A Parallel Comparator of Documents

Sonia Alouane Ksouri[1], Minyar Sassi Hidri[2], Kamel Barkaoui[3]
[1,2]Université Tunis El Manar
Ecole Nationale d'Ingénieurs de Tunis
LR-SITI-ENIT
BP. 37, Le Belvédère 1002, Tunis, Tunisia
[1,3]CEDRIC-CNAM
Rue Saint-Martin Paris 75003, France
{[1,2]sonia.alouane,minyar.sassi}@enit.rnu.tn,[3]kamel.barkaoui@cnam.fr

***Abstract*—Documents, sentences and words clustering are well studied problems. Most existing algorithms cluster documents, sentences and words separately but not simultaneously. However, when analyzing large textual corpuses, the amount of data to be processed in a single machine is usually limited by the main memory available, and the increase of these data to be analyzed leads to increasing computational workload. In this paper we present a parallel fuzzy triadic similarity measure called PFT-Sim, to calculate fuzzy memberships in a context of document co-clustering based on a parallel programming architecture. It allows computing simultaneously fuzzy co-similarity matrices between documents/sentences and sentences/words. Each one is built on the basis of the others. The PFT-SIM model provides a parallel data analysis strategy and divides the similarity computing task into parallel sub-tasks to tackle efficiency and scalability problems.**

Keywords-Text Mining, Document co-clustering, Fuzzy sets, multi-thread architecture, Parallel computing, Three-partite graph.

I. Introduction

In recent years, text mining has gradually become a new research topic. The study of text clustering has thus attracted wide attention. It arranges document items into several groups so that similar items fall into the same group. Several methods have been proposed, first to analyze these textual corpuses by classifying them and, later, to facilitate their exploitation. Their aim is to organize documents into groups (or clusters), where each cluster represents a different topic. They are based on three concepts: data representation model, similarity measure and clustering model. In this paper, we focus on the preprocessing step which includes data representation and similarity computing.

Most of the proposed models were originally designed for relatively small data sets. However, in recent years, clustering algorithms has been extended to work efficiently in large data sets. When used to classify these data sets, clustering models are very computing demanding and require high-performance machines to get results in reasonable time. Thus, scalable parallel computers can provide the appropriate setting to execute clustering of algorithms in order to extract knowledge from large-scale data repositories.

To achieve a more accurate document clustering, a more informative feature word-sentence has been considered in recent research work [1]. While considering three levels (documents-sentences-words) based on k-partite graph [2], to represent the data set, we are able to deal with a dependency between all of them. The sentence-word-based document similarity (or triadic co-similarity) considers a weighting scheme, based on a fuzzification controller process, for computing the document co-similarity.

The contribution of this research is to combine fuzzy sets [3] in the document preprocessing step describing the design of a parallelization strategy, while taking into account three abstraction levels (Documents-Sentences-Words) to more precisely determine the pertinent memberships. The fuzzy similarity matrices are used to calculate Parallel fuzzy triadic similarity, called PFT-Sim, between documents, sentences and words based on parallel algorithms. Hence, these similarities are considered as a preprocessing step to clustering and it becomes possible to choose the clustering method such as K-means [4], Fuzzy-C-Means (FCM) [5] etc. that is best suited to co-cluster the data.

The rest of the paper is organized as follows: in section 2 we highlight related work on distributed text mining. In section 3 we describe the parallel fuzzy triadic computing for document similarity. Section 4 presents the task-based parallel computing. Section 5 describes the data-based parallel architecture. Section 6 concludes the paper and mentions our future work.

II. Distributed text mining

Analyzing huge textual corpuses is the source of two challenges. Firstly, the amount of data to be processed in a single machine is usually limited by the main memory available. Secondly, the increase of the amount of data to be analyzed leads to an increasing computational workload. Thus, it is imperative to find a solution which overcomes the memory limitation (by splitting the data into several pieces) and to markedly reduce the runtime by distributing the workload across available computing resources (CPU cores or cloud instances).

Recently there has been an increasing interest in parallel implementations of data clustering algorithms. In [6], a distributed programming model and a corresponding implementation called MapReduce has been proposed. It allows efficient

parallel processing of data in a functional programming. In [1]a preprocessing and indexing methods for phrases, paired with new search techniques for the top-k most interesting phrases in ad-hoc subsets of the corpus are developed.

