Pergamon

Information Processing & Management, Vol. 32, No. 4, pp. 459–476, 1996

0306-4573/96 $15.00+0.00

0306-4573(95)00075-5

DESIGN AND IMPLEMENTATION OF A TOOL FOR THE AUTOMATIC CONSTRUCTION OF HYPERTEXTS FOR INFORMATION RETRIEVAL*

(Received 2 October 1995; accepted in final form 6 November 1995)

MARISTELLA AGOSTI, FABIO CRESTANI and MASSIMO MELUCCI
Dipartimento di Elettronica ed Informatica, Università di Padova, Italy

Abstract—The paper describes the design and implementation of TACHIR, a tool for the automatic construction of hypertexts for Information Retrieval. Through the use of an authoring methodology employing a set of well known Information Retrieval techniques, TACHIR automatically builds up a hypertext from a document collection. The structure of the hypertext reflects a three level conceptual model that has proved to be quite effective for Information Retrieval. Using this model it is possible to navigate among documents, index terms, and concepts using automatically determined links. The hypertext is implemented using the HTML hypertext mark up language, the mark up language of the World Wide Web project. It can be distributed on different sites and different machines over the Internet, and it can be navigated using any of the interfaces developed in the framework World Wide Web project, for example NETSCAPE. Copyright © 1996 Elsevier Science Ltd

1. INTRODUCTION

A number of different and important aspects need to be addressed in order to combine both the functionalities and capabilities of hypertext systems and IR systems. This combination should give final user the possibility of navigating, browsing and searching a large collection of textual document using a network of links implementing the semantic connections between the documents themselves. Many of these aspects have been pointed out by Agosti (1993) where it has been stressed that *automatic authoring* is a particularly crucial process. The necessity of automating the authoring of the collection resides in the fact that often the collection is large, and it is not feasible nor reliable to build up manually the network of links among documents; links that together with documents create and make available to the final users the so-called *IR hypertext.*

IR hypertext is a document base that allows access to documents mainly by navigation and browsing. An IR hypertext is composed of nodes, stores of information, and links; connections between nodes. The user navigates from node to node using links. The series of navigational choices which are made, leads hopefully, through the document base to the desired information.

This paper addresses the topic of the design and construction of IR hypertext systems presenting related work both about hypertext and IR in Section 2 and about automatic construction of hypertexts in Section 3. In Section 4 the conceptual architecture of IR hypertext is presented. The design of IR hypertexts of Section 5 is based on such architecture.

An IR hypertext can be implemented using the above presented methodology by choosing and using an appropriate set of implementing techniques. The adopted techniques are presented in Section 6. Section 7 reports on the design and implementation of TACHIR: a Tool for the

* A version of this paper was presented at RIAO94 conference in New York. (Agosti *et al.*, 1994). This paper supersedes the conference paper in critically relating the work of the authors to previous work in the area of automatic construction of hypertexts for information retrieval and in the completeness of the presentation of both methodological and implementation characteristics of the developed tool. The TACHIR prototype can be seen at work at the following URL: http://www.dei/unipd.it/english/irpd/activities/tachir/intro.html

Automatic Construction of Hypertexts for IR. The evaluation of IR hypertext systems still remains an open issue, however Section 8 gives some initial thoughts. Section 9 concludes the paper giving directions for future work in the area.

2. HYPERTEXTS FOR INFORMATION RETRIEVAL

Though early work on navigation and browsing of document bases in IR dates back to the seventies (Oddy, 1975), there are only very few IR systems that allow access to documents or to the indexing structure of documents by means of browsing (Frisse, 1988; Thompson, 1989). Fairly recently, there has been a new impulse in this research direction with new approaches and new prototypes being proposed.

A survey of the use of hypertext methods and capabilities in IR is outside the scope of this paper. However, we would like to bring the reader's attention to a few works that we consider seminal in the direction of enabling the user to access IR objects by navigation and browsing.

In Agosti *et al.* (1991), an architecture and a new functional model have been introduced to overcome major limitations of hypertext systems in relation to IR operations. The model has been named EXPLICIT, because its main focus is on the "explicit" presentation to the user of the network of index terms and concepts that are used for the representation of the document collection. EXPLICIT incorporates some important IR functions and assists the final user by means of a new type of associative IR; two most important features of such a model are a semantic association (Agosti & Marchetti, 1992) and an associative reading function.

In Dunlop (1991) and later in Dunlop and van Rijsbergen (1993), a combined model of IR, which encompasses the principles and benefits of both free text retrieval and hypermedia, is presented. This model gives the users access to large document bases with limited structure which can be browsed whatever its topology is. The model approximates the content of documents that cannot be directly retrieved by content (e.g. images), in fact it makes use of contextual information extracted with this purpose from a hypermedia network. Moreover, this model has been only partially evaluated because of the lack of methods for IR hypertext systems evaluation. However, the experiments and observation of a prototype system have shown that use of context information from hypermedia network to retrieve non-textual nodes by query is effective.

In Salton *et al.* (1994), a range of different IR text manipulation methods and strategies are addressed. In particular, the use of the global–local text comparison methods for text structuring seems a very interesting automatic authoring technique. This is achieved by using a hierarchical text decomposition to successively refine the query and as a consequence the coverage of the retrieved items. However, such a structuring is only performed at run time to produce a better ranking of the retrieved items, and it disappears as soon as the query has been processed. There is no static structure produced by this process that a user can navigate to refine and clarify his information needs.

Although it is often thought that only hypertext can provide browsing, it must be noted that the ability to move between related documents can also be provided by IR systems supporting relevance feedback van Rijsbergen (1979). Unlike hypertext, which generally has fixed links, relevance feedback allows the user to dynamically create links at run time by searching for documents similar to some others marked as relevant. However, browsing by means of relevance feedback is a very complex process and most of the existing IR systems supporting relevance feedback do not have a good user interface for browsing, as pointed out by Aalbersberg (1992). Only by keeping the user interface and the interaction with the user at the simplest level, is it possible to employ effectively relevance feedback techniques. In addition, a user could find it very useful also to be able to browse through the indexing items (index terms, concepts, thesaurus, etc.) and this cannot be provided by systems using relevance feedback.

The approach presented in this paper aims at enabling the users to navigate and browse the document base in a natural way, navigating not only through documents but also through the indexing items. The structure upon which the user navigates is built automatically from a collection of documents.

In Agosti and Crestani (1993), we proposed a design methodology for automatically constructing an IR hypertext by putting together various well established IR techniques of linking IR objects. This methodology, briefly described in Section 5, is based on the conceptual model presented in Section 4.

3. RELATED WORK ON AUTOMATIC CONSTRUCTION OF HYPERTEXTS

Automatic authoring has been addressed by researchers since the early days of hypertexts. The increasing availability of online textual document collections, whose size is too large to enable a manual authoring, is the main reason for which fully automatic, or partially automatic authoring techniques are currently being studied and implemented. In this Section, without even attempting to provide a comprehensive survey of all the work done within the field of automatic authoring, we refer to some of the contributions that are most relevant to this paper.

