An increase in the number of terms in the ontology, increases the number of pairs (keyword, term) that should be computed. The processing time for the whole collection was about 3 hours.

The clustering phase is carried out offline, but is repeated many times with different parameters, until the optimal clustering schema is achieved. Clustering quality is validated using several internal measures [18]. The clustering time depends on the number of documents and on the algorithm used and can be accelerated by caching information that is calculated repeatedly. In the current implementation we pre-calculate the similarity-matrix between all pairs of documents in the set and store it in main memory. For the running example, if the similarity is stored as a byte number approximately 33.6 Gigabytes ($190000^2/2$) of memory are needed to store the matrix. We decide to cluster only the 4000 documents having at least 2 incoming links. The similarity matrix was computed in 38 seconds and the set was clustered, into 36 clusters in 0.8 seconds, using DBSCAN.

6.2. Keyword extraction from incoming links - evaluation

6.2.1. Page characterization

In order to demonstrate the capabilities of the information extraction module in characterizing the contents of a web document, we selected 50 URLs and obtained their incoming links characterization using at most 100 incoming links ([14]) for each URL. We aggregated keyword appearances by source URL (`weightedSourceKeywords`- how many source URLs use each keyword in the description of the target URL) and kept the top 10 keywords for each target URL. This description, and the descriptions of Altavista and Google for the same pages were presented to a group of testers, which rated their quality (1 - very bad, 5 - very good). In more than 50% of the cases THESUS' descriptions were considered the best of the three. The average rating for THESUS results was 3.7 out of 5, outscoring the other two systems (Altavista – 1.9, Google – 3.4). Some of the URLs used in the test, where listed by Google and Altavista's directory and were described *by human editors*, whereas THESUS' descriptions were *automatically created* and were not based on the contents of the page.

6.2.2. Group characterization

Demonstration of the group semantics operators

In order to demonstrate the use and value of the operators that assign semantics to group of pages, we performed an experiment on the homepages of certain London museums, listed in Google: http://directory.google.com/Top/Regional/Europe/United_Kingdom/England/London/Arts_and_Entertainment/Museums

If we call the set of the homepages' URLs {U}, the set of pages that point to them {I} and the set of pages pointed by them {O} then we get the following results:

- weightedGroupKeywords({I},{U}) = museum 681, london 264, home 185, freud 102, belfast 100, hms 99, national 96, maritime 92, link 87, soane 84, holmes 81, victoria 80, albert 80, …
- weightedTargetKeywords({I},{U}) = london 17, museum 14, website 13, site 12, visit 11, home 11, web 10, world 10, information 10, history 9, library 9, page 9, art 8, british 8, national 8, …
- weightedGroupKeywords({U},{O}) = museum 56, link 40, war 24, london 17, information 16, collections 16, shop 15, education 15, imperial 14, news 13, exhibitions 13, soane 12, …
- weightedSourceKeywords({U},{O}) = museum 11, exhibition 9, collections 9, shop 9, exhibitions 9, online 8, home 7, news 7, events 7, visit 7, information 7, …

The first operator counts the number of keyword occurrences in the links of the first 100 pages that point to {U}. The second operator counts the number of pages that point to {U} using a certain keyword. In order to find the incoming links of a page, we used Google's backward links service. The third operator counts the number of times a keyword appears in all the outgoings links of pages in {U}. The last operator counts the number of pages in {U} pointing to a page using each keyword.

Comparing the two former sets of keywords we see that keywords with a high value of TIMES in the first set do not appear in the second set. These are keywords that appear many times but only in links to the same page (e.g. the names of each museum appears in most of the links to its homepage). Although, *weightedGroupKeywords* assigns high values to these keywords, *weightedTargetKeywords* counts them only once, since they characterize only one page of the target set. The contents of the second set, characterize the group of URLs {U} as a whole. The two latter sets show the Museums' interest as those are expressed in the outgoing links of their homepages. The words *exhibition, collection, online* and *shop* represent things that are usually found in the website of a Museum. In order to perform more thorough semantic operations, we need to construct an ontology on *culture* for instance.

Efficient characterization of groups of pages

In this experiment, we extract the keywords characterizing sets of pages listed under the same topic in Google's directory and measure their relevance to the topic. *WeightedGroupKeywords* and *weightedTargetKeywords* operators are employed for the characterization. The experiment is performed on 30 Google topics. For different values of N (number of keywords extracted), we calculate the percentage of keywords provided by THESUS that are on-topic; that are contained in the path description provided by Google. The results are depicted in Figure 5.

