Semi-Global Alignment of Lines and Columns for Robust Table Extraction from periodic PDF Documents

Jorge Kinoshita,

***Abstract*—In a PDF document there is no meta information that describes a table structure; therefore, it is difficult to extract data from a table in PDF documents, mainly when the table is not isolated in a single column page. The extraction is harder when the table is in a multi-column page or it appears beside another table. Given a table tab1 (manually extracted from the document D1.pdf, ex: a balance sheet of 1998) and a PDF document D2.pdf, (ex: the annual report of 1999) we propose to automatically extract a table tab2 (ex: the balance sheet of 1999) inside D2.pdf through a dual (lines and columns) semi-global alignment between tab1 and D2.pdf, even when tab2 is in difficult places. This is specially useful for extracting data from financial reports because a table (e.g. a balance sheet) in a given year is very similar to the corresponding one in the next year. To our knowledge, this paper is the first to propose data extraction from tables in PDF documents through a dual semi-global alignment.**

I. Introduction

It is hard to extract data from tables in Portable Document Format (PDF) files because they have two kinds of information: text and images. All meta information such as where begins and ends a table, a column, a line, or even, a cell are lost when a PDF is generated. Tables (ex: a balance sheet) in financial reports documents, which are generally delivered in PDF, contains the same pattern from year to year. These tables may not differ very much because financial data are generally duplicated in two consecutive years: there are small differences such as some lines may be inserted or deleted, or yet, an item specification may be renamed. We took this feature as a hint to use the alignment between two consecutive financial reports in order to extract data from these tables. Many aligned tables may be merged in a single table in order to avail a company. We propose to manually mark a table tab1 (mark a rectangle, i.e., the upper left corner and the bottom right corner of a balance sheet) in the first document D1.pdf (figure 1, align tab1 with D2.pdf (Ex: a part of D2.pdf in figure 2), get tab2 (table I), align tab2 with D3.pdf, repeat it, get all the tables aligned and join all of them in a single merge-table. The main purpose of this paper is to show how to sequence and align strings from tab1 and D2.pdf by:

1) Retrieving all the strings from tab1. For the first document D1.pdf, we have to manually select a rectangle that specifies where tab1 is located in D1.pdf.
2) Retrieving all the strings from D2.pdf (see figure 2).
3) Sequencing and aligning the sequence of strings from tab1 with the sequence of strings of D2.pdf (see table I).

To our knowledge this is the first paper that uses a dual semi-global alignment in order to make a robust data extraction from PDF documents and therefore, deemed to be the best contribution of this paper. The section II contains related work. The section III contains our alignment algorithm used to extract table from a PDF file. The section IV shows test results. The section V finishes the paper enforcing its robustness.

II. Related Work

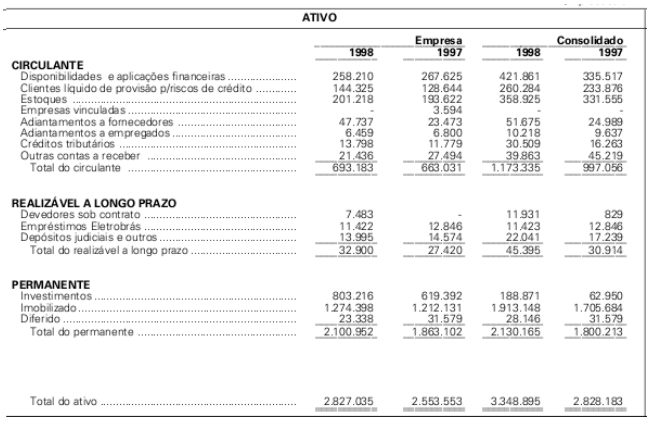
The papers were arranged in 3 areas: 1. Extraction of strings and tables from a single PDF document. 2. Extraction of tables from a series of web pages. 3. Alignment.

