Unilingual alignment: a building block for digital resources constitution

Julien Bourdaillet and Jean-Gabriel Ganascia
Université Pierre et Marie Curie – Laboratoire d'Informatique de Paris 6 – 4 place Jussieu, 75005 Paris, France – julien.bourdaillet@lip6.fr, jean-gabriel.ganascia@lip6.fr

We present an application named MEDITE issued from a collaboration between literary and AI scholars. Its goal is to facilitate the text alignment work by automating it and producing parameterized alignments according to user preferences. Textual genetic criticism is a discipline that studies drafts led by authors during the writing process. MEDITE's first aim is to align two linearized transcriptions of such drafts (two texts) in order to expose invariants and differences between them. We discovered that for texts with a lot of repetitions, existing aligners (version comparison tools) failed to perform correct alignments. We present algorithms based on homology detection that address this problem and their evaluation compared to other related software. Our tool is now intended to be used in different projects related to the constitution of digital resources.

Keywords: textual comparison, sequence alignment, edit distance with moves.

1 INTRODUCTION

MEDITE has been designed as an application to assist philologists in their practice of textual genetic criticism [2]. Genetic criticism is part of the humanities and was developed thirty years ago as an important original French school of literary study [1]. This discipline introduced a temporal dimension in literary criticism by studying not only the final version of a literary work but also writers' drafts in order to highlight the genesis of the text. It seeks to understand how a text is produced but remains close to the aesthetics of the work. Philologists suggest interpretative hypotheses when they read the final version of a text, which they corroborate (or invalidate) through the study of previous versions. This study is based on text version comparison and considers every modification between two versions. These modifications need to be character based because a writer can proceed by one- or two-character long modifications, which can seriously alter the sentence meaning, especially for a morphologically rich language such as French.

Techniques arising from genetic criticism have been applied to epistemology as in the following example. Claude Bernard was a nineteenth century physiologist who contributed to the birth of modern medicine. In order to study the evolution of his medical theories, philologists want to compare his experiment notebooks and their synthesis written some years later. The notebooks relate observations in a telegraphic style while observations are written in an academic style in the synthesis (in which new ideas are also inserted). An example of comparison of these two texts using MEDITE is given in Figure 1 and using Microsoft Word in Figure 2. It can be seen that MEDITE identifies considerably more invariants (in black and white) between the two texts than Word, resulting in a better alignment. Furthermore, the visualization interface impacts on the readability of the alignment.

Comparison and visualization problems are common in existing file comparison tools. These tools are generally descendants of *diff* in which two files are compared line by line and a list of inserted and deleted lines is produced. This kind of program comes from the community that created Unix where their main interest was source code comparison. For this task, line by line comparison is sufficient because program structure is very constrained and the syntax is strong. This results in well-organized texts (i.e. source code) and the assertion ``one line, one instruction'' is generally verified. Most of the modifications occurring between two versions are line modifications. The limits of these comparers appear with texts such as those of Claude Bernard because intra-line modifications are not well identified. For example, the modification of one character in a line will lead to a ``deletion of one line, insertion of one line'' analysis. This is acceptable for source code but is a bad result for natural language. The precision of detections is of crucial importance for genetic criticism and this is not addressed correctly by existing aligners.

Furthermore, Claude Bernard's texts contain a lot of repeated text blocks. In the left text of Figure 1, the word *mouvements* is repeated three times and it is repeated more times in the whole text. With simple alignment algorithms, several repetitions may not be found, resulting in missing invariant or moved blocks in the final alignment. This is due to the fact that invariants (and moves) between the two texts are blocks repeated at least twice and if some repetitions are missed then invariants will be missed. Similar problems existed in previous versions of MEDITE: when processing Claude Bernard's texts, our results were similar to those of Word. We present here an algorithm that addresses these problems.

This task can be defined as unilingual textual alignment that compares two related texts, written in the same language, and identifies invariants and differences between them. More precisely, the term alignment refers to the identification of these invariants and their pairing. Once identified, differences can be deduced, but there is no one way of doing this. We also address the move detection task, but since moves can be seen as a deletion plus an insertion, this task involves ambiguity.