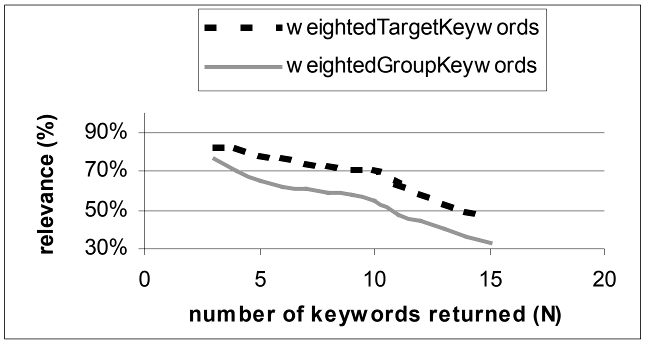


Figure 5. Relevance percentage of group characterization

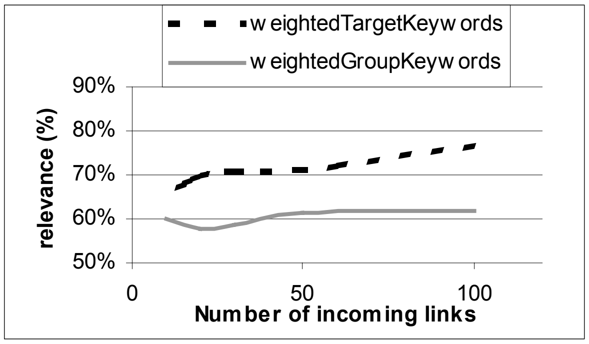


Figure 6. How relevance is affected by the number of incoming links

The examples justify the intuition, that THESUS operators allow the characterization of a set of pages based on incoming links to a very satisfactory degree compared to characterization assigned by humans (which is about 80% - i.e. 8 out of 10 keywords provided by THESUS are included on the human characterizations). Moreover, aggregation of the results per target page (operator *weightedTargetKeywords*) yields better results than aggregation per keyword (*weightedGroupKeywords*). The above results indicate that an increase on the number of keywords used for the characterization of a page decreases the total relevance and suggests keeping at most 10 keywords to characterize the set.

It is noteworthy that, in the results presented above, a maximum of 100 incoming links was used to characterize each page in a set. Visiting 100 documents in order to characterize one is not efficient, even when the documents have been pre-fetched or preprocessed. In another test, we characterize the same sets of pages, using fewer incoming links. We take into account the top 7 keywords (having 75% relevance according to results of Figure 5). Figure 6 illustrates how the number of incoming links affects the relevance of the extracted characterization to the topic. The figure shows that a reduced amount of incoming links (i.e. 20 links) gives comparable results to the ones achieved by taking into account the 100 incoming links (less than 10% loss in accuracy in both operators).

6.3. Testing THESUS ability to discover thematic subsets

In order to test THESUS ability in clustering a set of pages, we cluster with THESUS, a set of URLs that have been manually clustered. Then we measure the quality of the clustering scheme using an external clustering quality measure, ***entropy*** [30] as defined in [31].

The core set of pages contains the URLs suggested by DMOZ directory [27] under a few categories in the Music→Styles path. The categories are: *hip-hop, experimental, world, blues, Indian music, pop, dance, filk, electronic, electronica, country, dance, polka, early music, new age, lounge, rhythm and blues, folk, classical, gamelan, opera, bluegrass, rock, easy listening* and contain 481 URLs in total. We characterized documents by extracting keywords from incoming links and enhancing information with semantics using a manually created ontology on music (with 177 different terms/concepts) and WordNet.

We measure the effectiveness of the two clustering algorithms (DBSCAN and COBWEB). The minimum value of ***entropy***, which is considered to be the best, is 0 and can be achieved when each cluster contains documents of one category only. The worst (maximum) value of entropy is log(N), where N is the number of different categories, when each cluster contains one document from each different category. The maximum entropy is compared to the entropy of each clustering scheme (with DBSCAN and COBWEB) for different values of N. The goal is to decrease entropy by putting documents of the same category to the same cluster, thus increasing the homogeneity of clusters.

In the following tables we evaluate the resulting clustering schemes. The table columns present the total number of documents to be clustered in each case, the number of different categories they fall into, the value of the minimum similarity parameter (for DBSCAN only), the number of documents assigned in a cluster (DBSCAN only), the value of entropy for the resulting clustering scheme, the maximum entropy for the set of documents and the decrease ratio we achieved using our system.

