TOWARDS AN OPTIMAL UTILIZATION OF VOLUNTEER GRID COMPUTING: A COMPARATIVE STUDY OF THREE HEURISTICS

Maher Khemakhem[1], Abdelaziz Dammak[2] and Taha Chaâbouni[1]

[1]MIRACL Lab, FSEG, University of Sfax, Tunisia
Maher.Khemakhem@fsegs.rnu.tn,
[2] FSEG, University of Sfax, Tunisia
Abdelaziz.Dammak@fsegs.rnu.tn
chaabounitaha@yahoo.fr

ABSTRACT

Volunteer Grids present very interesting and attractive infrastructures that reduce, drastically, the response time of several greedy algorithms and applications such as the Arabic OCR (Optical Character Recognition) based on the Dynamic time Warping (DTW) algorithm. Intensive experiments performed on such infrastructures confirm their ability to provide enough computing and storage powers which can be exploited and used to substantially speed up the execution time of the Arabic OCR based on the DTW algorithm. Amongst the advantages of such infrastructures we can mention first their cost which is almost zero, since they are, commonly, obtained by federating, through the Internet, several computers which are geographically dispersed. Second, their flexibility since they can be simultaneously used by different distributed applications. As a consequence, these infrastructures drain, today, more and more researchers to discover what they can do with in order to respond better to the ever increasing user needs in terms of computing and storage powers. Unfortunately, and to the best of our knowledge, there is no previous research work which attempted to solve the problem of the optimal utilization of the computing power provided by such infrastructures. This is due, surely, to the complex structure of such infrastructures where the federated resources are highly volatile as we will explain next.

Consequently, we present in this paper a new approach followed by a comparative study of three heuristics which attempt to reach an optimal utilization of any provided computing power by any given volunteer grid. This approach exploits and uses volunteer grids during, rather, some predefined timeslots where tasks are assigned optimally or pseudo optimally over the corresponding available computers. Our proposal supposes, indeed, that we have, already, the forecast state of the volunteer grid to be used in the form of timeslots obtained by a reliable statistical study done, previously, on this grid.

KEYWORDS

Volunteer grids, heuristics, pseudo optimal assignment, independent computing tasks and timeslots.

1. INTRODUCTION

Volunteer grids can be considered as the only way to, drastically, speedup the response time of several complex (greedy) algorithms and applications without any prior cost or investment [12]. Indeed, conducted experiments have shown and confirmed that such infrastructures constitute a very promising and interesting way to, substantially, speedup, the Arabic OCR based on the DTW algorithm where the corresponding tasks are independent from each other. A volunteer grid can be viewed, in fact, as a huge virtual computer resulting of the voluntary federation of a big

number of computers and resources interconnected through the internet [1], [7] and [8]. As a result of this federation, owners of these federated computers and resources can have, at will, enough computing power and storage. Conducted experiences revealed that these kinds of infrastructures present several advantages such as the possibility of a simultaneous utilization of the corresponding resources by the authorized users compared to dedicated grids where only one utilization is allowed at the same time. This leads to a good flexibility and may be to a good (optimal) utilization of the federated resources of any given volunteer grid. Moreover, in such infrastructure any given authorized user, who is at the same time the resource donator, utilizes his computer as he wants and not as the remaining users want, this means that every donator can login or logout the grid at will. Unfortunately, this leads, automatically, to a high volatility of the corresponding resources and, consequently, to frequent faults. Hopefully, fault tolerance is not difficult to reach, the common solution consists of assigning any given task to more than one computer (worker) of the grid; owning the fact that users can have, at any given time, a big number of available computers. Unfortunately, this solution leads, surely, to a very important power loss of these available resources. In this paper we attempt to reduce this power loss, by an assignment of independent tasks within predefined timeslots to the available computers of such grids. In fact, these timeslots can be obtained by a good statistical study of all subscribed computers to any given grid. The principle of our attempt is based on forecasting the availability of every subscribed computer within some specific timeslots. In fact, we suppose that we own an auto training knowledge base where we have exactly significant statistics about the availability timeslots of every computer or resource of the grid and the corresponding computing power; such that we can forecast, at any given predefined timeslot, which computers are available to participate to any given work. Thus, if we own a set of tasks then we will be able to assign them in a good way over the available computers of the grid. Moreover, we suppose next that our volunteer grid is composed of some specific fixed and devoted nodes (computers) which are always active (connected to the grid server, these constitute the kernel of the grid) and a big variable number of volunteer computers each of which can be active (connected and ready to participate in the work at hand) or inactive (disconnected or connected but not ready to participate in the work at hand) depending on its owner. We suppose, also, that our set of tasks is structured as follows:

- It is composed of subsets each of which is stored in a specific fixed node of the grid and can be reached, only, by a part of the grid computers and not by all of them. It means that during the assignment process, tasks of any given subset can be assigned, only, to the reachable and active volunteer computers;

- Any given task of the input set can be executed by any available computer during any timeslot.

