The Use of Information Retrieval Tools in Automatic Document Modeling and Recognition

A. Belaïd and A. David

LORIA, Campus Scientifique B.P. 239
F-54506 Vandoeuvre-Lès-Nancy Cedex France
E-mail:{Abdel.Belaid, Amos.David}@loria.fr

Abstract

A combination of information retrieval (RI) tools and structural recognition (SR) tools are used for the automatic construction of a generic model of documents. The automatic construction takes as input a structured database of the documents which is converted into SGML, giving a logical representation of their content. the RS tools are applied to the postscript representation of the documents to produce their physical characteristics. We have coosen bibliographic reference database to illustrate our proposals.

1. Introduction

Document analysis and retrieval require an *a priori* knowledge on a class of documents to be analyzed and managed. This knowledge is represented in form of a model. Manual extraction of this knowledge is very tedious and its automatic extraction very difficult because of the heterogeneity of documents. Document modeling implies its representation as containers and as information contents.

Considering documents as containers, the difficulty of modeling lies more in the extraction of the components within the document image than in the definition of the document's structure. In fact, document components are often related to a hierarchical decomposition of the document and can be associated with its *structure components*. Formalisms such as SGML, XML or DSSSL exist today for different kinds of document. These formalisms reveal the structural organization of the documents. The existence of structured document databases constitute a real help for the constitution of such *structure models*. This is the case for important bibliographic references now distributed and described in a particular format [1, 2]. This format can be exploited by digital library tools like Balise1 for a rapid generation of a structure model for the references.

Considering documents as information contents, modeling is more difficult because the information that can be extracted depends on each application and cannot be so easily generalized as for the containers. Since all the possible information contents cannot be specified in the model, the common method used consists in statistics expressing frequencies and co-occurrences [3] between some *keywords*.

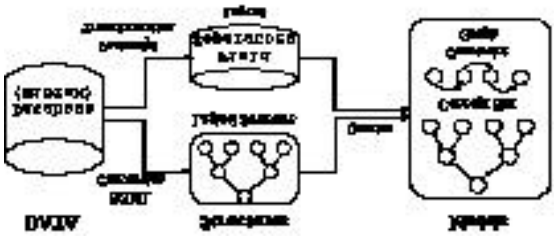
In our application, we use an important collection of nomralized bibliographic references in BIBTEX format. In this application, the *logical structure* of a document is directly derived from BIBTEX. The physical delimiters and the relationships between containers are extracted from a postscript analysis of the bibliographic references obtained by LaTeX. Two types of content descriptions were proposed in two applications : (1) by giving some information on a field content in terms of presence of typographical elements or some keywords (eg. initials and connector for authors, empty words for the title, etc.), (2) by representing all the instances of the sub- fields linked by co-occurrences.

1. System overview

Figure 1. gives an overview of the model constructionThe database is converted into SGML format (called logical structure) that can be used by tool packages for item extraction and statistics In order to obtain the items' separators, a postscript file of the database is generated and transformed into a more simple SGML format. The SGML format represents only text and typographic style (called physical

1 A commercial product by Berger-Levrault (AIS)

structure) of the references. Then, the physical and the logical structures are matched in order to obtain the physical separators (typography and textual separators). processd.



. **System overview**

Statistics of the occurrence and co-occurrences of a reference components can be calculated at the end of the process by using some specific retrieval and analysis tools.

Different kinds of models can than be built from these information files such as either a detailed structure hierarchy or a more concrete graph more close to the layout Structure.

The SGML tags make the extraction of field names easy. The result file is enriched by the use of LEX in order to point out some sub-fields non given directly in the initial format (e.g. the list of authors, the title words, the conference title words and the editor names). From this format, the structure hierarchy (fields and sub-fields) is extracted. One of our tools is applied on BIBTEX format to produce the field or subfield surrounded by specific tags. This procedure has been applied to 5400 bibliographic references. The execution time is about two minutes without any optimization of the algorithm.