One of the earliest work in the field of the automatic transformation of text into hypertext is reported in Furuta *et al.* (1989). That paper illustrates the methodology and the implementation of a technique for converting a regularly and consistently structured document into a hypertext. Regularly and consistently structured documents are those having a well-identified and fixed structure, for example, bibliographic cards or manual pages. The resulting hypertext is made of nodes corresponding to the document parts connected by means of structural links. The methodology is based on the reasonable assumption that there is a close relationship between the physical components of a document and the hypertext nodes. From an IR point of view, such structure-based hypertextual organization should provide with a better understanding of the semantic content of documents.

The authors claim that their methodology is well suited for medium-grained documents that are regularly and consistently structured, for example, the collection of dissertation abstracts they used for their experiments. Larger or less regular and consistent documents, such as those addressed by the work presented in this paper, would require some manual intervention to catch content-based links, that is to say links non-explicitly inserted in the documents and that are meant to represent semantic aboutness. It is these kind of links that are the most interesting from an IR point of view, since we are mainly interested in a semantic navigation and browsing of the document collection.

The difficulty in the automatic construction of content links has been addressed by Salton and Buckley, (1989) at the beginning of a work further carried out also in other directions, like passage retrieval Salton *et al.*, (1993), theme extraction and text summarization Salton *et al.* (1994). The algorithm proposed by the authors for determining content links is based on the evaluation of similarity between documents and/or parts of documents (sentences). The experiments were carried out by partitioning a textbook into smaller segments. This work leaves open a few questions:

- The resulting hypertext should be tested to see if it is useful for the IR purposes, both from a system and user's point of view.
- Is this technique effective for documents covering heterogeneous subjects? The problem requires further attention but it seems possible to anticipate that the effectiveness of a technique for the automatic construction of hypertexts depends on the extent of the subjects of the documents.
- The goodness of the resulting hypertext is related to the good tuning of some parameters, for example the similarity threshold; the way these parameters change the effectiveness of the hypertext should be evaluated.

Rada (1992) addresses the combination of structural links and content links. The author distinguished first-order and second-order hypertext. The former uses only structural links based on the document markup and determined by the document author. Structural links of this kind include links connecting outline headings, citations, cross-references, and indices. In second-order hypertext, links are not explicitly put in the text by the author, but are detected using some automatic procedures. In that work, second-order links were set up between index terms using

co-occurrence data. The use of first- and second-order links in the same hypertext enables to reflect both the structural schema of the source documents, that is the document author's schema, and a second alternative schema, reflecting the way index terms are distributed across the documents. Alternative outlines are different views of the same documents that users can employ to improve their understanding during browsing, since alternative outlines offer different semantic points of view on the same document. As the author suggests, some more work should be carried out to test which type of hypertext, first- or second-order hypertext, the user appreciates better.

The approach proposed in Smeaton (1995) is based on node–node similarity. This approach is different from other approaches based on IR techniques because it uses the overall hypertext topology as a decision support for link setting. A measure of the hypertext topology is used to assess the degree of hypertext compactness. The lower number of jumps from one node to another the user has to do to access the desired information, the more compact the hypertext. This measure of the hypertext compactness is employed to decide whether a similarity-based link should be added or not. The major contribution of this work is in proposing guidelines to control the automatic construction of the hypertext. What remains to be discussed is the relevance of the hypertext topology to the hypertext effectiveness. As the authors highlighted, some very compact hypertext may result in the user being disoriented because of too many links. Moreover, a compact hypertext is not always desirable; in a hypertext with a large number of links the user can be helped to browse, by providing more information about links, such as for example information about the link type.

The problem of discovering link types has been addressed by Allan (1995). The proposed technique provides a way of setting up links between passages of documents. The novelty of this work is that classical IR techniques are employed to determine the type of relationship incurring in a hypertext whose nodes are topics. Allan also addressed the problem of the number of links. To reduce the number of links and to make easier the visualization of the resulting graph, some link-merging techniques are suggested and described. The techniques proposed by Allan for the automatic link type identification are based on some values calculated on the merged links. For example, to identify a summary link, we can compute the amount of unlinked text that was added to a link endpoint during the link merging. The author points out a few directions that should be followed by further research in automatic authoring based on IR techniques. Among them additional work should be done with regards to heterogeneous documents, or documents that have been written with a poorly regular writing style; most proposed techniques for automatic authoring are based on the assumption that documents are quite well-segmented into passages.

4. A CONCEPTUAL ARCHITECTURE FOR IR HYPERTEXTS

It has been long recognized that the complexity of data modelling in IR is mainly due to the complex nature of the relationships between the different IR objects. These objects are: the documents and the *auxiliary data*. The auxiliary data are all those objects that are used to index the documents and are: index terms, classification structures, thesauri, etc. The complexity of IR modelling lies more in the modelling of these auxiliary data and in the relationships between the auxiliary data and the documents than in the modelling of the documents themselves. In fact, the meaning of an auxiliary data becomes fully defined only by means of the semantic relationships existing between this auxiliary data and other auxiliary data. Because of the importance of fully understanding the semantics of auxiliary data for the effectiveness of an IR system, the need for a IR conceptual modelling tool becomes essential. A modelling tool provides the necessary frame of reference for the understanding of the semantics of the relationships between IR objects. Using a conceptual paradigm as a modelling framework, in Agosti *et al.* (1990) we proposed the *conceptual architecture* depicted in Fig. 1.

The conceptual architecture is structured on three levels.

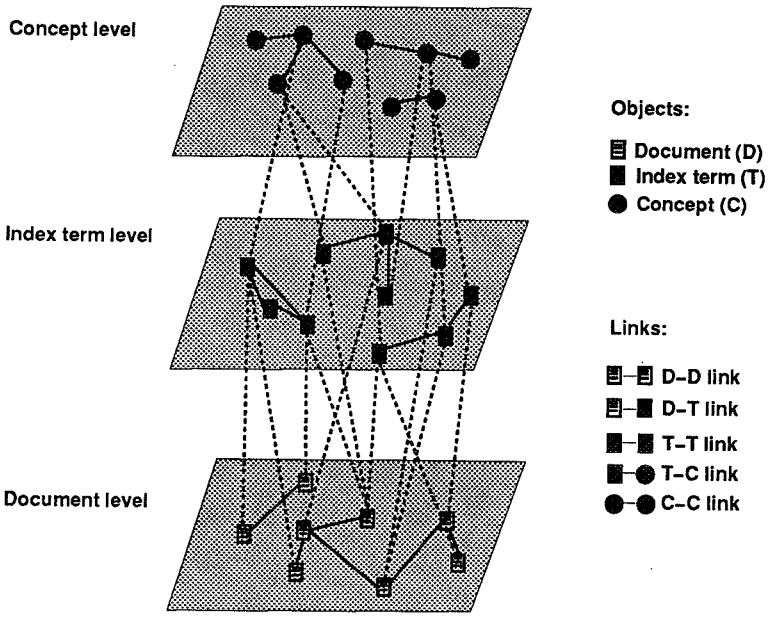

Fig. 1. The conceptual architecture.