Of course, such a problem is NP hard owing the fact that the grid computers are heterogeneous in terms of computing power and tasks are, also, variable in terms of computing power requirement, [5], [6], [13] and [14]. Thus, the idea is to use a good heuristic, for the assignment process of tasks, to reach a pseudo optimal utilization of our volunteer grid resources.

Consequently, we report in this paper results of a comparative study of three heuristics which show and confirm, in particular, that we can reach very interesting and promising utilization of volunteer grid resources without computing power loss.

The remainder of this paper is organized as follows: in the next section, we give a brief overview of grid computing. The third section details the mechanism of the Arabic OCR based on the DTW algorithm which constituted the main motivation of this work. The problem presentation and the

proposed approach are described in the fourth section. The studied heuristics, obtained results and their comparison are detailed in the fifth section. A conclusion and some perspectives of this work are presented at the last section.

2. GRID COMPUTING AND VOLUNTEER GRID

Ian Foster and Carl Kesselman are among the first who worked on grid computing. To identify the problem and referring the definition of a grid electricity, they have proposed in their book "The Grid: Blueprint for a Future Computing Infrastructure" a general definition of the Grid computing by presenting it as: a hardware and software infrastructure providing dependable access, consistent at high penetration rate (pervasive) and cheap at computing power, [8].

A grid computing is a hardware and software infrastructure that can federate (interconnect) over communication networks several heterogeneous resources belonging to different institutions (universities, governmental administrations …), [1], [8], [10], [11], [12] and [15].

A grid computing provides a transparent accessibility to its different interconnected resources. The idea behind the grid computing concept or architecture has several motivations. Among these motivations is making easy and transparent the sharing of resources between institutions. Two other important motivations must be reported here; the first one concerns the federation of the computing power of the interconnected computers to provide enough power to achieve properly:

- Distributed Supercomputing;
- High-Throughput Computing;
- On-Demand Computing ;
-

The second one concerns the federation of the storage power of the interconnected computers to provide enough storage space to achieve properly Data intensive Computing.

- There are only two differences between a grape and a grid computing; the first one concerns the localization of the interconnected computers. In a grape, the interconnected computers and resources belong to the same institution. However, in a grid computing the interconnected computers and resources belong to several institutions. The second one concerns the number of interconnected computers and resources which are generally bigger in grid computing than in grapes.

2.1 Dedicated vs. Volunteer grids

Dedicated grids are commonly composed of a fixed and large number of federated computers which can be used only by a fixed number of authorized users for some very specific purposes, [11]. Such infrastructures require a prior reservation by their authorized users in order to allow for each of them the utilization of the entire corresponding power. Consequently, optimizing the utilization of any dedicated grid computing power is easy to reach [2], [3], [4] and [13]. However, volunteer grids are composed of a big variable number of connected and federated computers belonging to volunteer donators (users, institutions, etc.), [7], [8], [9] , [12] and [15]. They aim at providing very large scale storage and computing power to their users (subscribers) for general purpose usage. Volunteer donators have just to subscribe their computers to be shared through some specific web sites (grid server) without any engagement from their side especially in terms of trust and security. Moreover, any donator (user) can login and logout, at will, without any prior notification to the other users yet connected to the same grid. This leads, automatically, to a very

high volatility of computers (nodes or workers) of the corresponding grid, and consequently, the optimization of the corresponding power utilization is too difficult to ensure.

3. ARABIC OCR BASED ON THE DTW ALGORITHM

The objective of this part is twofold; the first is to explain the mechanism of the Arabic OCR based on the DTW algorithm, and the second is to show, at the same time, the robustness and the complex computing of the DTW procedure.

Words in any cursive writing, such as the Arabic language, are inherently written in blocks of connected characters. While the segmentation of the text into blocks of connected characters is a preliminary phase to the recognition process, a further segmentation of these blocks into separate characters is usually adopted. Indeed, many researchers have considered the segmentation of Arabic words into isolated characters before performing the recognition phase, [10], [11] and [12]. The crux of the viability of the use of DTW technique is its ability and efficiency to perform the recognition without the prior segmentation of blocks into separate characters.

