

# Document Structure Matching for Heterogeneous Corpora

Ludovic DENOYER

Guillaume WISNIEWSKI

Patrick GALLINARI

Université Paris 6

LIP6

8 rue du capitaine Scott  
75015 PARIS – France

Ludovic.denoyer@lip6.fr

wisniewski@poleia.lip6.fr

Patrick.gallinari@lip6.fr

## ABSTRACT

Querying heterogeneous XML document collections is an open problem. This will require building some sort of correspondence between the DTD of the different sources. We consider here the problem of matching the structure of XML documents from different sources. We introduce for that a stochastic structured document model and describe preliminary experiments performed on the INEX collection.

## Keywords

### 1. INTRODUCTION

For the 2002 and 2003 editions, the Initiative for the Evaluation of XML retrieval (INEX) has been based on a homogenous document collection of IEEE scientific articles. This XML collection corresponds to a single DTD which is the union of the different journals DTDs. In most practical applications of XML retrieval, collections will be heterogeneous with documents coming from different sources with different DTDs. Handling heterogeneous collections is then a new challenge for XML IR which is the topic of a new explorative track at INEX 2004.

Heterogeneity has been considered for a long time in IR but only for plain text documents. In the database community, querying heterogeneous sources has also been a major concern for many years in different application domains like data integration, data warehousing, web services, etc. This has been explored more recently for semi-structured data and XML databases. Schema matching, i.e. identifying the semantic correspondences between elements of two schemas – or DTDs for XML - has been identified as a key operation for this problem. This is the first step for constructing complete mappings between two representations. Automatic or semi automatic schema matching has motivated a growing number of academic works recently [1]. Generally, it is supposed that

different sources are available, each with a known schema. Different schema element characteristics are usually considered for the matching process: tag names in XML, data instance characteristics (special symbols, number of characters, etc), data type, metadata, etc. The match operator is often learned from labeled examples using some basic machine learning techniques. Test sets for evaluation are usually relational tables or semi-structured web data. Such sources usually have a poor textual content and meaningful tag names.

Some of these ideas from schema matching could be helpful for XML IR. The context is however different and specific IR solutions should be developed. For example, for XML documents, tag names often have a weak semantic and few if any associated descriptive metadata, different tags may denote similar objects (e.g. the “item” tags in INEX), most important in many cases the DTD need not to be known. The document tree could be rather large and the relative importance of structural and content information is different than for databases. In many cases, the structure information is poor. Although some structural query elements (e.g. authors, dates) should be considered strictly, many others should be considered as vague constraints

Solutions to be developed for querying heterogeneous XML collections will also have to establish some correspondence between the structural elements of the different documents. The work described here is a preliminary attempt towards this direction.

We focus on the matching of document structures for IR on XML collections. Our guess is that, like for schema matching in databases, this is an essential operation for developing IR systems for heterogeneous XML collections. As an example, suppose one has developed an IR engine for some specific collection. This IR engine will use a mediated DTD and/or predefined tag names corresponding to this collection. One possible solution for querying new documents in the same domain is to transform these

documents into this mediated format. This is the point of view we adopt here.

We propose a stochastic structured document model for this matching task. This model will be trained using samples from a collection expressed in a mediated DTD. After training, it is able to take as input a new document, and to output a structured representation of this document, were the document elements have been tagged according to the mediated format. A characteristic of this model is that it takes into account both the structure and the content of the collection documents.

The matching problem is introduced in section 2. The model itself is described in section 3, preliminary experiments on the INEX collection are presented in section 4.

## 2. DOCUMENT STRUCTURE MATCHING

The general matching problem we want to deal with is the following : given a mediated DTD which is used for querying a collection of XML documents, how to transform new documents in this mediated DTD so that they could be queried using the same search engine ? This transformation should respect the semantic of the document elements.

The document structure matching problem considered here is a simplified instance of this general problem. We want to associate each new document with the mediated format without changing the document tree structure. This transformation will associate each document tag with a tag of the mediated DTD, it will not merge nor cut document elements.

We will suppose that there exists a set of documents already expressed in the mediated DTD. When the latter is a specific collection DTD, this set will be the collection itself. Otherwise, it is supposed that a basic transformation has been manually defined for transforming an initial set of documents. For example for the INEX collection, different tags have been defined as equivalent.