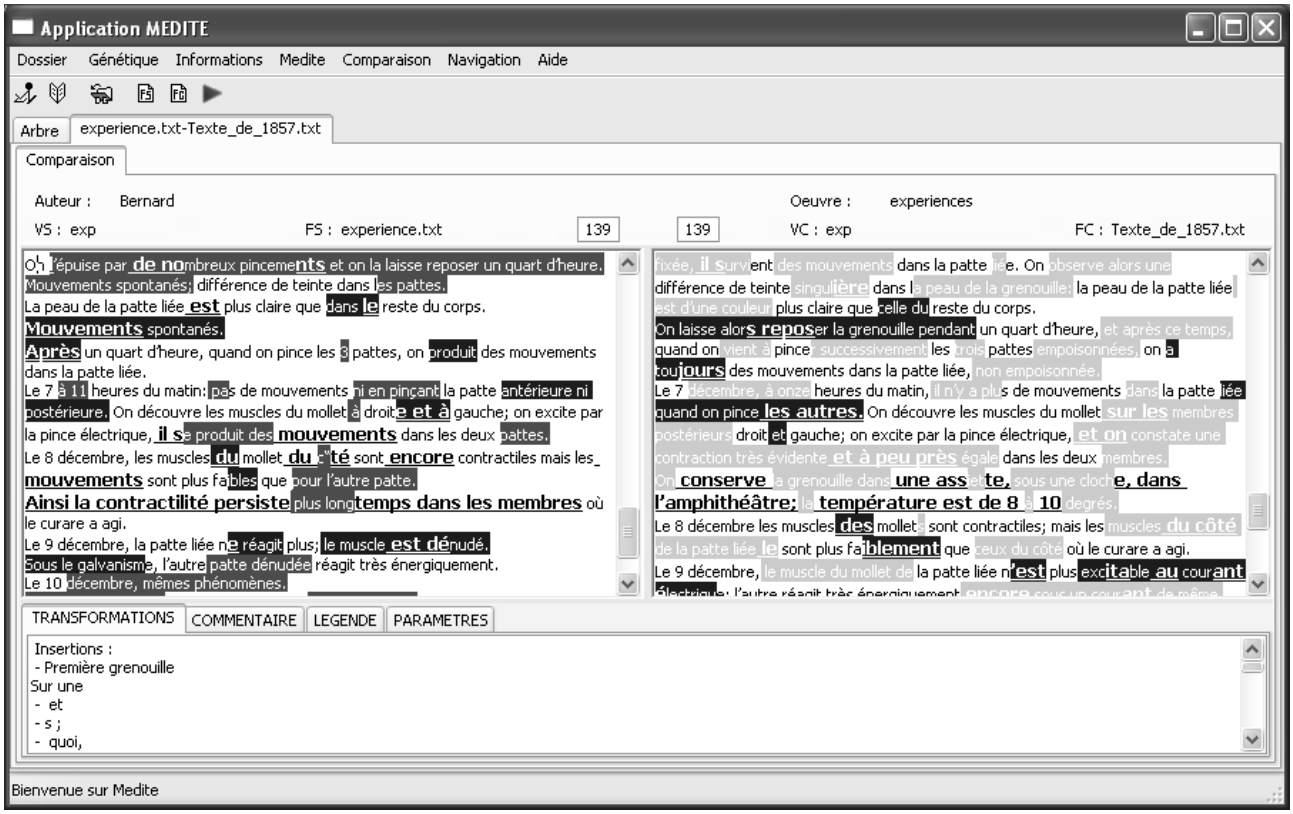

Fig. 1. The alignment of Claude Bernard's texts using MEDITE

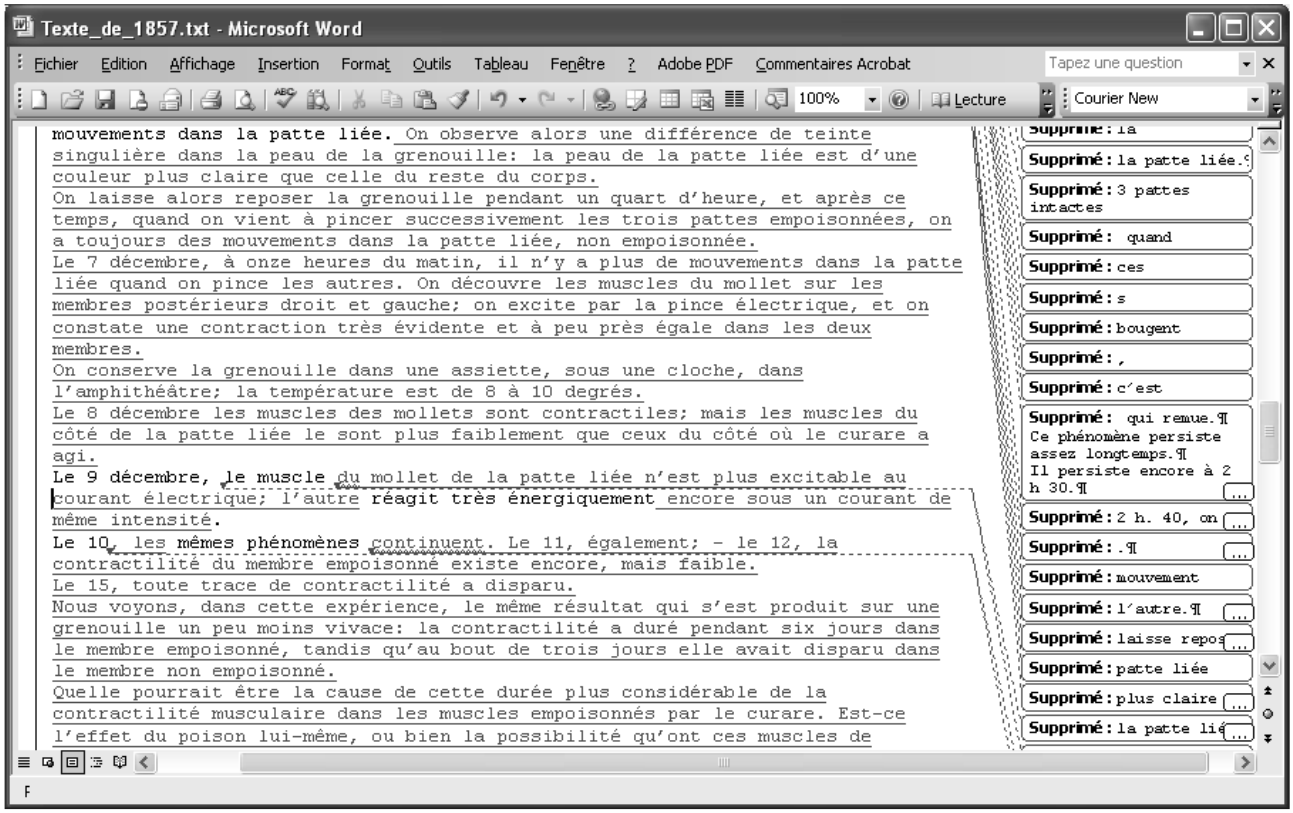

Fig. 2. The alignment of Claude Bernard's texts using Microsoft Word

2 ALGORITHM

A sequence alignment algorithm, based on the edit distance with moves conceptual frame, has been developed [5]. It detects deleted, inserted, replaced, moved and invariant character blocks and aligns pairwise these last three block types. The first algorithm step detects maximal exact matches (MEM): homologies between the two texts which can't be extended to the left or to the right without losing identity. MEMs are either invariant or moved blocks and are identified by browsing the space of possible alignments by an A* procedure which minimize the size of moved blocks. This whole process is then applied recursively between each pair of aligned invariant blocks in order to detect smaller blocks. Finally, as deleted, inserted and replaced blocks are non repeated blocks, they are deduced from the alignment.

2.1 Invariants and moves extraction

The first phase proceeds in three steps: searching for anchors in sequences, aligning them in order to determine invariants and moves, and finally processing recursively between invariants.

Anchors are homologies under a certain similarity criterion. Our criterion is exact homology, where the two substrings must match exactly. Firstly, a generalized suffix tree [4] is built over the two sequences in order to extract all the homologies. A minimum size parameter is chosen by the user (by default, five characters long). This method generates overlaps between homologies. However, it is necessary to resolve them in order to obtain a proper partition of the complete sequence of homologies in disjoint blocks. We use a heuristic based on a property of natural language: if the overlap contains separators, it is better to cut it on one of them, since an inter-word cut is preferable to an intra-word cut. Most of the time this condition is verified, but if not the block is cut arbitrarily.

In the second step, blocks must be aligned to determine which are invariant and which are moved. The space of all possible alignments is browsed by an A* procedure using an alignment cost function which is a heuristic based on a measure *M* maximizing invariant block size and minimizing moved blocks size. During the search, an alignment cost is decomposed into the cost of already aligned blocks plus an estimation of the cost of the remaining blocks to be aligned such that $\text{cost} = \text{cost}_{ab} + \text{cost}_{rb}$ where cost_{ab} is the sum of the size of moved blocks in already aligned blocks and cost_{rb} is the sum of the size of the blocks in the symmetric difference between remaining blocks to be aligned in the two sequences. Because it will not be possible to align these remaining blocks in the rest of the alignment process, they will be considered as moved in the final alignment and counted as a penalty due to *M*.

