A Table Detection Method for Multipage PDF Documents via Visual Seperators and Tabular Structures

Jing Fang, Liangcai Gao[a]
Institute of Computer Science & Technology,
Peking University, Beijing, China
fangjing, gaoliangcai @icst.pku.edu.cn

Kun Bai
IBM Research T.J. Watson Research Center
19 Skyline Dr., Hawthorne NY, USA 10532
kunbai@us.ibm.com

Ruiheng Qiu
State Key Laboratory of Digital Publishing Technology,
Beijing, China
qiurh@founder.com.cn

Xin Tao, Zhi Tang
Institute of Computer Science & Technology,
Peking University, Beijing, China
taoxin, tangzhi@icst.pku.edu.cn

***Abstract*—Table detection is always an important task of document analysis and recognition. In this paper, we propose a novel and effective table detection method via visual separators and geometric content layout information, targeting at PDF documents. The visual separators refer to not only the graphic ruling lines but also the white spaces to handle tables with or without ruling lines. Furthermore, we detect page columns in order to assist table region delimitation in complex layout pages. Evaluations of our algorithm on an e-Book dataset and a scientific document dataset show competitive performance. It is noteworthy that the proposed method has been successfully incorporated into a commercial software package for large-scale Chinese e-Book production.**

Keywords- table detection; table spotting; PDF documents; separators; ruling lines

I. INTRODUCTION

Table, as an efficient and compact means to present data and two-dimensional relationship information, has been widely used in different kinds of documents. Table recognition has become an important task in the field of document structure recognition, and has attracted a good number of researches in the past two decades.

The most straightforward motivation of our work stems from the requirement of electronic book (e-Book) reading on handheld devices. The ubiquitous e-Books are part of the next generation web, which has promoted development of web publishing industry. Along with the advancement of wireless network and mobile technology, handheld devices (e.g. smartphones, Kindle, iPad) have rapidly gained popularity and become a common platform for rendering e-Books. Due to the limited screen sizes of these devices, e-Book documents usually need to be re-flowed and re-composed to provide readers a friendly reading experience.

Firstly, the table regions should be detected and separated from other elements within a page. Then they could be isolated as independent objects to be rendered on handheld device screens. In a further step, the detailed structure of tables, i.e. columns, rows and cells should be recognized. In this way, an oversize table can be laid out in several continuous screen pages, or users can hide certain columns or rows according to their reading preference. In this paper, we focus on the first and the most crucial step--table boundary detection. Structure recognition will be discussed in future work.

As an open document format, PDF has been exploited extensively in web publishing. However, existing approaches for table detection in image-based documents do not work reliably on PDF files, whose internal and low-level information can be mined deeply. This paper proposed a table detection method targeting at PDF documents. The method has been incorporated into a commercial software of Founder Corporation[b], and has been used to detect and re-flow tables in about two millions of e-Books in the last several months.

The rest of the paper is organized as follows. Section II reviews several relevant studies in table detection, especially those on PDF files. Section III firstly gives an overview of the proposed solution and then presents each step in detail. Section IV demonstrates the experimental results. Conclusion and future work are included in Section V.

II. RELATED WORKS

A good number of research efforts have been made on table detection so far. Among them surveys provided by Zanibbi et al. [1] and Silva et al.[2] both summarized table detection approaches in detail.

However, the majority of researches on table detection concentrated on image-based documents, such as one of the pioneering works T-Rect and T-Rect++ systems proposed by Kieninger et al.[3, 4]. Although PDF format, as a higher level of document representation, becomes increasingly important, there is much less prior works toward PDF documents (e.g. [5-8]).

pdf2table system, proposed by Yildiz et al.[5], is the first relative research carried out on PDF documents, performing two tasks: table detection and table decomposition. The table regions were spotted by detecting and merging multi-lines with more than one text segments. Similarly, Oro et al.[7] classified the lines into three classes: text lines, table lines and unknown lines, according to the number of

[a] Liangcai Gao is the corresponding author
[b] http://www.apabi.cn/English/index_en.html

1520-5363/11 $26.00 © 2011 IEEE
DOI 10.1109/ICDAR.2011.304

segments in the lines. Then, the continuing table lines or unknown lines were combined to form tables. But they both made a single-column page assumption for input documents.