The existence of such a training corpus is necessary when the transformation has to take into account the elements content. This is different from many databases applications where only the structural information (schema) is considered.

In the proposed approach, the DTD of new documents need not be known. The matching will be inferred directly from the structural and content information of the document itself.

## 3. DOCUMENT MATCHING MODEL

The proposed approach relies on learning a stochastic model of the documents on the training collection. Both a stochastic grammar of the document structure and a model

of the content information associated to each tag in a given context will be learned.

When a new document is processed, it is transformed into the most probable structured document representation according to the stochastic model.

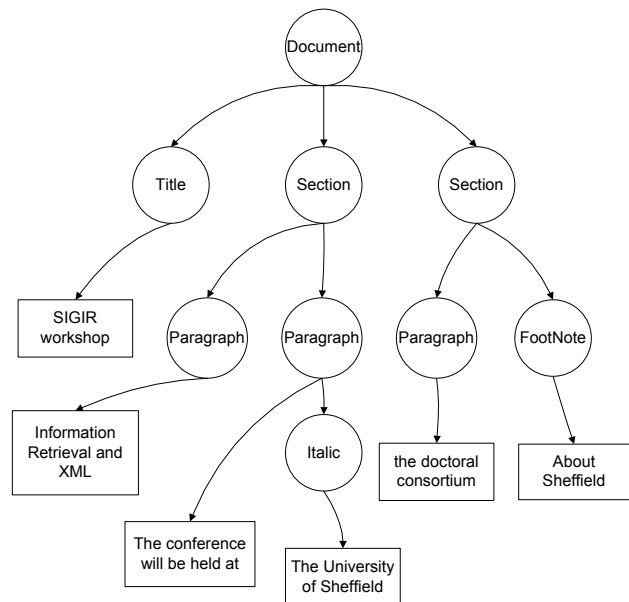
The effect of this transformation will be to associate each document element with a tag name from the mediated DTD. This association is not a simple tag  $\rightarrow$  tag correspondence, but depends on the structural and content contexts of the element. The stochastic model is able to learn simultaneously the transformations for different initial DTDs. When processing a new document, the most probable transformation will then be chosen.

### 3.1 STRUCTURED DOCUMENT

We consider a structured (XML) document as a tree. Each tree node  $n$  corresponds to a structural entity of the document and contains two types of information:

- A tag information ( $t_n$ ) from a discrete set of tags  $T$  defined by the mediated DTD e.g. (*part, section, title,...*).
- A content information ( $c_n$ ), we restrict ourselves here to textual content so that  $c_n$  is a sequence of words from a vocabulary  $V$ .

Figure 1 gives an example of such a structured document.



**Figure 1 : A tree representation for a structured document. Circle nodes correspond to structural information (tags) and square nodes to textual information.**



### 3.2.3 FINAL PROBABILITY

With these simplifying assumptions, the document probability (1) becomes:

$$P(d/\theta) = \prod_{n_i} \left( P(\text{children } g(n_i)/t_i, \theta) \prod_{k=1}^{k=n_i} P(w_{n_i}^k / t_{n_i}, \theta) \right) \quad (2)$$

The corresponding Bayesian network for the document of Figure 1 is shown in Figure 3.

