Arabic Cursive Characters Distributed Recognition using the DTW Algorithm on BOINC: Performance Analysis

Zied TRIFA, Mohamed LABIDI and Maher KHEMAKHEM
Department of Computer Science, University of Sfax
MIRACL Lab
Sfax, Tunsia
trifa.zied@gmail.com, mohamedlabidi@yahoo.fr
maher.khemakhem@fsegs.rnu.tn

***Abstract*— Volunteer computing or volunteer grid computing constitute a very promising infrastructure which provides enough computing and storage powers without any prior cost or investment. Indeed, such infrastructures are the result of the federation of several, geographically dispersed, computers or/and LAN computers over the Internet. Berkeley Open Infrastructure for Network Computing (BOINC) is considered the most well-known volunteer computing infrastructure. In this paper, we are interested, rather, by the distribution of the Arabic OCR (Optical Character Recognition) based on the DTW (Dynamic Time Warping) algorithm on the BOINC, in order, to prove again that volunteer computing provides very interesting and promising infrastructures to speed up, at will, several greedy algorithms or applications, especially, the Arabic OCR based on the DTW algorithm. What makes very attractive the Arabic OCR based on the DTW algorithm is the following, first, its ability to recognize, properly, words or sub words, without any prior segmentation, from within a reference library of isolated characters. Second, its good immunity against a wide range of noises. Obtained first results confirm, indeed, that the Berkeley Open Infrastructure for Network Computing constitutes an interesting and promising framework to speed up the Arabic OCR based on the DTW algorithm.**

Keywords*— *Volunteer Computing; BOINC; Arabic OCR; DTW algorithm;

I. INTRODUCTION

Arabic OCR based on the DTW algorithm provides very interesting recognition and segmentation rates. One of the advantages of the DTW algorithm is its ability to recognize, properly, words or connected characters without their prior segmentation. In our previous studies achieved on high and medium quality documents [2], [3], [5] we obtained an average of more than 98% as recognition rate and more than 99% as segmentation rate. The purpose of the DTW algorithm is to perform the optimal time alignment between a reference pattern and an unknown pattern in order to ease the evaluation of their similarity. Unfortunately, the drawback of the DTW is its complex computing [6], [7], [8]. Consequently, several solutions and approaches have been proposed to speed up the DTW algorithm, [1], [7], [8], [4], [9], [5].

In this paper we show and confirm, through an experimental study, how volunteer computing, which present the advantage to be costless, can speed up, substantially also, the execution time of the Arabic OCR using the DTW algorithm. More specifically, we show how BOINC can achieve such a mission.

The reminder of this paper is organized as follows; section (2) describes the Arabic OCR using the DTW algorithm. Section (3), gives an overview on volunteer computing especially BOINC (Berkeley Open Infrastructure for Network Computing). The proposed approach and the corresponding performance evaluation are detailed in Section (4). Conclusion remarks and some future investigation are presented in section (5).

II. MECHANISM OF THE ARABIC OCR BASED ON THE DTW ALGORITHM

Words in Arabic are inherently written in blocks of connected characters. We need a prior segmentation of these blocks into separated characters. Indeed many researchers have considered the segmentation of Arabic words into isolated characters before performing the recognition phase. The viability of the use of DTW technique, however, is its ability and efficiency to perform the recognition without prior segmentation [4, 2].

We consider in this paper a reference library of R trained characters forming the Arabic alphabet in some given fonts, and denoted by C_r : r=1, 2,…, R. The technique consists to use the DTW pattern method to match an input character against the reference library. The input character is thus recognized as the reference character that provides the best time alignment, namely character A is recognized to be C_k if the summation distance S_k corresponding to the matching of A to reference character C_k satisfies the following equation [4, 5].

$$S_k = min_{1 \le r \le R}\{S_r\} \quad (1)$$

Let T constitutes a given connected sequence of Arabic characters to be recognized. T is then composed of a sequence of N feature vectors T_i that are actually representing the concatenation of some sub sequences of feature vectors representing each an unknown character to be recognized. As portrayed in Fig.1 text T lies on the time axis (the X-axis) in such a manner that feature vector T_i is at time i on this axis.