Finally, these two steps are repeated recursively. The difference comes from input sequences. We loop over the alignment resulting from step two and consider the subsequences between each pair of aligned invariant blocks. Then these subsequences are used as input of the first step.

The output of the recursive steps one and two is an alignment for the two subsequences. Invariants and moves identified with this alignment are included in the main invariant and moved blocks. This recursive step enables to find alignments which would otherwise have not been found.

2.2 Alignment building

The first phase produces two sets of invariant and moved blocks. Text between two invariant blocks in the first sequence is a deleted block; text between two invariant blocks in the second sequence is an inserted block. As moved blocks can be seen as a deletion plus an insertion, all moved blocks (identified during the first phase) overlap a deleted or an inserted block. Hence they can either be considered as moved blocks or as part of deleted or inserted blocks.

Two heuristics are used to determine substituted blocks and moved block types. If two blocks (deleted and inserted) occur between the same invariant blocks, we examine the ratio between their sizes. If it reaches a certain threshold, they are considered as two substituted blocks, and are replaced by two paired substituted blocks. By default the ratio is set to 0.5 and the user is free to modify it. For instance, if a deleted block *'He saw me' occurs* between two invariant blocks of the first text and an inserted block *'I saw him' occurs* between the same blocks of the second text and their size ratio exceeds the threshold, then they are both identified as substituted blocks, meaning that 'He saw me' has been replaced by 'I saw him' in the text.

In a similar way, for each deleted or inserted block, we examine the ratio between their size and moved blocks overlapping them. If it is above another threshold, blocks are split in different blocks of relevant types. For instance with an inserted block 'This book is fundamental' and an overlapping moved

block ‘is fundamental’, if the threshold is reached, this will result of two non-overlapping blocks: ‘This book’ of type inserted and ‘is fundamental’ of type moved. The ratio is also set to 0.5 by default and the user can modify it. Finally this process results in an alignment between the two sequences.

3 RESULTS AND DISCUSSION

3.1 Benchmark

MEDITE has been compared with six aligners, the most famous being the one present in Microsoft Word. For each application, three file comparisons were made, where three points were tested (identified with capitalized letters below).

The first comparison is between two versions of a short story by Pascal Quignard entitled ``Bernon l’Enfant". Small modifications of some characters were introduced throughout the text. Lexical words were changed, misspellings corrected and words moved. The goal is to find such modifications. Paragraphs must be aligned (A); word modifications must be found (B) and character modifications must be found (C).

The second comparison is between a news agency dispatch and an article which is rather different but derived from it. Two paragraphs were kept with some internal modifications, and the remaining text was replaced completely by another one. The two paragraphs must be aligned (D); modifications inside these paragraphs must be found (E) and similar lexical words must be found (F).

The third comparison is the one described in the introduction. Texts from Claude Bernard's experiment notebooks and their synthesis must be aligned. This task is very hard because the existing content remained the same but the form changed and new content was inserted. Paragraphs must be aligned (G); word groups must be aligned (H) and isolated words must be aligned (I).

The results of this experiment are presented in Table 1. Paragraph alignment (A) is correct for all the applications. Only four applications detect word changes in test B and only MEDITE and Compare It detect character changes (C). The others detect character changes as word changes, whereas often only one or two characters have been modified. By contrast MEDITE focuses on the modified characters.

For the second comparison, only DiffDoc and MEDITE align the two paragraphs (D) and find small internal modifications (E). All the other applications fail to detect this. This test is useful because the longest invariant sequence is 752 characters long for two texts of 14 Ko and 18 Ko, and so represents about 5% the size of each file. As it doesn't change, we could except all software to find it but only two of them do. Because the theme of the two texts is related, common lexical words are used in the remainder of the texts but only MEDITE aligns them correctly (F).

The third comparison is the hardest one. Paragraphs are aligned correctly only by DiffDoc, MEDITE and Word (G). Several word groups are aligned by DiffDoc and Word but a lot are missed (H). We know they are missed because MEDITE detects them. As DiffDoc and Word miss numerous word groups, they miss isolated word changes whereas MEDITE aligns them pairwise correctly (I). The absence of these alignment anchors results in a bad alignment because a lot of information is not discovered and it impacts on the readability of the alignment. Our result can be viewed in Figure 1. The less the texts are aligned the less the visualization is good. In earlier versions of MEDITE we had similar problems but the introduction of recursion in our algorithm enabled us to address them.

