HAL open science

Case-based reasoning for invoice analysis and recognition

Hatem Hamza, Yolande Belaïd, Abdel Belaïd

To cite this version:

Hatem Hamza, Yolande Belaïd, Abdel Belaïd. Case-based reasoning for invoice analysis and recognition. 7th International Conference on Case-based Reasoning - ICCBR 2007, Aug 2007, Belfast, United Kingdom. pp.404-418, 10.1007/978-3-540-74141-1_28 . inria-00176632

HAL Id: inria-00176632

https://inria.hal.science/inria-00176632v1

Submitted on 4 Oct 2007

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Case-based Reasoning for Invoice Analysis and Recognition

Hatem Hamza[1,2], Yolande Belaïd[2], and Abdel Belaïd[2]

[1] ITESOFT, Parc d'Andron, Le Sequoia. 30470 Aimargues. France.
[2] University Nancy 2, LORIA. 54506 Vandoeuvre-les-Nancy, France.
{hamza,ybelaid,abelaid}@loria.fr

Abstract. This paper introduces the approach CBRDIA (Case-based Reasoning for Document Invoice Analysis) which uses the principles of case-based reasoning to analyze, recognize and interpret invoices. Two CBR cycles are performed sequentially in CBRDIA. The first one consists in checking whether a similar document has already been processed, which makes the interpretation of the current one easy. The second cycle works if the first one fails. It processes the document by analyzing and interpreting its structuring elements (adresses, amounts, tables, etc) one by one. The CBR cycles allow processing documents from both knonwn or unknown classes. Applied on 923 invoices, CBRDIA reaches a recognition rate of 85,22% for documents of known classes and 74,90% for documents of unknown classes.

Key words: case-based reasoning, document case, structure case, invoice analysis, invoice interpretation, structure extraction.

1 Introduction

Form and invoice analysis systems in real production chains are often faced with a huge quantity of documents requiring a high processing speed and a continuous adaptation capacity to the structure variation. The manual and even semi-automatic solutions which consist in building manually the model of each set of new documents can no longer be used because of the heavy modeling phase they require [1].

Invoices have variations depending on many factors : the company issuing the invoice, the client, etc. Most current information to be extracted are: addresses (delivery, billing...), total amounts and table lines showing details of services, purchased products... Two types of documents occur in invoice processing:

- documents of known class i.e similar documents have been already processed;
- new documents from an unknwon class.

In two documents from the same class, information blocks (addresses, amounts, tables...) are organized in the same way and have the same relative positions in the documents. However, their absolute positions vary from a document to another, depending on the specific content of each document. Figure 1 shows two

documents from the same class where the information to be extracted are gray tone or boxed. We can see that the absolute position of the total amount zone changes between the documents.

For a new document from an unknown class, the problem consists in building a generic and reliable model for all the documents of this class. Figure 2 shows two invoices from different classes. We can clearly see that they have different structures. The paper is organized as the following: section 2 presents some related works. Section 3 introduces the use of CBR the system. Sections 4, 5 and 6 present CBRDIA's architecture. Finally, the seventh section shows the obtained results and their interpretation.