Document level

This level contains the elementary objects of interest: the documents (D). Each document has its own identity and status; the identity of the document is independent of the way it is represented or structured. Documents can be related to each other by means of bibliographic citation or by similarity relationships (D–D links).

Index term level

This is the level of the index terms (T). Index terms are linked to the documents they are meant to index (D–T links), and each index term individualizes a set or class of documents on the basis of their semantic content. The classification used is polythetic: a document can be related to many different index terms thereby it can belong to several different classes. Index terms can also be related to each other by similarity relationships (T–T links).

Concept level

Concepts (C) are sets or classes of index terms. A concept is a higher level object than an index term, so a concept can be connected to several index terms (via T–C links) that are meant to represent it. Concepts are linked to each other according to their semantic relationships (C–C links). The structure on this level, consisting of concepts and their relationships, can represent a classification system or thesaurus for the particular application domain of the document collection.

This conceptual model provides a very powerful framework for navigating and browsing among IR objects. However, it would be of little use if it could not be implemented or if it needed to be implemented by hand. We therefore developed a methodology for the automatic construction of an IR hypertext based upon this conceptual architecture. This methodology is briefly reported in the next section. We refer to Agosti and Crestani (1993) for a complete treatment of the methodology.

5. THE DESIGN OF AN IR HYPERTEXT

The starting point of the design process is the usual set of IR raw data: a document collection; with documents available as individual unrelated objects.

The design process is divided into five phases. Each phase is concerned either with the construction of a level of the conceptual model or with the setting of a network of links within or between levels (see Fig. 1). With the exception of the first phase, which must be performed before any others, there is no strict or unique order for the performance of the remaining phases. Some phases can also be performed in parallel for a faster construction of the hypertext.

5.1. The construction of the index terms level

During this phase nodes on the index term level (T nodes) are created by extracting terms from documents. The technique most often used for this task is called *indexing*. The indexing process, that can be performed manually or automatically, is a very complex process that has been studied for long time in IR (see for example Frakes & Baeza-Yates, 1992; Salton, 1968; van Rijsbergen, 1979). It constitutes one central issue of the IR research because it is the technique by which document information content is represented. It is by this process that individual or groups of terms found in documents become index terms, assuming a representational power that places them on a higher level of abstraction than the documents.

5.2. Determining associations between documents and index terms

The process of determining associations between documents and index terms is strictly related to the above described process of assigning index terms to documents. Once an index term is assigned to a document it is also linked to that document. In this way it is possible to determine the D–T links of the conceptual model.

5.3. The construction of the concepts level

There are no well established techniques for determining a set of concepts from a document collection. Most of the time it is necessary either to build manually the set of concepts, a job that is very time consuming and that needs to be performed by experts of the application domain, or use an (or part of an) existing thesaurus, if available. The field of Information Science is very much concerned with the construction of such a set of concepts and very detailed instructions are given to guide application domain experts in this task (see for example Aitchison & Gilchrist, 1987; Srinivasdan, 1992).

5.4. Determining associations between concepts

From an IR point of view there is no operative advantage in having a set of application domain concepts if they are not connected to each other according to their semantics. It is by looking at the relationships a concept has with other concepts that we can understand the "meaning" of the concept in the context of the application domain. When this meaning has been fully understood, it is also possible to understand the "usage" of the index terms connected to the concept. In fact, index terms just represent the way the concept has been addressed in the documents belonging to a particular collection. The way a concept has been addressed by authors of documents in the collection could differ from the way a user of the IR system addresses it. Using a very precise term in addressing a concept increases the precision of the retrieval. However, the user could be interested in considering a concept in a loose way. This can be done using index terms expressing concepts semantically related to the concept which is

central to the user's information need. In this way it is possible to increase the recall of the retrieval.

The utility of having a tool which provides for each concept, a set of semantically related concepts, has long been recognised in IR. A *thesaurus* is a tool which provides for each term in a specific application domain, a set of terms related to it by some well defined semantic relationships. Due to its nature, the structure of associations represented in a thesaurus can be directly mapped into a network structure. This can be achieved by mapping concepts to nodes (the C nodes) and relationships to links (the C–C links).

As we pointed out in the previous section, sometimes a thesaurus on the specific application domain is not available. In this case it becomes necessary to build up manually the network of concepts. To do so, after having identified a set of concepts of the particular application domain, we need to determine their semantic relationships. The fundamental types of semantic relationships commonly represented in a thesaurus are: scope, equivalence, hierarchical and associative relationships. They provide a useful frame of reference on the kind of relationships to be taken into consideration for a manual construction of a network of concepts. Other specific types of relationships must be taken into consideration to model the relationships between concepts in a well defined application domain.

5.5. Determining association between index terms and concepts

The semantic association between index terms and concepts can be built using different formal approaches. Proposed approaches are mainly based on *associative information retrieval*, as it has been presented in the past as a possible alternative to exact-match retrieval. The work on associative retrieval refers back to work carried out in the 1960s (Jones *et al.*, 1968) concerning term associations. Mathematical background was provided by the work on automatic abstracting and indexing (Edmundson & Wyllys, 1961), where a word *frequency* in a text had been linked to the word *significance*. The mathematical formulation evolved in time, incorporating a probabilistic approach. Whilst the probabilistic approach does not require any *a priori* classification of the text to be retrieved, the approach presented in Agosti and Marchetti (1992) is based on an associative algorithm which makes use of preference relations, but which requires a preliminary identification of the index terms in the text. These terms can be presented to the user to visualize the representation of the semantic content of the documents.

5.6. Determining associations between index terms

There are many techniques for identifying relationships between index terms (T–T links). One of these is to use the concept network to relate index terms by means of objects on a higher level of abstraction. It is also possible to use statistical techniques based on term occurrence inside documents to determine their level of similarity and then use a cut-off value to establish links between them. Both these techniques must be carefully tailored to the particular application and the particular document collection, since the distribution of terms tends to vary in different document collections. Examples of automatic construction of similarity matrices for index terms are reported in Crouch (1990), Qiu and Frei (1994), and Salton and McGill (1983).

5.7. Determining associations between documents

For an automatic set up to links between documents (D–D links), it is possible to use statistical techniques very similar to those employed for the construction of links between index terms. Other techniques for setting up a network of related documents make use of bibliographic citations or co-authoring. Bibliographic citations can be used to build up a network implicitly assuming that the documents cited by a document must be somehow related to the citing document.

It must be noted that most operational IR systems use only the D–T links in the retrieval process. These are represented in the inverted file structure which is the most common storage structure in IR. Only very few operational IR systems enable the user to take advantage of relationships like those established by C–C and T–C links, and they are used only as an aid to query formulation. Relationships like those represented by T–T and D–D links are used only in few experimental IR systems (see for example Salton, 1971), however they have never been used for browsing.

6. THE IMPLEMENTATION OF AN IR HYPERTEXT

An IR hypertext can be implemented using the above presented methodology by choosing and using an appropriate set of implementing techniques. In order to chose the right set of techniques many aspects must be taken into consideration, such as the size of the application domain, the size of the document collection, the user preference between accessing document by query or browsing, the desired fan-out of the nodes of the hypertext, and so on.

