INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET AUTOMATIQUE

Information Retrieval and Link Authoring in an SGML-based Editor

Anne-Marie Vercoustre, Craig A. Lindley

N° 2591
Juin 1995

PROGRAMME 2
Calcul symbolique,
programmation
et génie logiciel

Rapport
de recherche

ISSN 0249-6399

Information Retrieval and Link Authoring in an SGML-based Editor

Anne-Marie Vercoustre[*], Craig A. Lindley[**]

Programme 2 — Calcul symbolique, programmation et génie logiciel

Projet Atoll

Rapport de recherche n°2591 — Juin 1995

Abstract: This document describes the integration of Grif, an SGML editor developed at Inria and marketed by Grif.Sa, with Sigma, a text retrieval tool developed by CSIRO. The integration provides Grif with flexible search and dynamic hypertext linking functions, and enhances the Sigma system to support search and display of SGML documents using a structured editor. The integration also clarifies the requirements for more generic facilities for document search, linking, and indexing for the respective systems as modular components of an open systems environment.

(Résumé : tsvp)

---

[*] Anne-Marie.Vercoustre@inria.fr, en détachement au CSIRO-DIT, de décembre 1993 à août 1994

[**] lindley@syd.dit.csiro.au, CSIRO Division of Information Technology, North Ryde, NSW, Australia

Unité de recherche INRIA Rocquencourt
Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex (France)
Téléphone : (33 1) 39 63 55 11 – Télécopie : (33 1) 39 63 53 30

Liens dynamiques et recherche textuelle pour un Editeur de documents SGML

Résumé : Ce rapport décritl'intégration de Grif, un éditeur de documents SGML, avec Sigma, un outil de recherche textuelle développé au CSIRO. Cette integration dote Grif d'une fonctionalité de recherche textuelle dans la base de documents qui peut être considérée comme des liens dynamiques. Inversement, le système Sigma est ainsi étendu à la recherche d'information dans des documents SGML et leur affichage par un éditeur structuré. Le papier met en évidence l'architecture de cette intégration et la communication entre les deux outils.

Information Retrieval and Link Authoring in an SGML-based Editor

Anne-Marie Vercoustre *
INRIA-Rocquencourt,
78153 Le Chesnay Cedex, France
Anne-Marie.Vercoustre@inria.fr

Craig A. Lindley
CSIRO, Division of Information Technology
North Ryde, NSW, Australia
lindley@syd.dit.csiro.au

Abstract

This document describes the integration of Grif, an SGML editor developed at Inria and marketed by Grif.Sa, with Sigma, a text retrieval tool developed by CSIRO. The integration provides Grif with flexible search and dynamic hypertext linking functions, and enhances the Sigma system to support search and display of SGML documents using a structured editor. The integration also clarifies the requirements for more generic facilities for document search, linking, and indexing for the respective systems as modular components of an open systems environment.

1 Motivations

Grif is an SGML editor providing authoring capabilities for writing documents according to a specific document model, or document type definition (DTD). It also offers hypertext capabilities for linking documents or parts of documents. For creating links the author must explicitly select a target object, requiring explicit knowledge of exactly what to link and where the target information is located (ie. which file or database).

Although explicit links may be required for precise references, handling them by hand rapidly becomes tedious and then infeasable as the size of the document or database, and therefore the number of links involved, increases. Moreover, the user may want to search for information freely and in different ways from those provided for by the author. In this case dynamic links are preferred. Integration with an information retrieval tool will help in both of these cases:

- to automatically retrieve a referenced object to be linked
- to provide dynamic linking between pieces of information semantically connected
- to provide dynamic linking in response to an arbitrary user query.

The integration of a text retrieval tool with Grif allows an author to choose to make static links as appropriate for the type of link or application, while supplementing all static links with interactive dynamic linking.

---

*In secondment to CSIRO-DIT, from December 1993 to August 1994.

Sigma is a retrieval tool for textual search within and among documents, based on statistical similarities. Currently Sigma can search on text documents where the elements to be retrieved are separated by specific markers. Sigma extensions to deal with SGML documents will increase the range of applications where Sigma could be used. Integration of Sigma with Grif supports retrieval of components of SGML documents, and provides a high level display for the retrieved information (see 3.1), as well as full authoring tools for SGML documents. Evaluation of the prototype system developed to demonstrate this integration highlights the need for modifications of the Sigma indexing system to make the software more flexible and dynamic. The extensions made to Grif are compatible with a variety of text retrieval engines, and so represent general purpose functions and interfaces in an open environment.

2 Functional requirements

The following functions are required:

- an indexing mechanism for SGML documents
- a search function for SGML documents
- display of a ranked list of references to retrieved SGML document parts, with support for the selection of references from the list
- a display of retrieved SGML document parts

Providing these functions by integrating Grif and Sigma will require:

- communication (interfaces) between Grif and Sigma
- a method of allowing the Sigma preprocessor to index documents initially represented within Grif tree structures
- extension of the Grif user interface to accommodate the new capabilities.

3 Sigma description

The Sigma software system is a set of C++ software routines for ranked information retrieval. The routines have been used for standard information retrieval, and also for automated link generation in hypertext and multimedia applications (Lindley et al, 1994). This section summarises the operation of the Sigma software and the retrieval methods upon which it is based.

Sigma retrieval takes place within the scope of a number of *source documents.* The source documents are each subdivided into *logical text units* (or LTUs), that are the individual targets of retrieval (ie. the items that are retrieved). LTUs are separated within source documents by appropriate markers.

3.1 Sigma Preprocessing

Sigma retrieval first requires preprocessing of the material that is to be retrieved (see Figure 1).

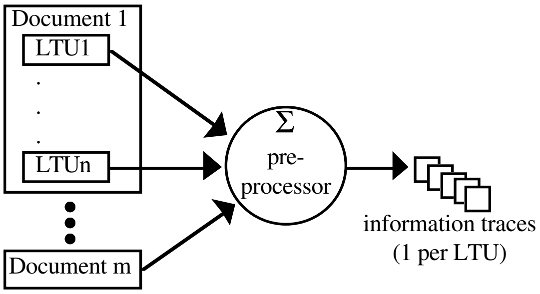


Figure 1. Sigma pre-processing.