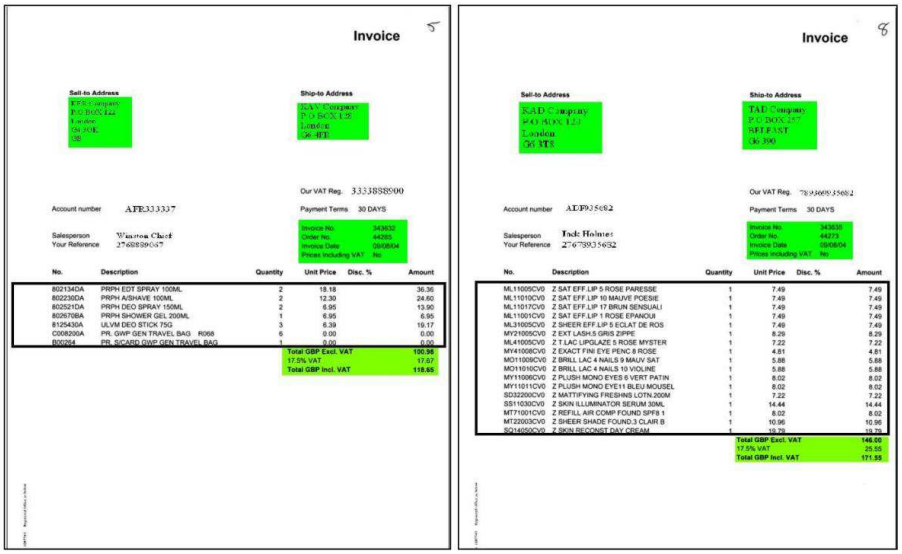


Fig. 1. Two invoices from the same class

2 Related Work

The most promising approaches are those which can process documents of either known or unknown classes [2] [3]. In [3], after a first step related to document classification, the document is interpreted via its structures (keywords) by combining two levels of knowledge: intra- and inter-classes knowledge. If the document class is recognized, the system looks for the solution using the intra-class knowledge (tags, relative positions of the related object, etc) and the inter-class knowledge (summarizing knowledge in different invoice classes). If the document is not recognized, then only inter-classes knowledge is used to interpret the extracted

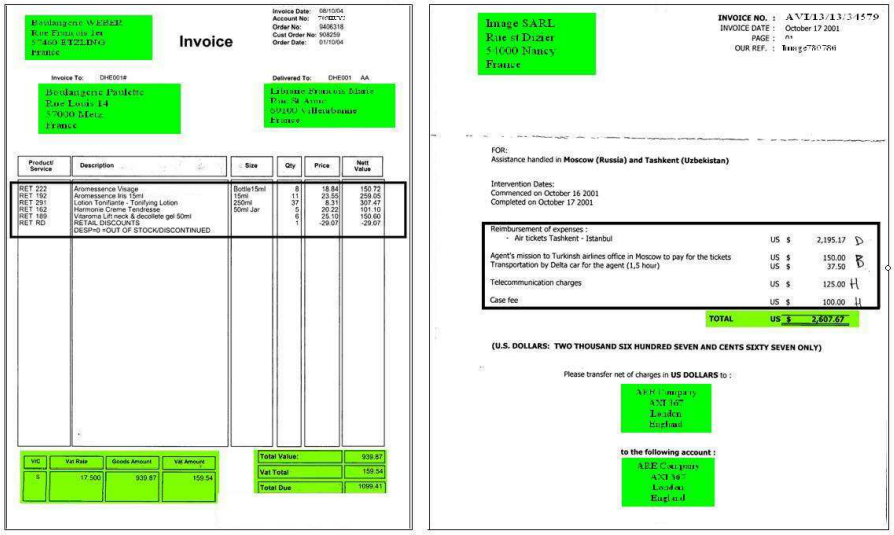


Fig. 2. Two invoices from different classes

information. The application of this approach is however limited to isolated keywords not taking into account more frequent and important structures in forms such as tables, addresses, etc. Concerning table analysis, Belaïd presented in [4] a morphological tagging approach for invoice analysis. This approach was used in order to tag table columns and fields. However, the processed tables are already extracted before tagging. Contrary to these methods, CBRDIA extracts and interprets data associated with both table lines and keywords. It can also process documents from both known and unknown classes.

To our best knowledge, no directly related work has been published in CBR field. However, we can link this work to other works on textual CBR (TCBR) [5] or on CBR in image processing [6]. In our approach, our cases will be represented either by strings or graphs. Cunningham [7] shows that the use of graphs in case representation can be useful in TCBR.

Another type of related works concerns systems using multiple CBR reasoners. In CBRDIA, we will use 2 CBR reasoners (one for invoices of known class, and another for invoices of unknown class). These reasoners will be sequential. Some other approaches [8] use parallel CBR reasoners in order to enhance the system performance.

3 Case-based Reasoning in our Approach

CBR is a solving strategy that uses previous experiences to process new problems that have not been processed before [9].

The problems we are facing in this work are the following:

- document structure extraction: this is a difficult and time consuming problem in industry. Structure extraction is done for every document in order to be interpreted. However, when a whole set of documents (coming from the same client) has the same structuring elements (example : a table, an amount block, and an address block), then the whole set can be represented by a generic model. This is generally done by a user who takes into account a certain number of documents in order to build such a model.
- document classification: there is a continuous flow of documents that have to be processed (read and interpreted). We do not know a priori to which class of documents the processed one belongs to. It is obvious that if the system has processed a similar document before, then it is a real waste of time not to take advantage of such a knowledge. Otherwise, a new document model has to be built in order to extract the desired information.
- document analysis and interpretation: this task (interpreting words, fields, or tables) is really hard. It has to be done either by a user who supervises every document, or automatically by a reliable system. For example, interpreting the word "total" means associating it with the numerical value related to it on the document. The system, in the interpretation phase should
  - generalize easily based on the previous document processing experiences;
  - understand the current document and make profit of the extracted and interpreted information it contains;
  - be as quick as possible. If possible, we have to avoid a classical training process as other machine learning techniques;
  - self adaptable to any new class of documents. In classical machine learning techniques (example: neural networks), as soon as a new class of data appears, these techniques fail generally in recognizing it. A new learning has to be done in order to overcome these difficulties. However, in CBRDIA, when the document class is completely new, the system can find solutions (partial solutions if not total ones) by trying to exploit the previous knowledge.