Liu et al.[6] proposed a table search engine system *TableSeer*. It crawls scientific PDF documents, identifies documents with tables, detects table regions, indexes them and enables end-users to search for tables. So far, it is the most complete system for table recognition targeting at PDF documents. The table detection part is implemented by labeling and merging sparse lines. However, as a searching system, *TableSeer* makes heavy demands on precision. It makes the assumption that all the tables have captions, which will discard tables without captions inevitably and leads to low recall rate.

Essentially, the approaches outlined above are based on the observation that table lines contain more than one text segments. However, utilizing purely content layout features has the following shortcomings: *i)* the text line segmentation is sensitive to predefined threshold, as well as spanning cells; *ii)* on a multi-table page, it is difficult to distinguish different tables from each other; *iii)* irregular tables such as sparse ones cannot be handled well. These factors may cause under-segment or over-segment of table regions.

From the essence of tables, we find that the more complex the layout of a table is, the more graphic lines are employed as borders and rules. Therefore, the graphic ruling lines and content layout should be treated together as important sources for spotting table regions. To our best knowledge, only Hassan et al. [8] detected tables in PDF files utilizing both ruling lines and content layout. However, their method utilizes these two sources isolatedly. Besides, since the detected graphic lines are not verified first in their method, the false positive lines may easily lead to false positive tables. Hence, the experimental results given by the paper is not quite satisfactory.

To solve the above problems, we propose a detecting method via both visual separators and tabular structures of contents. The separators refer to not only graphic lines but also white spaces to handle unruled tables. Besides, we analyze page layout to determine page columns in the first place to assist table detection in multi-column pages. Generally, our method has the following advantages:

1) Benefits of graphic ruling lines are fully reflected in complex page layout (e.g. multi-column pages) and irregular tables (e.g. sparse tables, nested tables, tables wrapped in body paragraphs). And these graphic lines parsed from PDF have accurate coordinates than those from images.

2) The white spaces are used as virtual separators for unruled tables. Since they are not as accurate as graphic lines, page columns are detected to assist table spotting, no matter the tables span the page columns or not.

3) Both candidate table contents and visual separators are verified before being used to spot tables, leading to higher accuracy.

III. PROPOSED SOLUTION

The proposed table detection approach consists of four steps: *i)* PDF parsing, *ii)* page layout analysis, *iii)* separator mining, and *iv)* table detection (shown in Fig.1). The task of Step 1 is to provide content streams of PDF documents, such as character and graphic objects. Step 2 analyzes page columns and Step 3 constructs graphic ruling lines and whitespace separators. In Step 4, we first validate the tabular contents and separators to filter out unnecessary sources and reduce false alarms, then detect table regions via visual separators, and finally present a post-processing step to discard false positive results. Each step will be elaborated in the following sections.

A. PDF Parsing

PDF documents are described by low-level structural objects such as a group of characters, lines, curves, images etc., and associated style attributes such as font, color, stroke, fill, and shape, etc. [9]. To parse those low-level objects, we utilize the PDF parser provided by Founder Corporation, which is developed according to the PDF specification [9]. For text objects, attributes like font, bounding box could be parsed, while the graph objects contain drawing and clipping instructions in groups of paths. Besides, the painting information like color space, joining styles are also available. The graphic objects are adopted for constructing ruling lines, which will be elaborated in Subsection III.C.

B. Page Layout Analysis

Most of existing approaches on table detection are based on single-column input assumption. Actually, multi-column layout in real-world documents is also common, in which tables may span the columns or only show up inside one column. This brings more challenges for table detection.

