Recognising Document Components in XML-based Academic Articles

Angelo Di Iorio, Silvio Peroni,
Francesco Poggi, Fabio Vitali
Department of Computer Science
and Engineering
University of Bologna (Italy)
{diiorio,essepuntato,fpoggi,fabio}@cs.unibo.it

David Shotton
Oxford e-Research Centre
University of Oxford (UK)
david.shotton@oerc.ox.ac.uk

ABSTRACT

Recognising textual structures (paragraphs, sections, etc.) provides abstract and more general mechanisms for describing documents independent of the particular semantics of specific markup schemas, tools and presentation stylesheets. In this paper we propose an algorithm that allows us to identify the structural role of each element in a set of homogeneous scientific articles stored as XML files.

Categories and Subject Descriptors

I.7.2 [**Document And Text Processing**]: Document Preparation—*Markup languages*; I.7.5 [**Document And Text Processing**]: Document Capture—*Document analysis*

Keywords

DoCO, XML, document components

1. INTRODUCTION

In most disciplines, academic texts have established models of organisation and structure which are followed, more or less strictly, by all scholars and contributors. Some structures are shared across disciplines and capture very common objects of a text (such as tables, lists, references, front matter, etc.), others are specialised for specific disciplines (such as program listings in computer science works, epigraphs in humanities, medical histories in medicine, and so on).

Markup languages, and in particular XML vocabularies, provide authors with constructs to linearise these structural components. They often express the same components with different elements. For instance, the element *para* in DocBook (a semantic markup language for technical documentation [12]), the element *p* of HTML [7], the element *block* of the legislative XML vocabulary called Akoma Ntoso [2], refer all to the same concept of one of a set of vertically-organised containers of text often called a paragraph.

DocEng'13, September 10–13, 2013, Florence, Italy.

http://dx.doi.org/10.1145/2494266.2494319 .

The idea of this work is to shift the analysis of scholarly documents to a higher level of abstraction, dealing with their structural components - such as paragraphs, lists, bibliographic references, etc. - independently of the elements and the format of the markup language they are written in. Our solution is to use a general, strong and shared conceptual model for the description of components, and to match the elements of each XML languages to it according to the best interpretation of their structural semantic roles.

The correct identification of logical components could be used to generate lists and summaries (including tables of content, list of figures, etc.) automatically, to render the content in a Web browser window, and to provide full-scale converters (or, in the worst case, robust stumps open to further development). The interesting aspect is that the same tools work for both *well-known* and *unknown XML vocabularies*. The abstract representation of a document and its components can also be exploited to improve the comprehension of its content, as remarked by [4], and to build *Semantic Publishing* [11] [10] applications. Verifying semi-automatically some structural requirements of scientific papers, such as those expressed in [3] for the inclusion of XML-based vocabularies in PubMed Central, is a further possible application. Finally, on top of the identification of specific and inter-connected constructs – for instance all those structures related to bibliographic references like lists of references, inline citations, citation contexts – it will also be possible to implement sophisticated (cross-language) services for accessing, querying and manipulating such content.

We propose an algorithm for the automatic identification of the structural roles of textual content of academic articles stored as XML files. Finally, we evaluate experimental results on real academic articles.

The rest of the paper is organised as follows. In Section 2 we give an overview of DoCO, our model for document components. In Section 3 we illustrate our algorithm for the automatic recognition of document components. In Section 4 we present the experiments on our algorithm and we discuss the outcome of these experiments. We conclude in Section 5 presenting some development we plan for the near future.

2. DOCUMENT COMPONENTS

Let us consider as an example a well-known component: the *paragraph*. Although the text it contains has meaning, a paragraph can be considered a pure structural component – i.e. a component that carries only a syntactic function.

This fits current usage: markup languages such as HTML and DocBook define a paragraph as a pure structural component, without any reference to a rhetoric function:

- "a run of phrasing content that forms a block of text with one or more sentences" [7];
- "Paragraphs in DocBook may contain almost all inlines and most block elements" [12].

