On tables of contents and how to recognize them

Article *in* International Journal on Document Analysis and Recognition (IJDAR) · May 2009

DOI: 10.1007/s10032-009-0078-8 · Source: DBLP

2 authors, including:

Jean-Luc Meunier
Naver Labs Europe
57 PUBLICATIONS **625** CITATIONS

SEE PROFILE

About Tables of Contents and How to recognize them?

JEAN-LUC MEUNIER AND HERVE DEJEAN

Xerox Research Centre Europe

6 chemin de Maupertuis

F-38 240 Meylan, France

Firstname.Lastname@xerox.com

ABSTRACT: We present a method for structuring a document according to the information present in its different organizational tables: table of contents, tables of figures, etc. This method is based on a 2-step approach that leverages functional and formal (layout-based) kinds of knowledge. The functional definition of organizational table, based on 5 properties, is used to provide a first solution, which is improved in a second step by automatically learning the form of the table of contents. We also report on the robustness and performance of the method and we illustrate its use in a real conversion case.

Keywords: Document structuring, Table of Contents recognition. Functional approach, Machine Learning

1. INTRODUCTION

The last few decades have seen the proliferation of electronically created documents and the increase of their usage in human activities. Unfortunately not all of these electronic documents allow for easy use beyond screen viewing or paper printing. Reasons for this restriction include, among others, the unavailability of the document in native format, the deprecation or disappearance of the original authoring environment, but also the case of scanned paper documents.

There is interest in converting those documents to a structured format. The motivations for converting documents are diverse, typically including the intent to reuse or repurpose parts of the documents, the desire for document uniformity across a database of information, facilitating document searches, and so forth. This is of particular importance for organizations that aim at optimizing their document processing, e.g. authoring/repurposing/publishing. Together with the advent of XML technologies, it creates a growing need for so-called legacy

document conversion tools and methods so as to transform unstructured print- or view-ready documents into appropriately structured documents that allow for further automatic processing.

The first steps of this conversion, such as geometrical analysis, OCR or logical analysis, have been well studied by the research and commercial communities, but further steps involving the reconstruction of higher level structure or semantic structure deserve more work.

We are interesting in this paper in the detection of one frequent tool used for helping reader in her reading: the table of contents. The purpose of such tool (with pagination, indexes) is to help the reader navigating through the book. If we have some evidence of its existence in the Early Middle Ages, its consistent usage occurs in Occident from the XIIth century, and soared up in the XVIth -XVIIth centuries (this navigation relies on pagination, which becomes reliable with the apparition of printing press).

Based on the observation that a table of contents reflects a logical organization of the content of the document and that unstructured documents often (at least for books) contain a table of contents, we are concerned in this paper with the detection and reconstruction of such tables of contents for structuring documents. This approach has already been explored in few previous works including [11], [12], [9], [6]. What motivates and differentiates the present work is the aim to develop a robust method generic enough to be applied on any document, without any specific knowledge about the document or the collection it may belong to. We therefore propose a generic characterization of a table of contents (hereafter ToC) together with a set of associated methods so as to detect a ToC from any given document, and eventually to recognize its logical hierarchical organization and to structure the document accordingly. The presented method is based on a two-step approach: a first step uses generic properties in order to recognize ToC, and a second step exploiting the specificities of the document layout in order to fine-tune this recognition process.

Furthermore, the same method can be used in order to detect other organizational tables such as tables of figure, tables of tables, etc. under the condition that the referred objects (figures, tables) have been identified.

The next section of this article explains the functional characteristics allowing us to identify a ToC. Then Section 3 firstly presents the two-step approach based on

functional and formal knowledge, and secondly details its implementation in the case of the recognition of table of contents. Section 4 presents the detection of other organizational tables based on the same method. Section 5 reports on the evaluation of this method, and eventually we discuss in section 6 our approach in view of the alternatives and future improvements.

2. CHARACTERIZING A TABLE OF CONTENTS

2.1 The Problem

Several approaches, reviewed in section 6, have been explored to determine the ToC of a document in the past. They exploit features specific to certain document classes, like the ToC layout, the page- or section-numbering scheme, or indentation.

We acknowledge that the set of features generally used to appropriately present the ToC to the reader is not extremely large, as it generally consists of a combination of different font sizes or indentations together with particular pieces of information such as heading name, numbering, dot leader and so on.

However, the observation of documents taken at random shows clearly a large number of combinations. Thus, writing a grammar that covers (almost) all the cases is extremely challenging. Everyday documents (books, journals, technical documents) illustrate the variety of ToCs: different numbering systems, different information present (headings or more), different ways to show to the reader the organization (font size, or indentation or typographical clues, e.g. all capitalized or not), presence of section numbering at some specific levels. For a specific collection, a descriptive approach can be effective, layout knowledge about this collection can be provided by rules or annotations (for learning approaches).

We believe that the approaches proposed so far do not permit in the general case to:

detect very accurately the ToC in absence of a priori knowledge about the specific layout used for representing the ToC;

determine the reference from ToC entries to document body (headings), with a higher precision than at the page level;

recognize other organizational tables, when their objects (for instance figure, equation) have been beforehand recognized in the document.