Let V represents a reference library of R trained characters C_r, $r=1, 2\ldots R$ defining the Arabic alphabet in some given fonts. We stress the fact that several fonts could be simultaneously considered. It suffices to get them trained which is easily done at the learning phase while constructing the reference library V. Let T represents a block of connected Arabic characters to be recognized. Then T is composed of a sequence of N feature vectors T_i which represent the concatenation of some sub sequences of feature vectors representing each an unknown character to be recognized. The text T is seen as laying on the time axis (the X-axis) in such a manner that feature vector T_i stands at time i on this axis. The reference library V is portrayed on the Y-axis (see Figure. 1), where the reference character C_r is of length l_r.

Let $S(i, j, r)$ represents the cumulative distance at point (i, j) relative to the reference character C_r. The objective is then to detect simultaneously and dynamically the number of characters composing T and recognizing these characters. There exists a number k of indices $(m_1, m_2,..., m_k)$ such that $C_{m1} \oplus C_{m2} \oplus \ldots \oplus C_{mk}$ correspond to 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), [12] and representing the optimal alignment is therefore of minimum cumulative distance that is:

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

This path is not continuous since it spans many different characters. We therefore must allow at any time the transition from the end of one reference character to the beginning of a new character. The end of reference character C_r is first reached whenever the warping function reaches the point (i, l_r, r) where $i = \left\lceil \frac{l_r+1}{2} \right\rceil, \ldots, N$. 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 the addition of the smallest cumulative distance among the end points found at time $(i-1)$. The resulting functional equations are:

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

with the boundary conditions :

$$S(i,1,\ r) = D(i,1,\ r) + \min_{\substack{1+\left\lceil \frac{lr+1}{2} \right\rceil \le i \le N \\ 1 \le j \le l_r \\ 1 \le r \le R}} S(i-1, l_k, k) \qquad (3)$$

To trace back the warping function and the optimal alignment path, we have to memorize the transition time from one reference character to the others [1], [3], [4]. This can easily be accomplished by the following procedure:

$$b(i,\ j,\ r) = trace \min_{\substack{1 \le i \le N \\ 1 \le j \le l_r \\ 1 \le r \le R}} \left\{ \begin{matrix} b(i-1,\ j,\ r), \\ b(i-1,\ j-1,\ r), \\ b(i-1,\ j-2,\ r) \end{matrix} \right\} \qquad (4)$$

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 in Figure. 1 by means of the two vectors VecA et 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 ⊕ C3.

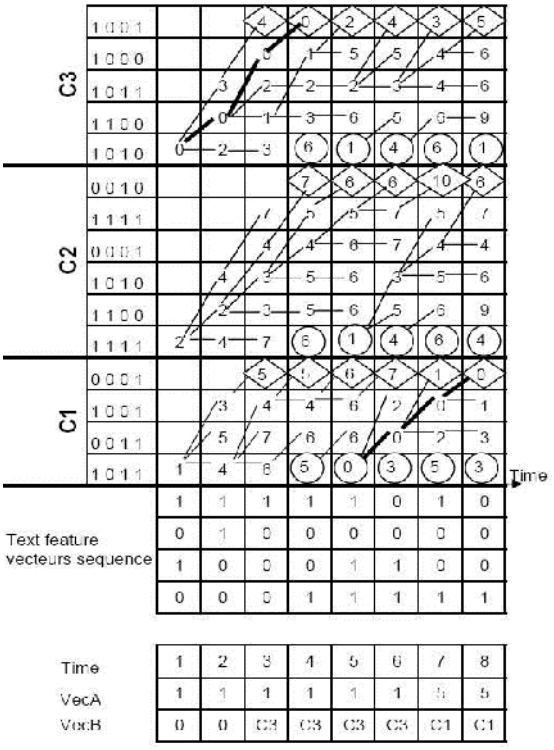

Figure.1 mechanism of the Arabic OCR based on the DTW algorithm

In our previous work [12] we have shown how volunteer grids constitute very interesting and promising infrastructures to speedup, substantially, the Arabic OCR based on the DTW algorithm especially if we want to computerize very big quantities of Arabic documents. In this paper, we attempt to solve the problem of the optimal utilization of such infrastructures owing the fact that this challenging problem has never been studied before to the best of our knowledge.