Here the term "block of text" and the verb "contains" emphasise the structural connotation of the paragraph, which is amplified by our direct experience as readers. Experience that implicitly tells us that a particular textual fragment shown in a book or in an HTML page is a paragraph rather than a chapter or a table.

The document component model we use in this work to define the various components of a document is the *Document Components Ontology* (*DoCO*)[1], an OWL ontology that has been developed so as to bring together the pure structural components of documents and their rhetorical components such as Introduction and Acknowledgments. *DoCO* imports other ontologies: the *Pattern Ontology* (describing structural patterns)[2] [5] and the *Discourse Element Ontology* (describing rhetorical components)[3]. This work focuses on a subset of DoCO that is enough to capture the most relevant part of the structure of scientific papers, i.e.: *paragraphs*, *footnotes*, *tables*, *figures*, *lists*, *bibliographic reference lists*, *front matter*, *body matter*, *sections*, *references*, *bibliographc references*, *bibliographies*, *article title* and *section titles*. More details about DoCO can be found in [9].

3. RETRIEVING TEXTUAL STRUCTURES

In this section we introduce an algorithm that takes as input a set of XML sources of scientific articles (that use the same vocabulary) and recognises the DoCO components introduced in the previous paragraph. This process is fully automatic and schema-independent, relying on no background information about the vocabulary, its meaning, its intended scheme and the actual textual content of the documents themselves. The DoCO classification has also been used as reference model within Utopia Documents [1].

Our algorithm works on XML documents and exploits structural patterns. It analyses each input document separately, and for each it performs the recognition of low-level structural patterns as presented in [5]. This analysis performed separately on each individual document can assign different patterns to the same element, since the document XML schema often allows authors to use the same element in different structural ways. Thus, the algorithm processes each element that has been assigned to more than one pattern separately and proceeds as follows[4]:

1. if an element were associated to exactly two different kinds of *po:Container*, then it applies the "majority wins" rule, otherwise it is associated to *po:Container*;
2. all the elements that are assigned to both the pattern *po:Container* (or its subclasses) and to the pattern *po:Popup* are always considered as *po:Container*;
3. if element *E* is associated to both pattern *P1* and *P2* and *P1* can be used in place of *P2* without changing the document structure, then *E* has pattern *P1*;
4. finally, a majority wins rule is applied to discriminate the remaining ambiguous scenarios.

Then the algorithm applies some heuristics in order to associate one of the textual structures introduced in Section 2 to each element of the document. These rules are applied in the order in which they are presented in the following.

Paragraph. Associate with *Paragraph* all those markup elements that were recognised as *po:Block* in the previous phase and that are the block element with most occurrences in the document.

Section. Associate with *Section* each markup element that contains at least one paragraph or one section, that was recognised as *po:HeadedContainer*, and that is not the XML document element of the document.

Section title. Associate with *SectionTitle* each markup element that is the header of a section (i.e. *po:isContainedByAsHeader*).

Body matter. Associate with *BodyMatter* the first markup element that is not the document element, that was recognised as *po:Container* (or any of its subclasses), that was not recognised as *Section* and is not contained (at any level) by sections, and that has children the largest number of element recognised as *Section* (note: it must always contain at least one section).

Front matter. Associate with *FrontMatter* the first markup element that is not the document element, that was recognised as *po:Container* (or any of its subclasses), that was recognised neither as *Section* nor *BodyMatter*, that is not contained (at any level) by sections and body matters, and that has children the smallest number of element recognised as *Section*. In addition, it must be placed before the body matter (if any).

Article title. Associate with *Title* the first markup element that was annotated with pattern *po:Field* or *po:Block* and that was not annotated with *Paragraph*.

Table. Associate with *Table* each markup element that contains at least two elements, that was not associated with any of the aforementioned structures, that was recognised as *po:Table*, that may have a *po:Container* as table header, and that has all the remaining child elements sharing the same name and pattern, which must be *po:Container* or any of its subclasses. In case of multiple descendant candidates, associate with *Table* only the upper element.