We present in the remaining article a method addressing these issues.

2.2 Our Two-step Method

The design of this method has been guided by the interest in developing a generic but nevertheless robust method that uses some intrinsic properties of the object known as a table of contents, in order to solve the layout variability problem stated before. We believe that the ToC function within the document imposes certain functional properties, which we can exploit to this end.
While performing well, this approach can still be complemented by exploiting the traditional layout information. This improvement is performed in a second step, where the formal regularities of the object provided by the functional step are extracted and exploited.
As evaluation will show, the two-step approach offers a robust solution for facing the layout variability issue. In the next sections, we develop the notions and use of functional and formal knowledge characterizing these two steps. This two-step method has been applied for other document component detection such as page number detection, page header and footer detection. It is explained in detail in [4].

2.2.1 Functional Knowledge

The functional knowledge, which we consider, relies on relations that document elements shares with the other elements in the document. As explained in [4], this kind of knowledge can not be used for recognizing all document elements, but is of first interest for an element such as table of contents. We define a ToC as a series of contiguous references to some parts of the document. This functional definition is very similar to the one given in [11]: *A TOC is simply a collection of references to individual articles of a document, no matter what layout it follows*.
A comparison with this work is given Section 6.
More precisely, this functional definition leads to define a ToC as an element following these properties:

1. Contiguity: a ToC consists of a series of contiguous references to some other parts of the document itself;
2. Textual similarity: the reference itself and the part referred to share some level of textual similarity;

3. Ordering: the references and the referred parts appear in the same order in the document;
4. Optional elements: a ToC entry may include (a few) elements whose role is not to refer to any other part of the document, e.g. decorative text;
5. No self-reference: all references refer outside the contiguous list of references forming the ToC.

This functional definition contrasts with definitions found in previous works, which are based on formal description of the ToC. For instance, [13] defines a ToC as "*nothing but text lines with a structured format.*"

Our hypothesis is that it is not required to describe the *structured format*, and that those five properties are sufficient for the entire characterization of a ToC, independently of the document class and language. In order to exploit such properties (esp. Property 2), the present method works at the document level (the whole document is required as input).

2.2.2 Formal Knowledge

But, if this functional approach allows a robust identification of table of contents, we acknowledge that the form of tables can also be informative. We are all usually able to recognize such tables at a first and quick glance only using their form and without having to check every reference. In order to improve the robustness of our method, layout information is then used in a second step. This layout information consists in the traditional characterization of textual elements OCR engines can provide such as page position, font name, font size, and information about the case (uppercase, lowercase, bold and italic cases). This kind of information is also easily extracted from digital formats (PDF for instance). Whereas such information is usually used in combination with a priori knowledge, through rules or annotations, so as to directly recognize elements, we use it without such *a priori* knowledge.

The underlying idea is that elements belonging to the same document objects (ToC entries for instance) should share some formal regularities. These regularities do not need to be known *a priori*: a smooth and efficient way to infer such formal regularities is to consider the output of the functional recognition as annotated data that will be used in order to train a classifier model learnt using traditional supervised Machine Learning methods. The classifier is then used in

order to improve the document elements recognized by the functional step (see Section 3.3.).

3. RECOGNIZING A TABLE OF CONTENTS

We now explain in detail this two-step method

3.1 Document pre-processing

Our input data usually corresponds to a document segmented into pages. Each page is composed of an ordered sequence of text blocks. Actually the segmentation in page is optional for the method while the notion of order (property 3) is necessary: the input block ordering must match the human reading flow of the document. In many cases, the proper reading flow must be determined, especially for multi-column documents. For this purpose, we use an algorithm based on the well-known XY-cut algorithm [13], which is able to segment them into bloks. This segmentation provides elements roughly corresponding to the notion of paragraphs. Note that the present method tolerates also a degraded input that only contains lines: the textual similarity (see 3.2.1) is then performed between lines instead of between paragraphs.

Page headers and page footers also deserves to be identified, especially for documents where page headers or page footers correspond to section headings (they can then degrade the link determination step [see Section 3.3]). They are often automatically detected and ignored during the next steps. The method for identifying headers and footers is detailed in [3].

3.2 Identifying the ToC Using Functional Knowledge

Four steps allow us to identify the area of the document containing the ToC text and the corresponding references. First a textual similarity is computed for all text fragments pair-wise. Then a list of ToC candidates is generated and scored. Their generation is based on the five previously properties section 2.2.1. Finally the best candidate is selected as ToC of the document and its references are determined.