Table1. Benchmark Results

Year	A	B	C	D	E	F	G	H	I	Total
MEDITE	1	1	1	1	1	1	1	1	1	9
DiffDoc	1	1	0	1	1	0	1	1	0	6
Word	1	1	0	0	0	0	1	1	0	4
Compare It	1	1	1	0	0	0	0	0	0	3
Visual Comparer	1	0	0	0	0	0	1	0	0	2
Araxis Merge	1	0	0	0	0	0	0	0	0	1
Beyond Compare	1	0	0	0	0	0	0	0	0	1

None of the applications except for MEDITE detects moved blocks, though we have already said that this is crucial for philology. For source code comparison, this is still the case. Detecting that a code line has

been moved from one function to another is an important piece of information. It is also important for any natural language text, because it makes possible to detect rearrangements of ideas, for instance.

3.2 Visualization

Figure 1 presents MEDITE's visualization interface. Although the figure is small it can be seen that the colors identify the different types of blocks well. Deleted blocks are red (or grey in the greyscale proceedings), inserted blocks are green (light grey) and substituted blocks are blue (dark grey) while invariant blocks remain black and white. These colors can be customized by the user. Moved blocks are underlined and have a bold font, enabling to represent those overlapping inserted, deleted or substituted blocks.

Applications tested in Section 3.1 have poor visualization in comparison to MEDITE. Not only do bad alignments result in bad visualization but in addition, graphical user interfaces (GUIs) are generally ill-suited. Another serious problem is that a lot of them present a merged text that mixes deletions and insertions: when texts are very different, the visualization is bad, as is the case with Word in Figure 2.

In MEDITE, when the user clicks on an invariant block its corresponding block is presented side-by-side on the other window. It is thus possible to browse the text in an intuitive way following the blocks the user is interested in. This differs from other applications, where scrolling bars are locked, so when big parts of text are deleted or inserted it is sometimes impossible to look at them side-by-side.

MEDITE also generates an HTML report which is a direct visualization of the bi-block list. Each block is displayed with its match and both are colored corresponding to their type. This kind of visualization can be useful especially for source code comparison.

4 CONCLUSION

This paper presents a textual alignment system and addresses the problem of sequence alignment when applied to natural language. We show that it can be very difficult and that results from existing aligners are not satisfactory when treating texts studied by textual genetic criticism where there are a lot of repeated blocks. Our experiments show that both existing algorithms and their visualization give poor results. Only two systems, DiffDoc and Word, compete with MEDITE but nevertheless their results are poorer. Further, we present a method to detect moved blocks in textual comparisons whereas none of the applications we tested was able to do this.

Because our algorithm is sequence-based, it is language independent and can process any language without language-specific resources. For instance, it processes Arabic texts. Further, it is character-based and so it detects intra-words modifications which is very useful for flexional languages. The moves detection is a kind of knowledge discovery as it exposes block correlations between the texts.

More generally, this problem is interesting because sequence alignment is an old problem. Texts resulting from genetic criticism have shown hard cases that were handled incorrectly by classic file comparison tools. In addition, and this was the original aim of this work which is now completed, natural language processing brings new facilities to researchers in textual genetic criticism via a tool such as MEDITE which is now used by philologists in textual genetic criticism and epistemologists in ideas' history. It enables them to study longer texts and to discover, more systematically, transformations between authors' draft. They can then establish diachronic corpus of an author's oeuvre. We now plan to embed MEDITE in a digital library platform named "Open Book" [3] and to start a similar project with the Bibliothèque Nationale de France.

REFERENCES

[1] Deppman J., Ferrer D. and Groden M., 2004. Genetic Criticism - Texts and Avant-textes. University of Pennsylvania Press

[2] Ganascia, J.-G., Bourdaillet, J. Alignements unilingues avec MEDITE, 2006. Actes des Huitièmes Journées Internationales d'Analyse Statistique des Données Textuelles.

[3] Ganascia J-G, The open book project, proceedings of the IPSI-workshop in Kronberg, 2004. ed. V. Milutinovic.

[4] Gusfield D. Algorithms on Strings, Trees and Sequences: Computer Science and Computer Biology, 1997. Cambridge University Press.

[5] Lopresti D.P. and Tomkins A., 1997. Block Edit Models for Approximate String Matching. Theor. Comput. Sci., 181(1), pp 159-179.