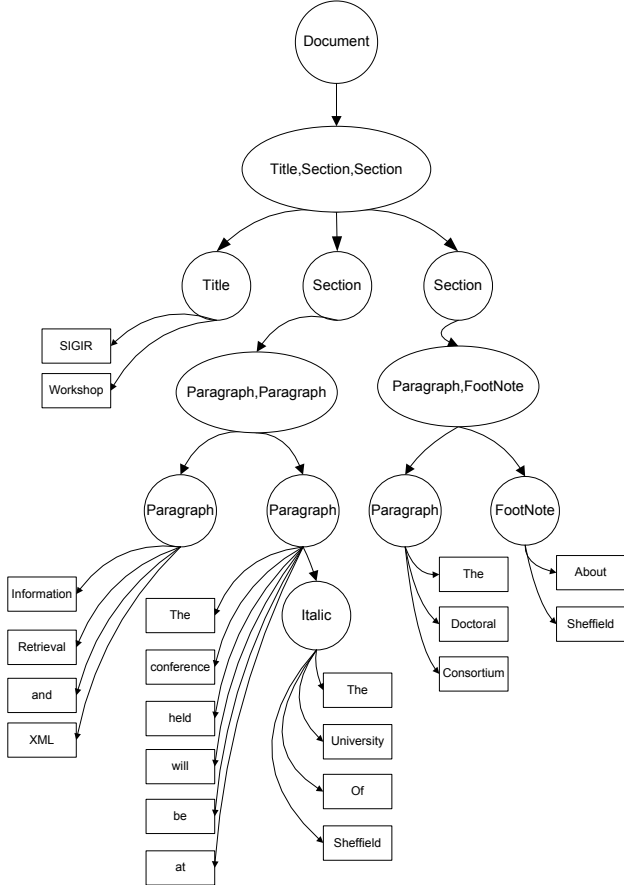


Figure 3: The final belief network corresponding to the structured document model

### 3.3 LEARNING

In order to learn the model, we have to estimate:

- $P(t_1, \dots, t_m / t, \theta)$  for each tag  $t$  and each possible sequence of  $t$  children tags  $t_1, \dots, t_m$
- $P(w/t, \theta)$  for each word  $w$  and each tag  $t$

over the training set  $D$  of structured documents.

For that we will maximize the log-likelihood of the structured document collection given the document model  $\theta$ :

$$L_D = \sum_D \left[ \sum_{n_i} \left( \log P(\text{children } g(n_i)/t_i, \theta) \sum_{k=1}^{k=n_i} \log P(w_{n_i}^k / t_{n_i}, \theta) \right) \right]$$

This amounts at solving the equation  $\nabla_{\theta} L = 0$  under equality constraint which ensure that probabilities sum to one. This system has an analytical solution. We used here the following robust estimators derived from this solution:

- $P(t_1, \dots, t_m / t, \theta) = \frac{N_{t_1, \dots, t_m}^t}{N^t} + \epsilon$
- $P(w/t, \theta) = \frac{N_w^t + 1}{N^t + |V|}$

where  $N_{t_1, \dots, t_m}^t$  is the number of nodes with tag  $t$  and with children  $t_1, \dots, t_m$  in  $D$ ,  $N^t$  is the number of nodes with tag  $t$  in  $D$ ,  $N_w^t$  is the number of times word  $w$  is appearing in a node with tag  $t$ .

### 3.4 INFERENCE

Once  $\theta$  is learned, a new document will be transformed into a structured document in the mediated DTD. For that we will have to find the most probable structure according to our model, i.e. to solve:

$$t_d = \operatorname{argmax}_{t_1, \dots, t_{n/d}} P(d/\theta) \\ = \operatorname{argmax}_{t_1, \dots, t_{n/d}} P(t_1, \dots, t_{n/d} / \theta) P(c_{n_1}, \dots, c_{n_{d/d}} / t_1, \dots, t_{n/d}, \theta) \quad (2)$$

Solving this equation is similar to the decoding step in Hidden Markov Models, except that we deal here with trees instead of sequences. The best tagged tree is obtained via a dynamic programming algorithm (not described here).

The complexity of this decoding step is  $O(|d| + |T| + |R|)$  where  $|d|$  is the number of nodes of the document,  $|T|$  is the number of possible tag labels and  $|R| = \operatorname{card}(N_{t_1, \dots, t_m}^t \neq 0)$  is the number of “derivation rules” learned. It is thus linear wrt  $|d|, |T|, |R|$ .

### 4. EXPERIMENTS

#### 4.1 INEX CORPUS

We performed preliminary experiments using the INEX collection of documents. This is a collection of XML

articles from 20 different journals and proceedings of the IEEE Computer Society. It is composed of about 12000 documents which represent more than 7,000,000 XML elements. Documents have been preprocessed using a stop-list and the Porter stemmer. All the words appearing in more than 20 documents were kept (about 27,000 words in all). The corpus was split in two, one part corresponding to the articles issued from the “*IEEE transaction ...*” journals (4233 articles) and the other part corresponding to articles from other IEEE journals (7874 articles). Training was performed on the second part and testing on “*IEEE transaction ...*” articles. This appeared as a natural split for the matching task since the two collections have different DTDs. Training allows learning the structure of the first collection. “*IEEE transaction*” articles were then formatted according to this learned structure. Note that only the content and tree structure information of the document to be formatted are used with this model, the tag names of the document in the test set are not used.