List. Associate with *List* each markup element that contains at least one other element, that was not already associated with *Table*, that was recognised as *po:Table*, and that has all the child elements sharing the same name and pattern, which must be one out of *po:Container*, *po:HeadedContainer*, *po:Record*, *po:Field* and *po:Block*.

Figure. Associate with *Figure* each markup element that was not previously annotated with any DoCO structure, that was associated with *po:Milestone* or *po:Meta*, and that has at least one attribute of which value is a valid URL ending with a image extension format.

Table box. Associate with DoCO *TableBox* each markup element that was not previously associated with any

[1]DoCO, the Document Components Ontology: http://purl.org/spar/doco.
[2]PO, the Pattern Ontology: http://www.essepuntato.it/2008/12/pattern.
[3]DEO, the Discourse Element Ontology: http://purl.org/spar/deo.
[4]In the following text we use the prefixes *po* to refer to entities defined in the Pattern Ontology, *deo* to refer to entities defined in the Discourse Element Ontology. Entities without prefixes are defined in the Document Components Ontology.

DoCO structure, that was recognised as *po:Container* (or any subclasses except *po:Table*), and that contains at most three elements, of which at least one was associated with *Table*. In case of multiple descendant candidates, associate with *TableBox* only the upper element.

Figure box. Associate with DoCO *FigureBox* each markup element that was not previously associated with any DoCO structure, that was recognised as *po:Container* (or any of its subclasses except *po:Table*) and contains at most three elements, of which at least one is either a *Figure*, or a *po:Block* containing only *Figure* and no text, or a *po:Container* containing no textual blocks and an element associated with *Figure*. In case of multiple descendant candidates, associate with *FigureBox* only the upper element.

Reference. Associate with *deo:Reference* each markup element that was associated with *po:Milestone*, that has an attribute x with value v equal or similar (e.g. "#" + v) to another attribute y of another element. The latter element must be also linked by the reference element through the DCTerms property *dcterms:references*.

Bibliographic reference list. Associate with *BibliographicReferenceList* the markup element that was associated with *List*, and that has all the children referenced by some reference. In case multiple elements satisfy the previous rules, consider as the bibliographic reference list that *List* that has at least one child referenced twice in the text.

Bibliography. Associate with *Bibliography* the markup element that was annotated with *Section* and either (a) contains a *BibliographicReferenceList* or (b) in which all the children except the section title are referenced by some reference. In case multiple candidates, consider only those which have at least one descendant referenced twice.

Bibliographic reference. Associate with *deo:BibliographicReference* the markup element that is a child of either a *BibliographicReferenceList* or *Bibliography* (excluding section titles), and that is itself a *deo:Reference*.

Footnote. Associate with *Footnote* each markup element that was not associated with any other class, and that was either a *po:Popup* or *po:Container*. In the former case, its closest ancestor annotated with *po:Block* must be also a paragraph, while in the latter case it must be referenced by an element associated with *deo:Reference*.

4. TESTING THE ALGORITHM

We executed our experiment on a set of real XML documents by performing a process consisting of four steps[5].

Gold standard synthesis. We studied the XML vocabulary originally used to mark up the documents, and associated each of its elements with one or more DoCO structures. This analysis was subjective and solely based on our understanding of the semantics of the element, its definition schema and its documentation.

DoCO mapping . The Java implementation of the algorithm described in Section 3 took as input one set documents and produced a map that associated each element with one or more DoCO structure. The algorithm associates structures to each instance of each element in the documents, but there is no effort to enforce one single assignment that holds for the whole vocabulary, since an element may be used with two or more rhetorical characterisation.

[5]All the materials and results of the experiments are available online at http://www.essepuntato.it/2013/doco/test.