We performed a set of experiments with different document collections, spanning over a few different application domains, and we devised a set of the implementation techniques that we consider to be quite effective. In particular, we decided to use well established IR techniques, where possible, to avoid developing new techniques that needed to be tested thoroughly. We believe that it is better to use sound and well experimented techniques than create new ones.

For indexing, that is for the extraction of index terms from documents (T nodes) and the lining of these nodes with the documents whose information content they are meant to represent (D–T links), we used the classical IR indexing technique that has been used for a long time by many researchers in the field (Salton, 1968; van Rijsbergen, 1979). According to this technique, indexing is divided into the following steps:

(1) Term extraction,
(2) stop words removal,
(3) conflation
(4) weighting.

In particular, we used van Rijsbergen's list of stop terms (1979), augmented with domain specific stop terms, the Porter's stemming algorithm (Porter, 1980), and the classical $tf \times idf$ weighting scheme (Robertson & Sparck Jones, 1976).

Regarding the concept level, we recognized the difficulty of building up a set of domain specific concepts (C nodes) and their relationships (C–C links).

We therefore decided to make use, whenever possible, of existing thesauri. The difficulty, then lies in translating the thesaurus into a hypertext format. This very much depends on the hypertext formalism used. Very often we had to perform this phase by hand since there is not a standard form for thesauri and therefore it was of no use trying to implement an automatic authoring procedure.

In order to link concepts with index terms (T–C links) we used the approach described in Agosti and Marchetti (1992), called "semantic association". This technique enables the automatic construction of links between nodes on the index term level and nodes on the concept level (T–C links). The cited paper reports a complete description of this complex technique.

In order to find associations between index terms (T–T links) we employed a technique that makes use only of information present in the index terms themselves. This technique does not involve the semantics of index terms but only information provided by statistical analysis of index term occurrence in documents (Salton, 1968). We decided to use this technique instead of more sophisticated ones, like for example the Expected Mutual Information Measure (van Rijsbergen, 1977) or the Latent Semantic Indexing (Deerwester *et al.*, 1990) because it is easier to implement and faster to use. Moreover, given that we intended to use associations between index terms only for navigation and browsing and not for query expansion, we decided to sacrifice some precision to have a faster association technique.

The same considerations guided us in the choice of the technique for the automatic

identification of associations between document (D–D links). Techniques based on the use of term distribution in documents provide fast and effective means of establishing relationships between documents (Salton, 1968). In a few cases, where bibliographic citations are available, it is also possible to use referencing between documents to establish an association between the citing and the cited document. This is based on the heuristic rule assuming that a document cited by another document must be somehow related to it.

7. TACHIR: A TOOL FOR THE AUTOMATIC CONSTRUCTION OF IR HYPERTEXTS

The process of creating a hypertext like the one we described in the previous section would be very tedious if it was to be performed by hand. Given the size of document collections commonly used in real IR applications the number of nodes and links would make it prohibitive to construct the network even partially by hand. Other problems may arise if the documents to be authored are large: in such a case straightforward techniques for segmenting documents or more complex passage retrieval methods should be considered. We therefore decided to develop a tool to help a user to automatically build IR hypertexts. The tool makes use of the methodology described above.

In the following sections we will report on the design and implementation of TACHIR: a Tool for the Automatic Construction of Hypertexts for IR. This tool is now at an advance prototype level and we are about to evaluate its effectiveness with regards to the automatic authoring of a collection of full text articles from a scientific journal.*

7.1. The design of TACHIR

We decided to design TACHIR by means of an *object oriented* (O–O) approach. The O–O approach, if used for designing purposes, could support the direct representation of two essential data abstractions: classification and generalization–specialization, providing a natural framework for IR data design operations (Agosti *et al.*, 1990). The classification–abstraction mechanism groups together similar objects in a class the generalization–specialization mechanism can model the relationships between classes producing a taxonomy of classes. The classes of a taxonomy are related by superclass/subclass relationships. A subclass inherits all the properties of the superclass and other properties can be added, if necessary, in order to specialize the subclass. A class is also an object, therefore it can be related to other objects or classes. Some properties represent intrinsic characteristics of the class itself because the class is indeed an object itself (properties of the class), while other properties are common to all objects that are elements of the class (properties of the objects). By viewing classes as objects, we can apply the classification mechanism to group classes into "meta-classes" and therefore assign specific properties to them (Loomis *et al.*, 1987). One of the most relevant features of the object orientation is class re-usability, which derives from the joining of encapsulation and inheritance features. Class re-usability is achievable through the extension and/or redefinition of the concrete kernel and abstract interface to the class.

The procedures implementing the conceptual architecture reported in Section 5 have been developed using C++ (Stroustrup, 1991) on a Sun Sparc Station. A *library of IR object classes* provides a set of C++ classes whose abstract interfaces encapsulate the concrete kernels and that can be linked to a specific application.

From a conceptual point of view the class library is the TACHIR backbone. IR objects implement the basic IR structures and the abstract interfaces of the library allows the use of IR functionalities. However, it is important to note that the class library includes classes that are independent of our methodology. We design these classes in order to be able to use them also

* This experimentation is going to be performed in the context of the IRIDES Project. The project is funded by the EU in the context of the IDOMENEUS Database, Information Retrieval, and Multimedia Network of Excellence. More information about IRIDES can be found at http://www.dcs.gla.ac.uk/irides/

in a generic IR framework and not only acting as the implementation tool of our methodology. According to this independence requirement, the class library has been developed as independent as possible by any specific application, so that an application designer could find and reuse the basic IR structures and functionalities for other kinds of implementations.

The class library is organized as a tree where each vertex represents a class and each edge corresponds to a specialization relation. The class *ir_object* is the root of the library. Figure 2 depicts, as an example, just a section of the complete class library. Each object of the conceptual architecture is implemented through an IR object that is instance of a class of the libary. A description of the details of the organization of the whole library is beyond the purpose of this paper, because of that, in the next few sections we report on the implementation of the architecture of TACHIR as a whole.

7.2. The implementation of TACHIR

The current implementation of TACHIR assumes that the user accesses the IR hypertext using a World Wide Web browser, like for example, such tools as MOSAIC or NETSCAPE. These distributed hypermedia browsing systems are based on the *World Wide Web* (WWW) technology (Berners *et al.*, 1992) developed in the framework of the WWW project. This project is a wide-area hypermedia IR initiative aiming to give universal access to a large volume of documents over the Internet. The adoption of these tools make the navigation on the IR Hypertext easier, since they provide a "point-and-click" interface with all the WWW built-in functions for browsing through a hypermedia.

The bridge between our conceptual model and WWW is the *hypertext mark-up language* (HTML) used by WWW for marking documents. HTML is a collection of tags used for describing the various components of a document, such as, title, authors, item lists, paragraphs, hypertext links and anchors, and so on. HTML enables the transformation of a flat text into a hypertext. This transformation is achieved by inserting HTML tags into the document to be transformed. In the hypertext area, this process is often known as "authoring".