For all these reasons, the choice of CBR was natural. In CBRDIA, two sorts of cases are defined: a document case and a structure case. As shown in figure 3, the flow of our approach is based on three main steps: problem elaboration, global problem solving and local problem solving.

Problem elaboration consists in indices extraction from the document. These indices are either keywords (KW) and their spatial relationships, or table rows. This problem is then solved using either global solving process or local solving one.

Global Solving (first CBR cycle on figure 3) consists in checking if a similar document case exists already. If yes, then the system solves the problem by applying the solution of the database case to this problem. Otherwise, the problem is decomposed into sub-problems, and solved via the solving of its sub-problems. The second CBR cycle corresponds to this step, and is called local solving. The use of global solving and local solving makes our system able to process any kind of invoice documents.

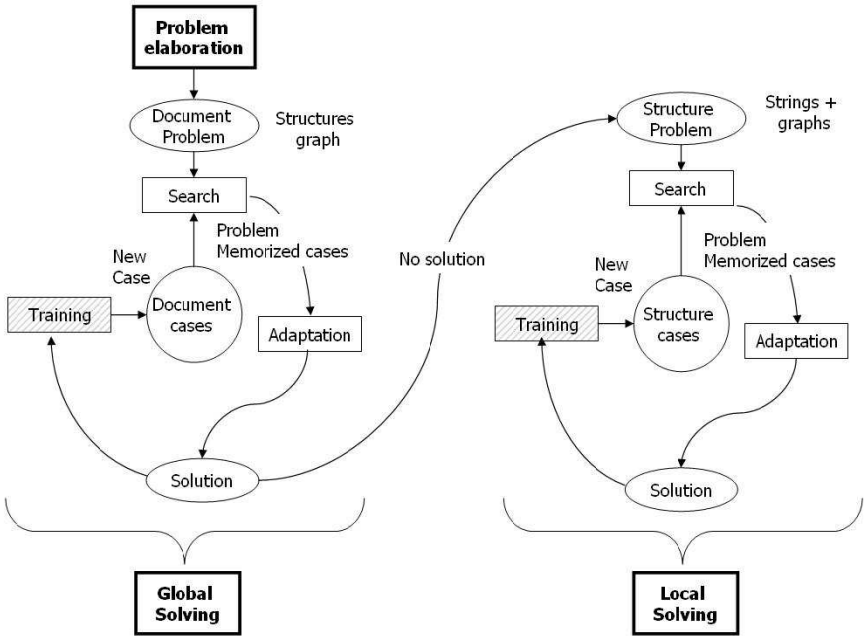


Fig. 3. CBRDIA flow

4 Problem Elaboration

The system requires the precise definition of the problem. This precision is required in every step of the flow (case retrieval, solution adaptation). The system input is a raw document given by OCR (optical character recognition). The OCR file written in XML contains the list of words and their coordinates. The document is represented by a set of words $D = W_i$, $i = 1..n$.

4.1 Data Extraction and Coding

First, each word W is given a list of attributes:

- position (coordinates in the document);
- KW: if a word in the current document matches a word in a predefined list of keywords, then it is tagged as a KW. This list is enriched gradually as new keywords are discovered;
- nature: represented by an alphabetic character. For example, 'A' for numerical, 'B' for alphabetical, etc.

In the next step, fields (F) are constituted from the set of words D by gathering neighbour words horizontally. Each successive pair of words $d(W_i, W_j)$ in F verifies $d(W_i, W_j) < \alpha$ where α is a threshold depending on the words' size. F is also characterized by a list of attributes:

– position;
– nature: the nature of a field is deduced from its words' natures. For example, if F contains an alphabetical and a numerical word, then it will be given the tag 'C' for alphanumerical.