4. THE PROBLEM PRESENTATION AND THE PROPOSED APPROACH

Recall that a volunteer computing grid consists of several interconnected volunteer computers through the Internet. Every volunteer computer must install a client program that is, usually, defined as a screen saver. So, when the computer is turn on and it is not in use (idle), the corresponding client program connects to the volunteer grid server and downloads a task to execute. Once the task is downloaded, the client program starts its processing. Of course, the execution time of the assigned task depends on the computing power of the host unit; the task processing will last for a period varying from one volunteer to another. After the processing, the client program sends the result to the grid server and gets a new task to execute. And the work cycle continue. The problem is that at any moment the user can interrupt the actual work; in such case, the entire intermediate outcome will be lost and the whole work must be restarted from the beginning. Consequently, we have both, time wasting and computing power loss. So far, the common adopted solution to this problem consists of assigning any given task to more than one worker (computer) of the grid to ensure its complete execution. Thus, if the size of tasks is big then the power loss is important. To solve this problem, we propose to use any given volunteer grid within predefined timeslots where we know, almost exactly, the number of workers which can participate to the work at hand such that each task will be assigned only to one worker. Recall that these predefined timeslots can be obtained by a careful statistical study of the grid to be used. This statistical study must take care about every worker of the grid especially for the following information:

- Its provided computing power;
- Its availability in timeslots (maybe per day);
- Which devoted node(s) can reach it?
- The transmission capacity of the channel connecting every worker to the reachable devoted node.

All these information are supposed to be stored in a knowledge base on the grid server in order to allow the generation of the predefined timeslots which we are looking for.

In fact, the consequent idea consists of assigning to every worker participating in the work the maximum of tasks at each predefined timeslot taking into account its computing power. In the remainder of this paper we consider the Arabic OCR based on the DTW algorithm as a case study.

Tasks in distributed Arabic OCR based on the DTW algorithm are composed of the following parts:

- The code to run (the DTW procedure);
- The input binary image file which contain the image of the Arabic document (text) to be recognized (computerized);
- The reference library (see Part. 3).

Of course, the only part which is specific and variable in each task is the input binary image file; the two others are unchangeable. That is why; we will suppose next that each grid worker (computer) owns these two unchangeable parts and, consequently, the assignment process will be limited to the binary files. Thus, our set of tasks is formed of binary files having variable sizes. Recall, also, that our grid is composed of some devoted nodes (computers), in addition to the server, which are always active (connected to the grid server, these constitute the kernel of the grid) and a big variable number of volunteer computers each of which can be active (connected and ready to participate in the work at hand) or inactive (disconnected or connected but not ready to participate in the work at hand) depending on the will of its owner. As mentioned in the introduction, our set of tasks is structured as follows:

- It is composed of subsets where each of which is stored in a specific devoted node of the grid and can be reached, only, by a part of the grid workers and not by all of them. It means that during the assignment process, tasks of any given subset can be assigned, only, to the reachable active workers. We note that any worker can reach at least one devoted node;

- Any given task of the input set can be executed by any available worker during any timeslot; which means that the tasks granularity should take into account the following items:
    - The workers computing power;
    - The data exchange (transmission) capacity of the grid network;
    - The timeslot duration.

Consequently, our goal is to find a good heuristic which is able to exploit optimally the provided computing power of the grid in order to execute the maximum of tasks during each timeslot owing the fact that our problem is NP-hard.

5. STUDIED HEURISTICS AND OBTAINED RESULTS

We have studied three heuristics in order to reach our expectation which consists of optimizing the utilization of the computing power of any given volunteer grid. For the three studied heuristics and for each timeslot we have used the following steps:

- Sort the available workers (which can participate in the work) according to their computing power in decreasing order in a specific array WA;
- Sort binary image files (tasks to be assigned) of all subsets according to their sizes in decreasing order without considering to their storage node in a specific array TA.

The functioning of each heuristic is detailed next in an independent manner.

5.1 The first heuristic

The principle of this heuristic can be illustrated by the following steps:

Step (FH) Load the most powerful worker (computer) not yet assigned, by browsing the array WA from the beginning, as follows:

1. Try to assign biggest tasks, yet unassigned one by one and on browsing TA from the beginning, to the current worker; this is possible for each of which if and only if the two following conditions are satisfied:

a. This worker is reachable by the given task node;

b. The remaining computing power of this worker, taking into account the data transmission time and the timeslot, allows this assignment.

This process ends when condition a. is satisfied and condition b. is unsatisfied. According to our previous considerations and hypothesis, this worker will be assigned, at least, by one task on browsing the array TA from the beginning. Go to step 2.