0.1 Separator Extraction

In order to have at our disposal all the possible optional fields, we generate automatically a fictitious base using a fictitious reference structure. In this base, the field contents are field names. This reference has been used for the model construction. Each reference is then formatted in Postscript by BIBTEX/LaTeX and converted into SGML. Then, all that is needed is to search for the field contents in order to locate them, and deduce the separa
torstors.

Figure 3. represents the result of the application of LaTeX in plain printing style on the fictitious reference of Figure 2.. It can be noticed that some fields are missed such as "editor" and "number". Also from the example, the separators found are ". " between AUTHOR and TITLE, " In </Times-Roman><Times-Italic>" between TITLE and BOOKTITLE, etc.

.

```
@InProceedings{order1,
        author          = “AUTHOR”,
        title           = “TITLE”,
        editor          = “EDITOR ”,
                        booktitle       =
“BOOKTITLE ”,
                        volume          =       “
VOLUME ”,
                        number          =       “
NUMBER ”,
                        series          =       “
SERIES ”,
                        pages           =
“PAGES ”,
                        address         =       “
ADDRESS ”,
                        month           =       “
MONTH ”
```

. **Fictituous reference**

<Times-Roman>AUTHOR. TITLE. In </Times-Roman><Times-Italic>BOOKTITLE</Times-Italic><Times-Roman>, volume VOLUME of </Times-Roman><Times-Italic>SERIES</Times-Italic><Times-Roman>, pages PAGES, ADDRESS, MONTH YEAR. ORGANIZATION, PUBLISHER, NOTE.</Times-Roman>

. **Representation of the fictitious refence**

0.1 Content analysis

We have developed some software tools for indicator calculations that can be used during document analysis. Using these tools on a database of bibliographic references, we can provide some general indicators in form of statistics of the database [3]. We present in this section the document format that we take as input, the main transformations on this input format and the reasons for these transformations.

We take as input a set of documents represented in SGML format. A document and its attributes are separated by a beginning and an ending markers. For example a document is enclosed within <doc> and </doc> markers, and each document is composed of the following attributes :

- *reference* enclosed within <ref> and </ref>
- *authors* enclosed within <aut> and </aut>
- *collection title* enclosed within <col> and </col>
- *keywords* enclosed within <mcl> and </mcl>
- *title* enclosed within <tit> and </tit>

In order to facilitate the processing of information on these documents, each of them is represented as an object.

The method we use for transforming the SGML format into the object representation is by attaching a set of instructions to each marker (beginning and ending markers).

When the ending marker for a document is encountered, all the values of all the object attributes will have been attached to the object. The </doc> marker invokes the book instance method for the generation of inverse lists. In

our application, we generate an inverse list for each attribute that can be used for request formulation and for the calculation of indicators.

An inverse list can be described as follows : given a value of an attribute, the inverse list of that attribute gives the objects in which the given value is assigned to the attribute.

0.1 Information analysis

Document access and analysis is based on a method we call *classification* with *constraints*. The result of any type of analysis provides the distribution of values assigned to the attributes.

Classification allows a user to specify the attributes to use for analysis. Attributes can be combined in three ways :

- *Only one attribute* : Let *L* be a set of all assigned values to the attribute *A* in the database. Specifying only one attribute, say *A*, for analysis, we suppose that the user wants to obtain the distribution of each element of *L* over the database objects. In other word, the user wants to obtain how many times each element of *L* is assigned to the attribute *A*. For example, if the user specifies *Author* as the only the attribute for analysis, we provide the list of all authors in the database and how times they are assigned as authors.
- *Two same attributes* : Let *L* be a set of all assigned values to the attribute *A* in the database. If two attributes, say *X* and *Y*, are specified for the analysis and that *X = Y*, then we suppose that the user wants to know the distribution of the co-occurrence of any two values of *L* for attribute *X* (or *Y*). For example, if the user specifies *X = Author* and *Y = Author* and *L* is a set of all the authors of the database, then we provide all the co-occurrence of any two authors that have written together and how many times they exist as co-authors.
- *Two different attributes* : Let *L* be a set of all assigned values to the attribute *A* and *R* a set of all values assigned to the attribute *B*. If the user specifies *A* and *B* (*A* different from *B*), then we provide all the existing co-occurrences of the elements of *L* and those of *R* in the objects of the database. For example, let *A = Author* and *B = Keyword*. In this case we provide the frequencies at which each keyword in the database is used by each author.