From fields, we extract horizontal and vertical lines (HL, VL). We use the fields' neighbourhoods and the fields' alignments to constitute HL and VL. A vertical line VL is a set of fields F vertically aligned. Two vertical fields $d(F_i)$ and $d(F_j)$ are in the same vertical line if $d(F_i, F_j) < \delta$ where δ is a threshold depending on the fields' size and position. Similarly, we use a threshold for horizontal fields. A line is described by the following attributes:

– position;
– pattern: a string composed of fields' tags list. For example, if the fields' tags in the line are: 'A', 'B', 'B' and 'C', then the pattern is "ABBC". These patterns will be very helpful in the extraction of tables.

Figure 4 shows an example of a field, a HL and a VL. After these elementary information are extracted, we can extract high level structuring elements (S) which can be either pattern structures (PS) when related to tables or keyword structures (KWS) when related to local arrangements of keywords (KW). The final document problem will be defined thanks to PS and KWS.

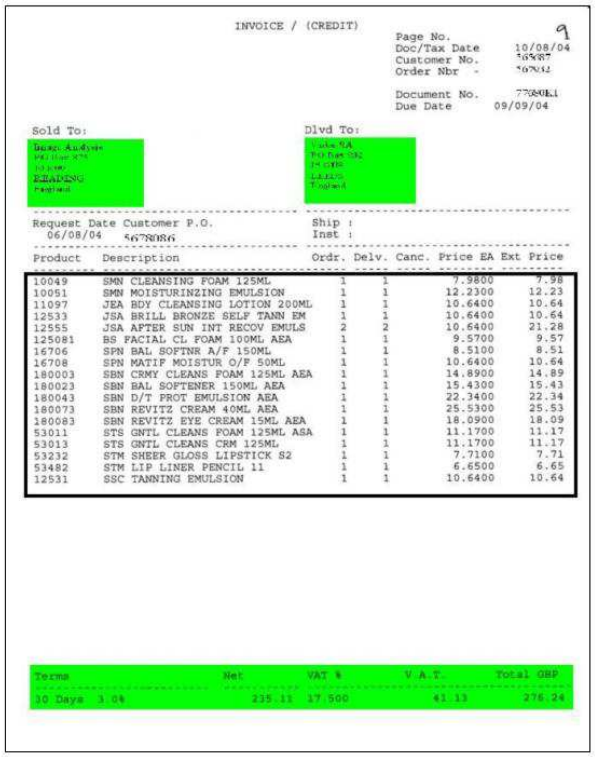
Fig. 4. A VL in the big box, a HL in gray tone, a field in the small box

4.2 Structures Extraction

Figure 5 shows a document containing 3 KWS and a PS.

PS Extraction. PS are a list of consecutive HLs having similar patterns. This is the case of a table. Figure 5 shows a document containing a PS composed of 18 HLs having the pattern "ABAAAA". The PS detection process contains three steps:

– For each HL, we constitute a list of HL neighbours HLN using edit distance on their strings (i.e. patterns). We use a threshold (usually equal to 1 in order to accept only 1 transformation between strings) between HL patterns to find neighbours;
– The list of each HL neighbours is studied based on the fields' positions. In figure 6, the edit distance between the patterns is null, as they represent



INVOICE / (CREDIT)

Page No. 9
Doc/Tax Date 10/08/04
Customer No. [illegible]
Order Nbr - [illegible]

Document No. [illegible]
Due Date 09/09/04

Sold To:

Dlvd To:

Request Date 06/08/04
Customer P.O. 56780886

Ship :
Inst :

Product	Description	Ordr.	Delv.	Canc.	Price EA	Ext Price
10049	SMN CLEANSING FOAM 125ML	1	1		7.9800	7.98
10051	SMN MOISTURINZING EMULSION	1	1		12.2300	12.23
11097	JEA BDY CLEANSING LOTION 200ML	1	1		10.6400	10.64
12533	JSA BRILL BRONZE SELF TANN EM	1	1		10.6400	10.64
12555	JSA AFTER SUN INT RECOV EMULS	2	2		10.6400	21.28
125081	BS FACIAL CL FOAM 100ML AEA	1	1		9.5700	9.57
16706	SPN BAL SOFTNR A/F 150ML	1	1		8.5100	8.51
16708	SPN MATIF MOISTUR O/F 50ML	1	1		10.6400	10.64
180003	SBN CRMY CLEANS FOAM 125ML AEA	1	1		14.8900	14.89
180023	SBN BAL SOFTENER 150ML AEA	1	1		15.4300	15.43
180043	SBN D/T PROT EMULSION AEA	1	1		22.3400	22.34
180073	SBN REVITZ CREAM 40ML AEA	1	1		25.5300	25.53
180083	SBN REVITZ EYE CREAM 15ML AEA	1	1		18.0900	18.09
53011	STS GNTL CLEANS FOAM 125ML ASA	1	1		11.1700	11.17
53013	STS GNTL CLEANS CRM 125ML	1	1		11.1700	11.17
53232	STM SHEER GLOSS LIPSTICK S2	1	1		7.7100	7.71
53482	STM LIP LINER PENCIL 11	1	1		6.6500	6.65
12531	SSC TANNING EMULSION	1	1		10.6400	10.64

