World of Computer Science and Information Technology Journal (WCSIT)
ISSN: 2221-0741
Vol. 1, No. 4, 132-137, 2011

Dynamic Time Warping Algorithm with Distributed Systems

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—Distributed computing is the method of splitting a large problem into smaller pieces and allocating the workload among many computers. These individual computers process their portions of the problem, and the results are combined together to form a solution for the original problem. At present, Distributed computing systems can be broadly classified into two methods, namely Grid computing and Volunteer computing. In this paper, we are interested by the distribution of the Arabic OCR (Optical Character Recognition) based on the DTW (Dynamic Time Warping) algorithm on the distributed computing systems such as Scientific Research Tunisian Grid (SRTG) and Berkeley Open Infrastructure for Network Computing (BOINC), and we present the performance analysis of an experimental study of our distribution in order, to prove again that such systems 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.

Keywords- Grid Computing; Volunteer Computing; Arabic OCR; DTW algorithm; SRTG; BOINC.

I. INTRODUCTION

Data and programs in centralized applications are kept at one site and this is conceived as a bottleneck in performance and availability of remote information in desktop computers. Distributed systems were emerged to remove this flaw. During 1990s, distributed systems were used for information exchange between remote desktop computers. In these years, they consisted of different computers connected to each other and located at geographically remote sites. This was the starting point for emerging concepts such as Peer-to-Peer (P2P) Computing [1], Agents [2], Grid Computing [3] and Volunteer Computing [4].

When viewed alongside heavily distributed computation systems such as grid computing and volunteer computing, the strength of many Arabic OCR techniques comes into question. The raw processing power available to grid computers makes circumventing certain Arabic OCR algorithm viable in regard to computing time.

Arabic OCR based on the Dynamic Time Warping (DTW) algorithm is a well known procedure especially in pattern recognition [5]. In fact, this procedure is the result of the adaptation of dynamic programming to the field of pattern recognition. The purpose of the DTW algorithm is to perform optimal time alignment between a reference pattern and an unknown pattern and evaluate their difference. Arabic printed cursive OCR based on the DTW algorithm provides very interesting recognition rates. Conducted experiments achieved on high and medium quality documents containing around 20000 Arabic words show that the recognition average rate is more than 98% and the segmentation average rate is more than 99% [6],[7]. Unfortunately, the underlying complex computing of this algorithm makes its execution time very slow and hence restricts its utilization.

This paper examines a small-scale implementation of a publicly available distributed computing system for computing a subset of the Arabic OCR based on the DTW algorithm.

The paper begins with an introduction to the Arabic OCR based on DTW algorithm. The next section introduces the Distributed Computing Systems. This is followed by a brief view on the grid computing SRTG and the volunteer computing BOINC. Finally, the results of the distributed implementation are analyzed and conclusions are presented.

II. THE DYNAMIC TIME WARPING (DTW) ALGORITHM

An OCR system is generally decomposed into four stages as shown on Fig.1. The first one concerns the acquisition of the text scanned image to be provided in the form of pixels or binary data. The second stage deals with the pre-processing of this raw data and mainly concerns filtering the scanned image, framing and positioning and the segmentation of the text. The pre-processing measurement vectors are however a completely inadequate support for the decision process. This is the task of the third stage which concerns the description and feature extraction, and hence the determination of characteristic fragments of the character or the group of connected (cursive)

characters to be recognized so that a certain combination of characteristic fragments can be assigned with adequate confidence by the decision process to a recognized class. The final stage forms the culminating point of the recognition process: the matter of decision on the correct classification of the unknown. What makes DTW an attractive algorithm to use in the recognition process is its ability to eliminate time differences between the characters or shapes to be recognized [6], [8].

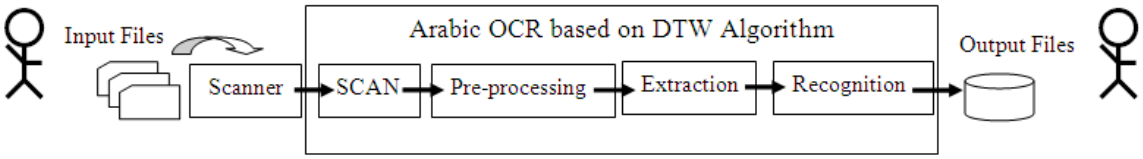

Figure 1. Arabic OCR system

Based on the dynamic programming path finding, DTW presents a computationally efficient algorithm to find the optimal time alignment between two occurrences of the same character and more generally between any two given forms.

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 on Fig.2 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) represents 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 $Cm_1 \oplus Cm_2 \oplus \ldots \oplus Cm_k$ represents 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:

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

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.2, 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 on Fig.2 inside diamonds and points at which transitions occur are within 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) [5]. The resulting functional equations are:

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

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:

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

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.2 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. The heavy marked path through the distance matrix represents the optimal alignment of text to the reference library. We observe that the text is recognized as C1 $\oplus$ C3.