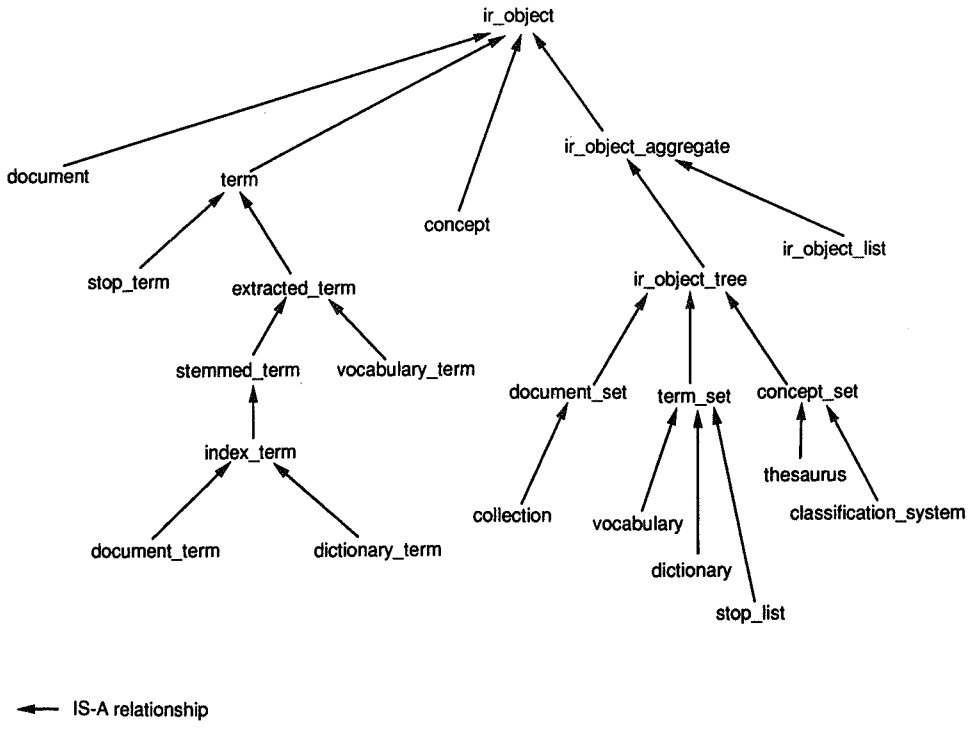

Fig. 2. IR object class library.

Using the class library described in the previous section TACHIR assists the user in the automatic authoring of an IR Hypertext from a collection of documents. In the following, we explain how this is achieved.

Input data. The input data of the automatic authoring process consist of a collection of documents, a stoplist, and a collection of concepts with their relationships.

The documents have to be written using the HTML mark-up language, to do so, we make use of the many converters to HTML that are publicly available, like for example, converters from LaTeX, Microsoft Word, ASCII text, etc. to HTML. Since the input documents are very often only textual, TACHIR has been developed to work with the current version of HTML (version 2.0) that addresses text only. The most recent extensions, enabling the writing of complex mathematical formulas, tables, and other advanced features, are not yet managed.

A document representing the HTML mark-up language is called a *HTML document.* Without entering into the details of the required format (see Melucci, 1994), we must observe that a HTML document must follow the following minimal structure:

⟨HTML⟩	beginning of the document
⟨HEAD⟩	beginning of the header
⟨TITLE⟩	begining of the document identifier; if empty TACHIR will automatically assign one
⟨/TITLE⟩	end of the title
⟨/HEAD⟩	end of the header
⟨BODY⟩	beginning of the content to be indexed
⟨/BODY⟩	end of the content
⟨/HTML⟩	ending of the document

For example let us consider the following HTML document:

⟨HTML⟩
⟨HEAD⟩
⟨TITLE⟩38.2,1⟨/TITLE⟩
⟨/HEAD⟩
⟨BODY⟩
⟨h1⟩
Tracking Design Changes with Formal Machine-Checked Proof/h1

⟨h3⟩ Paul Curzon⟨br⟩ University of Cambridge Computer Laboratory, Pembroke Street, Cambridge, UK. ⟨br⟩ email: pc@cl.cam.ac.uk WWW:⟨A
href="http://www.cl.cam.ac.uk/users/pc/"⟩
http://www.cl.cam.ac.uk/users/pc/⟨/a⟩⟨/h3⟩

Designs are often modified for use in new circumstance. If formal proof is to be an acceptable verification methodology for industry, it must be capable of tracking design changes quickly. We describe our experiences formally verifying an implementation of an ATM network component, and on our subsequent verification of modified designs. Three of the designs verified are in use in a working network. They were designed and implemented with no consideration for formal methods. This case study gives an indication of the difficulties in formally verifying a real design and of subsequently tracking design changes.⟨p⟩⟨hr⟩ ⟨a href="default.html"⟩⟨img src=
../Icons/ManLeftArrow.gif border=0⟩⟨/a⟩

⟨/BODY⟩
⟨/HTML⟩

TACHIR will identify this document as *38.2_1*, and the text that lies within ⟨*BODY*⟩ and ⟨*/BODY*⟩ will be indexed. With such tags, TACHIR will treat the document content as full text and index it accordingly, without giving to its individual parts: title, abstract, and author, any specific semantics. HTML tags are ignored during indexing, but TACHIR preserves the layout and then HTML tags are not removed from the text. The document identifier will act as an anchor and will be used to link to this document index terms and/or other documents within the

hypertext.

As we reported above, a practical way of avoiding to identify the set of concepts and relationships is by using an existing thesaurus or classification schema. In a few of our applications we made use of the *ACM classification scheme*. We produced a HTML version of the scheme and we fed it into TACHIR as the concept level of the TACHIR conceptual architecture. An example of a section of the ACM classification scheme is reported in the following:

```
⟨HTML⟩
⟨HEAD⟩
⟨TITLE⟩
Extended Computing Reviews Classification Scheme
⟨/TITLE⟩
⟨/HEAD⟩
⟨BODY⟩
⟨UL⟩
⟨LI⟩A. GENERAL LITERATURE
⟨UL⟩
⟨LI⟩    A.0 GENERAL
⟨UL⟩
⟨LI⟩      biographies/autobiographies
⟨LI⟩      conference proceedings
⟨LI⟩      general literary works (e.g. fiction, plays)
⟨/UL⟩
⟨LI⟩    A.1 INTRODUCTORY AND SURVEY
⟨LI⟩    A.2 REFERENCE (e.g., dictionaries, encyclopaedias, glossaries)
⟨LI⟩    A.m MISCELLANEOUS
⟨/UL⟩
⟨LI⟩  B. HARDWARE
⟨U⟩
⟨LI⟩    B.0 GENERAL
⟨LI⟩    B.1 CONTROL STRUCTURES AND MICROPROGRAMMING
⟨LI⟩      (D.3.2)
⟨UL⟩
⟨LI⟩      B.1.0 General
⟨/UL⟩
⟨LI⟩      B.1.1 Control Design Styles
⟨UL⟩
⟨LI⟩        hardwired control
⟨LI⟩        microprogrammed logic arrays
⟨LI⟩        writable control store
⟨/UL⟩
```

Main features and characteristics of TACHIR. Figure 3 illustrates the TACHIR architecture and processes. This figure refers to the conceptual architecture reported in Fig. 1, where the tasks performed are grouped in the IR Object Builder, and the data processed in the Hypertext Author.