Input parameters for the preprocessor include:

- the name of a file that lists all of the source documents to be referenced by a particular index. The source documents may be located in any accessible place within a network environment.
- the name of the index file that is to be generated.

Preprocessing involves the application of:

1. *conflation algorithms*, that carry out word stemming on the input text files to overcome problems with different word endings, variations in spelling, punctuation, and hyphenation.
2. *stop lists*, to filter out terms within the input text files that add little to the mapping operation, such as conjuncts, articles, etc.. Stop lists may be general, or application-specific.
3. *synonynm* lists to ensure that similarity matching is not limited to identical word forms, but captures identity of meaning. Synonym lists may be general, or application-specific.
4. information trace generation algorithms, to produce information traces for each LTU in the text base. The generation of information traces is the only mandatory aspect of preprocessing.

The Sigma information traces are fundamental to the retrieval process. To generate an information trace, a pre-filtered document is parsed into character substrings of length n, referred to as *ngrams* (a default value of n = 3 has been found to provide the best results). The number of occurrences of each unique ngram are counted. The information trace for an LTU contains a list of each ngram that the LTU contains, along with the frequency of occurrence of each ngram within that LTU. The information trace for a document set (or database) contains the total number of occurrences of each unique ngram within the document set (this depends upon the kind of similarity metric used; not all similarity metrics use such a database-wide information trace). The information traces for an LTU or a document set also include the *relative frequency* of each ngram within the LTU or document set, respectively. Currently, the global information trace (spanning the whole document set covered by an index) is generated dynamically when the Sigma process is invoked.

The output of the Sigma preprocessor is an index file that contains:

- an information trace for each LTU, along with a reference to the LTU (eg. path name and offset within the text file that an LTU belongs to)
- possibly an information trace for the whole document set covered by the index, depending upon the similarity metric used and whether this global trace is generated statically or dynamically.

3.2 Sigma Retrieval

The index produced by the Sigma preprocessor is used for information retrieval (see FigureD 2).

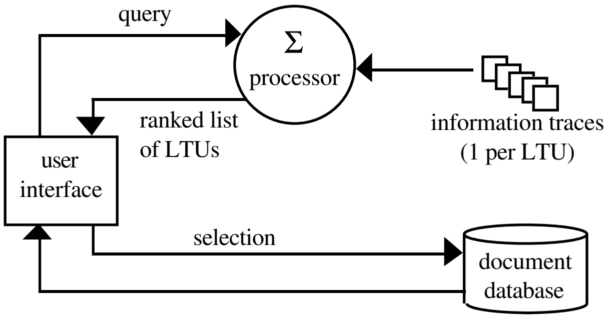


Figure 2. Sigma retrieval process.

Input parameters used by the retrieval process are:

- the name of the index file
- a threshold for minimum item similarities

The algorithm used by the retrieval process is:

- accept an input search string
- apply conflation algorithms, stop-list filtering, and synonym expansion to the input string
- generate an information trace for the input string
- generate similarity measures that represent the similarity of the input string to each of the LTUs within the target document set, as a function of ngram frequencies within the search string, the LTUs, and (possibly) the overall document set

- return a list of LTUs in decreasing order of similarity to the input string, for those LTUs with a similarity above the specified similarity threshold. The returned list includes the path name to the associated item in the document set.

The similarity calculation can be made using a number of methods (see Teufel, 1989, and Teufel and Schmidt, 1988). The current CSIRO Sigma algorithms use a De Heer metric. A De Heer metric is an example of a *direct similarity measure* thatmeasures the similarity between a list of key terms and a document. The De Heer metric is:

$$YD1 = \frac{\mid p(q) \bigcap p(a) \mid}{\mid p(q) \mid} * z(q,a) * e^{(1-a(q,a))}$$

where: YD1 is the measure of similarity between a particular ngram and a particular LTU

p(q) is the set of ngrams of the query

p(a) is the set of the ngrams within an LTU

z(q,a) is the average frequency of ngrams that are in both the query and the text database

e$^{(1-a(q,a))}$ is a weighting function

Once a list of items has been returned, a particular item can be selected (eg. by a user) and retrieved from the storage system. This level of operation is accomplished by the software within which the Sigma routines are embedded. The Sigma routines have successfully been embedded as automated link generation functions within a hypermedia system (Lindley et al, 1994), and for supplementary support within a combined hypermedia and knowledge based decision support system (Kumar and Lindley, 1994).

Note that the Sigma retrieval system can be regarded as an instance of an information retrieval system based upon weighted models (type SW) as described by Mazur (1994) when:

- each ngram is regarded as a single descriptor, and an ngram/descriptor is regarded as a taxonomic category such that every document *d* associated with the descriptor t contains at least one occurrence of the ngram type identified with t.

- there are no generalisation relationships between descriptors (ie. between ngrams).

- the relative frequency of an ngram t in a document *d* is equated with the weight *w* of t in *d*.

4 Grif as an SGML editor

Grif is an SGML WYSIWYG Editor developed and marketed by Grif.Sa, France, based on the Grif prototype developed by Inria and CNRS. Grif supports the production of documents that conform to the SGML standard and contain text, tables, graphics, mathematical formulae, etc..

Grif's interactive user interface assumes no prior knowledge of the SGML language. It gives users a “natural “ view of documents (wysiwyg), hiding the SGML tags unless they are required for specific tasks. When the document is edited, grif uses a Document model (DTD) that specifies the possible structure and presentation of the document, displayed in logical views.

When a document is edited, all commands are contextual and guarantee that the document is always valid and in conformance with its model.

4.1 Grif: an open and extensible system

Grif is based upon an object-oriented architecture in which each element of a DTD is considered as an object. These objects can react to stimuli (commands) received from the user via the editor.

Openness is provided at several level:

- creation of new DTDs and their presentation models: the Grif Builder allows for adding new kinds of document or their presentations. Compiling the new specifications will produce the adequate tools for these new types of document. “Documents“ must be understood here in a broad sense including “data structures“ used by external applications.