In [7], an approach has been proposed which applies the distributed programming paradigm MapReduce to advance performance of suitable text mining tasks. It showed that distributed memory systems can be effectively employed within this model to preprocess large data sets by adding layers to existing text mining infrastructure.

However, in many applications, databases involving more than two types of interacting objects, or simply related, are also frequent. A simple way to represent such data sets is to use as many matrices as there are relations between the objects. Then, one could use classical co-clustering methods to separately cluster the objects occurring in the different matrices.In this way,however, interactions between objects are not taken into account, thus leading to a loss of information. Therefore, handling the views together, referenced as the multi-view clustering task, is an interesting challenge in the learning domain to resolve the limits of classical clustering.

Many extensions to the clustering methods have been proposed to deal with multi-view data. In [8], they describe an extension of k-means (MVKM) and of EM algorithms using a multi-view model. In [9] and [10] the authors build clusters from multiple similarity matrices computed along different views. In [11], a co-clustering system called MVSC has been proposed. It permits a multi-view spectral clustering while using the co-training that has been widely used in semi-supervised learning problems. The general idea is to learn the clustering in one view and use it to label the data in another view so as to modify the graph structure (similarity matrix).

Closer to our approach, some works aim at combining multiple similarity matrices to perform a given learning task. The MVSim architecture [12] which is an extension of the X-Sim algorithm [13], adapts the previous algorithm to the multi-view context. The basic idea is to create a learning network isomorphic to these data set structures.

III. PFT-Sim: Parallel Fuzzy Triadic Document Similarity

Document clustering should be based not only on analysis of single words, but of sentences as well. Sentence-based analysis means that the similarity between documents should be based on matching sentences rather than single words only. Parallel algorithms are able to work on partitioned data. In this context,we propose a parallel architecture where we combine fuzzy sets [3] in the document preprocessing step while taking into account the three abstraction levels Documents-Sentences-Words to more precisely determine the pertinent memberships [14].

Given $\widetilde{SD} = [\mu]_{ji}$ a fuzzy Sentences$\times$Documents similarity matrix of J (sentences) by I (documents) $(i = 1..I,\ j = 1..J)$, representing the membership degrees associated to the i^{th} document according to the j^{th} sentence and $\widetilde{WS} = [\mu]_{kj}$ a fuzzy Words$\times$ Sentences similarity matrix of K (words or terms) by J (sentences)$(k = 1..K,\ j = 1..J)$, representing the membership degrees associated to the k^{th} word to the j^{th} sentence. Let H be the number of local data sources, to each one corresponds one local matrix $\widetilde{SD}^{(h)} = [\mu]_{ji}$ and $\widetilde{WS}^{(h)} = [\mu]_{kj} (h = 1, ..., H)$. These values are determined by proceeding to a fuzzification controller process. It converts crisp values to fuzzy ones. The conversion to fuzzy values is represented by the membership functions [15]. The mathematical formulations of these functions are given in the following equations.

$$\widetilde{SD}_i = [\mu]_{ji} = \begin{cases} 1, & if\ \ SD_{ji} \geq L_i \\ \frac{SD_{ji} - U_i}{U_i - L_i}, & if\ \ L_i < SD_{ji} < U_i \\ 0, & if\ \ SD_{ji} \leq L_i \end{cases} \quad (1)$$

and

$$\widetilde{WS}_j = [\mu]_{kj} = \begin{cases} 1, & if\ \ WS_{kj} \geq L_i \\ \frac{WS_{kj} - U_i}{U_i - L_i}, & if\ \ L_i < WS_{kj} < U_i \\ 0, & if\ \ WS_{kj} \leq L_i \end{cases} \quad (2)$$

For each iteration t $(t = 0..it)$, we process three fuzzy co-similarity matrices of Document$\times$Document called $\tilde{D}_2^{\{t\}}$, Sentence$\times$Sentence called $\tilde{S}_2^{\{t\}}$ and Word$\times$Word called $\tilde{W}_2^{\{t\}}$ containing $[\mu]_{lm}^{\{t\}}$ having respectively $(l = 1..I,\ m = 1..I)$,$(l = 1..J,\ m = 1..J)$ and $(l = 1..K,\ m = 1..K)$. While taking into account these notations we can, for example, express fuzzy co-similarity matrix of documents in the 2^{nd} site during the 3^{rd} iteration as $(\tilde{D}^{(2)})^{\{3\}}$.