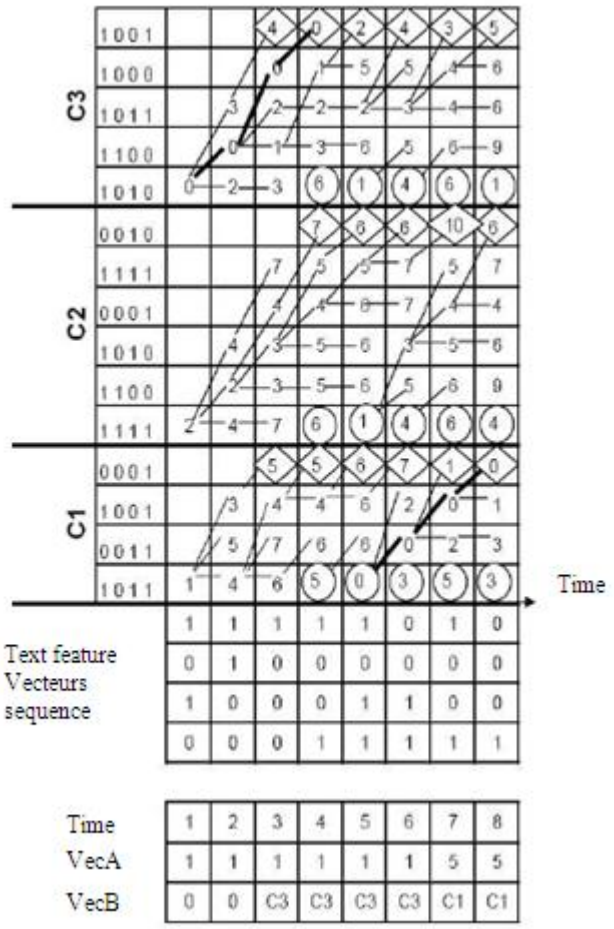

Figure 2. The DTW mechanism.

III. DISTRIBUTED COMPUTING

Distributed Computing is the natural frame for the solution of numerical problems where a task can be divided into independent pieces, and whose ratio of computation to data is high. Every work unit is sent to a different computer, while the central system collects and analyzes the results [9]. At present, Distributed computing systems can be broadly classified into two systems, namely Grid computing and Volunteer Computing. Examples of such systems include: The Scientific Research Tunisian Grid (SRTG) as Grid Computing and Berkeley Open Infrastructure for Network Computing (BOINC) as Volunteer Computing.

Grid computing and Volunteer Computing share the goal of better utilizing existing computing resources. However, there are profound differences between the two paradigms. Grid computing involves organizationally owned resources: supercomputers, clusters, and PCs owned by universities, research labs, and companies. These resources are centrally managed by IT professionals, are powered on most of the time, and are connected by full time, high-bandwidth network links. In contrast, public resource computing or volunteer computing can provide more computing power than any other supercomputer, cluster, or grid, and the disparity will grow over time. Most participants are individuals, who connected to the Internet by telephone or cable modems or DSL, and often behind network-address translators (NATs) or firewalls [10].

A. Grid Computing and SRTG

Grid computing can be defined as the coordinated resource sharing and problem solving in dynamic, multi institutional collaborations [11]. More simply, Grid computing typically involves using many resources (computer, data, I/O, instruments, etc.) to solve a single, large problem that could not be executed on any one resource. As a matter of fact, various Grid application scenarios have been explored within both science and industry. These applications include compute-intensive, data-intensive, sensor-intensive, knowledge-intensive and collaboration-intensive scenarios and address problems ranging from fault diagnosis in jet engines and earthquake engineering to bioinformatics, biomedical imaging, and astrophysics [12]. This huge ability of sharing resources in various combinations will lead to many advantages such as increase the efficiency of resource usage, facilitate the remote collaboration between institutions and researchers, give to users a huge computing power, and give to users a huge storage capacity.

The Scientific Research Tunisian Grid (SRTG) is implemented by the research team UTIC [13]. It is similar to the XtremWeb-CH [14] which is an improved version of XtremWeb [15]. The main goal of the SRTG is to provide to Tunisian researchers an effective experimental framework to achieve their different needs such as the deployment of greedy applications and their corresponding performance evaluation.

B. Volunteer Computing and BOINC

Volunteer computing is a form of distributed computing in which the general public volunteers processing and storage resources to scientific research projects. Early volunteer computing projects include the Great Internet Mersenne Prime Search [16], SETI@home [4], Distributed.net [17] and Folding@home [19]. 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.

