The Journal of
Mathematics and Computer Science

Available online at
http://www.TJMCS.com

The Journal of Mathematics and Computer Science Vol .1 No.4 (2010) 305-312

Gravitational Attraction Search with Virtual Mass (GASVM) to solve Static Grid Job scheduling Problem

Amin Jula[1,*], Narjes Khatoon Naseri[2], Amir Masood Rahmani[3]
Islamic Azad University, Mahshahr Branch, amin.jula@gmail.com
Islamic Azad University, Shoushtar Branch, narges.naseri85@gmail.com
Islamic Azad University, Tehran Science and Research Branch, rahmani74@yahoo.com

Received: September 2010, Revised: November 2010
Online Publication: December 2010

Abstract

Achieving the most grid computing efficiency requires optimized job scheduling, that is a problem with vast search space and Attaining optimal solutions using deterministic algorithm is extremely difficult or impossible. Besides, Falling in the trap of local minima is considered to be one of the problems existing in gravitational attraction search. GASVM proposed two modifications. First, defining virtual mass (VM) for K best solutions. For each solution, VM is defined depends on mass and ranking in the sorted list of solutions. VMs will increase gravitational mass of proper solutions and attract others to them.

Second, we calculate gravitational force of just K proper solutions on the others to prevent current good solutions, more searching about, and attracting other solutions in the direction of them. In each modification, we obtain K by using roulette wheel algorithm. Analyzing the results of GASVM executions shows that the proposed

---

[1,*] Professor and Head of Hardware and IT Department
[2] Professor
[3] Professor and Head of Computer and Mechatronics Department

algorithm is able to achieve its intended aims to modify gravitational attraction search algorithm.

Keywords: gravitational attraction search algorithm, static job scheduling in grid, Newton's gravitational law.

1. Introduction

Grid Computing was suggested as substituted for usual super computers in the last decade 1990 to solve special problems which required mass numerical calculations and get access to a higher volume of the distributed data. The main purpose was that the fast enough Networks that are geographically dispersed, using suitable software, share their processing resources and data resources management into a single system to face with problems which each of these networks alone are not able to deal with them [1].

Today increase in the efficiency of grid is a considerable problem which requires scheduling efficiency and accurately. Jobs enter to grid are different and in proper workload. Deal with these different conditions needs different strategies and scheduling algorithms such as FPLTF, WQR, RR, Max-min, min-Max, FCFS [2]. Grid job scheduling problem focuses on response time. The purpose of job scheduling is optimal assigning jobs to resources. Nowadays, due to the increased dimensions of problems and the importance of speed to get to the proper responses, classical methods are not advanced enough to solve the problems and the random algorithms are mostly use to solve problems. Due to these circumstances heuristic search algorithms have gained noticeable growth. Heuristic algorithms are those which have come through inspiration of physical and biological processes in nature and most of them act as group population. Genetic Algorithm is an example of this type which has been inspired by the science of Inheritance and evolution 1975. Simulated annealing algorithm has been inspired by thermo dynamics 1983, immunity algorithm with defensive human system simulation 1986, algorithm for searching ant colony by making use of ant behavior to search for food 1991, and particle swarm optimization through imitating social behavior of birds 1995 are just a few examples to be mentioned [3-7].

In the same direction gravitational attraction search algorithm was offered by Webster through the inspiration of Mass and energy of gravitation force based on simulation of existing laws in nature [8]. The issue will be deal with in part three in details.

In this article static grid job scheduling will be studied to optimize by improved gravitational attractions search algorithm. The structure of the article will be continued like that in part two grid and its job scheduling will be reviewed. Gravitational Attraction Search Algorithm has been introduced in part three, and in part four the suggested algorithm will be offered, and finally in part five the results of suggested algorithm will be described and part six constitutes general conclusions.

2. Grid Job Scheduling

Cooperative system for distributed geographical resources to solve problems has been called grid computing. One of the significant features of grid computing is the absence a central unit to control the operations regarding input and output in computer at each hour. This issue has made proper use of the network difficult. There are a variety of units with different functions operating in grid network and scheduler is one of these units. Scheduling is considered important because the systems in real world are distributed physically or operatively in a heterogeneous way.