Terms	Net	VAT %	V.A.T.	Total GBP
30 Days 3.0%	235.11	17.500	41.13	276.24

Fig. 5. An invoice containing 3 KWS and a PS

the same string "ABB". However they do not correspond to the same PS because of the difference of the spatial positions. To avoid such confusions when the edit distance is null, we take into account patterns' fields positions as the following. For every list HLN we compute a new matching value. This value depends on the number of exact vertical alignment of fields having the same tag. The final matching value is the ratio in (1):

$$RT = \frac{|matching\ fields|}{|fields\ in\ HLN|} \text{where } |X| \text{ is the number of elements in} X \ . \quad (1)$$

The higher RT is, the more probable HLN is a PS. In fact, if RT tends to 1, then two possibilities exist:

- $RT = 1$, HLN is a singleton (this case will be eliminated because it is meaningless for table), or HLN is a perfect table;
- $RT < 1$, meaning the case of a possible table.

– After processing the whole document, the chosen HLN is the one maximizing RT. PS is then the best HLN candidate. This method can extract tables only when there are at least two table lines in the document.

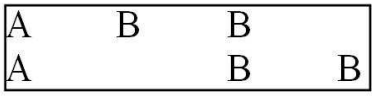


Fig. 6. Two patterns with edit distance=0

KWS Extraction. Keyword structures (KWS) are local arrangements of keywords (KW) like "road", "zip-code", "name", etc for an address. These KW occur frequently in administrative documents and can be in several languages. KW are extracted thanks to a specific software developped by ITESOFT. Its details are outside the aim of this paper. KW can be written in different manners but have always the same meaning. For example, "total", "tot", "total amount" represent the same information but are written differently from an invoice to another. In order to avoid confusions and to be able to propose general cases, KW with the same meaning are given the same KW tag. We use graphs to represent this keyword association (keywords in vertices, and relative spatial relationships on edges). This association maintains the real positions of KW in the document as well as the semantic proximity between them. We preferred using relative positions instead of absolute positions when tagging the edges in order to have a better generalisation of a case. For example when a homogeneous set of documents is processed, it is usual that the absolute positions of a KW changes from a document to another one. However, its relative position to the other KW in the document does not change.

4.3 Document Structure Extraction

A document structure is a gathering of all its sub-element structures (PS and KWS). We use a graph for its representation. In order to have a harmonious graph representation, useful for future comparisons, we consider all the vertices visible at the same level. This means that the difference between vertices is characterized by edges which are either "spatial" (left, right, top, bottom) when they designate spatial relationships or "contain" when they designate a structure component (as between a KWS and each of its KW). This kind of graph representation gives flexibility to CBRDIA as it is just articulated around KWS and PS relative positions. It is also helpful for document comparisons (see 5.1). We preferred the graph representation to the vector representation as the latter does not take into account any position in the document. Moreover, classical vector representations do not give any information about the position in the document, nor about the relative position with another KW. In addition to all that, [10] [7] show clearly that using the graph representation gives much better results in document classification than the vector based representation. All these reasons lead us to represent KWS and documents in graphs.

4.4 Problem Enriching Using a Set of Homogeneous Documents

In the previous sections, we introduced problem elaboration starting from one single document. This can be sufficient when the extraction is done easily and when the proposed document does not contain any noise (very good OCR results, tables with multiple lines). However, this is not usually the case. In many cases, extracting a document problem without checking whether this problem is representative of the set of documents it comes from can cause many errors in the solving process.

In this paragraph, we show how to extract and to enrich a problem starting from a set of documents. A set of documents means a set of homogeneous documents i.e they are all issued by the same company, they have the same physical structure. The similarity of documents in the same set is high and processing one document can help a lot in processing the remaining documents. In order to help having available and representative problems, we use a whole set of documents in the problem elaboration process. The final problem is representative of all the problems of the processed documents. The system extracts the problem from each document in the set and adds this problem to the previous ones. As the document problems are graphs, the final problem is a graph representing all the extracted graphs. Such a graph can only be the Minimum Common Supergraph (MCS) of all the extracted graphs. Formally speaking, strating from a set of graphs G_i, $i = 1..n$ an MCS is a graph such that it has a subgraph isomorphism with every graph in G_i. This MCS is very helpful. It represents the whole class of documents and allows a better generalization in both the steps of elaboration and solving.