To handle these cases, we firstly detect page columns based on whitespace analysis algorithm proposed by Breuel [10]. It exploits a priority queue of pairs (r, O), where r is a rectangle and O is a set of obstacles overlapping r. Pairs are iteratively extracted. If the set of obstacles is empty, the maximum white rectangle still needs to be discovered; otherwise, one of its obstacles is chosen as a pivot and the rectangle is split into four neighbor regions as sub problems. White rectangles are returned in order of decreasing area.

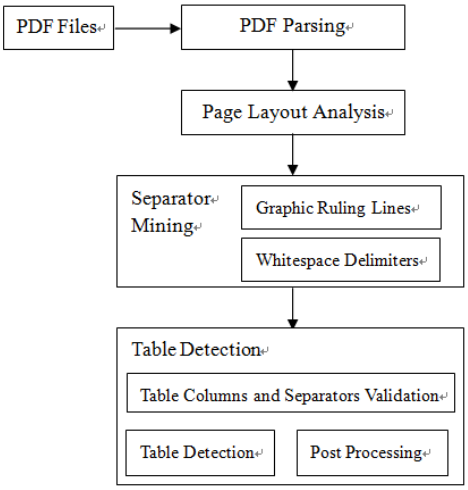


Figure 1. Workflow of our table detection method

We implement this branch-and-bound method with characters as "obstacles". Then, referencing those loosely generated whitespace rectangles, characters are aggregated to text segments. Through experiment we find that the original method is sensitive to the stopping rule. Hence, considering layout similarity characteristic of multi-page documents, i.e. columns are usually repetitive on continuous pages and tall unlike other kinds of white spaces, we adjust the algorithm in the following way:

1) Taking N pages before and after the current page (N=5 in this paper) and generate loose white spaces.

2) Calculating the cumulative length of all vertical white spaces on the same X-position;

3) Treating the obvious peaks as the potential column positions and keeping the white spaces on these positions instead of page height to tolerate span-column elements.

C. Separator Mining

A table is a grid of cells separated either by graphic ruling lines or whitespace delimiters. In order to handle both ruled and unruled tables, we construct both separators in this paper.

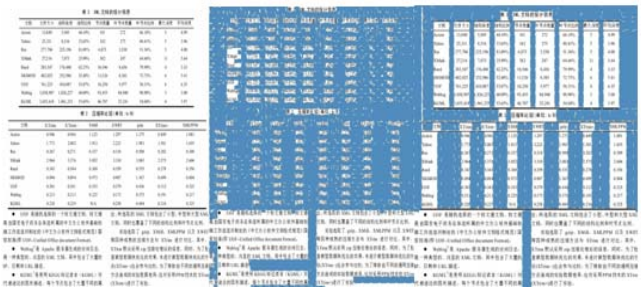
1) Graphic ruling lines

Table ruling lines in PDF documents are mostly composed of graphic objects. Each object is a serial of path instructions, such as operators *m*(moveto), *l*(lineto), *re*(rectangle), *c, v, y* (cubic Bezier curve) are used to draw lines, rectangles and curves [9]. Thus, sub-paths of straight lines are acquired by dividing the rectangles and delimitating draw-line paths when slop changes. Afterwards, cluster algorithm is utilized to form integrated graphic lines in both horizontal and vertical directions.

Different from [8], we refine these raw lines for ruled table detection instead of using them directly because of the following reasons: *i)* graphic lines are usually constrained by the clipping path that limits the regions of the page affected by painting operators [9]. Only elements inside the clipping area are applied to the page; *ii)* there are graphic objects filled or stroked with white color, which are invisible and meaningless from readers' point of view; and *iii)* other types of graphic lines exist, such as header or footer separators, matrix braces, figure components, etc., which should be discarded to reduce the false alarms. Therefore, we first refine the graphic lines by taking clipping path and color space into consideration, then further validation will be carried out in Subsection 3.D to remove false positive table ruling lines.