3.2.1 Links computation

Firstly, links are defined between each pair of text blocks in the whole document satisfying a textual similarity criterion. Each link includes a source text fragment and a target text fragment. As mentioned earlier, these fragments can correspond to line or paragraph depending on the level of pre-processing. Figure 1 shows a toy example of a text fragment to text fragment similarity matrix for a document with 15 text fragments. A bold **x** indicates a level of similarity between two fragments above a given matching threshold, σ, an important parameter of the method.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1					**x**		x								
2							**x**						x		
3								x			**x**				
4														**x**	
5						x									
6									x						
7										x					
8														x	
9															
10												x			
11															
12															
13															
14															
15															

Figure 1: Example of text-fragment to text-fragment similarity matrix for a document with 15 text fragments

We use two similarity measures depending on the nature of the document. If the document is a digital one (usually a PDF file from which the text has been extracted), the similarity measure is the Jacquard coefficient: we use the ratio of words shared by the two fragments, considering spaces and punctuation as word separators. If the document required to be ocr-ed, the similarity measure has to take into account possible OCR errors: we introduce here a normalized edit-distance based measure between fragments (usually words). This measure is not systematically used because of its computational cost (the first measure is faster). Whenever the distance is above the similarity threshold σ, a pair of symmetric links is created. In practice, 0.4 is a good threshold value to tolerate textual

variation between the ToC and the document body while avoiding too many noisy links (as discussed in the Evaluation section). The computation of links is quadratic to the number of text blocks and takes most of the total computation time. However, searching for the ToC in the N first and last pages of the document leads to linear complexity without loss of generality.

3.2.2 Candidates generation

Secondly, all possible ToC candidate areas are built. A brute force approach works fine. It consists in testing each text block as a possible ToC start and extending this ToC candidate further in the document until it is no longer possible to comply with the five properties identified above. A ToC candidate is then a set of contiguous text blocks, from which it is possible to select one link per block so as to provide an ascending order for the target text blocks. In this whole process, we account for the possible presence of optional elements in the ToC by relaxing the contiguity property. We have chosen to introduce an additional parameter that defines how many consecutive optional elements can be tolerated in a ToC

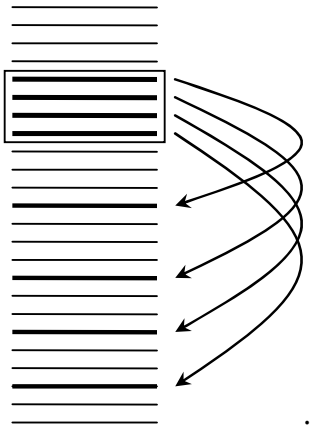

Figure 2: a TOC candidate is composed of a set of contiguous elements from which it is possible to select one link per block so as to provide an ascending order for the target text blocks

3.2.3 Candidate ranking

Thirdly, we employ a scoring function to rank the candidate tables of contents. The highest ranked candidate table of contents is then selected for further processing. Currently, the scoring function is the sum of entry weights, where an entry weight is inversely proportional to the number of outgoing links. This entry weight characterizes the certainty of any of its associated links, under the

assumption that the more links initiate from a given source text block, the less likely that any one of those links is a "true" link of a table of contents.

3.2.4 Final Link Determination

Once the highest ranked table of contents candidate has been selected, we select the best link for each of its entries by finding a global optimum for the table of contents while respecting the five ToC properties. A weight is associated with each link, which is proportional to the similarity level that led to the link creation. A Viterbi best-path algorithm [5] is adequate to effectively determine the global optimum. Figure 3 illustrates an example, in which the table of contents includes text blocks #1, #2, #3 and #4. The text block #1 is the source text block for two possible links: (#1, #5), and (#1, #7). The weight for the (#1, #5) link is 0.3, while the weight for the (#1, #7) link is 0.4, and so on for other text blocks. Any links which would violate the non self referencing property are suitably omitted. An arrow indicates the possible choice complying with all properties for #i+1 if #j was chosen as the target for #i. The bold value in parentheses indicates the best achievable score when selecting one possible target for an entry of the ToC given previous selection for previous entries.

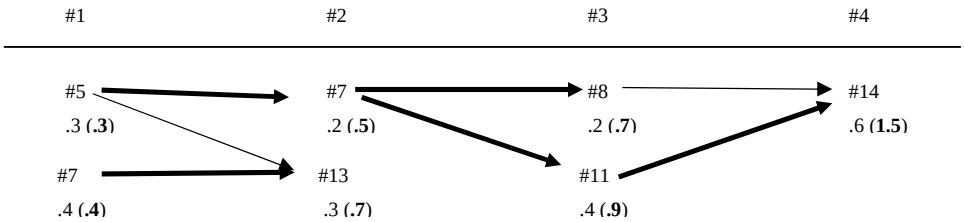

Figure 3: a trellis for the Viterbi shortest path

Viterbi algorithm consists in finding the best choice for item #i given best choice for the previous ToC item (the score of a path is the sum of its node weights). This is achieved by maintaining the best possible score at each stage together with the corresponding #j as illustrated Figure 4. The dashed arrow shows the global optimum.

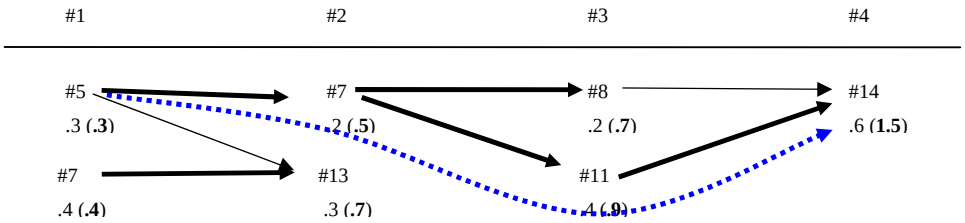

Figure 4: the shortest path is given by the dashed arrow.