BOINC (Berkeley Open Infrastructure for Network Computing) is a middleware system for volunteer computing. BOINC is being used by a number of projects, including SETI@home, Climateprediction.net [5], LHC@home [20], and Einstein@Home [18]. Volunteers participate by running BOINC client software on their computers. They can attach each computer to any set of projects, and can control the allocation of resources among projects.

IV. DISTRIBUTED ALGORITHM PERFORMANCE

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 a given distributed computing systems such as the SRTG and BOINC to speed up the DTW algorithm? 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 first among some computers interconnected to the SRTG and second among some volunteer computers which are already subscribed to our project over BOINC.

A. *The DTW data Distribution over SRTG*

SRTG is composed of several institutions heterogeneous computers interconnected trough the Internet. One of these computers is named the coordinator and the remaining one is named worker. The coordinator is responsible of the management of the recognition process and the coordination among workers. The coordinator is working as a web service. Thus if we need to launch on the SRTG a distributed Arabic recognition process, we have first to log into the coordinator, ask it about the number, the computing capacity and the Operating System of available workers. Then, we have to fix the target workers that will participate in the work and finally we have to prepare the different files (in XML format) [5] required to achieve this task. These files which include the data to be processed (the binary sub image) and the code to be executed by every worker must be sent to the coordinator. After receiving these files, the coordinator assigns them to the target workers. After achieving the recognition process, every worker must turn back obtained results (recognized sub texts) to the coordinator. The coordinator must turn back to the user the totality of received results from workers.

Our experiment aims to implement the proposed approach and to prove that the speedup factor increases with the number of workers used. We have considered the following conditions:

- The studied application was implemented in the "C sharp" language.
- We have used 9 dedicated homogeneous workers having the exact same configuration: 3GHZ CPU frequency, 512 Mega Octets RAM and running Windows XP-professional.
- We have used a text corpus formed of 7000 Arabic words randomly chosen which were scanned using an HP scanner with a resolution of 300 dpi (dots per inch).
- We have considered also a reference library composed of 103 characters representing approximately the totality of the Arabic alphabet (including the characters shape variation according to their position within words).
- The grid network capacity was around 100KBs.
- The XML file has been generated manually.

Fig. 3 and Fig.4 illustrate the obtained results of the described experiment.

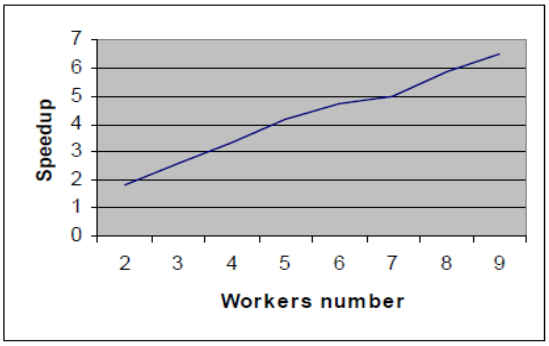

Figure 3. Speedup of the distribution of 7000 Arabic words.

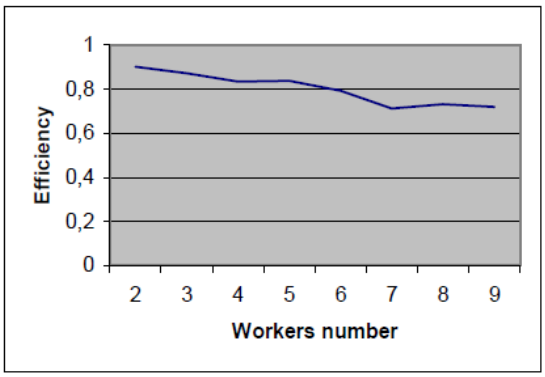

Figure 4. Efficiency of the distribution of 7000 Arabic words.

These figures show in particular that:

- The speedup factor increases as the number of workers increases;
- The efficiency factor is always >0.58, it means that more than 58% of the computing power of the workers participating in the work is used;
- If we use 9 workers then the speedup factor reaches the value 6.7 which will lead to the recognition of more than 250 Arabic characters per second. This is a very interesting result given that currently commercialized systems have approximately the same speed but less recognition rate c,f., [21] compared to our approach especially for medium and low quality texts (documents).

B. *The DTW data Distribution over BOINC*

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 [22].

Throughout our experiment to prove that BOINC can constitute an interesting and promising framework to speed up the Arabic OCR, 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 average 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.

Fig.5, Fig.6 and Fig.7 illustrate the obtained results of the described experiment.

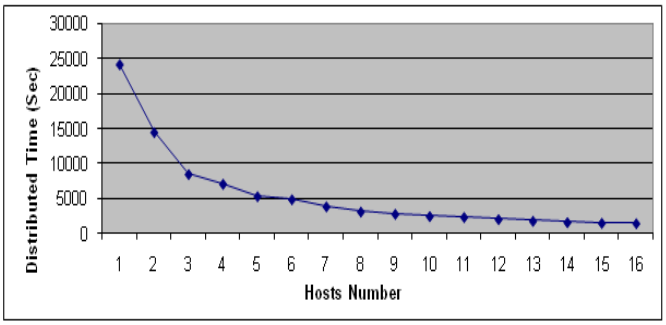

Figure 5. Distributed execution time of 100 Arabic pages.