IV. Task-based parallelism

There are two main problems of processing huge corpuses: the long processing time and the huge amount of memory required. Fortunately, corpus processing has in general a parallel nature and the majority of matrix operations can be easily decomposed into operations performed on matrix parts. In this section we will present parallel algorithms to compute the fuzzy co-similarity matrices. A natural way of task distribution among multiple threads during matrix construction is the definition of the matrix columns as sub-matrices treated in parallel on each local site.

Before proceeding to parallel fuzzy triadic computing, we must initialize Document $\times$ Document, Sentence $\times$ Sentence and Word $\times$ Word matrices with the identity ones noted $\tilde{D}_2^{\{0\}}$, $\tilde{S}_2^{\{0\}}$ and $\tilde{W}_2^{\{0\}}$. We consider only the similarity between a document (resp. sentence and word) and itself as maximal. All others values are initialized with zero. Figure 1 shows the general chart of the task-based parallelism. It shows two levels of parallelism, the first one concerns the processing of each of $\tilde{D}_2$, $\tilde{S}_2$ and $\tilde{W}_2$ simultaneously. The second level of parallelism concerns the task distribution of matrix construction considering all sub-matrices (columns) in parallel.

A. $\tilde{D}_2^{\{t\}}$ Parallel Computing

Usually, the similarity measure between two documents D_l and D_m is defined as a function that is the sum of the similarities between shared sentences. Our idea is to generalize this function in order to take into account the intersection between

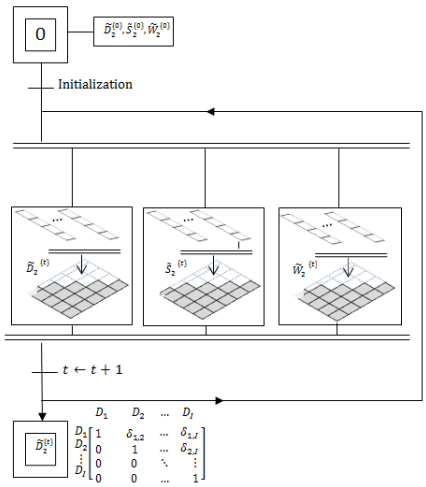

Fig. 1. Task-based parallelism.

all the possible pairs of sentences occurring in documents D_l and D_m. In this way, not only can we capture the fuzzy similarity of their common sentences but also the fuzzy ones coming from sentences that are not directly common in the documents but are shared with some other documents. For each pair of sentences not directly shared by the documents, we need to take into account the fuzzy similarity between them as provided by $\tilde{S}_2^{(t-1)}$. Figure 2 shows the $\tilde{D}_2^{(t)}$ computing process.

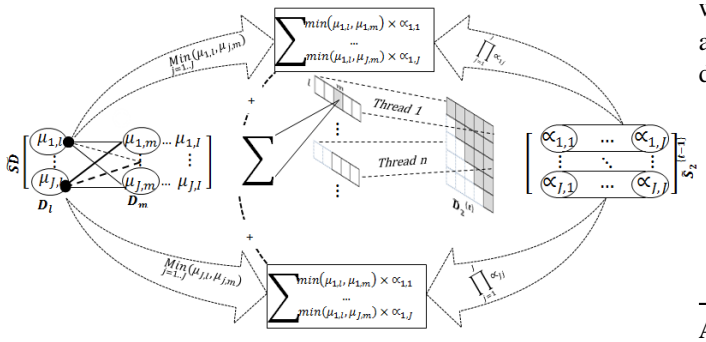

Fig. 2. $\tilde{D}_2^{\{t\}}$ computing process.

Thus, $\delta_{l,m}^{\{t\}}$, except the case $l = m$, can be formulated as follows:

$$\delta_{l,m}^{\{t\}} = \sum_{i=1}^{J}\sum_{j=1}^{J} \min(\mu_{il}, \mu_{jm}) * \alpha_{ij}^{\tilde{S}_2^{\{t-1\}}} \quad (3)$$

The $\tilde{D}_2^{\{t\}}$ parallel computing is presented in algorithm 1.