To account for optional elements (property 4), we simply allow an arrow to "jump over" a series of consecutive sources. The trellis Figure 5 illustrates this when allowing one hole. Computation of the best score at each stage is done identically. The number of maximum allowed holes is a parameter of the method.

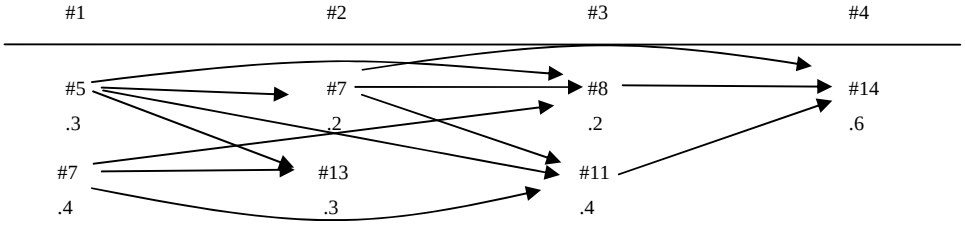

Figure 5: Trellis when holes are permitted.

Multiple global optimums may exist given a trellis, in particular when multiple targets with identical similarity levels exist for one source text. For the latter case, we systematically choose the first target.

Alternatively, it is possible to determine the links of all candidates before ranking them, as the computational cost is order of magnitude lower than computing the links themselves. Doing so permits to refine the score of a candidate. For instance, depending if an entry got a link selected or not, its weight would count as positive or rather negative.

We will discuss the evaluation of this functional approach Section 6, but as the reader will see it, it achieves very good perform for both detecting ToC entries and links between ToC entries and the referred blocks in the document. The next section discusses a way to improve this performance using the form of the ToC.

3.3 Using Formal Knowledge

Since the *functional* approach reaches a high average success rate per document, we propose to learn the *form* of the ToC from the output of the *functional* method and then refining these results by using the learned model on the same document.

Note that no manual annotation is done at anytime and that the learned model is used only once, for the document being processed.
We aim at learning a binary classifier of the links: Does a text block "belong to the ToC" or not. Once the best ToC and its links have been produced by the *functional* approach, the following steps occur:

1. Select a set of positive and negative samples from the best functional ToC
2. Learn a model for classifying ToC links between the ToC entries and the referred text block in the document;
3. Measure the model quality using an N-fold validation; Stop here if the model is not satisfying a minimal quality;
4. Compute a prediction for each link outgoing from a ToC entry;
5. Select again the best links for the Toc using the model prediction (i.e. re-apply the Viterbi with the prediction as weight instead of the textual similarity).

We now describe each of these steps.

3.3.1 Select a set of positive and negative samples from the best functional ToC

For each ToC entry of the best functional ToC, if it has a selected link, then this link becomes a positive sample. We get then one positive sample per ToC entry, not taking into account the optional elements. In order to gather negative samples, we consider the links that were not selected. We choose a number of links from this set of negative candidates. The more samples we have the better but on the other hand it can be critical to have a balanced or near balanced training set, with for instance 2 negatives for one positive sample. Other more sophisticated algorithms can be of course implemented for negative sample selection. If we consider the example shown Figure 4, we can see that our current method for selecting negatives examples does not provide any negative ones for the ToC entry #4, since there is only one possibility (#14). Negative examples could then be drawn in the page where the fragment #14 occurs. We did not investigate further such possibilities.

3.3.2 Train a machine learning algorithm

For our experiment, we have used a logistic regression algorithm, which accepts continuous features.

In order to model our data (pairs of text blocks), we do use layout and typographic features, together with the textual similarity level which remains also very informative.

Given a text block, we can build a feature vector including:

- x, y, width, height of the text bounding box in the document body (normalized on [0,1])
- Boolean indicating if all characters are uppercase for both link ends
- Boolean indicating if all characters are lowercase both link ends
- Font-size (normalized on [0,1])
- Font-color (three RGB values on [0,1])

And finally for each pair (source block, target) we build a feature vector as follow:

- Features of the source block
- Features of the target block
- Textual similarity between the two text blocks

We did not include other typographic (font-name, -style, etc) at first stage, but they can easily be integrated.

3.3.3 Measure the model quality using an N-fold validation

In order to improve over the results of the functional method, we validate the model over the training set. If the validation shows a performance below a certain threshold, we do not go further with this method and retain the functional method result. This test also allows for an automatic quality assurance of the functional step: since it is difficult to learn some formal regularities form the candidate ToC, we can assume that this difficulty may be due to a noise level higher than usual.

In practice, given the small training set size, we do a N-fold with somehow large value of, e.g. N=10, and require the N-fold validation to reach a certain success rate, e.g. 60%. The success rate we use is the F1 measure on the positive class.