2. Try to complete the remaining load of the current worker by assigning to it some smallest task, yet unassigned, on browsing TA from its end in increasing order and one by one((, t)).This is possible, if and only if the two following conditions are satisfied:

   a. This worker is reachable by the given task node;

   b. The remaining computing power of this worker, taking into account the data transmission time and the timeslot, allows this assignment.

   Till the worker saturation (the condition b. is unsatisfied).

3. Go to Step (FH).

5.2 The second heuristic

The principle of the second heuristic can be illustrated through the following steps:

Step (SH) Load the most powerful worker (computer) not yet assigned, by browsing the array WA from the beginning, as follows:

1. Assign one and only biggest task, yet unassigned, by browsing TA from the beginning which is stored on a reachable node.

   Go to step 2.

2. Try to complete the remaining load of the current worker by assigning to it some smallest task yet unassigned on browsing TA from its end, in increasing order and one by one, this is possible, if and only if the two following conditions are satisfied:

   a. This worker is reachable by the given task node;

   b. The remaining computing power of this worker, taking into account the data transmission time and the timeslot, allows this assignment.

   Till the worker saturation (the condition b. is unsatisfied).

3. Go to Step (SH)

5.3 The Third heuristic

The principle of this second heuristic can be illustrated through the following steps:

a. Assign tasks to workers iteratively as follows:

   For a given task, try to find a worker which can handle it without overflow on timeslot otherwise we pass to the next task

Repeat iterations till the last task.

5.4 The studied example

A deep statistical study allows us to generate the following case:

- The number of devoted nodes is 18;
- The number of volunteer workers (computers) is 1830;
- The total provided computing power by these 1830 workers is equal to: 9,83 E+09 Machine instructions during, only, one timeslot. The computing power is generated automatically in such a way that the corresponding computing power is enough to compute at least the current reachable task and the cumulative computing power of all workers is less or equal to 9,83 E+09;
- Tasks are generated automatically, in such a way that the corresponding size can be handled by any unassigned worker. We have done vary the total size of tasks to be assigned from 3,67 E+09 to 1,59 E+10 Machines instructions.

5.5 Obtained results

The numerical results are summarized in Figure.2 and detailed in Figure.3, Figure.4 and Figure.5. Figure.3 and Figure.4 illustrate the percentage of the used computing power versus the total size of tasks to be assigned. We observe from Figure.4 that the heuristic 3 (illustrated by the yellow curve) gives the best results in terms of optimality of grid utilization followed by the heuristic 1 (illustrated by the blue curve) which gives almost the same results. If we refer to the Figure.3 in addition to the Figure.2, we observe that if the total size of tasks is smaller than the total provided computing power, then our three studied heuristics give the same results which correspond to a perfect assignment. In fact, these results can be considered very interesting for several greedy algorithms and applications, especially, if our grid provides enough (a very big) computing power. This, fortunately, corresponds, exactly, to the current trends of volunteer grids.

Figure.5 illustrates the execution time of the three studied heuristics versus the total size of tasks to be assigned. We observe that the heuristic 1, which gave the best results in terms of optimal utilization of the grid, gives the worst results in terms of execution time lustrated by the yellow curve. Moreover, we observe that the corresponding execution time increases, substantially, with the increase of the total size of tasks to be assigned. On the other hand, the remaining heuristics give, almost, the same execution time which can be considered acceptable.

Consequently and in light of these obtained results, we can confirm that the heuristic 1 is the most interesting one in order to exploit and use in a good manner the total provided computing power of any given volunteer.

6. CONCLUSION AND PERSPECTIVES

In this paper, a new approach has been proposed and studied to solve the problem of the optimal utilization of volunteer grid computing power in order to speed up, substantially and for free, several greedy algorithms and applications such as the Arabic OCR based on the DTW algorithm. Our proposal is based on the exploitation and utilization of such infrastructures during some predefined timeslots obtained by a statistical study of the target grid. Moreover, we have studied three heuristics which are able to guarantee a good utilization of such infrastructures through a pseudo optimal assignment of, more specifically, independent tasks relating to the Arabic OCR

based on the DTW algorithm. Obtained numerical results show and confirm that our proposal can constitute a very interesting step towards the optimal utilization of volunteer grids.

Several investigations are under studies, especially, the way to achieve a reliable statistical study of a given volunteer grid such as BOINC (Berkley Open Infrastructure for Network Computing). We also can implement our adopted heuristic on such infrastructure in order to ascertain the viability of our proposal.