2) Whitespace delimiters

To get whitespace delimiters, the algorithm of [10] is also adopted. The "obstacle" in this step is text block aggregated from text segments. Specifically, we construct an adjacent matrix graph with all the text segments and utilize depth-first search to aggregate text blocks. In this way, the white spaces can be less trivial than character-generated ones and more similar with real table lines. The difference is illustrated in Fig. 2, in which Fig.2 (b) uses characters as "obstacles" and Fig.2 (c) uses text blocks. The white spaces in the latter are more obvious to be delimiters.



(a) (b) (c)

Figure 2. Whitespace delimiter analysis with different "obstacles" input. (a) the original page, (b) use characters as "obstacles", and (c) use text blocks and graphic lines.

D. Table Detection

1) Preprocessing

a) Validation for table content

The aggregated text blocks contain both body paragraphs and table column fragments. In order to reduce unnecessary detection and false alarms, we now present a method to remove the body text paragraphs.

Guided by practical experiences and observations, we concluded that the left and right sides of text paragraphs are either next to the page boundary or page column space (shown in Fig.3). However, tables do not follow with this pattern, since they have at least two columns or even more. We verify each text block with this rule and most body paragraphs can be filtered out. Noises are tolerant to remain, such as small figure notation text blocks.

b) Validation for visual separators

As mentioned in Subsection 3.C, graphic lines should be validated and refined to discard false positive ones. Firstly, after table content validation, the graphic lines far from the candidate table content are discarded. Then we utilize layout features of those table content as: there should be more than one text segment above and below the same horizontal graphic line, which are also overlap correspondingly. The Whitespace delimiters are handled similarly. After this verification, most false horizontal separators are removed.

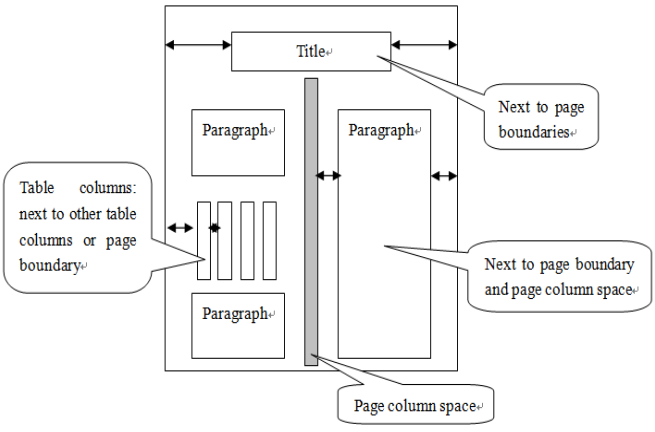


Figure 3. Neighborhood detection for table column validation

2) Table Detection Solution

a) Ruled tables

According to people's reading experience, if only horizontal ruling lines are detected, the table columns are often delimitated by obvious vertical white spaces. Therefore, we rank the horizontal graphic lines and begin with the longest one to examine whether it crosses multiple vertical whitespace separators. If so, the longest vertical white space will be treated as the table height, and the width of this horizontal line represents the table width. All the other lines and vertical white spaces inside this table region are removed from the candidate sources to avoid forming overlap tables. The termination condition is that there is no source graphic lines exist. Otherwise, if both horizontal and vertical graphic lines exist, the same method is executed as outlined above, using vertical lines instead of vertical whitespace delimiters.

b) Unruled tables

In real-world documents, completely unruled tables are rare but still exist (see example in Fig. 4). Those tabular data are usually typesetted regularly with clear whitespace delimiters, not only in vertical direction but also in horizontal direction. Whitespace delimiters not only distinguish different tables in multi-table pages but also separate tables from other logical components. Please note that we only keep the horizontal white spaces wider than the dominant interline spacing. Then the similar step can be carried out as we did for the ruled tables. In addition, since the white spaces recognized are not as accurate and creditable as graphic ruling lines to determine the wide range of tables, we create a constraint that if the spotted table crosses the page columns, it should be segmented along the column space.