A. Extraction of strings and tables from a single PDF document

A PDF document is created to be printed. It contains strings and images. We found some programs that retrieve strings from PDF files in:

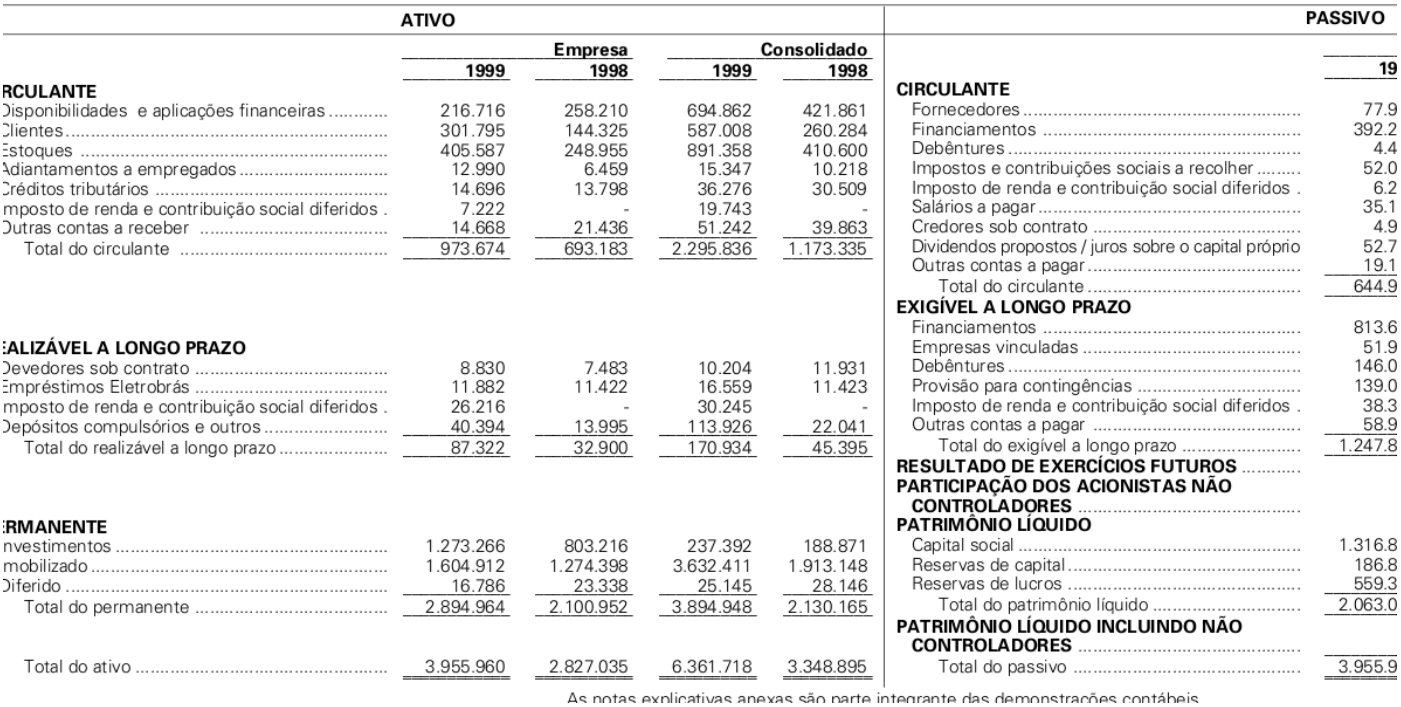
- a XML file, contains for each string, its position (page number, Cartesian coordinates, width and height). Ex: command: `pdftohtml -xml`.
- a free text with strings, spaces and line feeds that tries to keep their original geometric positions. Ex: `pdftotext - layout`.

Jorge Kinoshita is with the Department of Electrical and Computer Engineering, Polythenic School, Sao Paulo, Brazil