Total size of assigned tasks found by heuristic 1	Percentage of used computing power by heuristic 1	Execution time of heuristic 1 (nanoseconds)	Total size of assigned tasks found by heuristic 2	Percentage of used computing power by heuristic 2	Execution time of heuristic 2 (nanoseconds)	Total size of assigned tasks found by heuristic 3	Percentage of used computing power by heuristic 3	Execution time of heuristic 3 (nanoseconds)
3,67E+09	0,374	1287	3,67E+09	0,374	1236	3,67E+09	0,374	1239
4,45E+09	0,453	1615	4,45E+09	0,453	1412	4,45E+09	0,453	1593
5,32E+09	0,542	1879	5,32E+09	0,542	1672	5,32E+09	0,542	2005
6,23E+09	0,634	2205	6,23E+09	0,634	1915	6,23E+09	0,634	2641
7,15E+09	0,728	2516	7,15E+09	0,728	2235	7,15E+09	0,728	3688
7,99E+09	0,814	3574	7,99E+09	0,814	3104	7,99E+09	0,814	5902
8,91E+09	0,907	3893	8,91E+09	0,907	3413	8,91E+09	0,907	8767
9,73E+09	0,99	3537	9,41E+09	0,957	2961	9,72E+09	0,989	39822
9,76E+09	0,993	3297	9,43E+09	0,96	3814	9,82E+09	0,999	579841
9,76E+09	0,993	3240	9,47E+09	0,964	3306	9,82E+09	0,999	875785
9,76E+09	0,993	3199	9,44E+09	0,961	3396	9,82E+09	0,999	1292112
9,76E+09	0,993	3472	9,47E+09	0,964	3425	9,82E+09	0,999	1552184
9,76E+09	0,994	2995	9,48E+09	0,965	3478	9,82E+09	0,999	1742426
9,77E+09	0,994	3057	9,51E+09	0,968	3526	9,82E+09	0,999	1983197
9,76E+09	0,993	3260	9,50E+09	0,967	3962	9,82E+09	0,999	2254855

Total provided computing power	9,83E+09	9,83E+09	9,83E+09	9,83E+09	9,83E+09	9,83E+09	9,83E+09	9,83E+09	9,83E+09	9,83E+09	9,83E+09	9,83E+09	9,83E+09	9,83E+09	9,83E+09
Total size of tasks	3,67E+09	4,45E+09	5,32E+09	6,23E+09	7,15E+09	7,99E+09	8,91E+09	9,73E+09	1,06E+10	1,14E+10	1,23E+10	1,32E+10	1,41E+10	1,50E+10	1,59E+10