- extension or tuning of the user interface: new menus may be added to the standard interface to provide new functionality. Markup callback facilities are provided through message-action association. Menu items may be restricted to be active when selecting specific classes of elements or documents (contextual commands).

- Development of new applications using the Grif Application Interface (GATE). With GATE it is possible to build new documents or update existing documents from applications. Documents may then act as rich interfaces for applications.

- Access to external data and external applications through a listener function.

All these Grif capabilities have been extensively used for the Grif-Sigma integration.

4.2 Tree representation of SGML Documents in Grif

Documents in Grif are internally represented as trees that reflect the logical structure as defined by the Document Model (DTD). Grif uses an SGML parser to build the tree representation for the document . The parser checks whether the document conforms to the DTD (the grammar for this document). When the document has been edited with Grif it is garanteed to conform to its model since Grif allows only for contextual and correct modifications.

Figure 3. shows un example of a document marked up using SGML.

```
<article status=“final“>
<title> Information Retrieval and Links Authoring in an SGML-based Editor
<\title>
<author> Anne-Marie Vercoustre
<author> Craig A. Lindley
<abstract> This document describes the integration of Grif, an SGML editor
developed at Inria and marketed by Grif.Sa, with Sigma, a text retrieval tool de-
veloped by CSIRO.
...
<section>
<title> Motivation<\title>
...
<body><paragr font=“default“> Grif is an SGML editor which provides author-
ing capabilities for writing documents according to a specific document model, or
document type definition (DTD) ...
<\body><\section>
...
<section>
<title> Sigma description<\title>
<body><paragr font=“default“> The Sigma software system is a set of C++
software routines for ranked information retrieval ...
<\body><\section>
<section>
<title> Grif as an SGML editor <\title>
<body><paragr font=“default“> Grif is an SGML WYSIWYG Editor developped
and marketed by Grif.Sa, France, based on the Grif prototype developped by Inria
and CNRS. ...
<\body>
<subsection>
<title> Grif an open and extensible system<\title>
<body><paragr font=“default“> Grif is based upon an object-oriented architec-
ture in which each element of a DTD is considered as an object.
<\body>
<\subsection><\section>
...
<acknowl>We are grateful to V.R Kumar for his support in providing Sigma in-
terfaces <\article>
```

Figure 3. An example of an SGML Tagged file.

Figure 4 shows an example of a tree document representation for the SGML document given in Fig.3

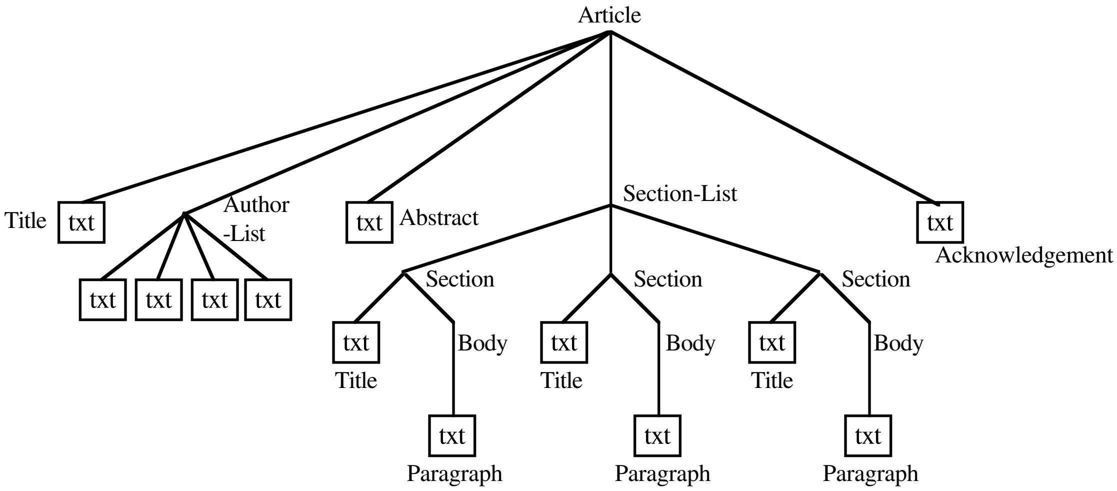


Figure 4. Tree document representation for the SGML document file given in Figure 3. The small boxes represent the textual content (i.e basic elements).

5 Analysis of Integration Requirements

The starting point for integrating Grif and Sigma was to change as little as possible of the existing software. In the first part of this section we describe how to extend Sigma from text retrieval to SGML documents using tagged text files. Several solutions for integration with Grif are considered within this framework; however, this approach raises many technical difficulties, so we then propose a stronger integration using Grif document representations.

5.1 Indexing of SGML documents

The extension of Sigma indexing to SGML documents will involve:

- use of SGML tags for deciding the granularity of chunks to retrieve and for identifying the elements[1].
- filtering of the list of tag names from SGML documents prior to preprocessing so that SGML markup is not used in the search and matching process
- re-indexing of updated documents

In order to simplify the above issues, indexing can be applied in a first phase to only one model of a document (i.e. to documents conforming to a specific DTD), with a new preliminary preprocessing step performed by Grif to create document sources for the Sigma preprocessor that include the appropriate LTU parsing and SGML component identification, but without SGML markup.

---

[1] Note that Sgml syntax does not require that each opening tag is closed with the corresponding ending tag; one closing tag may close a hierarchy of hanging open tags.

5.2 Display of a ranked list of retrieved elements

Currently Sigma provides a ranked list of retrieved elements and the first line of each element is displayed to the user (eg. using the Hyperbrowser) to allow the user to select and view desired items. With Grif the list of elements can easily be defined and displayed as a *path* . As in the Intellitext system described by Jansen (1993), a path is a list of references to typed elements such as sections, figures, or notes within a document or among different documents. In Grif, a path is implemented as an instance of a general path type defined as an SGML document model.