ATIVO	Empresa		Consolidado	
	1998	1997	1998	1997
CIRCULANTE				
Disponibilidades e aplicações financeiras	258.210	267.625	421.861	335.517
Clientes líquido de provisão p/riscos de crédito	144.325	128.644	260.284	233.876
Estoques	201.218	193.622	358.925	331.555
Empresas vinculadas	-	3.594	-	-
Adiantamentos a fornecedores	47.737	23.473	51.675	24.989
Adiantamentos a empregados	6.459	6.800	10.218	9.637
Créditos tributários	13.798	11.779	30.509	16.263
Outras contas a receber	21.436	27.494	39.863	45.219
Total do circulante	693.183	663.031	1.173.335	997.056
REALIZÁVEL A LONGO PRAZO				
Devedores sob contrato	7.483	-	11.931	829
Empréstimos Eletrobrás	11.422	12.846	11.423	12.846
Depósitos judiciais e outros	13.995	14.574	22.041	17.239
Total do realizável a longo prazo	32.900	27.420	45.395	30.914
PERMANENTE				
Investimentos	803.216	619.392	188.871	62.950
Imobilizado	1.274.398	1.212.131	1.913.148	1.705.684
Diferido	23.338	31.579	28.146	31.579
Total do permanente	2.100.952	1.863.102	2.130.165	1.800.213
Total do ativo	2.827.035	2.553.553	3.348.895	2.828.183

Fig. 1. tab1 - A piece of 1998-Gerdau-Report.pdf with the table ATIVO.



ATIVO	Empresa		Consolidado		PASSIVO	
	1999	1998	1999	1998		19
RCULANTE					**CIRCULANTE**	
Disponibilidades e aplicações financeiras	216.716	258.210	694.862	421.861	Fornecedores	77.9
Clientes	301.795	144.325	587.008	260.284	Financiamentos	392.2
Estoques	405.587	248.955	891.358	410.600	Debêntures	4.4
Adiantamentos a empregados	12.990	6.459	15.347	10.218	Impostos e contribuições sociais a recolher	52.0
Créditos tributários	14.696	13.798	36.276	30.509	Imposto de renda e contribuição social diferidos	6.2
mposto de renda e contribuição social diferidos	7.222	-	19.743	-	Salários a pagar	35.1
Outras contas a receber	14.668	21.436	51.242	39.863	Credores sob contrato	4.9
Total do circulante	973.674	693.183	2.295.836	1.173.335	Dividendos propostos / juros sobre o capital próprio	52.7
					Outras contas a pagar	19.1
					Total do circulante	644.9
					EXIGÍVEL A LONGO PRAZO	
ALIZÁVEL A LONGO PRAZO					Financiamentos	813.6
Devedores sob contrato	8.830	7.483	10.204	11.931	Empresas vinculadas	51.9
Empréstimos Eletrobrás	11.882	11.422	16.559	11.423	Debêntures	146.0
mposto de renda e contribuição social diferidos	26.216	-	30.245	-	Provisão para contingências	139.0
Depósitos compulsórios e outros	40.394	13.995	113.926	22.041	Imposto de renda e contribuição social diferidos	38.3
Total do realizável a longo prazo	87.322	32.900	170.934	45.395	Outras contas a pagar	58.9
					Total do exigível a longo prazo	1.247.8
					RESULTADO DE EXERCÍCIOS FUTUROS	
					PARTICIPAÇÃO DOS ACIONISTAS NÃO CONTROLADORES	
ERMANENTE					**PATRIMÔNIO LÍQUIDO**	
nvestimentos	1.273.266	803.216	237.392	188.871	Capital social	1.316.8
mobilizado	1.604.912	1.274.398	3.632.411	1.913.148	Reservas de capital	186.8
Diferido	16.786	23.338	25.145	28.146	Reservas de lucros	559.3
Total do permanente	2.894.964	2.100.952	3.894.948	2.130.165	Total do patrimônio líquido	2.063.0
					PATRIMÔNIO LÍQUIDO INCLUINDO NÃO CONTROLADORES	
Total do ativo	3.955.960	2.827.035	6.361.718	3.348.895	Total do passivo	3.955.9

As notas explicativas anexas são parte integrante das demonstrações contábeis

Fig. 2. D2.pdf - A piece of the 1999-Gerdau-Report.pdf with 2 tables: ATIVO and PASSIVO, to be aligned with the ATIVO table of the 1998-Gerdau-Report.pdf.

- a free text with strings that does not keep their geometric positions: Ex: `pdftotext`; `acrobat reader`.