Results comparison . The two sets of assignments achieved from the two sets of documents were then automatically compared. We measured their agreement in terms of *true positives* (TP), *false positives* (FP) and *false negatives* (FN), and we derived *precision P*, *recall R* and the *F1-score*. We calculated P as `TP/(TP+FP)`, R as `TP/(TP+FN)`, and the F1-score as `2*P*R/(P+R)`.

Discussion. We tested the algorithm on a set of 117 scientific papers encoded in DocBook format, taken from Balisage Proceedings (`http://www.balisage.net`).

We evaluated the outcome of the algorithm against our previously-prepared DoCO mapping. The table shown in Fig. 1 summarizes our comparison through the values of TP, FP, FN, precision, recall, F1-score. It also shows the names of the elements correctly associated with each DoCO structure (TP), and the names of those elements belonging to the FP and FN sets, useful for the following discussion. A point worth highlighting is that the FP and FN assignments present (13% and 12% of the total, respectively) involved only 10 out of the 16 structures taken into consideration.

The overall results, shown in the last row of the table, were encouraging, since the overall values of precision and recall were quite high (0.887 and 0.890, respectively).

For the 4 DoCO structures *Reference*, *Title*, *SectionTitle* and *FrontMatter* (25% of the total of 16 structures examined), the heuristics worked very well on this dataset, giving a perfect match between the outcome of the algorithm and the assignments in the gold standard.

Another clear situation was that no element was assigned to the 2 DoCO classes *BibliographicReferenceList* and *BodyMatter*. This is what we expected, since no element corresponding to these classes had been identified in the preliminary human analysis. However, the absence of false positives for these assignments confirms that the rules employed in our algorithm for such structures are accurate and reliable.

However, in other cases many more options are available to the users in the DocBook vocabulary, some of which were not covered by the heuristics implemented in our algorithm. Consider, for instance, the results for the elements related to bibliographies: the values of precision, recall and F1-score for *Bibliography* and *BibliographicReference* are considerably lower than for other structures, and there is a strong connection between these values. The *Bibliography* is in fact a special *Section* whose content is exclusively made of references. The presence of blocks that are not recognized as bibliographic references, or that do not contain bibliographic references at all, causes the whole section to be mis-classified.

5. CONCLUSIONS

In this paper we proposed an algorithm for the automatic identification of some textual structures in academic articles stored as XML files. We evaluated the algorithm outcomes of a testing session involving real academic articles. Starting from the encouraging tests results, we plan to refine the heuristics we developed, as discussed in more detail in the online materials. Our goal in this work was to first identify the most common DoCO structures within real documents and the extent to which they could be automatically recognised, and second to discover where the current heuristics were failing, so that we could refine them. In future, having made these refinements, we plan to perform an exhaustive analysis on other datasets and involving other DoCO structures. The comparison with the results of similar tools is also

Type	TP	TP elements	FP	FP elements	FN	FN elements	Prec.	Rec.	F1
bibliographic referencelist	0		0		0		-	-	-
paragraph	11562	para	2363	td bibliomixed	1036	para	0.830	0.917	0.871
figurebox	381	figure mediaobject	175	mediaobject imageobject listitem equation	282	figure	0.685	0.574	0.625
list	624	orderedlist itemizedlist keywordset	176	tr variablelist	107	orderedlist itemizedlist	0.78	0.853	0.815
table	135	informaltable variablelist tbody	62	orderedlist table itemizedlist figure thead	41	informaltable tbody variablelist	0.685	0.767	0.723
bibliography	49	bibliography	1	section	54	bibliography	0.98	0.475	0.640
section	1928	section appendix bibliography	0		117	abstract	1.0	0.942	0.970
reference	3408	xref	0		0		1.0	1.0	1.0
title	117	title	0		0		1.0	1.0	1.0
figure	564	imagedata	0		4	imagedata	1.0	0.992	0.996
bodymatter	0		0		0		-	-	-
sectiontitle	1928	title	0		0		1.0	1.0	1.0
frontmatter	117	info	0		0		1.0	1.0	1.0
footnote	392	footnote	24	listitem	59	footnote	0.942	0.869	0.904
bibliographic reference	1032	bibliomixed	2	section	965	bibliomixed	0.998	0.516	0.680
tablebox	52	table	13	figure	67	table	0.8	0.436	0.565
TOTAL	*22289*		*2816*		*2732*		*0.887*	*0.890*	*0.889*