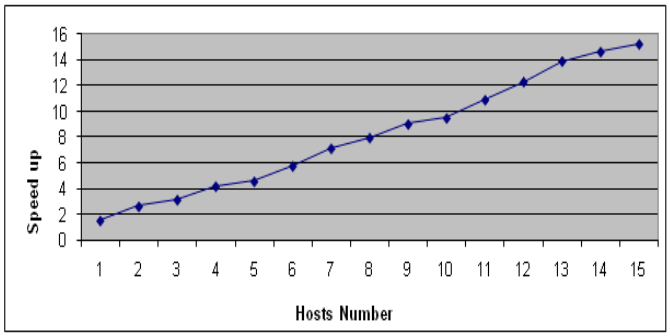

Figure 6. Speedup of the distribution of 100 Arabic pages.

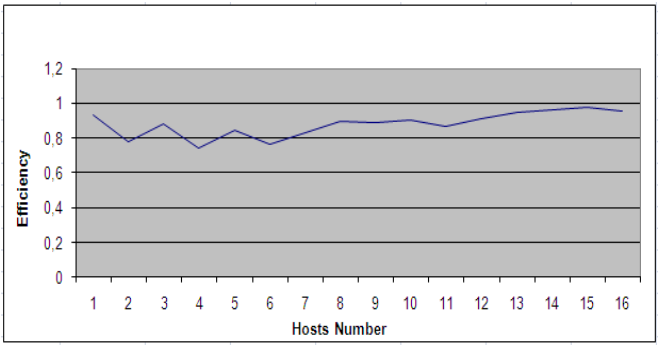

Figure 7. Efficiency of the distribution of 100 Arabic pages.

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 one computer was approximately 6.20 hours and the average test time for sixteen 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.
- The efficiency factor reaches the value 0.95 which means that the computing power of each dedicated worker is used for more than 95%.
- 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.

Consequently, obtained results confirm that distributed computing systems and more specifically grid computing and volunteer computing present a very interesting framework to speed up the Arabic optical character recognition based on the dynamic time warping algorithm.

V. CONCLUSION

The mechanics of distributed computing are straightforward, and platforms like SRTG and BOINC have made running them quite practicable. The size and big amount of computing of some applications like Arabic printed cursive characters Recognition using the Dynamic Time Warping (DTW) means that they must run on clusters for the foreseeable future. Even when the parallelization is possible, many design parameters must be established in order to construct a usable experiment. We have explored some of those parameters here. In future work, we intend to develop an autonomic computing architecture to distribute the complex application of the printed Arabic Optimal Character Recognition.

REFERENCES

[1] D. S. Milojicic, V. Kalogeraki, R. Lukose, K. Nagaraja, J. Pruyne, B. Richard, S. Rollins, and Z. Xu. Peer-to-Peer Computing. In *Proceedings of the Second International Conference on Peer-to-Peer Computing*, pages 1–51, July 2002.

[2] G. Tesauro and et al. A Multi-agent systems approach to autonomic computing. In *IBM Press*, pages 464–471, March 2004.

[3] Ian Foster, Carl Kesselman, and Steven Tuecke., The Anatomy of the Grid Intl J. Supercomputer Applications, 2002.

[4] 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

[5] 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

[6] N. Abedi, M. Khemakhem, Reconnaissance de caractères imprimés cursifs arabes par Comparaison dynamique et modèle caché de Markov Proc. GEI2004, Monastir, Tunisia, March 2004.

[7] M. Khemakhem and A. Belghith., The DTW Algorithm for Distributed Printed Cursive OCR within A Multi Agent System, Proc, ACM, ICICIS Cairo, Egypt, on March 14-18, 2007.

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

[9] I. Foster and C, Kesselman. Globus: A metacomputing infrastructure toolkit. Intel Supercomputer Applications, 11(2), p. 115-128, 1997.

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

[11] J. Nabrzyski, J. M. Schopf, J. W. Eglarz Grid Resource Management: State of the Art and Future Trends. Kluwer Academic Publishers, 2003.

[12] I. Foster, C. kesselman The Grid: Blueprint for a New Computing Infrastructure. 2nd Ed, Morgan Kaufmann, 2004.

[13] http://www.esstt.rnu.tn/utic/gtrs/

[14] http://www.xtremwebch.net

[15] http://www.xtremweb.net

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

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

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

[19] 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.

[20] LHC@home, http://athome.web.cern.ch/athome/

[21] CiyaICR product : http://www.ciyasoft.com/.

[22] 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"

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 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 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.