IR Object Builder. The task of the *IR Object Builder* is to construct the three levels of the IR Hypertext as described in Section 5. The IR Object Builder performs the indexing of the collection and the representation of the application domain knowledge. Indexing consists of stop word removal, conflation, and index term weighting. The representation of the application domain knowledge consists of the transformation of a knowledge base, in our case a classification scheme, into a network of concepts.

The algorithms implemented by the IR Object Builder are classical IR algorithms that have been used for long time in many IR systems to perform stop words removal, conflation, and index term weighting. Documents are represented as sets of weighted index terms using a

formalism very close to the one used in the IR Vector Space Model (Salton & McGill, 1983).

In the following the main IR Object Builder procedures are described

- *Stop list removal:* reads a file containing a list of stop words, and removes them from the input documents.
- *Conflation:* reduces words into their stems, using Porter's stemming algorithm (Porter, 1980). The original Porter's stemming procedure has been modified by introducing a "stemming degree" ranging from 0 (no stemming) to 9 (full stemming); the designer of the specific IR hypertext can fix the value of the stemming degree when running the application in order to set the desired level of stemming according to fewer (high level of stemming) or more (low level of stemming) index terms.
- *Weighting:* assigns weights to stemmed words. Each weighted stem is assigned two

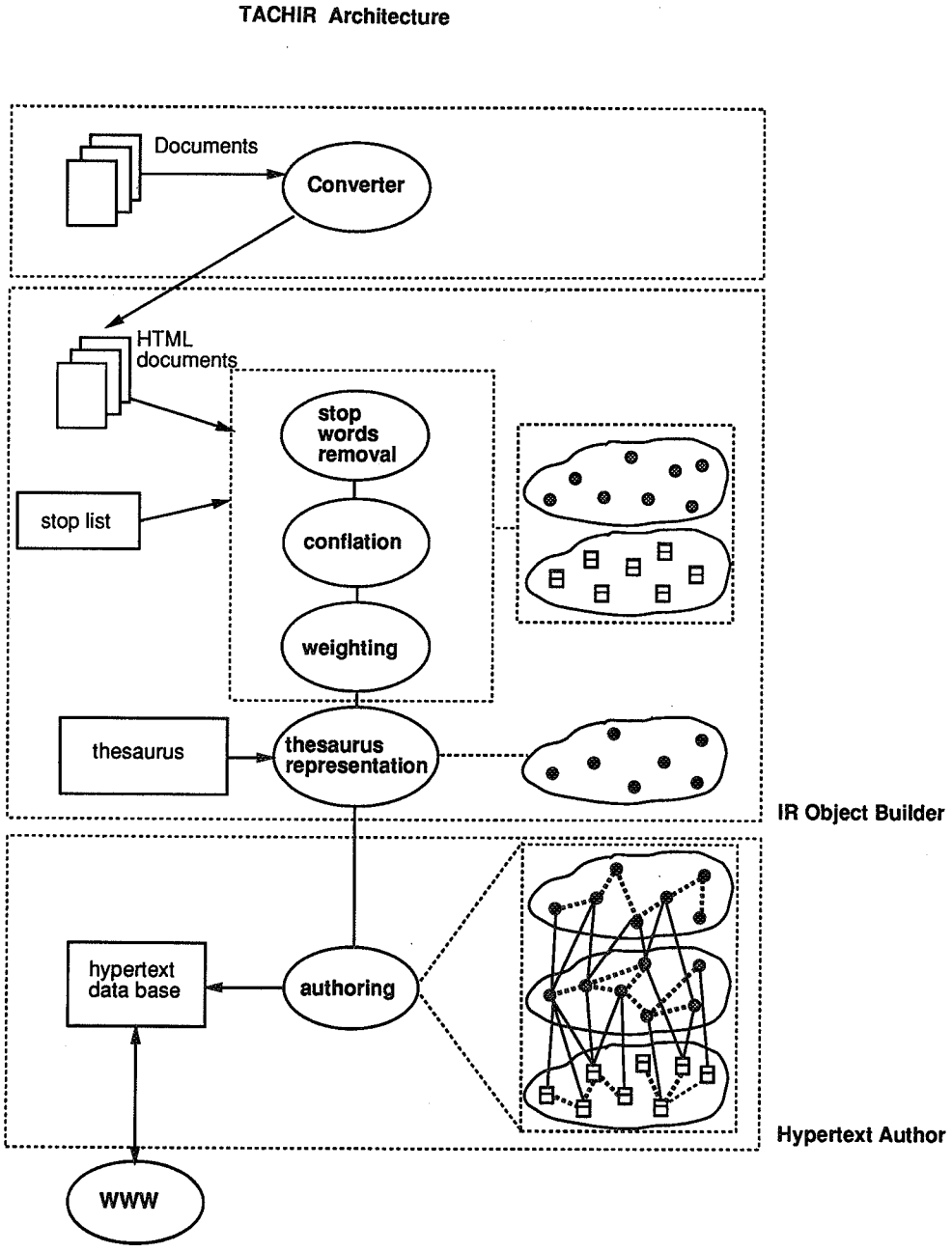

Fig. 3. TACHIR architecture.

weights: the intra-document weight and the intra-collection weight. Cut-off values are set up in the form of weight thresholds to get rid of stems that are either too rare or frequent in the document collection (Salton *et al.*, 1975).

- *Construction of the concept level:* TACHIR reads a file containing a set of concepts and relationships that represents the application domain knowledge and produces a network of concepts. In most of our applications of TACHIR we made use of the ACM classification scheme.
  In this case TACHIR recognizes the hierarchical structure by relating each concept to its narrower and broader concepts. Thus the scheme is translated into a tree that can be navigated across the hierarchy.

All the parameters used by TACHIR in the process of creating the IR Hypertext nodes can be set or modified by the application designer to customize the generation of the IR Hypertext.

The whole algorithm performed by the IR Object Builder can be summarized as follows:

```
read the file including the stop list

for each input file
  for each document
    conflate words and extract index terms
    create an object representing the document
    create objects representing index terms

read the file with the ACM classification scheme

for each concept of the ACM scheme
  conflate and extract index terms
  recognize and relate broader and narrower concepts
  create objects representing concepts
  create objects representing index terms

build the document level of the hypertext

build the index term level of the hypertext

build the concept level of the hypertext.
```

Hypertext author. The task of the Hypertext Author is to link the three hypertext levels. The Hypertext Author builds the hypertext by setting up links between the nodes representing documents, index terms, and concepts.