5.3 Displaying a retrieved element

It is not generally desirable from a users viewpoint to diplay the SGML markup during retrieved document display, although this may be useful under some circumstances. If a display system is used that does not handle SGML markup appropriately, an "on the fly" translater could be used to remove low level tags and apply some pretty printing heuristics. (Note that SGML documents do not contain any information on the way documents may be displayed, except fonts as attributes. Other descriptions, known as *presentation models,* must be used by SGML displayer systems.) This has the disadvantages of losing the ability to use ancillary presentation information defined in terms of SGML structures, and of not providing the option for viewing the SGML markup.

A better alternative is to use an SGML display system like Grif for displaying SGML documents or document components. Grif has the particular advantage of providing the user with an optional window for viewing the SGML markup, as well as the primary structured presentation interface. A major issue in using Grif with Sigma is how Grif will be given the reference to the part of a document identified by Sigma.

5.4 Designation of an element in an SGML document

As seen in the previous section, a protocol must be defined between Sigma and Grif to identify a selected element within a document. SGML/HTML protocols for specifying targets of links within documents cannot be used because they are based on named and predefined target anchors.

In the current version of Sigma, the retrieved elements are referred to by the numbers of the lines enclosing the corresponding text within a file[2]. In contrast, Grif uses an SGML parser to built an internal tree representation of a document. Hence it is necessary in integrating the Grif and Sigma systems to maintain the relationship between coordinates (ie. enclosing line numbers) in the Sigma source file and the corresponding subtree positions in the Grif document representation. Some potential methods for doing this are:

1. add attributes to the tree structure in Grif for recording the enclosing line number

---

[2] This information is not generally sufficient to identify the selected element in the case of SGML documents because several tagged elements may share the same subset of lines (for example two very short paragraphs may be kept on the same line).

of each substructure. The attributes must be set during source parsing.This solution would require modifications of the tree constructor connected to the SGML parser, a solution that may not be accessible in the commercial version of Grif.

2. use a skeleton of the SGML document as a protocol, i.e an SGML document containing all of the tags of the initial document with no content other than the text corresponding to the retrieved element(s). Grif would load both the full document and the skeleton and match the tree of the skeleton against the full tree for identifying the place(s) of interest. This may be computationally expensive in both the generation of the skeleton files and in matching to a currently loaded document.
3. use a protocol containing the text of the element retrieved and the name of the document that contains it. Once Grif has loaded the document, it will search for the textual string using its internal search mechanism. The efficiency of this solution must be evaluated; an efficiency improvement can be achieved by filtering the search with the tag name (added to the protocol). This approach will not properly handle cases where text components are duplicated.
4. record the Grif tree position as part of the path name of an LTU in the Sigma index. This means that the tree position must be provided to the Sigma preprocessor, so the method cannot be used to access documents that have not received some form of additional preprocessing by Grif. This method has been used for the demonstration prototype as the most appropriate and viable method of integrating the Grif and Sigma systems without extensive modifications.

Note that in all cases the document must be reindexed when it has changed; when saving a document Grif can easily call Sigma for reindexing the document. All of the approaches above have the same limitation that they will not work for retrieving elements within a currently loaded, modified but unsaved document - Grif would not display the right element because of the shift of line numbers or changes in the tree structure. This issue is considered below (see section 8).

In general this problem highlights a deficiency of SGML in not recognising the sequential order of document components as part of the logical structure of a document. If the SGML notation allowed for recording this information it could be used by both the Grif and Sigma systems for indexing LTUs or other document structures.

5.5 Encoding SGML Document Component Locations

The usual method for designating subtrees of a given tree is to use pointers towards nodes of the tree. An alternative is to designate a subtree or any collection of subtrees by the path from the root node of the document to the root node of the subtree. This way of addressing subtrees, called a *selection* path[3] has been implemented in Centaur, a generic programming environment developed at Inria (Clement and Hascœt, 1988). A *selection* is the path from

---

[3] This selection path must not be confused with the Path structure used in Intellitext and Grif and refered to in section 5.1.2; even though they both allow for keeping track of places in a graph, they correspond to completly different structures.

the top of the tree to the top of a subtree. The path to a particular child node from its immediate parent node is given by a number corresponding to the rank of that particular child node. A selection therefore consists of a sequence of numbers indicating the rank of each branch taken in a traversal of the tree from the root to the target node. For example, for the tree structure on Figure 5, the selection path for element g is given by 1.3.all, where "all" represents selection of the target subtree having the node g as its root.

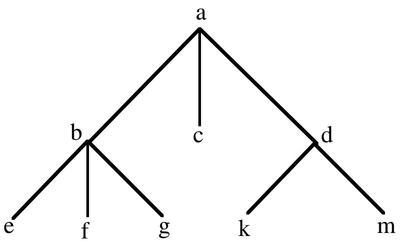


Figure 5.

In considering how the selection path must be encoded, it is necessary to consider the representation and coding/decoding efficiency of a path. In the present case, it was decided to encode the path as an integer. To facilitate this, the tree structure can be interpreted as a binary tree, with binary "0" representing a left branch and binary "1" representing a right branch. A total selection path can then be interpreted as a binary number that can be represented in its decimal integer equivalent. This method consistently handles both fixed-arity structures and lists of arbitrary length. For example, the tree of Figure 5 can be represented as shown on Figure 6.

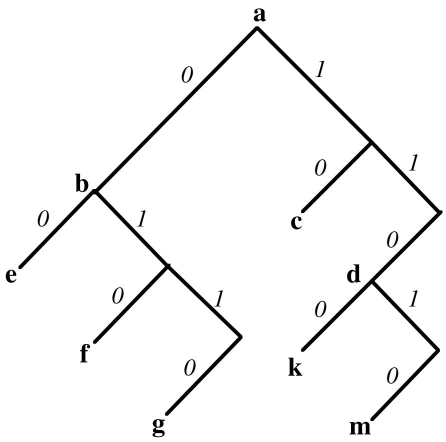


Figure 6.

In this case, the selection path to node g is 01101, where the terminal "1" stands for "all". The binary string is reversed for ease of processing, ensuring an integer value beginning with "1" (to accurately encode the tree depth to the target), so this example is encoded as the decimal number 22.