The reference library is portrayed on the Y-axis, where reference character C_r is of length l_r, $1 \leq r \leq R$. Let S (i, j, r) represent the cumulative distance at point (i, j) relative to reference character C_r . The objective here is to detect simultaneously and dynamically the number of characters composing T and recognizing these characters. There surely exists a number k and indices (m_1, m_2, ..., m_k) such that $Cm_1 \oplus Cm_2 \oplus \ldots \oplus Cm_k$ represent the optimal alignment to text T where $\oplus$ denotes the concatenation operation. The path warping from point (1, 1, m_1) to point (N, l_{mk},k) and representing the optimal alignment is therefore of minimum cumulative distance that is:

$$S(N, l_{mk}, k) = min_{1 \leq r \leq R}\{S(N, l_r, r\} \quad (2)$$

This path, however, is not continuous since it spans many different characters in the distance matrix. We therefore must allow at any time the transition from the end of one reference character to the beginning of another reference character. The end of reference character C_r is first reached whenever the warping function reaches point (i, l_r, r), i $= \left\lceil \frac{l_{r+1}}{2} \right\rceil$,...,N. As we can see from Fig.1, the end of reference characters C_1, C_2, C_3 are first reached at time 3, 4, 3 respectively. The end points of reference characters are shown in Fig.1 inside diamonds and points at which transitions occur are within a circle. The warping function always reaches the ends of the reference characters. At each time i, we allow the start of the warping function at the beginning of each reference character along with addition of the smallest cumulative distance of the end points found at time (i - 1) [4,5]. The resulting functional equations are:

$$S(i,j,r) = D(i,j,r) + min_{\substack{1 \leq i \leq N \\ 1 \leq j \leq l_r \\ 1 \leq r \leq R}} \begin{Bmatrix} S(i-1,j,r), \\ S(i-1,j-1,r), \\ S(i-1,j-2,r) \end{Bmatrix} \quad (3)$$

With the boundary conditions:

$$S(i,1,r) = D(i,1,r) + min_{\substack{1+\lceil \frac{lr+1}{2} \rceil \leq i \leq N \\ l \leq j \leq l_r \\ l \leq r \leq R}} S(i-1, l_K, k) \quad (4)$$

To trace back the warping function and the optimal alignment path, we have to memorize the transition times among reference characters. This can easily be accomplished by the following procedure:

$$b(i,j,r) = trace\ min_{\substack{1 \leq i \leq N \\ 1 \leq j \leq l_r \\ 1 \leq r \leq R}} \begin{Bmatrix} b(i-1,j,r), \\ b(i-1,j-1,r), \\ b(i-1,j-2,r) \end{Bmatrix} \quad (5)$$

Where trace min is a function that returns the element corresponding to the term that minimizes the functional equations. The functioning of this algorithm is portrayed on Fig.1 by means of the two vectors VecA and VecB, where VecB(i) represents the reference character giving the least cumulative distance at time i, and VecA(i) provides the link to the start of this reference character in the text T . The heavy marked path through the distance matrix represents the optimal alignment of text T to the reference library. We observe that the text is recognized as C1 $\oplus$ C3 [5].

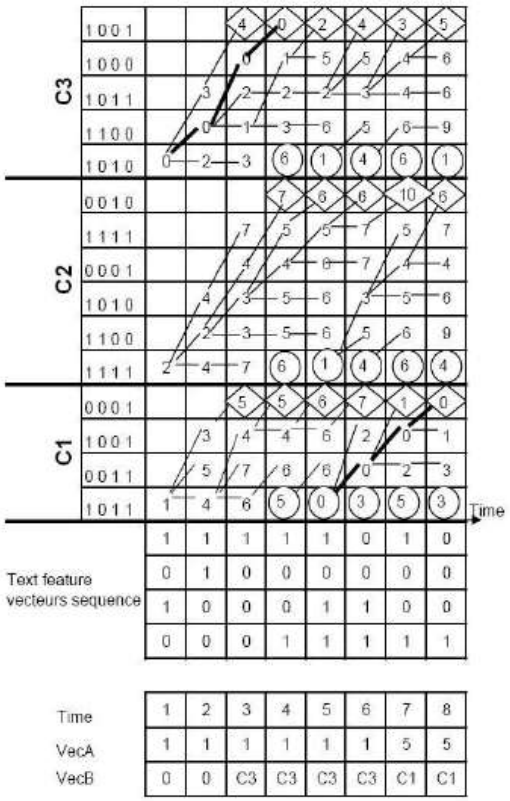

Figure 1. DTW Wrapping mechanism.