Documents	Categories	MinSim	Documents Clustered	Clusters	Entropy	Maximum entropy	Entropy decrease
400	23	0.55	115	8	0.96	1.36	29%
400	11	0.8	24	3	0.65	1.04	38%
191	22	0.55	74	8	0.76	1.34	43%
191	7	0.8	16	2	0.47	0.85	44%
457	27	0.65	132	6	1.13	1.43	21%
457	5	0.8	10	2	0.47	0.70	33%
182	23	0.6	71	7	0.79	1.36	42%
						Average decrease	36%

Table 1. Comparison of the DBSCAN results for various input parameters (minDocs=3) using ontology terms

Documents	Categories	MinSim	Documents Clustered	Clusters	Entropy	Maximum entropy	Entropy decrease
396	8	0.95	18	7	0.16	0.90	82%
396	13	0.85	31	4	0.67	1.11	40%
295	12	0.95	17	4	0.60	1.08	45%
295	20	0.8	48	6	0.90	1.30	31%
						Average decrease	49%

Table 2. Comparison of the DBSCAN results for various input parameters (minDocs=1) using keywords

Documents	Categories	Clusters	Entropy	Maximum entropy	Entropy decrease
25	2	7	0.11	0.30	63%
37	3	6	0.27	0.48	43%
55	4	12	0.22	0.60	64%
78	5	7	0.45	0.70	35%
100	10	33	0.58	1.00	42%
160	24	56	0.57	1.38	59%
244	26	85	0.59	1.41	58%
				Average decrease	52%

Table 3. COBWEB results (using concepts) for an increasing number of documents

6.3.1. Evaluation of the results

We clustered documents using DBSCAN and semantics first (Table 1) and then DBSAN and extracted keywords. Clustering using the semantics is significantly faster than clustering using keywords, since the similarity between ontology concepts is pre-calculated once for all the 177x177 combinations (using Wu and Palmer's measure on the ontology), whereas the similarity between keywords must be calculated for every new pair of keywords. The latter computation significantly slows down the process since it involves accessing WordNet. In both cases, we notice a significant decrease in the entropy of the system. Decrease is higher when using keyword characterization (49% of the maximum entropy, Table 2) rather than using the respective semantics (36%, Table 1).

In COBWEB documents are clustered based on their semantics. The entropy is again lower than the maximum entropy (a decrease of 52%). As can be deduced from the results in Table 3, after a certain number of documents (approximately 100) have been clustered, entropy remains constant. This means that the clustering algorithm needs many documents as input in order to perform well.

6.4. Conclusion

When comparing our results on various datasets (ranging from hundreds to thousands of pages) to those provided by web search engines or web directories, we come to the following conclusions:

1- Querying links semantics yields correct information on the semantics of pages, at least comparable to those established with full text querying, as proved in sections 6.2 and 6.3.

2- Defining a measure of similarity between web pages, which is based on semantics conveyed by the links between pages, lets us discover clusters of pages that have similar semantics and have similar connectivity features.

7. CONCLUSIONS

7.1.1. Summary – Contributions

Searching in the WWW is a task of very high importance, in social and financial terms, since hundreds of millions of users worldwide, with diverse profiles, are searching for pages relevant to their requests. Currently this task is mainly carried out by submitting keyword queries to search engines. The search criteria are based on the pages' contents ignoring the additional semantics emanating from links. The results contain pages that exactly match the majority of terms in the query. Information retrieval algorithms rank the results based on the distance of the query to the document contents. However, document authors may add misleading contents to their documents aiming to manipulate ranking algorithms and affect search engines results. With hyperlink information, the introduction of such bias becomes harder.

In his paper we capitalize on this observation. We present THESUS model, language and prototype system, which can be applied for selecting, processing and querying thematic subsets of the WWW. The salient features of THESUS language are a. it extracts and exploits semantics from links b. it enables querying and organization of pages based on aggregate connectivity features and link semantics. We also present the implemented THESUS system that collects URLs based on a set of keywords, extracts keywords from the collection's incoming and outgoing links (by processing link's neighboring text in the source URL), maps extracted keyword to semantics and populates a relational database with all this information. A clustering module is able to detect semantically cohesive groups of the initial document set that have similar semantics, assigned by incoming links.