For table extraction, the geometric information is very important. However, `pdftohtml` and `pdftotext` does not always work as expected by the user. The command `pdftotext - layout` committed errors such as deleting table cells, or even the whole table, which prevented us of using this software. The command `pdftohtml -xml` generally outputs one string to each table cell. However, in some cases, it outputs one string to two table cells (for example, it extracted the string "1998 1997" from table tab1) or two strings to one table cell. Although, `pdftohtml -xml` does not always work as expected, we considered it the best one to our purposes. In table extraction from a single PDF document, there is no information whether a string belongs to a table or to a text column; thus a large amount of heuristics may be necessary to automatically extract data from tables. Some papers ([7], [5],[3]) deal with recognizing tables from PDF documents. In general, first lines are recognized, then, columns; and finally tables are assembled. If a table were in a multi-column page or beside another table, this process can lead to many errors; for example, taking a text column as a column of the table; or yet, taking the

tables ATIVO and PASSIVO as a single one (see figure 2). We did not find any paper that deals with the table extraction from a series of PDF documents (such as financial reports).

B. Extraction of tables from a series of web pages.

Web pages, mainly an output from a cgi-script, where information is structured in a repetitive way may be aligned according to their tree structure (parsing their HTML trees and aligning them [8], [10], [9]) or no (as a flat text [1]). None of these papers mention the use of the semi-global alignment: the basic algorithm used in this paper. A closer paper is [8] that makes a partial alignment of trees when extracting data from the web. First, a web page is segmented (using also geometrical information) and then partially aligned. It does not discuss global, local and semi-global alignment as we do here. Our work, however, does not align trees because this meta information (HTML tags that describes tables) does not appear in PDF documents.

C. Alignment

Alignment is closed related to change, or to edit a string1 (a sequence of characters) into another string2 [2]. There are 4 edit operations: **replace** a character of string1 to another character in string2, **insert** a character in string2, **delete** a character of string1, **match** (copy) a character of string1 into string2. A weight is assigned to each operation, for example, match = +1; replace = -1; insert = -1; delete = -1. For example: There are many edit operations to change "CAT" into "BATOU" (example: delete C,A,T; insert B,A,T,O,U). The sum of the weight of each operation is called the similarity; in this case, -8. The alignment algorithm finds the edit operations that yields the highest similarity (ex: replace C,B; match A,T; insert O; insert U; with similarity = -1). A classical algorithm is based on dynamic programming which uses an alignment matrix (see [2]). There are 3 kinds of alignment: global, local e semi-global. An alignment algorithm receives two strings str1 and str2 as input. **Global alignment** finds the best similarity between str1 and str2. In the global alignment from [CAT] to [BATOU], the best similarity is -1. The Needleman–Wunsch algorithm resolves the global alignment using dynamic programming. **Local alignment** finds the best similarity between any substring of str1 and any substring of str2. In the local alignment from "CAT" to "BATOU", the best similarity is 2 (aligning C[AT] with B[AT]OU: match A,A; match T,T). In our case, there is no way to guarantee that any of these substrings is the whole tab1. The Smith–Waterman algorithm resolves the global alignment using dynamic programming. **Semi-global alignment** finds the best similarity between a string str1 and any substring of str2, i.e., it works well if we have a good idea of what we are looking for inside str2; that is, a balance sheet (tab2) as a substring of a financial report (D2.pdf). For example: The best similarity of the global alignment between the whole string CAT and any substring of BATOU is 1 (aligning [CAT] with [BAT]OU: replace C,B; match A,A; match T,T). Semi-global alignment is used mainly in biology, but we found few papers using it in computer engineering; the closer to our work is [6] that uses semi global alignment to aid a robot read text, because the camera can focus on a part of the text, however, it does not deal with PDF documents, neither table alignments.