Bunke [11] introduces the notion of weighted MCS. It is an MCS where the vertices and edges have weights corresponding to their frequencies in the set G_i. These weights can be useful as they allow the distinction between real information and noisy information. A noisy information (a vertex which can be in our application a keyword, a KWS or a PS) is usually characterized by a very low frequency. By using a threshold, we can filter the undesired information.

The redundancy of some information in the problem enriching phase can also help finding and modelling future solutions. For example, the redundancy of the field "phone number + XX.XX.XX.XX.XX" can be helpful in the solving process. If this redundancy is detected, the solution of the KW "phone number" is known in advance, and it becomes unnecessary to look for it once the solving phase starts. similar ideas concerning problem enriching are under study.

4.5 CBRDIA Cases

CBR requires the definition of cases: a problem and its corresponding solution. Let C be a case, C= {P_C, S_C}. According to the problem elaboration step, two cases are possible:

1. Structure case which can be a KWS case or a PS case

- KWS case: where P_KWS is the graph of keywords contained in a structure and S_KWS is the interpretation of each KW. For example, the solution of the KW "street" is the name of the street and the number corresponding to the address (example: 12 Decker street). In this case, KWS solution is the set of KW solutions. S_KWS = {S_KW} where KW is a particular case of KWS containing just one keyword;
- PS case: where P_PS is the pattern (e.g "ABBB") representing the table and S_PS is the interpretation of each table column.

2. Document case: P_DC is the document graph and S_DC is the solution of all its structures: S_DC = {S_KWS, S_PS}. P_DC consists in the graph of all the structures of the document. The graph vertices are the structures or the kewords contained in the structures. Two different edge labels exist. If the edges connect between structure vertices or between KW inside a KWS, then their labels are spatial labels (above, below, right, left) representing the relative positions of structures (arrows in full line on figure 7). Otherwise, their labels are "contain" type, meaning that a certain information is contained in a structure (arrows in dotted line on figure 7).

Figure 7 shows a simple example of a graph of a document problem.

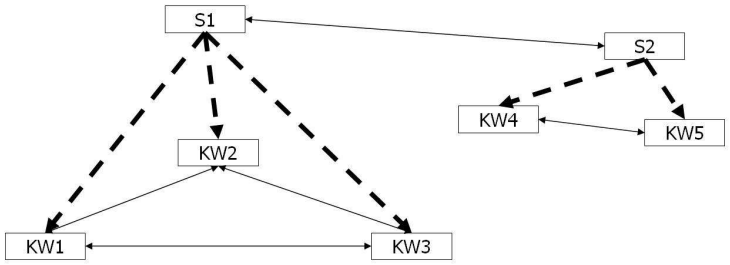


Fig. 7. Example of a document graph. KW1 to KW5 are keywords. S1 and S2 are two KWS.

5 Global Solving

5.1 Similar Case Retrieval

For graph comparison, many measures can be used [10]. However, as we are not only looking for accuracy, but also for a fast processing, we used Lopresti's method called graph probing [12]. It is a fast and accurate technique to compare graphs by measuring their degree of dissimilarity. In his paper, Lopresti applied his method successfully on document graphs containing simple structures of lines and words. In order to compare labelled and directed graphs, two probes PB1 and PB2 are measured:

1. PB1 = the frequency of each vertex in the graph;
2. PB2 = the frequency of each vertex' edge structure.

PB1 and PB2 are measured for the studied graph P_DC (G_1) and the Document Database (D_DB) graph (G_2). The probing distance P between the graphs G_1 and G_2 is then (2):

$$P(G_1, G_2) = (PB1(G_1) - PB1(G_2)) + (PB2(G_1) - PB2(G_2)) \ . \qquad (2)$$

It has to be noticed that if $P(G_1, G_2)$=0, G_1 and G_2 are not necessarily isomorphic. However, graph probing gives an approximation of the edit distance. We notice here that other types of distances are being studied.

Graph probing is then used to compare graphs and to retrive the nearest case in D_DB.