Figure 1: The outcomes of the evaluation of the Balisage set.

needed. There are very interesting solutions that cannot be discussed here because of space limits – e.g. ParsCit [8], which identifies logical structures through a per-line based aggregation algorithm, and AUTOBIB [6], which extracts bibliographic information and rebuilds bibliographic records through a Hidden Markov Model.

6. REFERENCES

[1] Attwood, T. K., Kell, D. B., McDermott, P., Marsh, J., Pettifer, S. R., Thorne, D. (2010). Utopia Documents: linking scholarly literature with research data. In Bioinformatics, 26 (18): i568-i574. DOI: 10.1093/bioinformatics/btq383

[2] Barabucci, G., Cervone, L., Palmirani, M., Peroni, S., Vitali, F. (2009). Multi-layer markup and ontological structures in Akoma Ntoso. In Proceeding of the International Workshop on AI approaches to the complexity of legal systems II (AICOL-II): 133-149. DOI: 10.1007/978-3-642-16524-5_9

[3] Beck, J. (2010). Report from the Field: PubMed Central, an XML-based Archive of Life Sciences Journal Articles. In Proceedings of the International Symposium on XML for the Long Haul: Issues in the Long-term Preservation of XML. DOI: 10.4242/BalisageVol6.Beck01

[4] De Waard, A. (2010). From Proteins to Fairytales: Directions in Semantic Publishing. In IEEE Intelligent Systems, 25 (2): 83-88. DOI: 10.1109/MIS.2010.49.

[5] Di Iorio, A., Peroni, S., Poggi, F., Vitali, F. (2012). A first approach to the automatic recognition of structural patterns in XML documents. In Proceedings of the 2012 ACM symposium on Document Engineering (DocEng 2012): 85-94. DOI: 10.1145/2361354.2361374

[6] Geng, J., Yang, J. (2004). AUTOBIB: Automatic Extraction of Bibliographic Information on the Web. In Proceedings of the 2004 International Database Engineering and Applications Symposium (IDEAS04): 193-204. DOI: 10.1109/IDEAS.2004.1319792

[7] Hickson, I. (2011). HTML5: A vocabulary and associated APIs for HTML and XHTML. W3C Working Draft 25 May 2011. World Wide Web Consortium. http://www.w3.org/TR/html5/ (last visited May 26, 2013).

[8] Luong, M., Nguyen, T. D., Kan, M. (forthcoming) Logical Structure Recovery in Scholarly Articles with Rich Document Features. In International Journal of Digital Library Systems, 1 (4): 1-23. DOI: 10.4018/jdls.2010100101

[9] Peroni, S. (2012). Semantic Publishing: issues, solutions and new trends in scholarly publishing within the Semantic Web era. Ph. D. Thesis. Department of Computer Science, University of Bologna, Italy. http://speroni.web.cs.unibo.it/publications/peroni-2012-semantic-publishing-issues.pdf

[10] Shotton, D. (2009). Semantic Publishing: the coming revolution in scientific journal publishing. Learned Publishing, 22 (2): 85-94. DOI: 10.1087/2009202.

[11] Shotton, D., Portwin, K., Klyne, G., Miles, A. (2009). Adventures in Semantic Publishing: Exemplar Semantic Enhancements of a Research Article. PLoS Computational Biology, 5 (4): e1000361. DOI: 10.1371/journal.pcbi.1000361.

[12] Walsh, N. (2010). DocBook 5: The Definitive Guide. Sebastopol, CA, USA: O'Really Media. Version 1.0.3. ISBN: 0596805029.