III. Table Extraction using the Alignment of Lines and Columns

We decided to use an open source library called `poppler` in order to extract strings from a PDF document because it is a very mature software. The command `pdftohtml -xml` makes use of this library and generates a XML output that shows for each string its geometric position in the document page. We had to modify the library. Generally `poppler` outputs one string for each table cell. However, sometimes `poppler` makes two kinds of error: 1. join two table cells or 2. split one table cell in two strings. We changed the source code of the library `poppler` to show the second kind of error because it leads to errors that are easier to deal: it is easier to fix errors in a spreadsheet with more columns (because two strings derived from an original table cell are in two columns) than to deal with interwoven columns. From now on, we are calling "cell" to each string that comes from `pdftohtml -xml` independently of it being a real table cell. We did not use the term "string" because it has been used in a broader sense in this paper. The alignment algorithms (ex: global, local and semi-global) are made to align strings which are uni-dimensional structures. It is possible to use these algorithms depending on how strings are extracted from a PDF document. A global alignment can be done for two documents which are very alike, i.e., which have same sequence of topics and tables. The Needleman–Wunsch algorithm deals with string and characters; here we deal with document (mapped as strings) and lines or cells (mapped as characters). This algorithm works fine when the sequence of the line and cells of a whole D1.pdf aligns with the sequence of lines and cells of a whole D2.pdf, but there are many cases in which it does not happen: let us suppose D1.pdf is a single column document but D2.pdf (with almost the same

sequence of topics) is a double column document: how can we guarantee a sequence of lines and cells that aligns D1.pdf to D2.pdf? A tool such as `pdftohtml -xml` only extracts strings (taken as cells) and their geometric positions, thus we have to rely on the geometry in order to align both documents. Another problem is that there is no guarantee that the sequence of topics and tables happens for two documents. For example: It is common that tables float in a document. A table may be placed in the top or bottom of a page depending on the free space, the size of the table, the layout of the page, etc. Thus, it is difficult that two documents have the same sequence of topics and tables. A semi-global alignment algorithm is useful for searching a given table tab1 in a document D2.pdf. It has lesser restrictions than the global algorithm because the sequence of topics does not matter, but even still we may find problems: let us suppose that tab1 is side by side another table as it may happen in figure 2. In this case, it is not possible to guarantee that we may find all the cells of tab2 together, concentrated as a sub sequence of cells in D2.pdf, because the cells of the table ATIVO may be mixed to the cells of table PASSIVO as in D2.pdf (figure 2), preventing a good alignment. The solution presented in this paper consists of using their geometric positions to semi-globally align tab1 and D2.pdf even when tab2 is in a very difficult location such as besides another table. The basic idea is to find an anchor cell, i.e., a pair of cells: cell1 of tab1 and cell2 of D2.pdf where cell1 best aligns with cell2 in D2.pdf and through them build the alignment. In order to accomplish this, it is necessary: 1 - organize the cells of tab1 and D2.pdf as a sequence of lines and columns,(section III-A) 2 - make the semi global alignment between lines and columns, (section III-B) 3 - take the anchor cell III-C and 4 - based on this anchor-cell alignment and in the sequence of lines and columns of tab1 and D2.pdf, derive other cells alignments (section III-D).

A. Sequencing the cells of the PDF document in lines and columns.

There are 3 ways to sequence the cells of tab1 and D2.pdf:

1) by the order provided by `pdftohtml -xml` but this order is not reliable, sometimes the software outputs the strings in the human order reading, other times, does not.
2) by line: Taking the cells line by line, first from top to bottom and then, from left to right. The line-sequence for D2.pdf (figure 2) is: (..., L, *Disponibilidades e aplicações financeiras*, *216.716*, *258.210*, *694.862*, *421.861*, L, *Fornecedores*, *77.987*, etc.), i.e., the cells from 2 tables (ATIVO and PASSIVO) are mixed.
3) by column: Taking the cells column by column, from left to right and then, from top to bottom. The basic idea is to order the sequence of cells according to the X-coordinate, but this may not work. Columns may be right aligned (ex: column of numbers), left aligned (ex: items description) or center aligned. What X-coordinate should we take in order to sequence the cells according to the columns? The basic rule for the column alignment is: given 2 cells A and B, if there is an intersection between the X-coordinates of A and B, then, the upper cell comes first. If there is no intersection between the X-coordinates of A and B, then the left cell comes first. Example: sequencing tab1 according to the columns yields: (CIRCULANTE, "Disponibilidades e aplicações financeiras", "Clientes líquido de provisão p/riscos de crédito", ...).

We sequenced the cells of tab1 and D2.pdf and arranged as lines and columns, thus we may see tab1 and D2.pdf as a spreadsheet similar to figure 3.