5.2 Solution Adaptation

When a similar case for P_DC (G_1) is found in D_DB (G_2), the adaptation consists first in finding for each structure in G_1 the corresponding structure in G_2. This is achieved by measuring the distance between the structures (PS or KWS) of G_1 and G_2. As the documents correspond to the same case (meaning they belong to the same set of documents), the system just copies the information about the nature (alphabetical, numerical) and the position (left on the same line, right on the same line, top on the line above) of each solution and looks for similar information in the current document. For example, if the solution corresponding to a KW "total" in G_2 case has the properties "real number + right", the system will look for a real number on the right of "total" on the same line (HL) in the processed document. If an answer exists, then it is proposed as a solution for this KW.

6 Local Solving

If no similar case exists in D_DB, the system builds a solution based on the structures already processed in others documents and stored in a special database S_DB (Structure cases database).

6.1 KWS Solving

The solving procedure acts as the following:

1. For each structure in the document, the nearest structure in the Structure Database (S_DB) is retrieved. P_KWS graph is compared to the KWS cases of S_DB. The solution of the nearest structure is adapted. Graph edit distance is used to find the nearest graphs in S_DB. Edit distance is used for graph comparison as we are really looking for graph isomorphism, or at least sub-graph isomorphism. As S_DB graphs are also small (no more than 5 vertices per graph in general), it is then better to use a more precise comparison

technique than to use a faster but less accurate one like graph probing. The cost function used to compute edit distance between graphs has uniform costs for both vertices and edges edit operations as both KW and their relative positions seem to have the same importance in the graph.

2. The nearest structures' solutions are now adapted to the document structures. As the cases in the S_DB have already a correct solution, the adaptation consists in taking the solution of each KW (case of KWS) and trying to find a corresponding solution in the processed document. If a complete solution is found for a structure, then, it can be stored in the S_DB. Otherwise, the following processing has to be done. For KW, some universal knowledge exists and it would be really a waste of time not to take advantage of it. For example, it is usual that the KW "total" is followed by a numerical. This numerical can be a real number or an integer depending on the document but its numerical nature is always valid. Following this logic, a rule basis detailing the general rules associated with keywords was built, in order to complete any partial solution of a KWS. This basis allows completing some missing KWS solutions. It has to be noticed that a rule basis is not able to solve complete cases as it does not take into account the context of the structure, and as its knowledge is very general and not related to any concrete case. If no solution can be found for a given structure, the system can ask the user to propose one.

The example on figure 8 shows a KWS which nearest KWS in S_DB solves four out of five KW. By using a rule basis, a complete solution can be found.

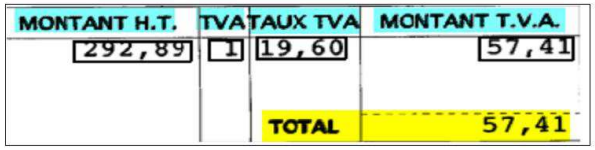


Fig. 8. A KWS. Only the KW Total is solved by the rule basis.

6.2 PS Solving

Each extracted PS (E_PS) is compared with the S_DB cases to retrieve the nearest structure. As PS are represented with strings, their patterns are compared using string edit distance. When a similar PS (C_PS) is found in S_DB, (same pattern, or with a maximum of one transformation), the table columns of E_PS are given the tags of C_PS, unless the rule between C_PS fields can not be applied on E_PS. In this case, the system tries to find the rule between E_PS fields by trying the rules in other close PS cases (close PS cases with more than one transformation) until a valid rule is found.

If no solution can be found, the user can also here propose one.

A perspective of our work is to use the table headers in order to interpret the table columns. In this way, PS cases could also be considered as KWS cases as we will use the KW found in the headers for the interpretation.

7 Experiments

7.1 The Database

The dataset is composed of 923 documents taken from different clients representing 325 different sets. The database set is divided in 2 groups :

- the first one contains 100 documents where each one has a similar case in the document database: this will help us testing the global solving;
- the second one contains 823 documents for which no associated case exists in the document database. Hence, local solving will be applied on these documents.

S_DB contained initially 300 structures. Only 20 of the tested structures have a complete similar case in S_DB. The remaining cases in S_DB are taken from several other documents which are not related to the tested documents. We have chosen to test our system in this way to show its ability to find a solution for a given problem even if it has never been studied before.

7.2 Measures

The results are described thanks to three different measures (3, 4, 5):

$$R = \frac{|right\ solutions\ in\ all\ documents|}{|desired\ solutions\ in\ all\ documents|} \quad . \tag{3}$$

$$R_KWS = \frac{|right\ KW\ solutions\ in\ all\ documents|}{|desired\ KW\ solutions\ in\ all\ documents|} \quad . \tag{4}$$

$$R_PS = \frac{|right\ PS\ fields\ in\ all\ documents|}{|desired\ PS\ fields\ in\ all\ documents|} \quad . \tag{5}$$

A right solution corresponds to a KW's solution or to a field in a PS that has been correctly extracted and interpreted.