These three types of classification can be combined with constraints that indicate the conditions that should be satisfied by the objects to be produced as results.

Constraints can be represented as
$C_1\ opb_1\ C_2\ opb_2\ ...\ C_n$ where
C_i = *{attribute, opc, value}* and
attribute is one of the document attributes,
opc is one of the comparison operators *[=, <=,>=, etc.]*
value the attributes's value
opb_i is a boolean operator *[and, or, except, etc]*

The C_is correspond to what is called criteria in information retrieval systems (IRS). In IRS, the criteria can be combined using boolean operators like *or, and, except*, etc. We use only the *and* boolean operator for combining criteria.

0.1 IR and indicator calculation

The algorithm we use is based on the use of symbolic indexed arrays and the facility of obtaining an object's attribute values in object representation. The symbolic indexed array also allows the assignment of any type of value including lists.

Supposing that the user specifies *(attribute1 = Author) and (attribute2 = Author)* as his classification requirement the algorithm is for processing this request is as follows :

```
For each o in the list of objects {
    put the authors of object o in A
    While there are more than one authors in A {
        put the first author in f
        put the remaining authors in A
        For each g in the list of authors in A {
            If the element (f,g) exists in the indexed array
            then append the object o to the element
            else create an element (f,g) in an indexed array
            } } }
```

Supposing that the user specifies *(attribute1 = Author) and (attribute2 = Keywords)* as his classification requirement the algorithm is for processing this request is as follows :

```
For each o in the list of objects {
    put the authors of object o in A
    put the keywords of object o in B
    Fore   each f in the list of authors A {
        For each g in the list of keywords in B {
            If the element (f,g) exists in the indexed array
            then {
                append the object o to the element
            } else {
                create an element (f,g) in an indexed array
            } } } }
```

The final result of the processing is presented in form of a list of the co-occurrences of the values in the two specified attributes. The list is presented in decreasing order of the number of objects assigned to each co-occurrence. If the user specifies (Author = Amos) as constraint, we use the inverse list to obtain the list of objects in which Amos corresponds is an author. Only the objects that satisfy the constraints are included in the final result. The comparison operators we use are

= *equal*
< less than
<= less than or equal to
> greater than
>= greater than or equal to
!= different from
* contain

The inverse lists are used for processing requests with only one attribute and the constraints. The object programming language we use provide symbolic indexed arrays which facilitates the management of inverse lists.

In the following, we present two examples of the results (or indicators) we obtained from the bibliographic references of our research laboratory. The bibliographic reference database contains 5400 references., (a) gives the frequencies of co-occurrence of keyword. We can see that intelligence_artificielle and système_expert takes the lead. (b) show the co-occurrence of an author and the keywords. This gives the keywords that an author uses most frequently.

(a) request -a1 keyword -a2 keyword

{ 92 <intelligence_artificielle><système_expert>}
{ 67 <apprentissage><intelligence_artificielle>}
{ 65 <intelligence_artificielle><représentation_connaissance>}
{ 54 <apprentissage><réseau_neuronal>}
{ 51 <représentation_connaissance><système_expert>}
{ 49 <intelligence_artificielle><langage_naturel>}
{ 45 <apprentissage><système_expert>}
{ 39 <intelligence_artificielle><robotique>}
{ 37 <développement_logiciel><génie_logiciel>}
{ 36 <robotique><réseau_neuronal>}
{ 35 <architecture_parallèle><parallélisme>}