Figure.2 Numerical results

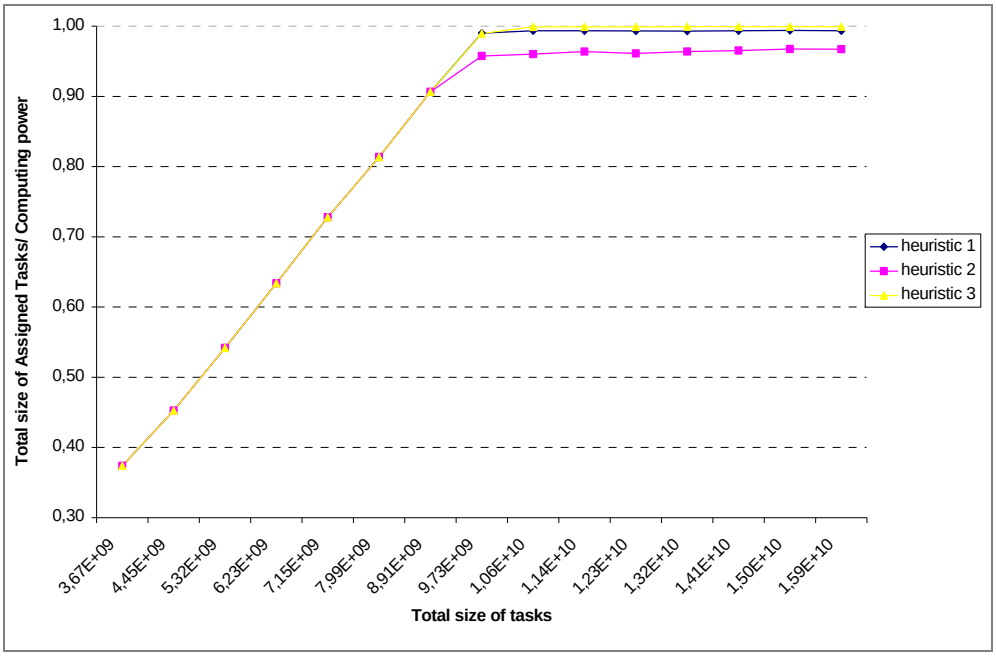

Figure.3 Percentage of used computing power

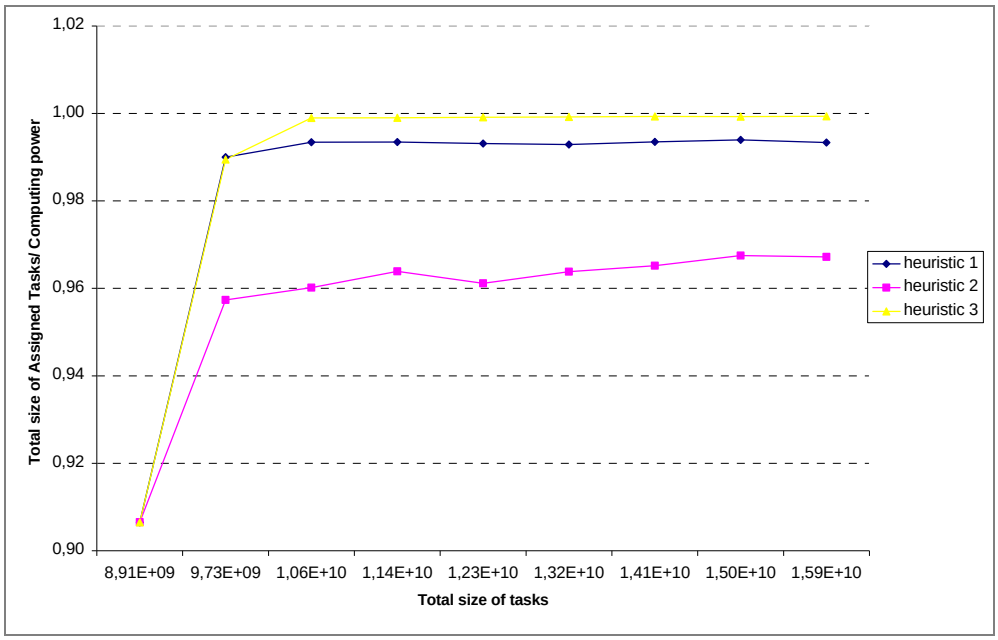

Figure.4 Zoom on the last part of Figure.2 (Percentage of used computing power)

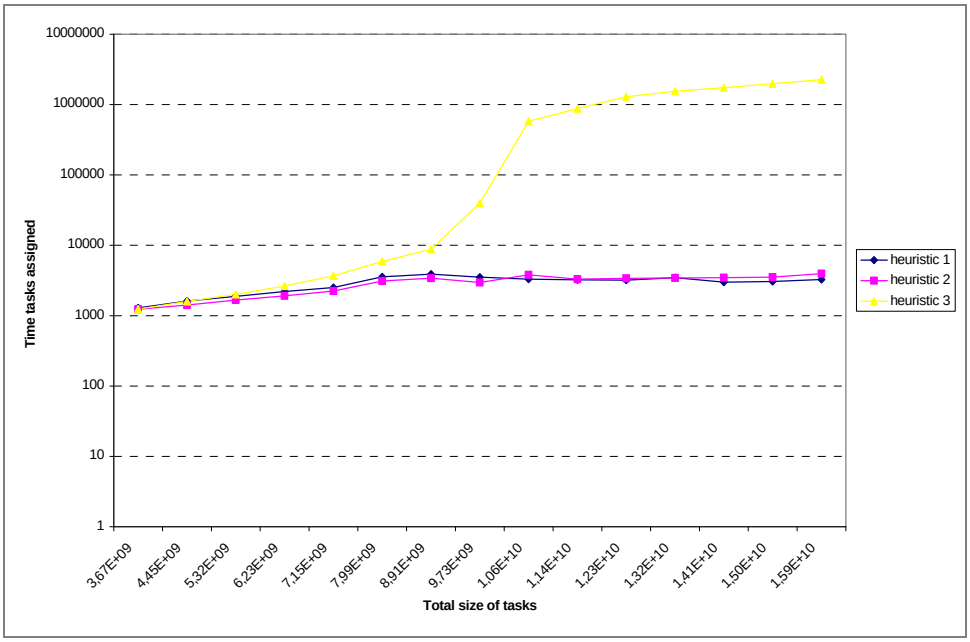

Figure.5 Execution time of the studied Heuristics

REFERENCES

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

[2] P. Brucker, A. Drexel, R. H. Möhring, K. Neumann, and E. Pesch, "Resource-constraint project scheduling: Notation, classification, models, and methods", European Journal of Operation Research, Vol. 112, N°. 1, pp. 3–41, Jan. 1999.