B. $\tilde{S}_2^{\{t\}}$ Parallel Computing

Similarly, for each pair of words not directly shared by the sentences, we need to take into account the fuzzy similarity between them as provided by $\tilde{W}_2^{\{t-1\}})$. The overall fuzzy similarity between documents S_l and S_m is defined in the following equation:

$$\alpha_{l,m}^{\{t\}} = Min[\sum_{i=1}^{I}\sum_{j=1}^{I} \min(\mu_{il}, \mu_{jm})\delta_{ij}^{\tilde{D}_2^{\{t-1\}}}$$

Algorithm 1 $\tilde{D}_2^{\{t\}}$ Parallel Computing

Require: $\widetilde{SD}$,$\tilde{S}_2^{\{t-1\}}$
Ensure: $\tilde{D}_2^{\{t\}}$
1: **for all** $l = 1..I$ **do in parallel**
2: **for** $m : l..I$ **do**
3: $\tilde{D}_2^{\{t\}}[l,m] \longleftarrow \delta_{l,m}^{\{t\}}$ equation (3)
4: **end for**
5: **end for**

$$\sum_{i=1}^{K}\sum_{j=1}^{K} \min(\mu_{il}, \mu_{jm})\omega_{ij}^{\tilde{W}_2^{\{t-1\}}}] \quad (4)$$

The $\tilde{S}_2^{\{t\}}$ Parallel Computing is presented in algorithm 2.

Algorithm 2 $\tilde{S}_2^{\{t\}}$ Parallel Computing

Require: $\widetilde{SD}$,$\tilde{D}_2^{(t-1)}$ $\widetilde{WS}$,$\tilde{W}_2^{\{t-1\}}$
Ensure: $\tilde{S}_2^{\{t\}}$
1: **for all** $l = 1..J$ **do in parallel**
2: **for** $m : l..J$ **do**
3: $\tilde{S}_2^{\{t\}}[l,m] \longleftarrow \alpha_{l,m}^{\{t\}}$ equation (4)
4: **end for**
5: **end for**

C. $\tilde{W}_2^{\{t\}}$ Parallel Computing

For each pair of words not directly shared by the sentences, we need to take into account the fuzzy similarity between them as provided by $\tilde{W}_2^{\{t-1\}}$. The overall fuzzy similarity between documents W_l and W_m is defined in the following equation:

$$\omega_{l,m}^{\{t\}} = \sum_{i=1}^{J}\sum_{j=1}^{J} \min(\mu_{il}, \mu_{jm}) * \alpha_{ij}^{\tilde{S}_2^{\{t-1\}}} \quad (5)$$

The $\tilde{W}_2^{\{t\}}$ parallel computing is presented in algorithm 3.

Algorithm 3 $\tilde{W}_2^{\{t\}}$ Parallel Computing

Require: $\tilde{S}_2^{\{t-1\}}$ $\widetilde{WS}$,
Ensure: $\tilde{W}_2^{\{t\}}$
1: **for all** $l = 1..K$ **do in parallel**
2: **for** $m : l..K$ **do**
3: $\tilde{W}_2^{\{t\}}[l,m] \longleftarrow \omega_{l,m}^{\{t\}}$ equation (5)
4: **end for**
5: **end for**

D. Task-Parallel Fuzzy Triadic algorithm

The Fuzzy triadic algorithm proposed is based on an iterative approach, in which each iteration t consists in evaluating the similarities according to documents/sentences/words three-partite graph. As it is shown in figure1 the steps computing,$\tilde{D}_2^{\{t\}}$, $\tilde{S}_2^{\{t\}}$ and $\tilde{W}_2^{\{t\}}$ are done in parallel. The global task-parallel fuzzy triadic similarity computing is described in algorithm 4.

Algorithm 4 Task-Parallel Fuzzy Triadic Algorithm

Require: $\widetilde{SD}$, $\widetilde{WS}$, It
Ensure: $\tilde{D}_2^{\{t\}}$, $\tilde{S}_2^{\{t\}}$, $\tilde{W}_2^{\{t\}}$
1: $\tilde{D}_2^{\{0\}} \leftarrow Identity$, $\tilde{S}_2^{\{0\}} \leftarrow Identity$, $\tilde{W}_2^{\{0\}} \leftarrow Identity$
2: **for** $t = 1..It$ **do**
3: $\tilde{D}_2^{\{t\}}$ parallel computing with $\tilde{S}_2^{\{t-1\}}$ (algorithm1)
4: $\tilde{S}_2^{\{t\}}$ parallel computing with $\tilde{D}_2^{\{t-1\}}$ and $\tilde{W}_2^{\{t-1\}}$ (algorithm2)
5: $\tilde{W}_2^{\{t\}}$ parallel computing with $\tilde{S}_2^{\{t-1\}}$ (algorithm3)
6: **end for**

V. Data-based parallelism

In order to reduce the complexity of the problem of treating huge databases, it is possible to split a given data matrix into a collection of smaller ones, each sub-matrix becoming a component of our network and processed as a separate view. The splitting strategy can be random or use a rapid clustering algorithm. The first alternative presents a solution which is based on no strategy. There is no guarantee to obtain an interesting matrix for an optimal processing by varying the number of splits randomly. Thus, we cannot deduct rules to be applied to this kind of problem.The second alternative to split a given data set, is to adopt the FCM algorithm [5].