(b) request -a1 author -a2 keyword

{ 13 <Nye,_Adrian><x_window>}
{ 12 <Rozenberg,_Grzegorz><réseau_pétri>}
{ 10 <O'Reilly,_Tim><x_window>}
{ 9 <Dautray,_Robert><calcul_numérique>}
{ 8 <Rozenberg,_Grzegorz><concurrence>}
{ 7 <Nye,_Adrian><fenêtrage>}
{ 6 <Diday,_Edwin><analyse_donnée>}
{ 5 <Habrias,_Henri><conception_système_information>}
{ 4 <Utkin,_Vadim_I.><commande_robuste>}
{ 3 <Shyamasundar,_Rudrapatna_K.><informatique_théorique>}
{ 2 <Collongues,_Alain><conduite_projet>}

1. Model Construction

Using these tools, we have developped several models for different kinds of references such as bibliographies, library catalogues and references in books and scientific papers. The resulting models depend on the structure of interest and on the strategy used for the recognition.

For the automatic recognition of scientific references, we used a concept network [4]. The structural part is similar to the generic structure described in section 2.1. Attributes are assigned to each node. We distinguish two types of attribut : *fixed*, i.e. computed during net creation, and *dynamic*, i.e. evolving during the treatment. The former corresponds to the node *name*, its *conceptual importance* and its

activation ratio. The latter gives the *activation value*. We will give in the following a short definition of these terms.

- **Node name** indicates the node category (such as *field, separator,* and *instance)* . This is obtained from the object description corresponding to the documents of the database as described above..
- **Conceptual importance** (CI) gives a value on its importance for the problem to be treated. It is obtained from the database as frequency analysis as described above.
- **Activation rate** (AR) indicates the interest to keep the current node activated or not during the system cycles. This is obtained by co-occurence analysis of the database as described above.
- **Activation value** (AV) allows to reveal the more important nodes to be considered during processing

As presented above, the co-occurrences are computed between terms in the same field or in different fields of the objects. The activation value of the separators are obtained from their co-occurences with the separated fields.

The importance of the approach we use for obtaining this attributs as presented in section 2. is that it provides a global analysis of the database.

1. Conclusion

We have shown in this paper our conceptual approach towards the automatic construction of a document model. We first presented how documents can be viewed conceptually. The resulting model highlights the most important elements to take into account during the process of analysis of a new document. We also illustrated with examples in real applications how the generic model can be constructed and used for document analysis.

Our objective is now to integrate this system into an information retrieval system (IRS). The resulting IRS will provide the necessary tools for document representation, provide direct access to the documents and provide fonctions for obtaining general indicators of the document base. Furthermore, we believe that IRS should contribute to the improvement of the generic model. A first example is the use of users' recommendations to take care of new information needs such as the integration of unspecified fields. A second example is the use of the recognized references to complete the existing databases. For this, there is need to verify if the reference exists. We can apply IRS tools to verify this existence of related references by request specifications on reference properties.

1. References

[1] F. Parmentier, Logical Structure Recognition of Scientific Bibliographic References, International Conference on Document Analysis and Recognition, Vol. 2, p. 1072-1086, ULM, Germany, 1997.

[2] A. Belaïd, Retrospective Document Conversion: Application to the Library Domain, International Journal on Document Analysis and Recognition, IJDAR, Vol. 1, N. 3, p. 125-146, 1998.

[3] A. David, Modélisation de l'utilisateur et recherche coopérative dans un sytèmes de recherche d'informations, ISKO '97 (Internation Society for Knowledge Organisation), Lille, France, 1997

[4] A. Belaïd, J. C. Anigbogu and Y. Chenevoy, Qualitative Analysis of Low-Level Logical Structure, In Electronic Publishing, Document Manipulation and Typography, Eds. Ch. Hüser, W. Möhr and V. Quint, Vol. 6, Issue 4, 1994.

[5] F. Parmentier, Spécification d'une architecture émergente fondée sur le raisonnement par analogie: Application aux références bibliographiques. PhD Thesis, University of Nancy HP, France, June 1997

[6] B. Michelet, l'analyse des associations, PhD Thesis, University of Paris VII, UFR of Chemistery.