- Documents are linked together and to the index terms that have been extracted from them. Index terms linked to documents are ranked by an intra-document weight, while documents are linked and ranked to each other by document–similarity.
- Index terms are linked to each other, to documents, and to the concepts of the application domain knowledge. Index terms linked to documents are ranked by intra-document weight, while index terms are linked and ranked to each other by term–term similarity.
- Concepts are linked to each other and to the index terms. Index terms linked to concepts are ranked by intra-collection weight. Concepts are linked together on the basis of the semantic relationships provided by the classification scheme, in the case of the ACM Classification scheme each concept is linked to a list of narrower concepts and to a broader concept (if they exist).

  Links express semantic or statistical relationships between the nodes depending on the kind of information used to determine them. Link ranking is performed in order to implement a form of preference relation between the source node and the destination nodes. Moreover, it is possible to cut off the number of links that are presented to the user by setting up a threshold on the document–document and term–term similarity. A proper use of the ranking enables us to present the user with the links that are most likely to be interesting. A few parameters enable us to tune TACHIR to obtain the best ranking performance for a particular application.

The algorithm performed by the Hypertext Author is summarized in the following:

```
for each document object
  identify documents with similarity over threshold
  create similar document list
  rank similar document list using similarity
  link similar documents in list
  identify index terms with intra-document weight over threshold
  create index term list
  rank index term list using the intra-document weight
  link index terms and documents

for each index term object
  identify index terms with similarity over threshold
  create similar index term list
  rank similar index term list
  link similar terms in list
  identify concepts where a index term with
    intra-collection weight over threshold occurs
  rank the index term list using the intra-collection weight
  link index terms and concepts

for each concept object
  link narrower concepts
  link broader concept.
```

Output data. TACHIR's output consists of the network structure depicted in Fig. 1. Conceptually, the structure is composed of nodes and links. There are two types of nodes, as depicted in Fig. 4.

- *hypertext nodes:* nodes representing documents, index terms, or concepts (the white boxes in Fig. 4). Each hypertext node contains data and anchors to link nodes, for example a document node includes a piece of text, and two anchors, one to a list of index terms, and one to a list of similar documents.
- *link nodes:* a link node (the grey boxes in Fig. 4) is a list of anchors to hypertext nodes, for example a link node implementing the relationship between a document and its index terms contains a list of anchors connecting the document node to the index term nodes.

In Section 5, we already stressed how it is almost always necessary to build "by hand" the semantic relationship between concepts. In this case, it is the user himself or a team of domain experts who has to build up the semantic structure which represents important and useful application domain knowledge. However, if this semantic structure is represented and stored in a machine readable form, TACHIR is able to build up automatically the network of concepts.

8. CONSIDERATIONS ON THE EVALUATION OF TACHIR

The elevation techniques that have been developed in IR are not directly usable in the evaluation of the characteristics of IR hypertext systems, because of the new element of interactivity introduced by the new systems. It is therefore necessary to develop new procedures and tools that establish a relationship between the evaluation of new interactive IR systems with previous evaluation work in IR.

Directions that we are investigating use a combination of methods as suggested in Jones *et al.* (1991).

Measure of overall system performance: aims to measure overall system performance through interaction tests, comparative studies, and controlled experiments.

Transaction log analysis: uses methods that are more user-oriented and that take into account all the data elicited by the system from the user during the interaction with the IR hypertext

system. These data are collected at interaction time by a transaction log mechanism. The data collected are later analysed using statistical and cognitive techniques.

9. CONCLUSIONS AND FUTURE WORK

In this paper we report on the design and implementation of TACHIR, a tool for the automatic construction of hypertext for IR. The tool uses a methodology we already presented in Agosti and Cresenti (1993) and Agosti *et al.* (1994). TACHIR constructs automatically an IR hypertext for a collection of textual documents and writes it using the HTML mark-up language. This enables the user to browse the IR hypertext using any of the interfaces and browsing tools developed in the framework of the WWW project.

At present we are studying ways of improving the user access to the hypertext nodes by adding techniques to query the hypertext. This would enable the user to access directly nodes that are good starting points for browsing and from which he could reach relevant nodes more

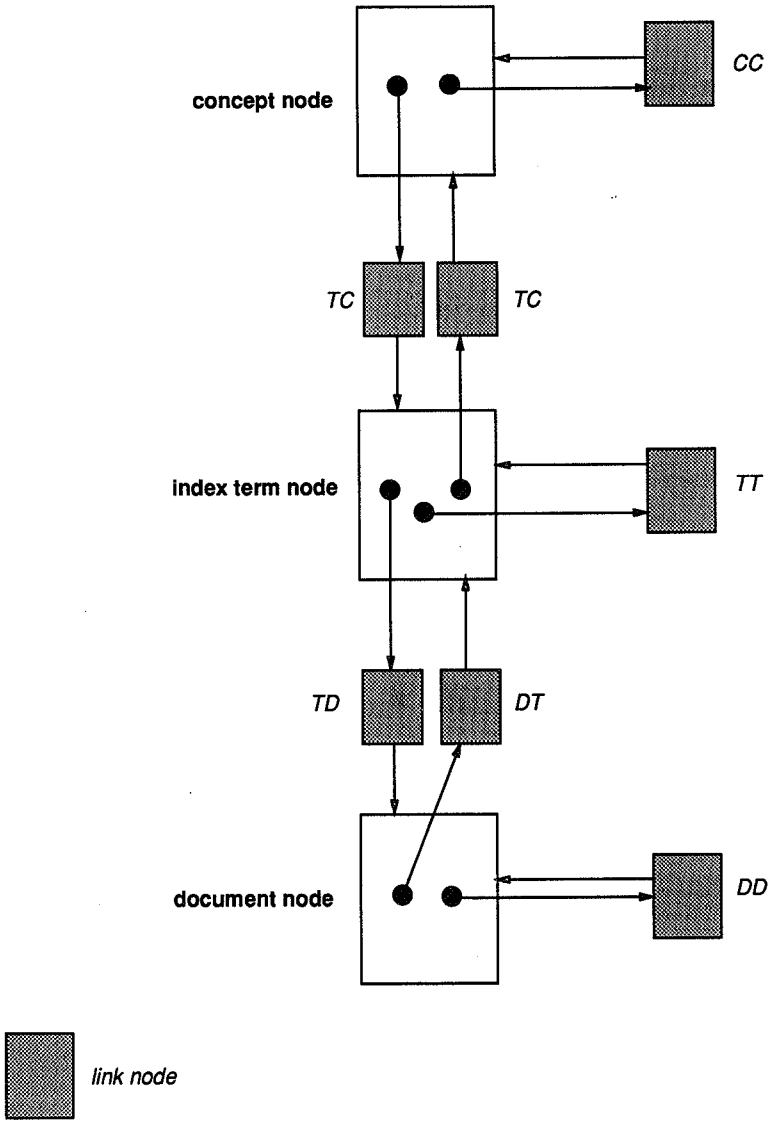

Fig. 4. Hypertext organization.

quickly. We are also studying techniques for using the hypertext to perform retrieval of relevant nodes by "constrained spreading activation". The aim is to spread the user relevance assessment, nodes that have been pointed out to be relevant, to nodes that are related to them with some degree of significance. The result would be a ranking of a large set of nodes of the IR hypertext that will be presented to the user using an appropriate interface. The user could then assess again the retrieved set and produce a new spreading of activation, thus making the retrieval process an iterative and interactive process.