7.1.2. Further Work

Further work will be devoted to the following:

- Proprietary web crawling based on semantics. The thematic crawler developed so far follows links to pages only when the extracted link information exactly matches the set of keywords of interest. It would be useful to be able to enhance the initial keyword set with semantics and follow links based on semantics and not on keywords. Current work on the web crawler also aims on providing an initial estimation on pages' importance using semantics and connectivity.
- Composite ranking of results: The ranking of the results should take into account two different issues: a) The importance of a page, as it is defined by a Page Rank [5]. b) The number of matching keywords between the query and the page's description emanating from incoming links.
- Link position in the page: The location of the source is interesting information. For instance many similar links are often located in a <LI></LI> list.
- Test other clustering algorithms. The tests performed with DBSCAN gave interesting results. However, a lot of pages were perceived as noise and consequently not clustered. Additionally, since DBSCAN is not an incremental algorithm, the clustering process should be repeated every time the initial document set changes. The experiments conducted with the COBWEB algorithm

[12] performed better compared to the manual classification. The ultimate goal is to be able to discover cohesive groups of pages that are semantically close and form semantic clusters; in other words to discover THESUs rather than constructing them.

- Design of a query module, giving the user the ability to perform advanced queries for thematic hubs, authorities, co-citations and couplings. This is for the moment feasible only at keyword level. However, we are working on employing link semantics in the search for strongly interconnected pages with similar semantics. This is the first step on integrating semantic information and link structure in the clustering algorithm.

REFERENCES

[1] R. Al-Halami, R. Berwick, In: Fellbaum C, Miller G. (eds), "*WordNet, an electronic lexical database",* MIT Press-Bradford Books, Cambridge, MA, ISBN 0-262-06197-X

[2] R. Armstrong, D. Freitag, T. Joachims, T, Mitchell, "*Web Watcher: A learning apprentice for the World Wide Web*". In Proceedings of the AAAI Spring Symposium on Information Gathering from Heterogeneous, Distributed Environments, pages 6-13, Stanford, CA, March 1995

[3] G. Arocena, A. Mendelzon, G. Mihaila. "*Applications of a Web Query language*" Proceedings of the 6th International WWW Conference, Santa Clara, California, April 1997.

[4] A. Broder, R. Kumar, F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A. Tomkins, and J. Wiener. "*Graph structure in the Web*". 9th International World Wide Web Conference, Amsterdam, The Netherlands, May 2000.

[5] S. Brin and L. Page. "*The anatomy of a large-scale hypertextual Web search engine*". In 7th International World Wide Web Conference, Brisbane, Australia, April 1998.

[6] S. Chakrabarti, M. Berg and B. Dom. *"Focused Crawling: A New Approach to Topic-Specific Web Resource Discovery",* In Proceedings of the 8th International WWW Conference, Toronto, Canada, May 1999

[7] S. Chakrabarti, B. Dom, D. Gibson, J. Kleinberg, P. Raghavan, S. Rajagopalan, "*Automatic Resource list Compilation by Analyzing Hyperlink Structure and Associated Text*", 7th International WWW Conference 1998

[8] S. Chakrabati, B. Dom, D. Gibson, J. Kleinberg, S. Kumar, P. Raghavan, S. Rajagopalan, A. Tomkins. "*Mining the link structure of the world wide web*". IEEE Computer, 32(8):60-67, 1999.

[9] The DARPA Agent Markup Language Ontology Library http://www.daml.org/ontologies/

[10] M. Ester, H.P. Kriegel, J. Sander, X. Xu, *"A density based algorithm for discovering clusters in large spatial databases with noise*", 2nd International conference on Knowledge Discovery and Data Mining ACM-SIGKDD, 1996

[11] T. Eiter, H. Mannila, *"Distance measures for point sets and their computation"*, in Acta Informatica Journal, 34, (1997)

[12] D. Fisher, "*Knowledge acquisition via incremental conceptual clustering*". Machine learning,2,139-172.

[13] A. Gionis, D. Gunopulos, N. Koudas, *"Efficient and Tunable Similar Set Retrieval".* In Proceedings of the ACM SIGMOD International Conference on Management of Data, Santa Barbara, California, May 21-24, 2001, pp~247-258

[14] The Google.com search engine: http://www.google.com/

[15] N. Guarino, *"Formal Ontology and Information Systems"* In: Procedings of the 1st International Conference on Formal Ontologies in Information Systems FOIS'98, Trento, Italy, June 1998. pp~3-15,IOS Press, Amsterdam

[16] M. Halkidi, Y. Batistakis, M. Vazirgiannis*, "On Clustering Validation Techniques"*, Journal of Intelligent Information Systems (JIIS), Vol. 17, No. 2- 3, pp. 107-145. 2001

[17] M. Halkidi, B. Nguyen, I. Varlamis, M. Vazirgiannis. "*THESUS: Organizing web document collections based on semantics*". To appear in Very Large DataBases Journal, special edition on Semantic Web.