III. VOLUNTEER COMPUTING AND BOINC

Volunteer computing is a form of distributed computing in which the general public volunteer make available and storage resources to scientific research projects. Early volunteer computing projects include the Great Internet Mersenne Prime Search [12], SETI@home [10], Distributed.net [11] and Folding@home [13]. Today the approach is being used in many areas, including high energy physics, molecular biology, medicine, astrophysics, and climate dynamics. This type of computing can provide great power (SETI@home, for example, has accumulated 2.5 million years of CPU time in 7 years of operation). However, it requires attracting and retaining volunteers, which places many demands both on projects and on the underlying technology.

The Berkeley Open Infrastructure for Network Computing (BOINC) is a framework to deploy distributed computing platforms based on volunteer computing. It is developed at U.C. Berkeley Spaces Sciences Laboratory and it is released under an open source license.

BOINC is the evolution of the original SETI@home project, which started in 1999 and attracted millions of participants worldwide [16].

This middleware provides a generic framework for implementing distributed computation applications within a heterogeneous environment. The system is designed as a

software platform utilizing computing resources from volunteer computers [14].

BOINC software is divided in two main components: the server and the client side of the software. BOINC allows sharing computing resources among different autonomous projects. A BOINC project is identified by a unique URL which is used by BOINC clients to register with it. Every BOINC project must run a host with the server side of the BOINC software.

BOINC software on the server side comprises several components: one or more scheduling servers that communicate with BOINC clients, one or more data servers that distribute input files and collect output files, a web interface for participants and project administrators and a relational database that stores information about work, results, and participants.

BOINC software provides powerful tools to manage the applications run by a project. For instance it allows to easily define different application versions for different target architectures. A "workunit" describes a computation to be performed, associating a (unique) name with an application and the corresponding input files.

Not all kind of applications are suitable to be deployed on BOINC. Ideally, candidate applications must present "independent parallelism" (divisible into parallel parts with few or no data dependencies) and a low data/compute ratio since output files will be sent through a typically slow commercial Internet connection [16].

In this paper, we show how volunteer computing such as BOINC can speed up the execution time of Arabic OCR based on the DTW algorithm.

IV. THE DTW DATA DISTRIBUTION OVER BOINC

The Arabic OCR based on the DTW procedure described in the preceding section presents many ways on which one could base its parallelization or distribution. The idea of the proposed approach is how to take advantages of the enough power provided by BOINC to speed up the execution time of the DTW algorithm?

A BOINC project uses a set of servers to create, distribute, record, and aggregate the results of a set of tasks that the project needs to perform to accomplish its goal. The tasks are evaluating data sets, called workunits. The servers distribute the tasks and corresponding workunits to clients (software that runs on computers that people permit to participate in the project). When a computer running a client would otherwise be idle (in the context of volunteer computing, a computer is deemed to be idle if the computer's screensaver is running), it spends the time working on the tasks that a server assigns to the client. When the client has finished a task, it returns the result obtained by completing the task to the server. If the user of a computer that is running a client begins to use the computer again, the client is interrupted and the task it is processing is paused while the computer executes programs for the user. When the computer becomes idle again, the client continues processing the task it was working on when the client was interrupted.

To be added into a BOINC project, applications must incorporate some interaction with the BOINC client: they must notify the client about start and finish, and they must allow for renaming of any associated data files, so that the client can relocate them in the appropriate part of the guest operating system and avoid conflicts with workunits from other projects [16].

We propose to split optimally the binary image of a given Arabic text to be recognized into a set of binary sub images and then assign them among some volunteer computers which are already subscribed to our project.

BOINC uses a simple but a rich set of abstraction files, applications, and data. A project defines application versions for various platforms (Windows, Linux/x86, Mac OS/X, etc.).

An application can consist of an arbitrary set of files. A workunit represents the inputs to a computation: the application (but not a particular version) presents a set of references input files, and sets of command line arguments and environment variables. Each workunit has parameters such as computing, memory and storage requirements and a soft deadline for completion. A result represents the result of a computation, it consists of a reference to a workunit and a list of references to output files.

Files can be replicated, the description of a file includes a list of URLs from which it may be downloaded or uploaded. When the BOINC client communicates with a scheduling server it reports completed work, and receives an XML document describing a collection of the above entities. The client then downloads and uploads files and runs applications; it maximizes concurrency, using multiple CPUs when possible and overlapping communication and computation.