B. Semi global alignment between lines and columns

The Needleman–Wunsch and Smith–Waterman ([2]) aligns two strings based on their characters. Here, we semi-globally align a sequence of cells cell1 (a line or a column) of tab1 to another sequence of cells cell2 of D2.pdf. In order to find the higher similarity between a line or a column, it is necessary to define the weights of the operations: replace/match cell1,cell2; insert cell2 and delete cell1.

We are assigning -1 to the delete and insert operations. For the replace/match operation, the weight will depend on the kind of cell1 and cell2. One table is composed of cells with letters (ex: *Fornecedores*) that generally is part of the table head and cells without letters, generally numbers. We want to align mainly the heads at tab1 and tab2, thus, we are going to assign a higher weight for the similarity of two cells with letters, but it depends on how similar cell1 and cell2 are. We define the weight of the match/replace operation, in this case, as:

-20 + (1 - distance($str1,$str2)/greater($str1,$str2))*40;

which yields a number between -20 and 20. where:

- greater($str1,$str2): length of the greater string: $str1 or $str2.
- distance($str1,$str2): edit distance between str1 and str2. The edit distance is a number that varies from zero (total match) to greater($str1,$str2).

The weight to replace/match a cell without letters (ex: *216.716*) to another without letter is 1. The weight to replace a cell with letter with a cell without a letter is -20 (therefore a delete or insert operation is performed).

Let us represent tab1 and D2.pdf as a spreadsheet in figure 3 where cells of tab1 are in columns A-G and the cells of D2.pdf are in columns H-N. Then we semi-globally align all pairs ((1,1),(2,1) ...) getting a line alignment score for each pair: line-score(1,1), etc. The same idea is done to align columns yielding: column-score(A,H),column-score(A,I), etc. In order to obtain a line-score, we assemble a semi-global alignment matrix with the cells in each line. For instance, for aligning (1,1) the matrix is created taking (A1, B1, C1, ... G1) x (H1, ... N1). The matrix is composed of pairs: (A1,H1), (A1,I1), etc. For each pair (A1,H1) we apply the weights for the operations: replace/match, delete and insert. The same procedure happens to find the column-scores.

We notice that our first idea was not to consider column-scores because it is possible to have some idea of the tab2 location using only line-scores. However this region is not very well defined. For instance, we take the line ("CIRCULANTE") in tab1 and align with the line ("CIRCULANTE", "CIRCULANTE") in D2.pdf where the first "CIRCULANTE" is from the ATIVO table and the second is from the PASSIVO table (see figure). It is not very clear which columns of D2.pdf corresponds to tab2. We thought of using some statistics and line alignments to discover where tab2 is placed; but an easier way to discover the tab2 location is to use column alignments too and through it, have an anchor cell, a cell1 in tab1 that aligns with a great confidence with a cell2 in D2.pdf.

C. Finding the anchor-cell: the best cell1 and cell2 alignment

A cell C9 belongs to a line 9 and a column C of tab1 (see figure 3). If both alignments 9-6 and C-K, aligns C9 to the same K6 cell in D2.pdf then we are going to define:

cell-score(C9,K6) = line-score(9,6) * column-score(C,K)

The anchor-cell may correspond to some special cells of tab1. In figure 2 there is a table which have the first column as item descriptions; all other cells are numbers. Due to the higher weights for aligning cells with letters (section III-B), the similarity of the first column will be higher, therefore, the anchor cell is taken from this column. Another example is a table which have an item description column and a line item description as heads. In this case, the anchor cell is very likely to be in the corner of these head line and head column. Example: The best anchor cell in the tab1-D2.pdf is 'Adiantamentos a empregados'. Through the line alignment 9,6 we obtain many other cell alignments; for instance (D9,L6), (E9,M6). This line alignment will be named the "guide line". Through the column alignment C,K we obtain many other cell alignments, for instance (C6,K3), (C7,K4). This column alignment will be named the "guide column". These other cell alignments will be used to generate the table alignment.