[18] M. Halkidi, M. Vazirgiannis, "*A Data Set Oriented Approach for Clustering Algorithm Selection*". In Proceedings of: Principles of Data Mining and Knowledge Discovery, 5th European Conference, PKDD 2001, Freiburg, Germany, September 3-5, 2001, pp. 165-179

[19] M. Henzinger, "*Hyperlink Analysis for the Web*", IEEE Internet Computing, Vol 5, No.1, 2001 pp. 45-50.

[20] T. Haveliwala, A. Gionis, P. Indyk, "*Scalable techniques for clustering the Web*", In Proc. of the WebDB Workshop, Dallas, TX, May 2000.

[21] M.M. Kessler, "*Bibliographic Coupling between Scientific Papers*". American Documentation, Vol.14, No.1, 1963

[22] J.Kleinberg "*Authoritative sources in a hyperlinked environment*". Journal of the ACM, Volume 46, Number 5, September 1999, pp~604-632

[23] The Kartoo System: http://www.kartoo.fr

[24] I. Niiniluoto, *"Truthlikeness"*, D. Reidel Pub. Comp., Dordrecht, Holland 1987

[25] The Northern line search engine: http://www.northernlight.com

[26] B. Nguyen, I. Varlamis, M. Halkidi, M. Vazirgianis, "*Construction de Classes de Documents Web"* In Premieres Journees Francophones de la Toile, June-July 2003, Tours, France.

[27] ODP - Open Directory Project, http://dmoz.org/

[28] T. Phelps, R. Wilensky, "*Robust Hyperlinks Cost Just Five Words Each*". UC Berkeley Computer Science Technical Report UCB//CSD-00-1091. Berkeley, CA. 2000

[29] H. Small, "*Co-citation in the scientific literature: A new measure of the relationship between two documents*", J. American Soc. Info. Sci., 24 (1973), pp 265-269.

[30] C. E. Shannon, "*A mathematical theory of communication*", Bell System Technical Journal, vol. 27, pp. 379-423 and 623-656, July and October, 1948.

[31] M. Steinbach, G. Karypis, and V. Kumar. "*A comparison of document clustering techniques*". In KDD Workshop on Text Mining, 2000.

[32] Vivisimo search engine: http://www.vivisimo.com/

[32] I. Varlamis, M. Vazirgiannis, "*Web document searching using enhanced hyperlink semantics based on XML*", In: In Proceeding of International Database Engineering & Applications Symposium, IDEAS '01,Grenoble, France, July 16-18, 2001, pp. 34-43

[33] WordNet web site: http://www.cogsci.princeton.edu/~wn/

[34] Z. Wu and M. Palmer *"Verb Semantics and Lexical Selection"*, In: Proceedings of the 32nd annual meetings of the associations for computational linguistics, Las Cruces, New Mexico, June 1994, pp~133-138

[35] O. Zamir, O. Etzioni, "*Web document clustering: a feasibility demonstration*", In: Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Melbourne, Australia,August 24-28 1998, pp.46-54.

APPENDICES

APPENDIX A. A summary of THESUS language definitions

	Datatypes	
doc	(URL, {keyword},{ontology_term})	THESUS' documents represented by their URL and a set of keywords/ontology terms.
link	(URLS, URLT, {keyword}, {ontology_term})	THESUS' links represented by source, target URL and a set of extracted keywords/ontology terms.
w_docs	{(URL, text)}	The web documents represented by their URL and text
w_links	{(URLS, URLT)}	The web links represented by the source and target URLs
	Auxiliary functions	
Getpos(text, URLT)	returns {pos}	Finds the positions of a URL inside a page
process(w_docs.TEXT, pos, M)	» {keyword}	Extract the keywords from an area of M characters around the positions of a URL
Getkeys({(keyword, times)})	» {keyword}	Returns only the keywords that appear in a set of pairs (keyword,times)
getTimes({(keyword, times)},keyword)	» {keyword}	Returns the number of times associated to a keyword in a set of pairs (keyword,times)
update({(keyword, times)}, keyword, N)	» null	Increases the number of times associated to a keyword in a set of pairs (keyword,times) by N
getSemantics ({keyword})	» (ontology_term}	Returns the corresponding ontology terms' set that can be deduced from the keywords set
	Operators	
Crawl	**fetch(URL)**	**Text :** the contents of URL
	groupMatch({URLexpr})	**{URL} :** URLs that match the URLexpr string
	crawl({SURL},N)	**{URL} :** the URLs that can are reached by starting from a URL in {SURL} and following at most N links
	thematicCrawl({SURL},{keyword},N)	**{URL} :** the URLs that can are reached by starting from a URL in {SURL} and following at most N links that contain at least on word from {keyword}
Hyperlink semantics extraction	**linkKeywords (URLS, URLT)**	**{keyword} :** the keywords from all links from URLS to URLT
	groupKeywords ({URLS}, {URLT})	**{keyword} :** the keywords from all links from any URL in {URLS} to any URL in {URLT}
	weightedGroupKeywords ({URLS}, {URLT})	**{(keyword,times)} :** the keywords and occurrences from all links {URLS}→{URLT}
	weightedTargetKeywords ({URLS}, {URLT})	**{(keyword,times)} :** the keywords and the number of pages in {URLT} characterized by them
	weightedSourceKeywords ({URLS}, {URLT})	**{(keyword,times)} :** the keywords and the number of pages in {URLS} that use them to characterize pages in {URLT}