In a first series of experiments, we kept the 139 INEX tags for training. This provides an evaluation of our method for a complex matching task. In a second series, we used a much smaller number of tags keeping only the 5 tags *headings, paragraph, section, list, misc*. The idea here is that only a small number of tag elements were used for querying XML documents in past INEX. We therefore wanted to evaluate the model for such a transformation.

## 4.2 EVALUATION MEASURE

In order to evaluate our model on the test collection, we used two measures:

- recall over the nodes. This is the percentage of nodes correctly labeled by the model.
- percentage of documents from the test corpus with more than x% correctly labeled nodes for different values x.

For each series, we performed experiments by using:

- only the structural information:  $t_d = \arg \max_{t_{n_1}, \dots, t_{n_{|d|}}} P(t_{n_1}, \dots, t_{n_{|d|}} / \theta)$ . This is called the **Structure model**
- only the content information:  $t_d = \arg \max_{t_{n_1}, \dots, t_{n_{|d|}}} P(c_{n_1}, \dots, c_{n_{|d|}} / t_{n_1}, \dots, t_{n_{|d|}}, \theta)$ . This is called the **Content model**
- both content and structural information (eq.2). This is the **Full model**

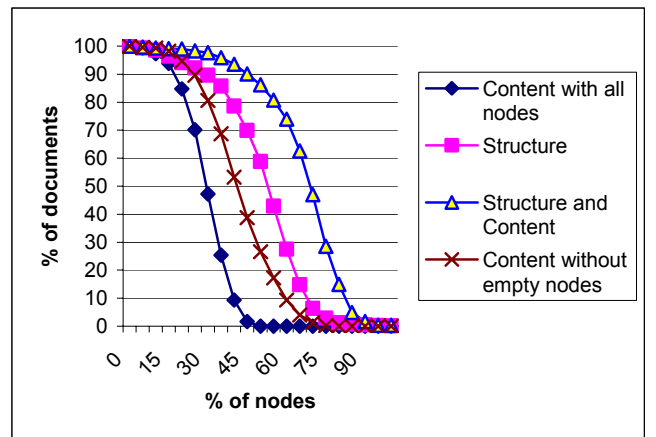
## 4.3 RESULTS

Recall for the different experiments are shown in Table 1. The full model is able to label correctly about 86 % of the nodes for the 5 tags setting and 65 % when using 139 tags.

Structure information by itself allows us labeling correctly respectively 73 % and 49 % of the tags. For the content model, we show two values which are respectively the recall over all nodes (Content with all nodes) and the recall over all nodes with a textual content. In the first case, the nodes with no content (about 25% of the nodes) will be considered as errors. The first value measures the usefulness of the content model for labeling the INEX corpus, while the second one gives an idea about its usefulness for a corpus with textual content at each node.

|                 | Content with all nodes | Content without empty nodes | Structure  | Structure and Content |
|-----------------|------------------------|-----------------------------|------------|-----------------------|
| Number of nodes | ≈5,000,000             | ≈3,500,000                  | ≈5,000,000 | ≈5,000,000            |
| 5 tags          | 58%                    | 81%                         | 72.90%     | 86.50%                |
| 139 tags        | 27.80%                 | 38.70%                      | 49.70%     | 65.30%                |

**Table 1: Recall for the structure, content and full models for the two experiments**



**Figure 4: Percentage of documents with more than % nodes correctly tagged (139 tags). For example, for the structure model, about 50% of the test documents have about 80% of their nodes correctly labeled.**

The curves in Figures 4 and 5 give the ability of the models for tagging correctly full documents. For the 139 tags setting, the full model has correctly tagged almost 90% of the nodes for about 40 % of the documents This is clearly not sufficient for a fully “automatic” labeling. The task is however difficult here since the number of tags is large. In such a situation, the stochastic model could be a component of a more complex labeling system. For the 5 tags experiments, this model has been able to correctly tag about 80 % document with an accuracy of about 85 %. This level of performance could be acceptable for many IR applications.