A. FCM-based split

The FT-Sim is the exploitation of the trial nature of the problem of similarity. That means the relationship between groups of sentences that occur in a group of documents and the relationship between groups of words that occur in a group of sentences. Thus, documents are considered similar and hence grouped together, if they contain similar sentences, and sentences in turn are considered similar and therefore grouped together, if they occur in similar documents, etc. The idea behind this method is proceeding a rapid clustering of sentences before construction of the sub-matrices for each core. We can already obtain groups of similar sentences. This will facilitate the following task which is the parallel co-similarity learning.

The aim of our proposition is the unsupervised data clustering while adopting a fuzzy partitioned-based strategy. Fuzzy clustering methods allow objects to belong to several clusters simultaneously, with different degrees of membership. The data set $X = \{X_1, .., X_N\} \subset R^M$ is thus partitioned into c fuzzy subsets. The result is the partition matrix $U = [\mu]_{ji}$ for $i = 1, .., N$ and $j = 1, ..., c$.

Thus, the FCM can be used to obtain a set of clusters with similar sentences. In this way we can exploit results to construct sub-matrices Documents $\times$ Sentences and Sentences $\times$ Words with similar sentences, with the aim to have more coherent matrices. Dividing a huge database into smaller ones can considerably reduce the time and the complexity of the computing. We must also consider the complexity of the FCM run, which does not constitute a problem, because FCM is considered as rapid compared with the other clustering methods. Our solution allows a gain in time of execution for the following task which is the co-similarity learning.

B. Splitting-based parallel architecture

Our proposition is to divide huge data set ($\widetilde{SD}$ or $\widetilde{WS}$) into several sub-matrices processed independently. These Sub-matrices are processed in parallel, on different local sites, and next the partial results are merged using controlled functions. Figure 3 shows the splitting-based parallel architecture.

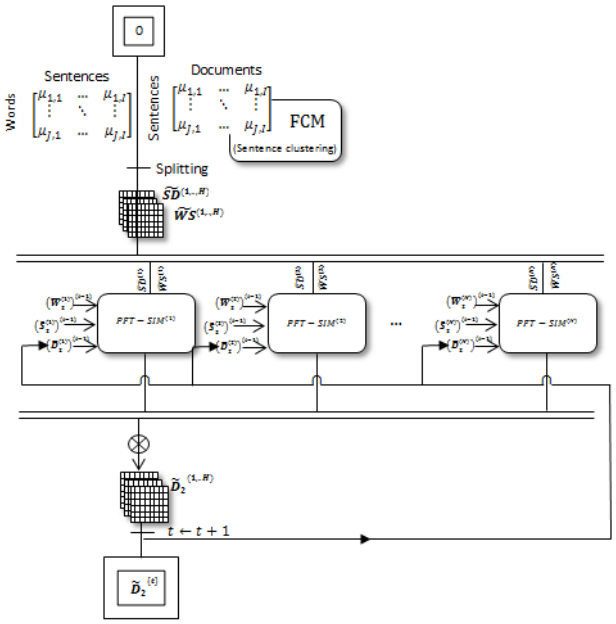

Fig. 3. Splitting-based parallel architecture.

In this topology, all local $FT-Sim^{(i)}$ instances $(i = 1..H)$ are run in parallel, then the similarity matrices $\widetilde{D}_2^{(i)}$ are simultaneously updated with an aggregation function. This policy offers the benefit that all the instances of $FT-Sim^{(i)}$ have the same influence.

The aggregation function takes H matrices $(\widetilde{D}_2^{(1)})^{\{t\}}$, $(\widetilde{D}_2^{(2)})^{\{t\}}$,..,$(\widetilde{D}_2^{(H)})^{\{t\}}$ issue from each data source i for a given iteration t. If a given document does not appear in a single local data source, then we assign its corresponding similarity measures directly in $\tilde{D}_2$. If a particular document appears in several different local data sources, we assign the minimum of all similarity measures relevant to this document to $\tilde{D}_2$ without taking into account the value of 0. The different steps of aggregation computing are presented in algorithm 5.