Acknowledgements—The authors would like to thank the reviewers for the useful comments on the earlier version of the paper. This work for partially funded by the Esprit IDOMENEUS Network of Excellence under the IRIDES Project.

REFERENCES

Aalbersberg, I. J. (1992). Incremental relevance feedback. In *Proceedings of ACM SIGIR* (pp. 11–21). Copenhagen.

Agosti, M. (1993). Editorial—hypertext and information retrieval. *Information Processing & Management, 29*(3), 283–285.

Agosti, M., Colotti, R., & Gradenigo, G. (1991). A two-level hypertext retrieval model for legal data. In *Proceedings of ACM SIGIR* (pp. 316–325). Chicago, October 13–16.

Agosti, M. & Crestani, F. (1993). A methodology for the automatic construction of a Hypertext for Information Retrieval. In *Proceedings of the ACM Symposium on Applied Computing* (pp. 745–753). Indianapolis, U.S.A.

Agosti, M., Crestani, F., Gradenigo, G., & Mattiello, P. (1990). An approach for the conceptual modelling of IR auxiliary data. In *Proceedings of the 9th Annual IEEE International Phoenix Conference on Computers* (pp. 500–505). Scottsdale, Arizona.

Agosti, M., Crestani, F., & Melucci, M. (1994). TACHIR: A tool for the automatic construction of hypertext for information retrieval. In *Proceedings of the RIAO 94 Conference*. New York, U.S.A.

Agosti, M., & Marchetti, P. G. (1992). User navigation in the IRS conceptual structure through a semantic association function. *The Computer Journal, 35*(3), 194–199.

Aitchison, J., & Gilchrist, A. (1987). *Thesaurus Construction. A Practical Manual.* (2nd edn). ASLIB, London.

Allan, J., & Gilchrist, A. (1987). *Thesaurus Construction. A Practical Manual.* (2nd edn). ASLIB, London.

Allan, J. (1995). Relevance feedback with too much data. In *Proceedings of ACM SIGIR*, Seattle, WA.

Berners, T. J., Caillau, R., Groff, J. F., & Pollermannes, B. (1992). *World Wide Web: the Information Universe* (Vol. 2, pp. 52–58) Westport, CT: Mecker Publishing.

Crouch, C. J. (1990). An approach to the automatic construction of global thesauri. *Information Processing & Management, 26*(5), 629–640.

Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T., & Harshman. (1990). Indexing by latent semantic analysis. *Journal of the American Society for Information Science, 41*(6), 391–407.

Dunlop, M. (1991). *Multimedia Information Retrieval.* PhD Thesis, Department of Computing Science, University of Glasgow, Glasgow, U.K.

Dunlop, M. & van Rijsbergen, C. J. (1993). Hypermedia and free text retrieval. *Information Processing & Management, 29*(3), 287–298.

Edmundson, H. P. & Wyllys, R. E. (1961). Automatic abstracting and indexing-survey and recommendations. *Communications of the ACM, 4*(5), 226–234.

Frakes, W. B. & Baeza-Yates, R. (Eds) (1992). *Information Retrieval: Data Structures and Algorithms.* Englewood Cliffs, New Jersey: Prentice Hall.

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

Furuta, R., Plaisant, C., & Schneiderman, B. (1989). Automatically transforming regularly structured linear documents into hypertext. *Electronic Publishing, 4*(2), 211–229.

Hancock-Beaulieu, M., Robertson, S. E., & Neilson, C. (1991). Evaluation of online catalogues: Eliciting information from the user. *Information Processing & Management, 27*(5), 523–532.

Jones, P. E., Curtis, R. M., Giuliano, V. E., & Sherry, M. E. (1968) *Applications of statistical associations techniques for the NASA document collection.* Technical Report CR-1020, NASA.

Loomis, M. E. S., Shah, A. V., & Rumbaugh, J. E. (1987). An object modelling technique for conceptual design. In *Proceedings of the European Conference on Object-Oriented Programming* (pp. 325–335). Paris: AFCET.

Melucci, M. (1994). *A tool for the Automatic Construction of Hypertexts for information retrieval.* Technical report, Dipartimento di Elettronica e Informatica, Università di Padova, October 1994.

Oddy, R. N. (1975). *Reference Retrieval Based on User Inducted Dynamic Clustering.* PhD Thesis, University of Newcastle upon Tyne, U.K.

Porter, M. F. (1980). An algorithm for suffix stripping. *Program, 14*(3), 130–137.

Qiu, Y. & Frei, H. P. (1994). *Constructing, Updating, and Employing a Similarity thesaurus.* Pers. Commun. 1994.

Rada, R. (1992). Converting a textbook to hypertext. *ACM Transactions on Information Systems, 10*(3), 294–315.

van Rijsbergen, C. J. (1977). A theoretical basis for the use of co-occurrence data in Information Retrieval. *Journal of Documentation, 33*(2), 106–119.

van Rijsbergen, C. J. (1979). *Information Retrieval* (2nd Edn). London: Butterworths.

Robertson, S. E. & Sparck Jones, K. (1976). Relevance weighting of search terms. *Journal of the American Society for Information Science, 27*, 129–146.

Salton G. (1968). *Automatic Information Organization and Retrieval.* New York: McGraw Hill.
Salton, G. (1971). *The SMART Retrieval System. Experiments in Automatic Document Processing.* New Jersey: Prentice–Hall.
Salton, G., Allan, J., & Buckley, C. (1993). Approaches to passage retrieval in full text information systems. In *Proceedings of ACM SIGIR* (pp. 49–58). Pittsburgh: ACM Press.
Salton, G., Allan, J., & Buckley, C. (1994). Automatic structuring and retrieval of large text files. *Communication of the ACM, 37*(2), 97–108.
Salton, G. & Buckley, C. (1989). *Automatic generation of content links for hypertext.* Technical report, Department of Computer Science, Cornell University, Ithaca, New York.
Salton, G. & McGill, M. J. (1983). *Introduction to Modern Information Retrieval.* New York: McGraw-Hill.
Salton, G., Yang, C. S., & Yu, C. T. (1975). A theory of term importance in automatic text analysis. *Journal of the American Society for Information Science, 26*(1), 33–44.
Smeaton, A. F. (1995). Building hypertext under the influence of topology metrics. In *Proceedings of IWHD Conference,* Montpellier, June 1995.
Srinivasdan P. (1992). Thesaurus construction. In Frakes, W. B. & Baeza-Yates, R. (Eds), *Information Retrieval: Data Structures and Algorithms* (Chap. 9). Englewood Cliffs, New Jersey: Prentice Hall.
Stroustrup, B. (1991). *The C++ Programming Language* (2nd Edn). Reading, MA: Eddison-Wesley.
Thompson, R. H. (1989). *The Design and Implementation of an Intelligent Interface for Information Retrieval.* Technical report, Computer and Information Science Department, University of Massachusetts.