7.3 Results

The results are given in table 1, they are very satisfying from an industrial point of view.

In global solving, the missing 14.78% correspond to 3.72% system errors and 11.06% OCR errors.

In local solving, for KWS, errors are due to:

Table 1. Results of CBRDIA for global solving and local solving

	R	R_KWS	R_PS
Global Solving	85.22%	79.77%	90.10%
Local Solving	74.90%	75.25%	74.80%

- 12.15% system errors (bad solution, no solution found, confusion with other solutions);
- 12.59% of OCR errors.

In local solving, for PS, errors are due to:

- bad detection of table lines (11.32%) (missing lines, no detection of table);
- OCR and segmentation errors (12.67%) (e.g : 12.T instead of 12.7. Two fields are fused together which implies a wrong tag);
- bad solution (1.21%) (fields are given bad solutions).

The OCR used in our system is a professional one used by ITESOFT. OCR errors are not just due to the software performance, but they also depend on:

- the quality of documents. In our dataset, we had about 11% of documents of very poor quality (this can be caused by the original quality of the document, or by a bad scanning);
- noisy information such as missing characters.

The difference between the results of global solving and those of local solving can be explained as the following:

- In global solving, the system is processing a similar document: it has the knowledge of what it is looking for in the document. The only sources of error can be:
  - a bad tagging (words, fields);
  - a bad PS extraction;
  - a missing KWS or a missing KW;
  - OCR errors.
- However, in local solving, in addition to all the previous sources of errors, we can also notice the bad extracted solutions. This can happen when the processed structures have no very close structures in S_DB. This can deteriorate the quality of the proposed solutions.

A special case of KWS was also tested (addresses). We tested our system for KWS solving on 30 documents containing addresses. We obtained 78.33% (118/150) of good results (150 being the number of processed KW in these address blocks). We can notice that this special case exists not only in invoice documents, but also in any other administrative documents.

8 Conclusion and Future Works

A CBR approach for invoice document analysis and interpretation was proposed in this paper. CBRDIA produces good results even if the documents have never been processed by the system before. This work is still under study in several ways. We are studying the improvement of problem elaboration especially in PS extraction. We are also focusing on S_DB and D_DB indexing in order to reduce the solving time. Finally, solutions' quality is the next step in our work. Enriching S_DB and D_DB requires having high quality solutions; otherwise, a lot of noisy cases can reduce the solving process efficiency. These studies should allow CBRDIA to have better results.

References

1. Yuan, J., L, Xu, Suen, C.: Form items extraction by model matching. In: ICDAR, France (1991)
2. Sako, H., M.Seki, Furukawa, N., H.Ikeda, A.Imaizumi: Form reading based on form-type identification and form-data recognition. In: ICDAR, Scotland (2003)
3. Cesarini, F., Francesconi, E., Gori, M., Soda, G.: Analysis and understanding of multi-class invoices. IJDAR **6**(2) (2003) 102–114
4. Belaïd, Y., Belaïd, A.: Morphological tagging approach in document analysis of invoices. In: ICPR. (2004) 469–472
5. Weber, R., Ashley, K., Bruninghaus, S.: Textual case-based reasoning. The Knowledge Engineering Review **20**(3) (2006) 255–260
6. Perner, P., Hlot, A., Richter, M.: Image processing in case-based reasoning. The Knowledge Engineering Review **20**(3) (2006) 311–314
7. Cunningham, C., Weber, R., Proctor, J.M., Fowler, C., Murphy, M.: Investigating graphs in textual case-based reasoning. In: ECCBR. (2004) 573–586
8. Cheetham, W., Shultz, J.: Using ensembles of binary case-based reasoners. In: ICCBR. (2005) 152–162
9. Aamodt, A., Plaza, E.: Case-based reasoning: Foundational issues, methodological variations, and system approaches. In: IOS press. (1994)
10. Schenker, A., Last, M., Bunke, H., Kandel, A.: Comparison of distance measures for graph-based clustering of documents. In: GbRPR. (2003) 202–213
11. H.Bunke, Foggia, P., Guidobaldi, C., Vento, M.: Graph clustering using the weighted minimum common supergraph. In: GbRPR. (2003) 235–246
12. Lopresti, D.P., Wilfong, G.T.: A fast technique for comparing graph representations with applications to performance evaluation. IJDAR **6**(4) (2003) 219–229