So, for a given iteration t, each instance $FT-Sim^{(i)}$ produces its own similarity matrix $(\widetilde{D}_2^{(i)})^{\{t\}}$. We thus get a set of output similarity matrices $\{(\widetilde{D}_2^{(1)})^{\{t\}}, (\widetilde{D}_2^{(2)})^{\{t\}}$,..,$(\widetilde{D}_2^{(H)})^{\{t\}}\}$ the cardinal of which being equal to the number of local data sets related to H. Therefore, we use the aggregation function denoted by $\bigotimes$ and developed in the aggregation function to compute a consensus similarity matrix merging all of the $\{(\widetilde{D}_2^{(1)})^{\{t\}}, (\widetilde{D}_2^{(2)})^{\{t\}}$,..,$(\widetilde{D}_2^{(H)})^{\{t\}}\}$ with the current matrix $\widetilde{D}_2^{\{t\}}$.

In turn, this resulting consensus matrix is connected to the inputs of all the $FT-Sim^{(i)}$ instances, to be taken into

Algorithm 5 Aggregation Function

Require: Collection of H matrices $\{(\widetilde{D}_2^{(1)})^{\{t\}}$,..., $(\widetilde{D}_2^{(H)})^{\{t\}}\}$
Ensure: $\widetilde{D}_2$
1: $I \leftarrow$ Compute the number of documents in $\{(\widetilde{D}_2^{(1)})^{\{t\}}$, $(\widetilde{D}_2^{(2)})^{\{t\}}$,..., $(\widetilde{D}_2^{(H)})^{\{t\}}\}$
2: Let $\widetilde{D}_2 = [\mu_{l,m}]$ ($l = 1..I$ and $m = 1..I$)
3: $\widetilde{D}_2 \leftarrow Identity$
4: **for** Each document D_l of $\widetilde{D}_2$ **do**
5: **if** D_l Appear in only one data set s **then**
6: $\mu_{l,*} \leftarrow \mu_{l,*}^{(s)}$
7: **else**
8: $\mu_{l,*} \leftarrow \min($ All $\mu_{l,*}^{(i)})$ $i \in \{$ sites where D_l appear$\}$ with $\mu_{l,*}^{(i)} \neq 0$
9: **end if**
10: **end for**

account in the $t + 1^{th}$ iteration, thus creating feedback loops allowing the system to spread the knowledge provided by each $(\widetilde{D}_2^{(i)})^{\{t\}}$ within the network. The parallel splitting-based steps are presented in algorithm 6.

Algorithm 6 Parallel splitting-based algorithm

Require: Collection of matrices $\widetilde{SD}^{(i)}$, $\widetilde{WS}^{(i)}$ $(i = 1..H)$, T
Ensure: $\widetilde{D}_2$
1: **for all** i **do**
2: $((\widetilde{D}_2^{(i)})^{\{0\}}$, $(\widetilde{S}_2^{(i)})^{\{0\}}$, $(\widetilde{W}_2^{(i)})^{\{0\}}) \leftarrow Identity$
3: **for** $i = 1..T$ **do**
4: Execute every $FT - Sim^{(i)}$ with $\widetilde{SD}^{(i)}$, $\widetilde{WS}^{(i)}$ and $t = 1$
5: $(\widetilde{D}_2)^{\{t\}} \leftarrow$ Aggregation of all $(\widetilde{D}_2^{[i)})^{\{t\}}$
6: Update each $(\widetilde{D}_2^{[i)})^{\{t\}}$
7: **end for**
8: **end for**

The complexity of this architecture is obviously related to that of the $FT - Sim^{(i)}$ algorithm. In the parallel splitting-based architecture, as each instance of $FT - Sim^{(i)}$ can run on an independent core, the method can easily be parallelized, thus keeping the global complexity unchanged (considering the number of iterations as a constant factor). So, the complexity of the aggregation function can be ignored.

By splitting a matrix, we lose some information. The solution does not compute the co-similarities between all pairs of sentences but only between the words occurring in each $\widetilde{SD}^{i}$. Thanks to the feedback loops of this architecture and to the presence of the common similarity matrix $\tilde{D}_2$, we will be able to spread the information through the network and alleviate the problem of inter-matrix comparisons. Thus, by using a parallel version of $FT - Sim^{(i)}$ on H cores, we will gain both in time and space complexity: indeed,the time complexity decreases, leading to an overall gain of $1/H^2$. In the same way, the memory needed to store the similarity matrices between words will decrease by a $1/H$ factor.