With this addressing method we can answer questions such as "are these two subtrees children of this third node" more easily than using pointers. Other advantages and an implementation for such addressing may be found in Clement and Hascœt (1988).

5.6 Interfaces issues

The interface between Grif and Sigma uses existing and clearly defined facilities: Sigma is made of a set of C functions while Grif offers an API (Application Interface) for interfacing with external applications (via C routines).

6 Implementation

The integration of the Grif and Sigma systems has involved the development of appropriate Grif functions that interact with generic sigma functions via the Grif Application Programmer Interface (API). For the prototype system described here, the generic Sigma routines were not modified, and inter-program data exchange occurs via files. The structure of the resulting system is shown in Figure 7.

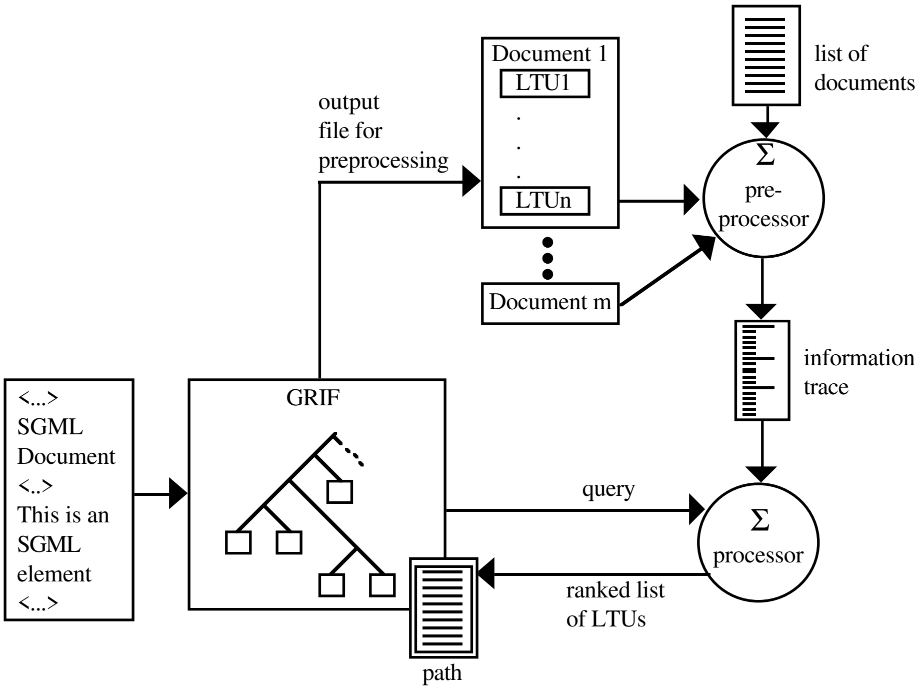


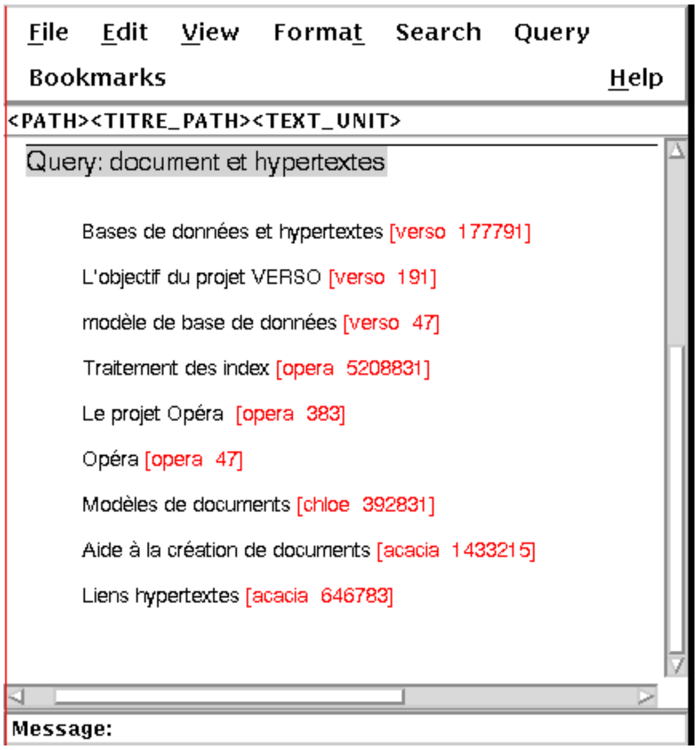
Figure 7. Grif-Sigma Interaction.

6.1 Indexing

Before retrieval is possible, it is necessary to generate a Sigma index for the target files. Currently preprocessing is an independent process. Each document to be indexed is loaded into Grif, where it is represented as a Grif tree structure; then, each sequential component (corresponding to an LTU) of the document is written without SGML markup into an external file for preprocessing. Within the external file each element is preceded by a line indicating its encoded position in the Grif tree structure (see section 5.5). The Sigma preprocessor uses another external file containing a list of all documents to be preprocessed. The preprocessor creates a Sigma index file containing an information trace for each LTU of each named document. The document name and Grif tree location of the associated LTU is recorded with each information trace within the index.

6.2 Searching and Displaying Answers

Once an index has been created, it is possible to initiate search and retrieval from Grif. The Grif user is presented with an interface for specifying a search query, or a text substring of a current element can be selected and used as a query in order to implement dynamic and automated hypertext linking (as described by Lindley et al, 1994). The query is passed via the Grif API to the Sigma processor, along with the name of the index. The Sigma processor generates an information trace for the query and then uses the Sigma index to generate a list of LTUs (ie. text components) ranked in decreasing order of similarity to the query. The list includes the title of each element (which may be a section title or the first line of the element), the name of its parent document, and the position of the element in the tree structure used by Grif to represent the document. The list is passed back to Grif via the Grif API. Grif then displays the list to the user using a document type called a *path*, which represents a list of references to objects of interest (eg. LTUs) in a database. An example of query-answers is shown in Figure 8. The query "Documents et Hypertext" (in French) has been input by the user in this place (the title of the Path document) and the titles of retrieved elements are listed underneath. The name between brackets is the name of the corresponding document while the figure is the encoded position within the document that should not be displayed to the end user.