3) Post processing

Due to the versatility of table formats in real-world documents, false positive tables are inevitable in the existing detection approaches. In our method, they are mainly caused by graphic figures and matrix formulas.

After examining the false positive tables, we find that although figures contain graphic lines and small text segments as notations, they seldom have the grid feature like tables. Besides, most tables do not contain curve line objects as graphic figures do. As for the matrixes, the graphic lines of them are usually two vertical lines, while tables seldom follow this pattern.

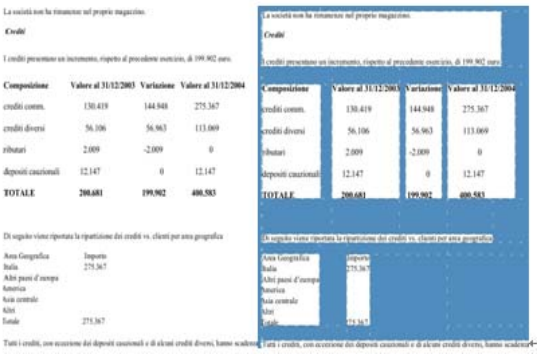


Figure 4. Multiple unruled tables on one page

Therefore, the following conditions are checked to verify whether the extracted tables are false positive ones or not:

1). There should be at least two rows and two columns in a table.

2). There should not be curve line objects in a table.

3). The separators and columns inside a table should be spaced at intervals.

4). The table should not contain only vertical ruling lines.

IV. EXPERIMENTAL RESULTS

We test our algorithm on two datasets. The first dataset (D1) includes 70 PDF e-Books provided by Founder Corporation. Those books were selected randomly from the two million e-Book library and manually examined to ensure each of them contains tables with various layouts. The second dataset (D2) is scientific documents provided by Liu, which is used in her previous work [6, 11]. We manually selected 70 documents with the same requirement as D1. In total, 3802 tables in D1 and 197 tables in D2 are evaluated. Since there is no existing ground truth, all the correctness comparisons are obtained from human understanding. We evaluate two performance metrics: recall (the percentage of the true objects that the method finds) and precision (the percentage of the objects that are in fact true).

Table I compares our evaluation results with that of [6]. The results are analyzed to explore the advantages as well as main causes for errors. Fig.5 provides some examples to illustrate the effectiveness of our methods in different cases, which cannot be handled well in existing methods.

A. Experimental Results and Analysis

From the data presented in the Table I, we conclude that our method has desirable improvement on recall rate over that of method [6], because they define captions as necessary attribute of tables. In real-world documents, captions may be absent or labeled with keywords different from their predefined list. Besides, in the e-Book dataset, we also get a higher precision, because sparse tables with a few cells are hard to be detected due to lacking of the sparse line feature proposed by Liu. However, ruling lines of these tables are more likely to be explored as boundaries. As a result, our method shows competitive performance.

However, errors are still inevitable in the following aspects: *i)* if the page column space is overlap coincidently with the table column spaces, the wrongly detected page column may cause over-segmentation of tables; *ii)* the whitespace separators are not as accurate as ruling line separators, which could result in tables being spotted partially; *iii)* some figures with tabular structure and only straight lines are still mistakenly detected as tables.