3.3.4 Compute a prediction for each link outgoing from a ToC entry

We consider in turn each entry in the ToC, including the optional elements (those that did not get any link selected), and for each of each outgoing link, we use the learned model to predict its class. The prediction typically is a probability.

3.3.5 Select again the best links for the Toc using the ML prediction (i.e. re-apply the Viterbi with the prediction as weight)

Using the exact same Viterbi shortest path algorithm used in the functional method, we select zero or one link per entry so as to maximize the sum of the weight of the selected link while respecting the *functional* constraints. The difference with the previous selection resides in the weights used: now it is the model prediction instead of the textual similarity (which is part of the feature set and may therefore play some important role in the prediction computation). A refined ToC becomes available.

4. Detection of other Organizational Tables

Some shrewd readers may draw that the presented method detects tables of contents, but also other tables: tables of figures, of tables, etc. Indeed, these *organizational tables* share the same properties as the ones described Section 2.2 used for table of contents identification. The most common organizational tables are tables of figures and of tables, but depending on the type of document, others can be found such as tables of Algorithms, of Equations, of Program files, of Laboratory Exercises. The entries of these tables generally correspond to the caption associated to the referred object. Such tables can be selected as ToC if their score is higher than the score of the true table of contents (typically when the organizational table is much longer than the ToC). In order to avoid such situation, we have to assign to the table candidates a *table type*. This table type will depend on the objects already recognized in the document: figures, tables, equations, etc. So as to correctly label the candidates, we rely on the following

strategy: a score is assigned to an organizational table candidate respective to a selected object type based on proximity of the associated linked text fragments of the organizational table to objects of the selected object type. For each table entry, a proximity score is computed between this entry and objects present in the page. We currently use the following score in order to compute this proximity:

$$L_{link} = 1 - \min_{L \in page}\left(\frac{\max(h, w)}{\max(H, W)}\right) \qquad (1)$$

where the coordinates h, w indicate the vertical and horizontal distances, respectively, between the linked text fragment T and the nearest object O on the page, H, W indicate the vertical and horizontal dimensions, respectively, of the page, and L_{link} is the proximity measure for the linked text fragment T. Note that the proximity measure of Equation (1) ranges between $L_{link}=0$ and $L_{link}=1$, with $L_{link}=0$ corresponding to a largest distance away on the page and $L_{link}=1$ corresponding to a zero distance (e.g., an overlap or contacting adjacency) between the linked text fragment T and the nearest object O. The score for a selected organizational table respective to a selected object type is then given by combining the proximity measures of the linked text fragments (given in Equation (1)), for example using a weighted sum:

$$(\text{Score})_t = \frac{1}{N} \cdot \sum_{n=1}^{N} (L_{link})_{n,t} \qquad (2)$$

where N is the number of linked text fragments associated with entries of the organizational table (or, correspondingly, N is the number of entries in the organizational table), the index n={1,…,N} ranges over all of the linked text fragments, t indexes the selected object type, $(L_{link})_{n,t}$ denotes the proximity measure L_{link} for the n^{th} linked text fragment respective to the nearest object of selected object type t, and $(Score)_t$ denotes the score for the organizational table respective to the selected object type t. Since L_{link} ranges between 0 and 1 and Equation (2) is normalized by the (1/N) factor, it follows that $(Score)_t$ given in Equation (2) also ranges between 0 and 1, with higher values indicating closer proximity between objects of the selected object type t and the linked text fragments associated with the entries of the organizational table.

The detection of such organizational tables allows for markup of the corresponding captions in the document body.

5. EVALUATION

We present in this section different evaluations for this method. We first use a small corpus to illustrate the main issues with the method and to also illustrate the effect of the formal step. Secondly we present a customer case where the method is used for marking up headings in technical documents. As evaluation measure, we use the traditional precision and recall as defined:

$$\text{precision} = \frac{number\ of\ correct\ links\ found}{number\ of\ \text{found}\ links} \qquad (3)$$

$$\text{recall} = \frac{number\ of\ correct\ links\ found}{number\ of\ total\ \text{expected}\ links} \qquad (4)$$

The F-1 score, which is the harmonic average of precision and recall, is also provided. According to these formulae, we evaluate the pair (toc entry, body entry), and not only the toc entry recognition.

Since this method relies on a previous segmentation, incorrect segmentation may be a source of errors. Typically, when a ToC entry spans over several lines, and the segmentation step is not able to recognize them as a single bloc, each part of the entry will be considered as a ToC entry. In the best case, only one part is recognized as ToC entry, and the rest is considered as hole. In the worse case, each part is recognized as ToC entry, which introduces errors (incorrect entries). We must add that the frequent tabular structure of the ToC eases the segmentation step, and OCR engines such as FineReader and OmniPage usually deal well with ToC segmentation so that this kind of errors is not so frequent.

5.1 Evaluation of the Formal Refinement

The purpose of this section is to illustrate with three documents, besides its robustness, the types of the main errors generated by the method, as well as the improvement due to the formal step. These three documents contain 194 ToC entries and 385 pages.

The ToC determination is then performed with the parameter set to their default value (similarity threshold σ: 0.4, nb consecutive holes: 3), for both the functional and formal methods. Table 1 shows the evaluation for the functional approach for these three documents.