A. Experimental Study

Significant reduction in the elapsed time, defined as the time elapsing from the start until the completion of the text recognition, can be realized by using a distributed architecture. This effect is known as the speedup factor. This factor is properly defined as the ratio of the elapsed time using sequential mode with just one processor to the elapsed time using the distributed architecture.

Next we consider only the case where the volunteer computers participating in the work are homogeneous. It means that all the corresponding interconnected computers are homogeneous in terms of computing power, hardware configuration and operating system. We ran several experiments on several specific printed Arabic texts.

Our experiments aim at proving that volunteer computing present, indeed, interesting infrastructures to speed up the execution process of the Arabic OCR based on the DTW algorithm.

During these experiments, we have considered the following conditions:

- The number of pages is 100,
- The number of lines per page is 7,
- The average number of characters per line is 55,

- The number of characters per page is 369,
- The reference library contains 103 characters,
- We have used 16 dedicated homogeneous workers having the exact configuration: 3GHZ CPU frequency, 512 Mega Octets RAM and running Windows XP-professional.

To reach our expectation, we have studied the effect of the distribution of approximately a hundred (100) of similar printed Arabic text pages over a variable number of volunteer computers.

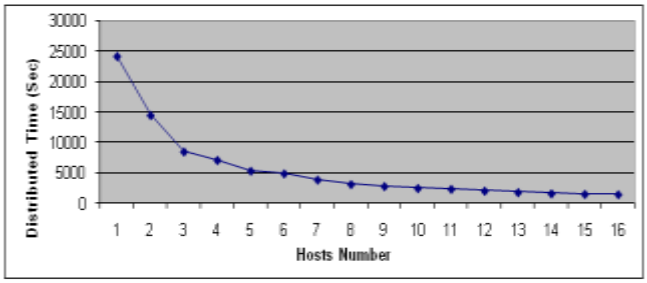

Figure 2. Distributed execution time.

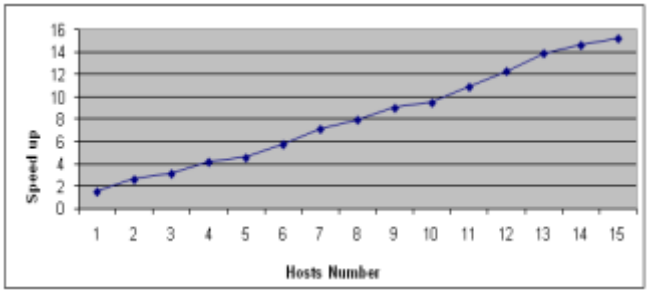

Figure 3. Speedup of the DTW algorithm.

Fig. 2 and 3 illustrate the obtained results of our experiment. These figures show in particular that:

- The execution time of the DTW algorithm decreases with the number of computers used. Each time you add a computer the execution time of recognition decrease. The average test time for 1 computer was approximately 6.20 hours and the average test time for 16 computers was 0.4 hours. It clearly shows an exponential decrease in the amount of time required to complete the tests.
- However, the speedup factor increases with the number of computers used.
- If we use 16 computers then the execution time reaches the value 1450 seconds and the speedup factor reaches the value 15. This result is very interesting, because in this case our proposed OCR system is able to recognize more than 830 characters per second, compared with the existing commercial Arabic OCR [2].

Consequently, volunteer computing constitute interesting infrastructures to speedup, drastically, the execution time of the Arabic OCR based on the DTW algorithm. Moreover, and thanks to the enough computing power provided by such infrastructures, we can think, now, about the improvement of the recognition rate of our system by adding to it some complementary approaches or techniques.

V. Conclusion and Perspective

This paper has shown how volunteer computing present interesting infrastructures to speed up, substantially, the execution time of the Arabic OCR based on the DTW algorithm. Indeed, conducted experiments confirm that such infrastructures can help a lot in building a powerful Arabic OCR based on the combination (integration) of some strong complementary approaches or techniques which require enough computing power.

Several investigations are under studies especially the way to exploit in a large scale the BOINC and the way to improve the recognition rate of the Arabic OCR based on the DTW algorithm in order to build a powerful Arabic OCR system.

References

[1] M. Khemakhem, A. Belghith, M. Labidi « The DTW data distribution over a grid computing architecture », International Journal of Computer Sciences and Engineering Systems (IJCSES), Vol.1, N°.4, p. 241-247, December 2007

[2] M. Khemakhem, A. Belghith: « Towards a distributed Arabic OCR based on the DTW algorithm », the International Arab Journal of Information Technology (IAJIT), Vol. 6, N°. 2, p. 153-161, April 2009.