VI. Conclusion

In this paper, a parallel fuzzy triadic similarity model for the co-clustering task has been proposed. It is based on a parallel architecture to tackle problems of big dimensions of matrix analysis and computational workload. This provides some interesting properties that allow a simple parallelization of the processes.

The idea is to iteratively take into account three parallel abstraction levels. The sentences consisting of one or more words are used to designate the fuzzy co-similarity of two documents. They give an ownership of words-sentences memberships in accordance with the size of a document. This can deal with the uncertainty associated with co-clustering. This ensures a good interpretation of the result of the co-clustering method which does not need to cluster the words-sentences to cluster the documents.

This model needs to take advantage of the multi-core/multiprocessor technologies which enable massively parallel processing. We propose to adopt the massively parallel application bus (MPAB) proposed in [16] which is a sort of ESB(Enterprise Service Bus). This will constitute our future work.

References

[1] S. Bedathur, K. Berberich, J. Dittrich, N. Mamoulis, and G. Weikum, "Interesting-phrase mining for ad-hoc text analytics," in *VLDB Endowment*, 2010, pp. 1348–1357.

[2] B. Long, X. Wu, Z. M. Zhang, and Z. Y. Philip, "Unsupervised learning on k-partite graphs," in *the* 12^{th} *ACM SIGKDD International Conference on Knowledge Discovery and Data mining*, 2006, pp. 317–326.

[3] L. A. Zadeh, "Fuzzy sets," *Information and Control*, vol. 8, pp. 338–353, 1996.

[4] J. B. MacQueen, "Some methods for classification and analysis of multivariate observation," in *the* 5^{th} *Berkeley Symposium on Mathematical Statistics and Probability*, 1967, pp. 281–297.

[5] J. C. Bezdek, "Fcm: The fuzzy c-means clustering algorithm," *Computers and Geosciences*, vol. 10(2-3), pp. 191–203, 1984.

[6] J. Dean and S. Ghemawat, "Mapreduce: Simplified data processing on large clusters," in *the* 6^{th} *Symposium on Operating Systems Design and Implementation*, 2004, pp. 137–149.

[7] S. Theubi, I. Feinerer, and K. Hornik, "Distributed text mining in r," in *Research Report Series Department of Statistics and Mathematics*, 2011, p. 107.

[8] I. Drost, S. Bickel, and T. Scheer, "Discovering communities in linked data by multi-view clustering," in *the* 29^{th} *Annual Conference of the German Classification Society, Studies in Classication, Data Analysis, and Knowledge Organization*, 2005, pp. 342–349.

[9] W. Tang, Z. Lu, and I. S. Dhillon, "Clustering with multiple graphs," in *the IEEE International Conference on Data Mining*, 2009, pp. 1016–1021.

[10] F. de Carvalho, Y. Lechevallier, and F. M. de Melo, "Partitioning hard clustering algorithms based on multiple dissimilarity matrices," *Pattern Recognition*, vol. 45, pp. 447–464, 2012.

[11] A. Kumar and H. Daume, "A co-training approach for multi-view spectral clustering," in *the* 28^{th} *International Conference on Machine Learning*, 2011, pp. 393–400.

[12] G. Bisson and C.Grimal, "Co-clustering of multi-view datasets: a parallelizable approach," in *the IEEE International Conference on Data Mining*, 2012, pp. 828–833.

[13] F. Hussain, *X-Sim: A New Cosimilarity Measure: Application to Text Mining and Bioinformatics*. Phd Thesis, 2010.

[14] S. Alouane, M. S. Hidri, and K. Barkaoui, "Fuzzy triadic similarity for text categorization: Towards parallel computing," in *the* 5^{th} *International Conference on Web and Information Technologies*, 2013, pp. 265–274.

[15] S. Kundu, "Min-transitivity of fuzzy leftness relationship and its application to decision making," *Fuzzy Sets and Systems*, vol. 93, pp. 357–367, 1997.

[16] R. Benosman, Y. Albrieux, and K. Barkaoui, "Performance evaluation of a massively parallel esb-oriented architecture," in *Service-Oriented Computing and Applications*, 2012, pp. 1–4.