Query: document et hypertextes

Bases de données et hypertextes [verso 177791]

L'objectif du projet VERSO [verso 191]

modèle de base de données [verso 47]

Traitement des index [opera 5208831]

Le projet Opéra [opera 383]

Opéra [opera 47]

Modèles de documents [chloe 392831]

Aide à la création de documents [acacia 1433215]

Liens hypertextes [acacia 646783]

```
Query: documents électroniques

opera:
                 Opéra                                (32.59)   3.
                 Grif, Didot, Modèle de Document      (21.06)   5.
                 Le Projet Opéra                      (17.26)   8.
                        Bi-Localisation               (4.42)    8.2.1.
                 Outils d'édition                     (4.12)    9.2.2.
                 Traitement des Index                 (8.25)    9.2.5.5.1.
                 Actions Internationnales             (3.96)    12.1.3.
                 Enseignement                         (9.57)    13.1.1.
verso:
                 modèle de base de données            (16.60)   5.
acacia:
                 Acquisition à partir de documents    (3.40)    8.1.5
                        Aide à la création de documents (4.28)  8.1.5.4.2
chloé:
                 Environnemenrts de programmation     (5.34)
                 Modèle de documents                  (9.79)    8.1.9.
```

Figure 9. a more structured display of answers

In Figure 9, the names of LTUs from the same documents are kept together and ordered according to the pre-order traversal of the tree. Indentation indicates the relative level in the tree when it applies. The last column in the figure gives the decoded path of the node.

For helping the user to select the more relevant parts of a document, the similarity measure may be provided (between parenthesis in the example). The similarity measure is not currently an output of the Sigma standard interface.

The issue arises of ordering a priori unordered levels, in particular the documents themselves. As suggested in (Fuller and al, 1992), the order could be based on the highest ranked nodes within each document, or the number of relevant elements, or some combination.

7 Evaluation

Evaluation of Effectiveness

Advantages:

- the approach developed here has not required the modification of any of the Sigma software, and has allowed a very neat integration via standard interfaces

- intercommunication via external files supports the further development of Sigma as a retrieval server, and the Grif-Sigma integration via interprocess protocols

Limitations:

- use of unmodified Sigma routines means that:

    - an external file must be used and maintained to list the source documents to be indexed

    - there is no incremental indexing or modification of individual information traces within an LTU, hence the index must be completely regenerated when additions or changes to documents are made

- indexing is done using the Grif representation; this will not allow the retrieval engine to be used from other applications using the SGML file as a standard.

- displaying the list of retrieved elements (eg. a list of section titles inside the full document) will be possible if the documents that contain these elements have been loaded into Grif, or this information has been added to the index. The first solution would use memory space and time, while the second would use secondary storage space. The current solution uses the source document generated by Grif, which is not longer needed by the integrated system after indexing for any purpose other than for the Sigma system retrieving titles.

Evaluation of the Prototype Design

It was not our intention here to evaluate Sigma as a search engine by comparison with other search engines. Rather we wanted to evaluated the influence of indexing granularity upon the *precision* and *recall* of answers. Grif-Sigma is a good platform for easily modifying the level of granularity of indexing.

We made series of test using a subpart of the Inria Activity Report 1993, a large set of French SGML documents. Some example queries for this collection are:

- what are the teams working in “hypertexts and electronique documents“?
- who has contracts with a particular industrial partner?
- what are the activities of a particular person?
- etc.

It appears that:

- when the granularity is small (indexing of sub-sections at any level), there is a possibility that the number of retrieved elements is greater than the design maximum, so some relevant documents are not selected.

- when the granularity is large, the precision of answers decreases since the weight of large elements overcomes the small elements. The weighting function used in the similarity measure must therefore be carefully adjusted for balancing its effect.

An interesting alternative would be to adjust the granularity level according to the size of the actual elements.

It should be noted that the queries described above were typically not well addressed using the built-in index attached to the Activity Report on the Web. For example the index does not retain any cross referenced information, or a list of industrial partners. Moreover, natural language queries and the display of the list of answers provides a much better interface than activation through an alphabetic index.

8 Connected Work

- Information retrieval for large SGML document databases is the subject of much current research and System development (MAESTRO, SIM, PAT) . The main focus is on the development of a query language for searching documents according to the structure of the documents and their textual content (Christophides V. and al., 1994), (Arnold-Moore T. and al., 1994), the storage of documents in databases, and the creation of appropriate indexes. Most systems have been designed for read-only documents, with the exception of SGML/Search (AIS Berger-Levrault, 1993), and offer their own user interfaces. The issue of connection with SGML clients and especially an authoring SGML tool has received little attention to date.

In most cases, the retrieved elements are displayed as if they were independent documents. In (Fuller M. and al., 1992), a partial table of contents is built showing the position of the retrieved elements in a full document. The text of a retrieved element is displayed without context, but browsing around is possible by requesting the database to display sibling or parent elements. No global view of the document is displayed.

In SIM, an HTML browser is used to display the document that is translated on the fly into HTML. The words that are part of the query are highlighted in the text by means of appropriated tags generated in the HTML source. Retrieval of document fragments is not currently possible.

Using Grif as a client in place of an HTML browser would work as well by generating a temporary SGML (or HTML) file, containing appropriate tags around the retrieved element. However this approach was not possible for the integration with Sigma since Sigma does not retain enough information for generating this SGML file.

- Hypertext Systems like I^{3}R (Croft and Turtle, 1989) or the medical handbook system described in (Frisse, 1989) use query-based retrieval for providing a list of ranked nodes for further browsing. Browsing along static links is then complemented or replaced by a direct search for information, providing a dynamic linking to relevant nodes. Dynamic linking can generally result from the interpretation of scripts, or link-resolving components, as proposed in (Tompa and al, 1993), where the interpretation of the link depends on the application; a traditional information retrieval function is only one example of such an application.Retrieval techniques and similarity measures between nodes have also been used for automatically creating links between anchors and nodes or between nodes.