Systems like controlling air traffic, astrology, medicine, biology, military force, and mobile belong to this category. Additionally scheduling is one of the principle subjects to get access to the maximum amount of efficiency belonging to grid computing [11].
Grid computing receives jobs from users and selects convenient resources for them according to information obtained from data service and creates a mapping job to resource based on objective functions and resources performance predictions. The problem consists of a number of independent jobs which must be scheduled; each job must entirely process completely in a unique resource, a number of machines which are candidate to use in scheduling, the workload of jobs, computing capacity of each machine, and the expected response time for computing.
In 1995 Voudouris and Tsang [10] suggested GLS algorithm for the first time to search in search space and solving the NP-hand problems. In 2004 Webster introduced it as a strong algorithm and called it GELS. This algorithm is based on gravitational attraction which causes object to attract one another in nature in such a way that the heavier object possesses higher gravitation force and the lighter objects are pulled toward the heavier objects. Certainly, the distance between two objects has an effect in attraction and between two objects with the same amount of weight and different distance from another object, object with a longer distance exert less amount of attraction. Gravitational Attraction Algorithm GAS makes use of the above natural laws for offering intelligent method [9].

3. GAS Algorithm

3.1 Main Idea

The search space of a problem is assumed as a multidimensional system with different solutions to the problem. Each point in space is a unique solution to solve the problem, and each solution possesses a "mass" through which the objective function will be computed. Any solution is better, more objective function value is generated, and thereby its mass will be more. Besides, search agents are collection of particles. After constituting search space, its rules will be recognized to govern it, with the assumption that only gravitation and motion laws govern it.
Gravitation law: Each particle in search space will attract other particles toward itself. The amount of this force is proportional to the gravitational property of the particle and inversely to the distance between the two particles.
Laws of Motion: the real velocity of every particle is equal to the sum of a coefficient of the previous velocity of the particle and the change of its velocity. Change in velocity, acceleration, is equal to the imposed force on the particle divided to its gravitation mass.

3.2 Gravitational Attraction Force

The search space is considered as a set of m particles. Position of each particle in search space is a point in space and is taken in account as a solution to the problem. This position is obtained from equation 1 in which position of particle i in dimension d is shown by x_d^i.

$$X_i = (x_i^1, ..., x_i^d, ..., x_i^D) \qquad (1)$$

In this system in time t, a force is imposed from particle I to each particle j in the direction of dimension d as much as $F_{ij}^d(t)$. The amount of this force is computed from equation 2 in which $G(t)$ is gravitation constant in time t, $R_{ij}(t)$ is distance between two particles i and j in that time and ε is a small number. To obtain distance between the particles Euclidean distance has been used, equation 3.

$$F_{ij}^{d}(t) = \frac{G(t) \times M_i(t) \times M_j(t)}{R_{ij}(t) + \varepsilon}(x_j^d(t) - x_i^d(t)) \tag{2}$$

$$R_{ij}(t) = \left\| X_i(t), X_j(t) \right\|_2 \tag{3}$$

The imposed force on the particle i in direction of dimension d in time t, $F_i^d(t)$, is equal to the total amount of the forces of other particles of the system imposed to it, equation 4.

$$F_i^d(t) = \sum_{j=1, j \neq i}^{m} r_j F_{ij}^d(t) \tag{4}$$

According to second newton's law, each particle gains acceleration in direction of dimension d, and this acceleration is proportional to the imposed force on the particle in the same dimension divided by the particle gravitational mass. The particle acceleration in direction of dimension d in time t is shown by $a_i^d(t)$ and it obtains from equation 5.

$$a_i^d(t) = \frac{F_i^d(t)}{M_i(t)} \tag{5}$$

Velocity of each particle is equal to the sum of a coefficient of the present velocity of the particle and acceleration of the particle that is obtained from equation 6. The new position of the particle i in the dimension d is equal to the sum of its present position and its velocity that computes by equation 7.

$$V_i^d(t+1) = r_i \times V_i^d(t) + a_i^d(t) \tag{6}$$

$$x_i^d(t+1) = x_i^d(t) + V_i^d(t+1) \tag{7}$$

r_i and r_j are the random numbers by uniform distribution in the interval of (0,1) which have been used to keep the random property of search. Equation 8 is used in order to adjust the gravitational constant. Gravitation constant tends to decline exponentially.