Table 1: Evaluation of three documents with the functional approach.

	# pages	**precision**	**recall**	**F1**	**#OK**	**#Err**	**#Miss**
Doc1	26	100.0	100.0	100.0	50	0	0
Doc2	114	100.0	98.8	99.4	82	0	1
Doc3	245	90.3	91.8	91.1	56	6	5
	Total	**96.9**	**96.9**	**96.9**	188	6	6

For documents 1 and 2, the results are excellent, whereas recall for document 3 is much lower. The missing toc entry in document 2 is due to an inconsistency in the Toc (the section title in the ToC differs from the title in the body). Such inconsistencies are not frequent, but nevertheless recurrent in documents. For document 3, 5 misses and 5 errors are due to the selection of bad links for some entries: due to this incorrect links, the correct ones for subsequent entries are then ignored due to the non crossing property. Note that one bad link for a correct entry generated one error and one miss. The last error is due to one page header present in the ToC, a two-page ToC, and considered as a ToC entry. Such error is recurrent, and in practice, header and footer detection considerably reduces noise. The next table shows the benefit of the formal step. This step is of course no able to find the missing entry for document 2 (inconsistency between ToC and body), but is able to correct the 5 bad links for document 3. No error is introduced for the first two documents which were almost perfect. Usually the formal step does not degrade quality, or marginally. For the third document, the bad link formerly created by the functional step is correct and all the subsequent impacted links are corrected. The remaining error is the same: due to the present of one page header in the ToC.

Table 2: improvement after the formal step. A gain of 2 points for the F1 score.

	# pages	**precision**	**recall**	**F1**	**#OK**	**#Err**	**#Miss**
Doc1	26	100.0	100.0	100.0	50	0	0
Doc2	114	100.0	98.8	99.4	82	0	1
Doc3	245	96.8 **(+6.5)**	98.4 **(+6.6)**	97.6 **(+6.5)**	60	2	1
	Total	98.9 **(+2.0)**	99.1 **(+2.2)**	99.0 **(2.1)**	192	2	2

In addition to describing the main type of errors we observed, and according to our experience, despite its small size this small set of document reflects well the behavior of the method in general.

5.4 Evaluation in a customer case

We present here an evaluation in a real conversion task with a larger dataset. This real task consists in structuring a collection of construction bids in divisions and sections. Even if these documents follow strict guidelines (the Construction Specification Institute MasterFormat [2]), they use various layout standards, depending on the institution issuing the call for bid. The paper documents are first scanned, OCR-ed and manually indexed at the division level. The TOC detector was used in order to segment each division into sections. The requirements for the structure within sections were minimal: a simple structuring in lines. Each call for bid has a table of contents from 12 to 183 entries (corresponding to sections). The test set was composed of three collections from different periods of time, from 1999 to 2007. During this period this Master Format was modified, but without impacting our method since the functional step does not make any assumption with respect to this format. Table 3 describes these different collections.

Table 3: description of the three collections following the Construction Specification Institute Master Format.

Collection	# documents	# pages	# sections
#1	32	13583	1964
#2	25	12183	949
#3	11	10202	865
TOTAL	**68**	**35968**	**3778**

This dataset represents 30% of the daily conversion (120,000 pages per day). The evaluation is not similar to the previous one which dealt with links between ToC and body entries. Here we only evaluation the detection of sections in the document body using the ToC. So the ToC evaluation is indirect but due to the amount of data used, we believe that this evaluation is very relevant to assess the ToC detector. Table 4 shows the results obtained for each collection. The overall behavior is similar to the previous evaluations. The functional step reaches a good quality (a F1 score around 90). The somehow low recall is generally due to tables of contents that do not cover the whole document (divisions 15 and higher can be missing in the ToC). The formal step behaves similarly to the previous evaluation, even if the improvement varies from 2 points up to 4.5 points.

Table 4: Evaluation without and with formal step.

	Collection	Precision	Recall	F1
Functional	#1	96.7	88.0	92.1
Functional+ Formal	#1	97.6 **(+0.9)**	90.9 **(+1.9)**	94.1 **(+2.0)**

Functional	#2	95.3	87.5	91.2
Functional+ Formal	#2	96.54 **(+1.1)**	90.7 **(+3.2)**	93.5 **(+2.3)**
Functional	#3	94.0	86.5	90.1
Functional+ Formal	#3	96.3 **(+2.3)**	92.7 **(+6.2)**	94.5 **(+4.5)**

We also wanted to measure the importance of the key parameter: the similarity threshold σ , with and without formal step. As Table 5 and Table 6 show it, the σ set up at 0.4 gets the best score, with the functional approach but also with the formal approach. As expected the higher the σ value, the higher the precision. Of course recall is impacted. More important, a precise estimation of this parameter does not seem important since other values such as 0.5 also provide acceptable results. The 0.4 value has also been validated upon other collections and is used as default value for this parameter. This somehow low value enables the ToC detector to detect links between ToC entries and body entries even for documents where the text segmentation is not perfect: a similarity is detected even between two portions of section titles. This also allows to cope with OCR noise and with some inconsistencies between ToC entries and body entries (one different word for instance). The constraints provided by the five properties used by the functional step allow the ToC detector to select the right links in most cases. Eventually, it is interesting to mention that tuning between precision and recall for specific applications and purposes can be done through this σ parameter.