In most hypertext systems the matching between the query and the nodes is done according to some information attached to the nodes (attribute contents, predicates). Although some systems also provide full-text retrieval, the main difference in our approach is that the retrieved elements are neither full nodes nor text strings in nodes, but logical elements as part of larger documents that provide natural contexts for further browsing.

- Hytime (DeRose S. and Durang G, 1994), the first official standard for describing the structure of time-based hypermedia, provides locators for making links to a specific portion of a tree designated by a path called here a *treeloc*. The specification of a *treeloc* is, for example:

<treeloc id=part1>

<marklist> 1 2 4 3 2 </maklist>

</treeloc>

1 2 4 3 2 is the path into the document and each number specifies the child number.

Optimisation of the path code is not an issue here since it is part of the Hytime or SGML document and not part of an index.

- The use of paths for referencing nodes is well known in the field of databases for hierarchically structured data. In Maestro the retrieved elements are referenced by their paths into the document collection; for example: Group_name Document_id Element [rank] Element [rank]. For the connection between the O2 Database [Bancilhon F. and al., 1992] and the World Wide Web, when an element of the database is set as the target of link by an external document, the Url contains the path from a persistent root to this element. In this case the encoding of the path has the form of a query into the database.

The storage of the node's ancestors in the node has also been proposed as more efficient technique for retrieving a subtree of a tree stored on a disc (J. Zobel and al., 1991). These ancestors can be stored by encoding into the node identifier, saving disk storage space and index lookup time.

If such path encoding is used at the database level, it becomes very convenient to use it for communicating with a client application as proposed in this paper.

9 Extensions

Searching Documents Undergoing Modification

At this point we have not solved the problem of the consistency of the index attached to documents being edited. Changes to text content will only affect the index entry for changed text, and changes to the text structure will only affect path names in the subtree under the parent node above the point of structural change. Hence accessing changing documents can be addressed by:

- incremental indexing and localised index modification by Sigma
- substitution: of selection paths in the case of structural changes or of information traces in the case of content changes

In both cases, a trace can be kept of all modifications to the document since the last retrieval request, using a *modification path.* (as in Clement and Hascœt, 1988). A modification path has the same structure as a selection path, but only records information about which positions have changed (due to insertion, deletion, or modification); all the elements containing these positions must be reindexed. A modification path can also be used to dynamically recalculate the positions of altered elements.

It may be argued that the use of labelled nodes will suppress the problem of updating subtree addresses since the corresponding addresses can be recalculated when loading a document. However, this approach has three main drawbacks: the labels have to be kept in the SGML source, the labels do not provide a way of keeping an incremental trace of the modifications (useful for lazy incremental reindexing), and the approach does not offer an easy way of checking the inclusion of different elements.

Generalised Indexing of SGML Documents

The prototype described here is limited to searching SGML documents that have been preprocessed by Grif. It would be useful to be able to use Grif to search and display parts of SGML documents that have been indexed by Sigma independently of Grif.

A distinction must be made between:

- searching within the scope of a single document (*internal* search) using a *local index*. The document can be loaded into Grif, the local index generated, and the Grif tree position used as the path name to access a target element.
- searching outside a currently loaded document (*external* search). The search can be conducted using an external index, with pathnames to retrieved objects specifying their location on the system. The parent document of a retrieved object would be loaded into Grif, and the target object found using the Grif internal search facilities.
- searching across the currently loaded document and among external documents (*mixed* search). This involves a combination of the previous two cases.

A mixed search suitable for use during editing would be to pre-index all the documents from the SGML tagged files, while building a dynamic local index for all documents being edited. This approach is used for the prototype described here, but without dynamically updating documents currently undergoing modification.

10 Conclusion

We have presented the integration of an SGML authoring and display system with a textual retrieval tool. The integration was implemented using the Grif API to extend the user interface and call Sigma routines.The integration architecture is general enough to be used for integrating Grif as a client for another Information Retrieval Tools or SGML document databases. The ranking or ordering of answers can also take advantage of their relative positions within the full document.

There are a number of areas of potential ongoing development work related to the work of the CSIRO Division of Information Technology, or more specifically related to the integrated Grif/Sigma system. These include:

- the incorporation of dynamic scoping within the retrieval process (eg. within a document, or across arbitrary document subsets)
- filtered retrieval
- Distributed search and information retrieval, including commensurable ranking

of locally retrieved elements, and conventions for the representation of document metaknowledge (for example taxonomies).

CSIRO is curretly studying a multi-agent architecture for distributed search and information retrieval (Lindley, 1995).

These extensions require a more modular organisation of the Sigma Index table(see tha Annexe), mainly for addressing the incrementality and global ranking issues. Taxonomies could also be described in SGML, so Grif-Sigma could be used for editing, displaying and searching for taxonomies.

Acknowledgements

The authors are thankful to Rosemary Irrgang for her work on the information retrieval engine and tests on the the weighting function used in the similarity measure.

References

AIS Berger-Levrault (1993), *SGML/Search, Description du Langage*, Internal document, 34 avenue du Roule 92200, Neuilly sur Seine,

Arnold-Moore T., Fuller M., Lowe B., Thom J. and Wilkinson R. (1994), "*The ELF data model and SGQL query language for structured document databases*", Technical report CITRI TR/94-13, Collaborative Information technology Research Institute, Melbourne, Australia.

Bancilhon F., Delobel C. and Kannelakis P., *Building an Object-Oriented Database System - The Story of* O_2, Morgan Kaufmann, 1992.

Botham A., Fuller M., Mackie E., Sacks-Davis R. and R. Wilkinson, "An SGML-based hypertext information retrieval system", *In Proceedings of the Fifjteenth Australian Computer Science Conference*, Hobart, Australia, 1992.

Clément D. and Hascouet L."Designating subtrees in Centaur", GIPE, Esprit Project No.348, January 1988.