D. Generating the Table Alignment.

We may deduce other cells alignments such as E6-M3 in figure 3 through the guide line and guide column. Due to the 9-6 alignment, if E9 aligns with M6, then column E aligns with column M. Due to the C-K column alignment, if C6 aligns to K3, then line 6 aligns with line 3. Therefore, we deduce that E6 aligns to M3. Thus, many other alignments are extracted, even with empty cells (a cell is inserted or deleted). If there were no cell in the intersection of line 3 and column M; then cell E6 would be deleted in D2.pdf, i.e., E6 aligns to an empty cell. Example: Table I shows some alignments that were extracted through this method. This method of aligning cells avoids some inconsistencies. For example, as "Empresas vinculadas" of tab1 did not column-aligned with any cell of D2.pdf, all cells corresponding to the line "Empresas vinculadas" in tab1 are simply removed. Some cells of tab1 may not align with the proper cells of tab2 in D2.pdf because their lines or their columns do not cross the guide line or the guide column. For instance the line of "1998 1997" cell in tab1 does not cross the guide line. The cell alignments found in the former phase correspond to places in the line alignment matrix and in the column alignment matrix when using a dynamic algorithm for semi-global alignment. For instance, the column alignment matrix has a cell that corresponds to

string1	string2
CIRCULANTE	CIRCULANTE
Disponibilidades e aplicações...	Disponibilidades e aplicações
Clientes líquido de provisão ...	
	Clientes .
Estoques	Estoques
Empresas vinculadas	
Adiantamentos a fornecedores	
Adiantamentos a empregados	Adiantamentos a empregados
693.183	973.674
1.800.213	2.130.165
2.828.183	3.348.895

TABLE I. PARTIAL RESULT OF THE TAB1-D2.PDF ALIGNMENT. IN COLUMN STRING2, THERE ARE THE STRINGS OF TAB2.

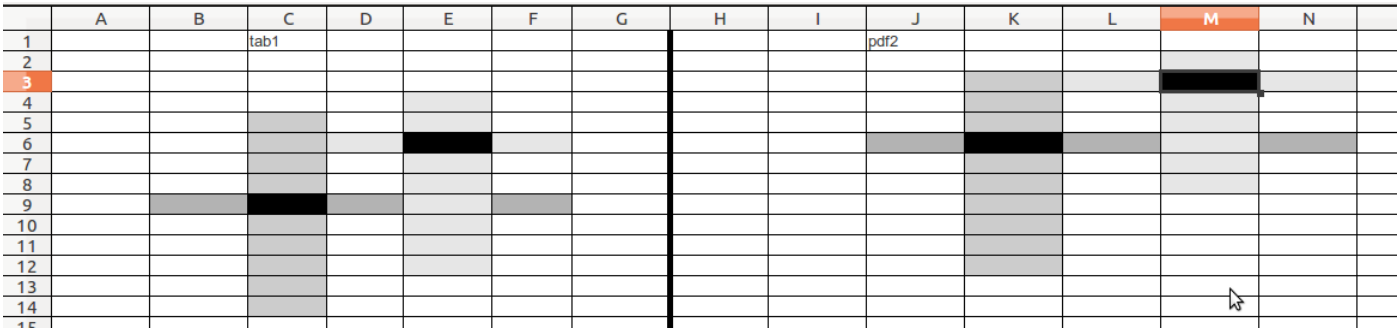


Fig. 3. Cell C9 of tab1 aligns to K6 of D2.pdf.

a reliable alignment of 258.210 in tab1 with 216.716 in D2.pdf. Based on this alignment, we can "stretch" the path, i.e., we may trace back through this alignment cell in the column semi-global alignment matrix discovering other alignments that could not be taken in the former phase. See chapter 11 of [2] for a reference about an alignment matrix.

IV. Tests

We applied the semi-global alignment to merge balance sheets from Gerdau. We expected that the algorithm would work fine in most cases. In the wrong cases, we expected that it would be easy to fix the merge-table through some macros of openoffice that merge or split lines and columns; but we had harder problems. We notice that:

- the semi-global alignment is very time and memory consuming because it assembles all matrices of lines and columns to all pages in a D2.pdf. In order to improve the algorithm, we automatically search for the page where the table is and apply the algorithm to this page only. We tested some mechanisms to automatically find the page that contains a given table. These mechanisms see pages and tables as bag of words. We concluded that it is better to use a ranker (pages that have more words of tab1) than a classifier (train a classifier to respond yes or no using pages that has the table and pages that has not the table. As table floats in the pages, the page text that contains the table is not very precise for training and testing).
- in some cases, the first or the last line of tab1 does not properly align to the correct line of D2.pdf. In our experiments, the first or the last line was not aligned to any line of D2.pdf. The consequence is that tab2 looses its beginning or its ending line becoming smaller than it should be. When aligning tab2 to a D3.pdf, the error is propagated. We tried some tricks to automatically fix this problem, but we noticed that the algorithm was becoming specialized in financial tables. Thus we decided to manually mark the beginning and ending of each table. This is the biggest problem in our algorithm because, although we always found a good anchor-cell in our tests, the automatic table extraction of many documents was not always well succeeded leading to error propagation.

V. Conclusion

To our knowledge, this is first paper to use semi global alignment to extract table (ex: a balance sheet from a certain year) from a series of PDF documents (ex: annual reports). We showed how to extract data from a balance sheet, even in very difficult cases: when a table is side by side another table or column. We accomplished it using two semi-global alignments: lines and columns. We intend to build a software to process PDF documents changing it as a sequence of lines and columns, enabling operations such as join lines, split columns, and even, building a merge-table. A critic to our work is that we tested the algorithm and we got some problems as in the section IV. To the future, we may find a way to mark the table region according to: 1. the geometry of the cells (a line or column of cells that are parallel to the discovered tab2). 2. lines and columns in the border of tab1 that were not aligned. In a former work [4], we show a method to align two strings of lengths m and n based on the global alignment algorithm but implemented using adaptative technology. It is seen that this algorithm could run in paralell and therefore, in a time proportional to m+n and not m*n. This result can easily applied to this work that uses the semi global alignment, a variant of the global alignment algorithm, however, [4] does not use semi global

References

[1] D. Guo, D. Li, M. Liu, Y. Liu, and S. Chen. The dynamic web pages information extraction algorithm based on sequence

alignment. *Computational Sciences and Optimization, International Joint Conference on*, 1:784–786, 2009.

[2] D. Gusfield. *Algorithms on strings, trees, and sequences: computer science and computational biology*. Cambridge University Press, New York, NY, USA, 1997.

[3] T. Hassan. Object-level document analysis of PDF files. In *Proceedings of the 9th ACM symposium on Document engineering*, DocEng '09, pages 47–55, New York, NY, USA, 2009. ACM.

[4] J. Kinoshita and R. L. Rocha. ALINHAMENTO DE DUAS CADEIAS USANDO UM TRANSDUTOR ADAPTATIVO. In *Memórias do WTA'2009 : terceiro Workshop de Tecnologia Adaptativa / Laboratório de Linguagens e Técnicas Adaptativas.*, pages 78–82, 2009.

[5] E. Oro and M. Ruffolo. PDF-TREX: An Approach for Recognizing and Extracting Tables from PDF Documents. In *Proceedings of the 2009 10th International Conference on Document Analysis and Recognition*, ICDAR '09, pages 906–910, Washington, DC, USA, 2009. IEEE Computer Society.

[6] B. Suresh, C. Case, A. Coates, and A. Y. Ng. Autonomous Sign Reading for Semantic Mapping. 2011 IEEE International Conference on Robotics and Automation, May 2011.

[7] B. Yildiz, K. Kaiser, and S. Miksch. pdf2table: A Method to Extract Table Information from PDF Files. *Proceedings of the 2nd Indian International Conference on Artificial Intelligence IICAI05*, 2005.

[8] Y. Zhai and B. Liu. Web data extraction based on partial tree alignment. In *Proceedings of the 14th international conference on World Wide Web*, WWW '05, pages 76–85, New York, NY, USA, 2005. ACM.

[9] S. Zheng, R. Song, J.-R. Wen, and D. Wu. Joint optimization of wrapper generation and template detection. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '07, pages 894–902, New York, NY, USA, 2007. ACM.

[10] P. Zigoris, D. Eads, and Y. Zhang. Unsupervised Learning of Tree Alignment Models for Information Extraction. In *Proceedings of the Sixth IEEE International Conference on Data Mining - Workshops*, pages 45–49, Washington, DC, USA, 2006. IEEE Computer Society.