Table 5: impact of the σ parameter with functional step only.

σ	Collection	Precision	Recall	F1
0.3	1	95.8	88.6	92.1
0.3	2	94.1	87.4	90.6
0.3	3	93.1	86.7	89.8
0.4	1	96.7	88.0	92.1
0.4	2	95.3	87.5	91.2
0.4	3	94.0	86.5	90.1
0.5	1	96.5	85.8	90.8
0.5	2	95.4	87.2	91.1
0.5	3	93.2	81.3	86.8
0.6	1	97.2	83.8	90.0
0.6	2	95.8	84.6	89.9
0.6	3	93.9	81.5	87.5

Table 6: Impact of the σ parameter with formal step.

σ	Collection	Precision	Recall	F1
0.3	1	96.7	91.3	93.9
0.3	2	95.1	90.4	92.7
0.3	3	95.1	93.3	94.6
0.4	1	97.6	90.9	94.1

0.4	2	96.4	90.7	93.5
0.4	3	96.3	92.7	94.5
0.5	1	97.4	88.3	92.6
0.5	2	96.5	90.3	93.3
0.5	3	96.1	88.0	91.9
0.6	1	97.8	86.4	91.7
0.6	2	97.6	88.4	92.8
0.6	3	95.7	87.2	91.2

As a matter of fact, the customer expectation was slightly higher than what the ToC detector could provide, and it required the introduction into the system of other components in order to improve both precision and recall. The main improvement was due to the detection of page numbering, since sections frequently started with a new numbering sequence. The detection of page header and page footer also improves quality by reducing noise for the ToC detector. Additionally an important requirement was to be able to perform an automated quality assurance in order to discarded processed documents. To achieve this, we use heuristics very specific to the collection, such as. These heuristics achieve a very good precision but a poor recall. In the end, we were able to reach a precision and recall around 97% covering 95% of the divisions, reaching the customer expectation.

6. RELATED WORK

We discuss here other work around the detection or analysis of table of contents. Lin [11,12] and Mandal et al. [13] each proposed a method for detecting ToC pages in a document. Lin's method is based on a similar functional approach and is applied to journals. It takes advantage of a combination of text matching, layout and page numbers to determine the pages holding the ToC as well as the starting page of each referenced paper. Mandal's method relies on a page-number-based heuristic and works on the page image, before segmenting the page content. The present method offers a means to go beyond the page level. It also differs from the cited works in that it remains fully independent of any layout or page/section-numbering scheme, which we see as a key advantage in term of robustness and generality, because the large variability of such schemes across collections is hard to capture in a systematic way.

Regarding further analysis of the ToC itself, after the ToC was identified by automatic or manual means, we noticed the works below. Two of them aimed at attaching semantics to the ToC entry constituents, which we do not do here. Satoh et al. [16] first proposed an electronic library framework including the automatic analysis of the ToC pages of journals in order to extract bibliographic information by categorizing each text block by means of a decision tree. Belaïd et al. [1] proposed a labeling approach of the ToC of scientific journal in the Calliope electronic library using dictionaries and collection-dependant contextual rules for part-of-speech tagging. The objective here is to analyze each element of the ToC, which is provided as input.

Lin et al. [10] acquired the logical structure of the ToC thanks to an in-depth analysis of its numbering scheme. Tsuruoka et al. [15] exploited the indentation and font size to classify ToC lines in different hierarchical groups. Feng at al. [6] exploited the indentation, page numbers and numbering scheme to compute the logical structure of a book. We view those approaches as less general than the one proposed here, because of the observed diversity of page/section-numbering schemes and ToC layout.

We also tested two commercial OCR engines. They can generate MS word format as final output. This output format may contain a document map section which corresponds to the document structure. They offer good results for very simple documents but their user quickly reaches their limit. As far as we can analyze their results they might use font information in order to select section headings.

7. DISCUSSION

The general characterization of a ToC, introduced in section 2, experimentally proved to be sufficient for both detecting the ToC and determining where each of its entries points to in the document body. The all five properties are necessary, since we know of documents that would defeat any reduced method.

We proposed a particular implementation leveraging the general characterization in order to validate it experimentally. Other implementations are certainly possible.

Although we tried to develop a system as generic as possible, it does not cover certain cases:

- when several ToCs are present in a document (for instance, one for the whole document and one per chapter), the ToC detector usually detects the largest one. One foreseen solution is to structure the document with the first ToC found, and to reapply the ToC detector on the new parts created. But this solution fails when the main ToC is not the largest one: it can happen that this kind of ToC only refers to chapter headings, and each chapter contains a ToC describing its organization.
- On contrary, when a document does not include a table of contents, our system will quite certainly wrongly identify a portion of text as table of contents. Some criteria can be then used in order to evaluate the quality of the selected table of contents such as coverage (proportion of document covered by the ToC), the number of holes in the ToC, or the average similarity between links and section headings. In addition the cross-validation of the formal pass will probably produce a low F1 measure.