Christophides V., Abiteboul S., Cluet S., Scholl M., "From Structured Documents to Novel Query Facilities3, in SIGMOD'94, P. 313-324, Minneapolis, Minnesota, May 1994.

Croft W.B. and Turtle H. , "A retrieval model for incorporating hypertext links", *Proceedings of Hypertext'89*, Pittsburgh, 5-8 November 1989.

DeRose S. and Durang G, *Making Hypermedia Work, A user's Guide to Hytime*, Kluwer Academic Publishers, Boston, Dordrecht, London, 1994.

Fuller M., Mackie E., Sacks-davis R. and Wilkinson R., *Coherent answers for a large structured Document Collection,* RMIT and the University of Melbourne, Technical Report 69, December 92.

Frisse M.E. 1988, "Searching for Information in a Medical Handbook", *Communication of the ACM*, 31, 880-886, 1988.

Grif.SA 1993, "*GATE, The Grif Application Programming Interface.*", Technical Documentation, Grif.SA, France.

Jansen B. and Bray G. 1993, “Context & Knowledge Types Vs Serendipity“, in *Proceedings of the 13th International Conference Avignon'93,* France, 1993, pp85-95

Jansen B. 1994, “Intellitext: An Environment for Electronic Books“, in *Editer les Manuscrips Completude et Lisibilité, Ecole Normale Superieure,* 5 rue d'Ulm, Paris, France.

Kumar V. R. and Lindley C. A. 1994 “Improving Decision Support Through Hypermedia“, *3rd ACM Golden West International Conference on Intelligent Systems*, Las Vegas, USA, 6-10 June.

Lindley C. A. 1995 “A Protocol and Architecture for Distributed Search and Information Retrieval“, not yet published.

Lindley C. A., Kumar V. R., Irrgang R. I. and Robertson J. R. 1994 “An Evaluation of Information Retrieval Methods and Semantic Network Processing for Automatic Link Generation in Hypermedia Systems“, *Second International Multimedia Symposium*, Perth, WA, 23-28 January.

Macleod I.A. 1990, Storage and Retrieval of Structured Documents, Information Processing & Managment, 26(2):196-208, 1990.

Mazur Z. 1994 “Models of a Distributed Information Retrieval System Based On Thesauri with Weights“, *Information Processing and Management*, 30(1), 61-77.

Quint V. and Vatton I. 1992, “CombiningHypertext and Structured Document in Grif“, in *Proceeding of the European Conference on Hypertext* (ECHT'92), eds., A. Rizk, N.Streitz, and J. André, 23-32, ACM Press, New York.

Teufel B. 1994“*Informationsspuren zum numerischen und graphischen Vergleich von reduzierten naturlichsprachlichen Texten*“ Informatik-Dissertationen ETH Zurich NR. 13. Verlag der Fachvereine Zurich .

Tompa F.W, Blake G.E and Raymond D.R, 1993, “Hypertext by Link-Resolving Components“, *Proc. of the Fifth ACM conference on Hypertext (Hypertext'93),* Seatle, 14-18 November, ACM Press.

Teufel B. and Schmidt S. 1988 “Full text retrieval based on syntactic similarities“, *Inform. Systems* 31(1).

Vercoustre A.M 1990 “Structured Editors-Hypertext Approach, Cooperation and Complementarity, *Proceeding of the International Conference on Electonic Publishing, Document & Typography* (EP'90), Gaithersburg, Maryland, PP.64-78, September.

Zobel J., Mackie E. and Sacks-Davis R., *Efficient Retrieval of Hierarchically structured Data,* RMIT and the University of Melbourne, Technical Report 91-5, April 92.

Annexe

This annexe describes a functionnal interface for Sigma that allows for incremental indexing from Grif and various strategies for ranking the retrieved elements.

1. Interface Functions

Sigma functions for global indexing:

Build_information_trace, parameters: index name, file list; - - a batch function

Sigma functions for local indexing:

Insert_LTU_trace, parameters: index name, text, reference to source (ie. doc name, path); – the text contains the title on the first line

Insert_doc_trace, parameters: index name, text, doc name; - - the text is the list of LTU text

Delete_doc_trace, parameters: index name, doc name;

Delete_LTU_trace, parameters: index name, reference to source (i.e path[4]);

Sigma functions for local searching:

query, parameters : local index name, filter, query string, similarity threshold
return : list of retrieved element (i.e list of paths, similarity mesure)

For querying documents that are currently edited with Grif. It may be reduced to the current document.

Sigma functions for global searching:

query_database, parameters : global index name, filter, query string, similarity threshold
return : ranked list of element (i.e path, title)

or

query_global, parameters : local index name, filter, query string, similarity threshold
return : list of retrieved element (i.e path, title, similarity mesure)

Grif API extension function:

ranking_elements, parameters : list of elements (i.e path, similarity mesure, title)
return : ranked list of elements (i.e path, title)

---

[4] here a path must be understood as both the name of the document and the encoded path withing the document;

ordering_elements, parameters : list of elements (i.e path, similarity mesure, title)
return : hierarchy of elements (i.e path, title)

2. Index table structures

Database_index : list of document_index;

Document_index : document name: string,
list of LTU_index;

LTU_index : path: string,
title: string;
list of (ngram, frequency);

Unité de recherche Inria Lorraine, Technopôle de Nancy-Brabois, Campus scientifique,
615 rue du Jardin Botanique, BP 101, 54600 Villers Lès Nancy
Unité de recherche Inria Rennes, Irisa, Campus universitaire de Beaulieu, 35042 Rennes Cedex
Unité de recherche Inria Rhône-Alpes, 46 avenue Félix Viallet, 38031 Grenoble Cedex 1
Unité de recherche Inria Rocquencourt, Domaine de Voluceau, Rocquencourt, BP 105,
78153 Le Chesnay Cedex
Unité de recherche Inria Sophia-Antipolis, 2004 route des Lucioles, BP 93, 06902 Sophia-Antipolis Cedex

Éditeur

Inria, Domaine de Voluceau, Rocquencourt, BP 105, 78153 Le Chesnay Cedex (France)

ISSN 0249-6399