[3] M. Khemakhem et al., Arabic Type Writen Character Recognition Using Dynamic Comparison, Proc. 1st Computer conference Kuwait, March. 1989.

[4] M. Khemakhem, A. Belghith, M. Ben Ahmed, Modélisation architecturale de la Comparaison Dynamique distribuée Proc, Second International Congress On Arabic and Advanced Computer Technology, Casablanca, Morocco, December 1993.

[5] M. Khemakhem and A. Belghith., A Multipurpose Multi-Agent System based on a loosely coupled Architecture to speed up the DTW algorithm for Arabic printed cursive OCR. Proc. IEEE AICCSA-2005, Cairo, Egypt, January 2005.

[6] A.Cheung et al. An Arabic optical character recognition system using recognition based segmentation, Pattern Recongnition. Vol. 34, 2001.

[7] G. R. Quénot et al., A Dynamic Programming Processor for Speech Recognition, IEEE JSSC, Vol. 24, N(F 9)q.20, April 1989.

[8] Philip G. Bradford, Efficient Parallel Dynamic Programming, Proc. the 30 Annual Allerton Conference on Communication, Control an Computing, University of Illinois, 185-194, 1992.

[9] C. E. R. Alves et al, Parallel Dynamic Programming for solving the String Editing Problem on CGM/BSP, Proc. SPAA'02, August 10-13, 2002 Winnipeg, Manitoba, Canada.

[10] D.P. Anderson, J. Cobb, E. Korpela, M. Lebofsky, D. Werthimer. "SETI@home: An Experiment in Public-Resource Computing". Communications of the ACM, November 2002

[11] Distributed.net, http://distributed.net

[12] GIMPS, http://www.mersenne.org/prime.htm

[13] S.M. Larson, C.D. Snow, M. Shirts and V.S. Pande. "Folding@Home and Genome@Home: Using distributed computing to tackle previously intractible problems in computational biology". Computational Genomics, Horizon Press, 2002.

[14] D.P. Anderson. "BOINC: A System for Public-Resource Computing and Storage". 5th IEEE/ACM International Workshop on Grid Computing, Nov. 8 2004.

[15] Einstein@Home, http://einstein.phys.uwm.edu/

[16] B.Antoli, F. Castejón, A.Giner, G.Losilla, J.M Renolds, A.Rivero, S.sangiaos, F.Serrano, A. Tarancón, R. Vallés and J.L. Velasco "ZIVIS: A City Computing Platform Based on Volunteer Computing"

[17] D.P. Anderson, E. Korpela, and R. Walton. "High-Performance Task Distribution for Volunteer Computing". First IEEE International Conference

[18] Gupta, S., Doshi, V., Jain, A., & Iyer, S. (2010). Iris Recognition System using Biometric Template Matching Technology. International Journal of Advanced Computer Science and Applications - IJACSA, 1(2), 24-28. doi: 10.5120/61-161.

[19] Padmavathi, G. (2010). A suitable segmentation methodology based on pixel similarities for landmine detection in IR images. International Journal of Advanced Computer Science and Applications - IJACSA, 1(5), 88-92.

[20] Miriam, D. D. H. (2011). An Efficient Resource Discovery Methodology for HPGRID Systems. International Journal of Advanced Computer Science and Applications - IJACSA, 2(1).

[21] Hamdy, S., El-messiry, H., Roushdy, M., & Kahlifa, E. (2010). Quantization Table Estimation in JPEG Images. *International Journal of Advanced Computer Science and Applications - IJACSA, 1*(6), 17-23.

AUTHORS PROFILE

Maher Khemakhem received his master of science and his PhD degrees from the University of Paris 11, France in 1984 and 1987, respectively. He is currently assistant professor in computer science at the Higher institute of Management at the University of Sousse Tunisia. His research interests include distributed systems, performance evaluation, and pattern recognition.

Zied Trifa received his master degree of computer science from the University of Economics and management of Sfax, Tunisia in 2010. He is currently PhD student in computer science at the same University. His research interests include Grid Computing, distributed systems, and performance evaluation.

Mohamed Laabidi received his master degree of computer science from the University of Economics and management of Sfax, Tunisia in 2007. He is currently PhD student in computer science at the same University. His research interests include Cloud and Grid Computing, distributed systems, and performance evaluation.