A key step is the determination of the links between ToC entries and elements in the document (property 2). In many documents, this similarity can be simply detected at the character/token level (and is currently implemented so). But in some rare documents, such as magazines, the ToC entries might correspond to a very short description of the section (a short sentence). In this case, property 2 is still valid, but the use of an edit-distance-based approach is no longer valid, and the link is almost impossible to establish without more elaborated linguistic techniques.

The method is sensitive to its input quality, namely its ordering, segmentation and page headers/footers. Indeed, if the ordering or paragraph segmentation of the input document does not reflect the human reading flow and paragraphing, the method may fail because properties 1, 2 and 3 would not fully apply anymore. This happens for document with chronological tables of contents. Page headers or footers also constitute an important source of noise to the method since they may recall the title of the current section (running titles). Unfortunately, this strategy does not work for certain types of documents where section headings are also used as headers (examples found in some technical documentation). Overall, we believe that the nature and quality of pre-processing required for the present method is acceptable and commonly applied in many document-understanding tasks.

An important yet missing step is the determination of the hierarchical levels of the ToC entries. Some experiments have been conducted which strongly rely on the assumption that elements of the same level share the same visual aspect. Entries are clustered according to their layout, and the hierarchical level is assigned using a heuristic. The quality of this method highly depends on the quality of the documents, especially for documents provided by OCR: in such case, information about layout may be noised, and other kinds of information have to be introduced in order to constraints the clustering operation.
The next enhancement for this method would be to lessen Property 3: *Ordering: the references and the referred parts appear in the same order in the document.* This would allow the detection of ToC such as the ones quite frequent in magazines and journals, which are first organized by topic, as in the *Communications of the ACM* where the topics are: articles, columns and departments. Currently our method selects as ToC only the longest

Both the evaluation reported here and other applications we performed on real cases are very satisfying. The research direction we are now taking in order to improve the conversion is based on the combination of and interaction between components such as segmentation or detection of page numbers. The use of redundant information should allows for an improvement of each component and thus of the overall conversion quality.

8. REFERENCES

1. Belaïd A., Pierron L., Valverde N., Part-of-Speech Tagging for Table of Contents Recognition, International Conference on Pattern Recognition, 2000.

2. Construction Specifications Institute, Project Resource Manual (PRM) : The CSI Manual of Practice (Hardcover), McGraw-Hill Professional, 2004

3. Déjean H., Meunier J-L., System for converting PDF documents into structured XML format, 7TH IAPR Workshop on Document Analysis Systems, Nelson, New Zealand, 13-15 February 2006.3

4. Déjean H., Meunier J-L, Logical document conversion: combining functional and formal knowledge, Proceedings of the 2007 ACM symposium on Document Engineering, DOCENG 2007, Winnipeg, Manitoba, Canada. DOI: http://doi.acm.org/10.1145/1284420.1284456

5. Forney G. D.. The Viterbi algorithm. Proceedings of the IEEE 61(3):268–278, March 1973.

6. He F., Ding X., Peng L., "Hierarchical logical structure extraction of book documents by analyzing tables of contents", Document Recognition and Retrieval XI, Proc. of SPIE-IS&T Electronic Imaging, SPIE Vol. 5296, 2004.

7. Ishitani Y., Document Transformation System from Papers to XML Data Based on Pivot XML Document Method, International conference on document analysis and recognition, ICDAR 2003
8. Jain A. K., Myrthy M. N., and P. J. Flynn. Data clustering: A survey. ACM Computing Survey, 31(3):264--323, 1999.
9. Le Bourgeois F., Emptoz H., Souafi Bensafi S., "Document Understanding Using Probabilistic Relaxation: Application on Tables of Contents of Periodicals", Proceedings of the Seventh International Conference on Document Analysis and recognition, ICDAR'01, 2001.
10. Lin C., Niwa Y., Narita S.. Logical Structure Analysis of Book Document Images Using Contents Information, ICDAR 1997
11. Lin X., "Detection and analysis of table of contents based on content association", International Journal on document Analysis and Understanding, Volume 8, Numbers 2-3, 2006.
12. Lin X., "Text-mining Based Journal Splitting", Proceedings of the Seventh International Conference on Document Analysis and recognition, ICDAR'03, 2003.
13. Mandal S., Chowbury S.P., Das A.K., Chanda B.. Automated Detection and Segmentation of Table of Contents Pages from Document Images, ICDAR 2003
14. Meunier J-L., Optimized XY-Cut for Determining a Page Reading Order, ICDAR 2005
15. Tsuruoka S., Hirano C., Yoshikawa T., Shinogi T.. Image-based Structure Analysis for a ToC and conversion to XML, DLIA workshop 2001.
16. Satoh S., Takasu A.,Katsura E., "An automated Generation of Electronic Library based on Document Image Understanding", Proceedings of the third International Conference on Document Analysis and recognition, ICDAR'95, 1995.