link analysis	**TAuthorities({URL}, N, {keyword})**	**{URL} :** the URLs that are pointed by more than 'N' pages with keywords in {keyword}
	THubs({URL}, N, {keyword})	**{URL} :** the URLs that point to more than 'N' pages with keywords in {keyword}
	TCocitations({URL}, N, {keyword})	**{URL} :** the URLs that are pointed by more than 'N' pages in {URL} with keywords in {keyword}
	Tcouplings({URL}, N, {keyword})	**{URL} :** the URLs that point to more than 'N' pages in {URL} with keywords in {keyword}

APPENDIX B. The algorithms for the main operators

```
crawl({URL}, N):-
  RES = {}
  if N<=-1
    ∀ d in  {URL}
      if w_links.URLT = d
        RES = RES ∪ w_link.URLS
      return crawl(RES, N+1)
      if N=0
      return {URL}
      if N>=1
        ∀ d in {URL}
          if w_links.URLS = d
            RES = RES ∪ w_link.URLT
      return crawl(RES, N-1)
```

```
linkKeywords (URLS, URLT):-
  let Stext =  select w_docs.text where w_docs.URL = URLS
  KEYS = {}
  ∀ d in getpos(Stext, URLT)
    KEYS = KEYS ∪ process(w_docs.TEXT, d, 100)
  return KEYS
```

```
groupKeywords ({URLS}, {URLT}) :-
KEYS = {}
  ∀ d in {URLS}
    ∀ e in {URLT}
      KEYS = KEYS ∪ linkKeywords(d,e)
  return KEYS
```

```
thematicCrawl({URL}, {keyword}, N):-
  RES = {}
  if N<=-1
    ∀ d in  {URL}
      if w_links.URLT = d AND linkKeywords(w_link.URLS,d)∩{keyword}≠∅
        RES = RES ∪ w_link.URLS
  return thematicCrawl(RES, {keyword}, N+1)
  if N=0
  return {URL}
  if N>=1
    ∀ d in {URL}
      if w_links.URLS = d AND linkKeywords(d,w_link.URLT)∩{keyword}≠∅
        RES = RES ∪ w_link.URLT
  return thematicCrawl(RES, {keyword}, N-1)
```

weightedGroupKeywords ({URLS}, {URLT}) :-

```
  WKEYS = {}
  ∀ d in {URLS}
    ∀ e in {URLT}
      ∀ K in linkKeywords(d,e)
        if getkeys(WKEYS) ∩ {K} ≠∅
          update(WKEYS,K,1)
        else WKEYS=WKEYS ∪ (K,1)
  return WKEYS
```

weightedTargetKeywords ({URLS}, {URLT}) :-

```
  WKEYS = {}
  ∀ e in {URLT}
    ∀ K in groupKeywords({URLS},e)
      if getkeys(WKEYS) ∩ {K} ≠∅
        update(WKEYS,K,1)
      else WKEYS = WKEYS ∪ (K,1)
  return WKEYS
```

weightedSourceKeywords ({URLS}, {URLT}) :-

```
  WKEYS = {}
  ∀ d in {URLS}
    ∀ K in groupKeywords(d,{URLT})
      if getkeys(WKEYS) ∩ {K} ≠∅
        update(WKEYS,K,1)
      else WKEYS=WKEYS ∪ (K,1)
  return WKEYS
```