TABLE I. EXPERIMENTAL DATA

Mehods	D1		D2	
	Precision	Recall	Precision	Recall
Method in this paper	96.13%	92.07%	94.42%	93.71%
Method in Liu[6]	93.56%	83.20%	96.28%	92.50%

B. *Examples*

As representative examples, Fig. 5(a) shows a ruled table wrapped in single-column body paragraphs. Without utilizing the ruling lines, the table region may become under segmented, joining the text paragraph on its left side. Fig. 5(b) shows partially ruled tables that span two columns in a double-column page. In this case, the horizontal ruling lines help us to determine the width range of the tables. Besides, the non-aligned vertical whitespace delimiters presented in this paper can distinguish two tables from each other, which is a challenging problem for existing sparse line based methods. Fig. 5(c) shows three ruled tables lying inside one column, and the two smaller ones are side-by-side. This case may easily lead to under-segmentation utilizing existing methods, especially for the two smaller ones. Finally, the third sparse table in Fig. 5(d) is also a challenge for detecting methods based on purely content layout information, because there is only one text segment in the lines, which cannot be defined as candidate table lines.

The experimental data and examples indicate that our table detection approach is robust and effective. It works well from simple grid tables to multi-column layout cases, from multi-tables in one given page cases to irregular layout tables' cases.

V. CONCLUSION AND FUTURE WORKS

In this paper, we propose a novel and effective table detection method in PDF documents via both visual separators and content layout analysis. We notice that utilizing purely content layout features cannot achieve satisfactory results for complex page layouts and irregular tables, where the graphic ruling lines are more likely to be explored as boundaries. Our approach has outstanding performance in these cases, even for tables without ruling lines. In the future, further structure recognition will be carried out on the basis of detected table regions.

ACKNOWLEDGMENT

We wish to thank Professor Soon Tee Teoh of SJSU and our colleague Yongtao Wang for their comments on this paper. This work is supported by Liangcai Gao's China Postdoctoral Science Foundation (No. 20100480125) and National Basic Research Program of China, also named "973 Program" (No. 2010CB735908).

REFERENCES

[1] R. Zanibbi, D. Blostein, and J. Cordy, "A survey of table recognition: Models, observations, transformations, and inferences," IJDAR, Vol. 7, Mar. 2004, pp. 1-16.

[2] AC. e. Silva, A.M. Jorge, and L. Torgo, "Design of an end-to-end method to extract information from tables," IJDAR, Vol.8, 2006, pp. 144-171.

[3] T. Kieninger, and A. Dengel, "A paper-to-html table converting system," Proc. of Document Analysis Systems (DAS'98), 1998.

[4] T. Kieninger, and A. Dengel, "Applying The T-Recs Table Recognition System To The Business Letter Domain," Proc. of International Conference on Document Analysis and Recognition (ICDAR'01), 2001, pp. 0518.

[5] B. Yildiz, K. Kaiser, and S. Miksch, "pdf2table: A Method to Extract Table Information from PDF Files," Proc. of Indian International Conference on Artificial Intelligence (IICAI'05), 2005, pp. 1773-1785.

[6] Y. Liu, K. Bai, P. Mitra, and C.L. Giles, "TableSeer: automatic table metadata extraction and searching in digital libraries," Proc. of Joint Conference on Digital Libraries, (JCDL'07), 2007.

[7] E. Oro, and M. Ruffolo, "PDF-TREX An Approach for Recognizing and Extracting Tables from PDF Documents," Proc. of the International Conference on Document Analysis and Recognition (ICDAR'09), 2009, pp. 906-910.

[8] T. Hassan, and R. Baumgartner, "Table Recognition and Understanding from PDF Files," Proc. of the International Conference on Document Analysis and Recognition (ICDAR'07), 2007, pp. 1143-1147.

[9] PDF Reference 1.7

[10] T.M. Breuel, "Two geometric algorithms for layout analysis," Proc. Of Document Analysis Systems (DAS'02), 2002, pp. 188-199.

[11] Y. Liu, K. Bai, P. Mitra, and C. L. Giles, "Improving the Table Boundary Detection in PDFs by Fixing the Sequence Error of the Sparse Lines," Proc. of the International Conference on Document Analysis and Recognition (ICDAR'09), 2009, pp. 1006-1010.

(a) (b) (c) (d)

Figure 5. Expetimental example illustrations