Both measures show that structure and content information are complementary and that considering them together increases significantly the performances.

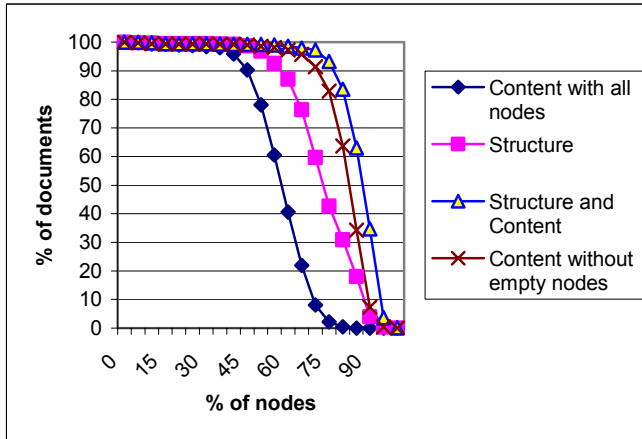


Figure 5: percentage of documents with more than % nodes correctly tagged (5 tags).

## 5. RELATED WORK

XML document matching as defined here shares similarities with schema matching in databases. Many schema matching approaches only consider schemas and make use of handcrafted rules for matching the elements of different schemas. Automatic methods have been developed for relieving this expensive process. [1] provides a survey and a typology of recent approaches to automatic schema matching.

The work closest to ours is probably that of Doan et al. [2]. They also considered transforming new DTDs into a mediated one using a 1-1 tag correspondence. They use a set of basic similarity measures and classifiers each operating on different schema element characteristics. These classifiers provide local scores which are linearly combined to give a global score for each possible tag. The final decision corresponds to the mediated tag with the highest score. Combining the different scores is a key idea in their approach. One of their classifiers operates on the element textual content. Up to our knowledge, this is the only system which takes into account this information for schema matching. They also make use of some basic relational information for scoring an element by considering the co-occurrence of parent-child tags. Compared to this approach, our model proposes a generative model of structured documents and focuses on structure and content which are the more important elements for XML IR.

Madhavan et al. [6] use many of the ideas developed in [2] for the more general problem of learning associations

between pairs of schema. They also compute different local scores which are then combined for the final decision. Other relevant but less related work is for example [3, 4, 5].

## 6. FUTURE WORK

This is preliminary work, the model still needs enhancements, more analyses and tests on other collections have still to be performed. The model could be enriched by using additional information (e.g. tag names, etc). However the proposed model has already shown very encouraging results on a medium size XML collection.

## 7. ACKNOWLEDGEMENTS

This work was supported in part by the IST Programme of the European Community, under the PASCAL Network of Excellence, IST-2002-506778. This publication only reflects the authors' views.

## 8. REFERENCES

- [1] Rahm E. and Bernstein P., 2001, A survey of approaches to automatic schema matching, *The VLDB Journal*, 10, 4, 334-350
- [2] Doan A., Domingos P., and Halevy A., 2003, Learning to Match the Schemas of Data Sources: A Multistrategy Approach. *Machine Learning* 50, 3, 279-301
- [3] Li W. and Clifton C., 2000, SEMINT: A tool for identifying attribute correspondences in heterogeneous databases using neural networks, *Data & Knowledge Engineering* 33, 1, 49-84
- [4] Berlin J. and Motro A., Autoplex, 2001: Automated Discovery of Contents for Virtual Databases. *Proceedings of COOPIS-01, Sixth IFCIS*, 108-122.
- [5] Melnik S., Garcia-Molina H., and Rahm E., 2000, Similarity Flooding: AVersatile Graph Matching Algorithm and its Application to Schema Matching, *18th International Conference on Data Engineering*
- [6] Madhavan J., Bernstein P., Chen K., Halevy A., 20003, and Shenoy P., Corpus-based Schema Matching, *Workshop on Information Integration on the Web at IJCAI 2003*.
- [7] Carrasco R. and Rico-Juan J., 2002, A similarity between probabilistic tree languages: application to XML document families. *Pattern Recognition*, 36,9, 2197-2199
- [8] Lewis D., Naive bayes at forty, 1998, The independence assumption in information retrieval, *European Conference on Machine Learning*.