[3] J. Blythe et al., "Task Scheduling Strategies for Workflow-based Applications in Grids", In the IEEE International Symposium on Cluster Computing and Grid (CCGrid'05), Vol. 2, 2005

[4] T. L. Casavant and J. G. Kuhl, "A taxonomy of scheduling in general-purpose distributed computing systems", IEEE Transactions on Software Engineering, Vol. 14, N°2, pp. 141–154, 1988.

[5] D. Hochbaum, "Approximation, Algorithms for NP-hard Problems", PWS Publishing Company, Boston, 1997.

[6] E. G. TALBI, "A Taxonomy of Hybrid Meta heuristics", Journal of Heuristics, Vol. 8, N° 5, pp. 541–564, 2002.

[7] Available at: Einstein@Home, http://einstein.phys.uwm.edu/

[8] I. Foster and C. Kesselman, "The Grid: Blueprint for a Future Computing Infrastructure", Morgan Kaufmann Publishing Company, San Francisco, 1999.

[9] Available at: LHC@home, http://athome.web.cern.ch/athome/

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

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

[12] M. Khemakhem and A. Belghith, "A P2p Grid Architecture For Distributed Arabic OCR Based On The DTW Algorithm", The International Journal of Computers and Applications (ACTA PRESS, IJCA), Vol. 31, N° 1, 2009.

[13] M. Maheswaran, S. Ali, H. J. Siegel, D. Hensgen, and R. F. Freund, "Dynamic mapping of a class of independent tasks onto heterogeneous computing systems", Journal of Parallel and Distributed Computing, Vol. 59, N° 2, pp. 107-121, Nov. 1999.

[14] JD Ullman, "NP-complete scheduling problems", Journal of Computer and System Sciences, Vol. 10, pp. 384-393, 1975.

[15] M. Khemakhem, A. Belghith: « Towards trusted volunteer grid environments », the International Journal of Communication Networks and Communications (IJCNC, AIRCC) Vol. 2, N°.2, March 2010.

[16] N. Muthuvelu et al, A Dynamic Job Grouping-Based Scheduling for Deploying Applications with Fine-Grained Tasks on Global Grids, *Proceedings of the 3rd Australasian Workshop on Grid Computing and e-Research (AusGrid 2005)*, Newcastle, Australia, January 30 – February 4, 2005.

[17] F. Dong and Selim G. Akl ,Scheduling Algorithms for Grid Computing: State of the Art and Open Problems, *School of Computing Queen's University* Kingston, Ontario Canada ,January 2006.

[18] M. Wieczoreket al, Taxonomies of the Multi-criteria Grid Workflow Scheduling Problem, CoreGRID *Technical Report* Number TR-0106, Aug. 30, 2007.

BIOGRAPHY:

Maher Khemakhem received his Master of Science, his Ph.D. and Habilitation accreditation degrees from the University of Paris 11 (Orsay), France respectively in 1984, 1987 and the University of Sfax, Tunisia in 2008. His research interests include distributed systems, performance analysis, grid and cloud computing, Networks security and pattern recognition.
E-mail address: maher.khemakhem@fsegs.rnu.tn.

Abdelaziz Dammak, received his Bachelor degree in applied fundamental mathematics, Master degree in Operations Research from Scientific & Medical University of Grenoble (France), PhD in applied mathematics from University of Paris-Sud, Orsay (France) and HDR in Quantitative Methods field at the Faculty of Economics and Management Sciences in Sfax. He teaches Operations Research at the University of Sfax. His research interests are in combinatorial optimization problems: graph theory, mathematical programming and development of heuristics. His research focus on university timetabling applied to Tunisian universities. He is author of different articles published in different international journals and the Proceedings of PATAT. E-mail address: abdelaziz.dammak@fsegs.rnu.tn.

Taha Chaabouni received his Master degree from the University of Sfax, Tunisia in 2010. He is currently preparing a Ph.D. at the University of Sfax, Tunisia. His research interests include cloud computing, distributed systems and performance analysis.
E-mail address: chaabounitaha@yahoo.fr