$$G(t) = \beta^{-\alpha \frac{1}{T}} \tag{8}$$

4. GASVM Algorithm

Initially, each job is allocated to a processor at random in such a way that each processor will have one or more jobs for execution. At this time one solution produced to solve the problem, and in the same way SolNo different solutions will be generated at random to solve the problem. Then mass of each particle will be computed and recognized mass of the related particle. If we call T_i the required time to execute all the appropriate jobs to processor i, mass of solution will be the largest T_i. The procedure of computing mass of particle is reflected in figure 1.

```
FinalMass = 0;
for (int pno = 0; pno < ProcessorNo; pno++)
{
    SumOfTimes = 0;
    for (int j = 0; j < TaskNo; j++)
        if (Sols[i].Pno[j] == pno)
            SumOfTimes += ( TaskTime[Sols[i].Tno[j]] );
    if ( SumOfTimes > FinalMass )
        FinalMass = TotalSum;
}
Sols[i].Mass = 1000 - FinalMass;
```

Figure 1. Calculating mass of solutions Pseudo code

Finally, the solutions sort descending based on mass and the solution with the largest mass as in the equation 9 will be recognized as the best. Since equation "a" has been known as the best, with the minimum amount of time for the biggest jobs in gravitational method, in must be related to the biggest particle.

$$best(t) = \max_{j\in\{1,\dots,m\}} fit_j(t) \tag{9}$$

As the best solution, a solution with minimal time to execute jobs and in gravitational attraction search the best solution is solution with largest mass, we can subtract the calculated number for mass from a fixed number and thus it will be a solution with minimum time for execution has maximum amount of mass.

Focused on gravitation search algorithm we realize that, due to being too improper solutions than suitable solutions, sum of gravitational masses of improper solutions exceeds the sum of gravitational masses of proper solutions, and consequently improper solutions attract proper solutions and proper solutions will be lost. We suggest that for a limited number of ideal solutions explain a field named "Virtual Mass". The amount of virtual mass of each solution will be defined as the basis of its quality compared with other solutions and its rank in the ordered set of solutions. The calculating method for virtual mass is reflected in figure 2.

```
Function CalculateVirtualMass ( )
Begin
For k=0 to NumK acquired by Roulette wheel
        Solution[ k ].VirtualMass = solution[k].RealMass * (α + ( 1- k / NumK ) )
End
```

Figure 2. Virtual mass pseudo code

Where α is a real random number in range (0, 1), k is rank of solution in the sorted set, NumK is number of solutions that virtual mass should be calculated for them. This increases the

gravitational mass of solutions and as a result, improper solutions will be absorbed by them. However, determining the number of solutions involved in this is very important. To dynamic determining the number of solutions that should have virtual mass Roulette Wheel algorithm has been used.
This algorithm generates a random number between 1 and sum of masses of all solutions. Then, to achieve this random number, algorithm attempts to calculate sum of masses of solutions starting from the best solution. Achieving the desired number in calculating the total condition will finish this process.

5. GASVM Algorithm execution results

In this part we will deal with the execution results of GASVM which has been suggested and the classic GAS for some sample experimental problems which are executed on personal computer under identical conditions.
The procedure was that the GASVM Algorithm executed five times to solve the same problem and their average calculated. Then the same problem solved through classic GAS and their average calculated. Afterwards From the average of results prepared charts for comparing performances.
The first problem is a static job scheduling with twenty processors and forty jobs. The problem can be assumed as a simple scheduling problem, but the results obtained in their comparison of the two algorithms are very effective and noticeable. All used runtimes in this problem are created at random by computer and in each execution more than 200 iterations have been operated and the results have been recorded. The obtained results reflected in figure 4 show that GASVM algorithm has been quickly obtained the proper solutions after 38 iterations and solutions are converged on it, while the classic GAS tries to reach more suitable solutions through gradual slope in the curve**.**
The other problem which is used for the comparison of GASVM algorithm and classic GAS has been solved is a static grid job scheduling problem with fifteen processors and sixty jobs. All used runtimes in this problem are created at random by computer like the previous problem. Each execution is stopped after 200 iterations and the results have been recorded. The difference between the former and later is increase number of jobs and reduction of processors which brings about a more difficult problem.
The results are reflected in figure 5 as a chart. Studying the chart shows that the classic GAS after a few iterations gets involved in the trap of local optima and obviously local optimization prevents us from finding real solutions, while the GASVM algorithm continues finding better solutions steadily after execution 200 iterations with an acceptable slope and doesn't fall in the trap of local optima.

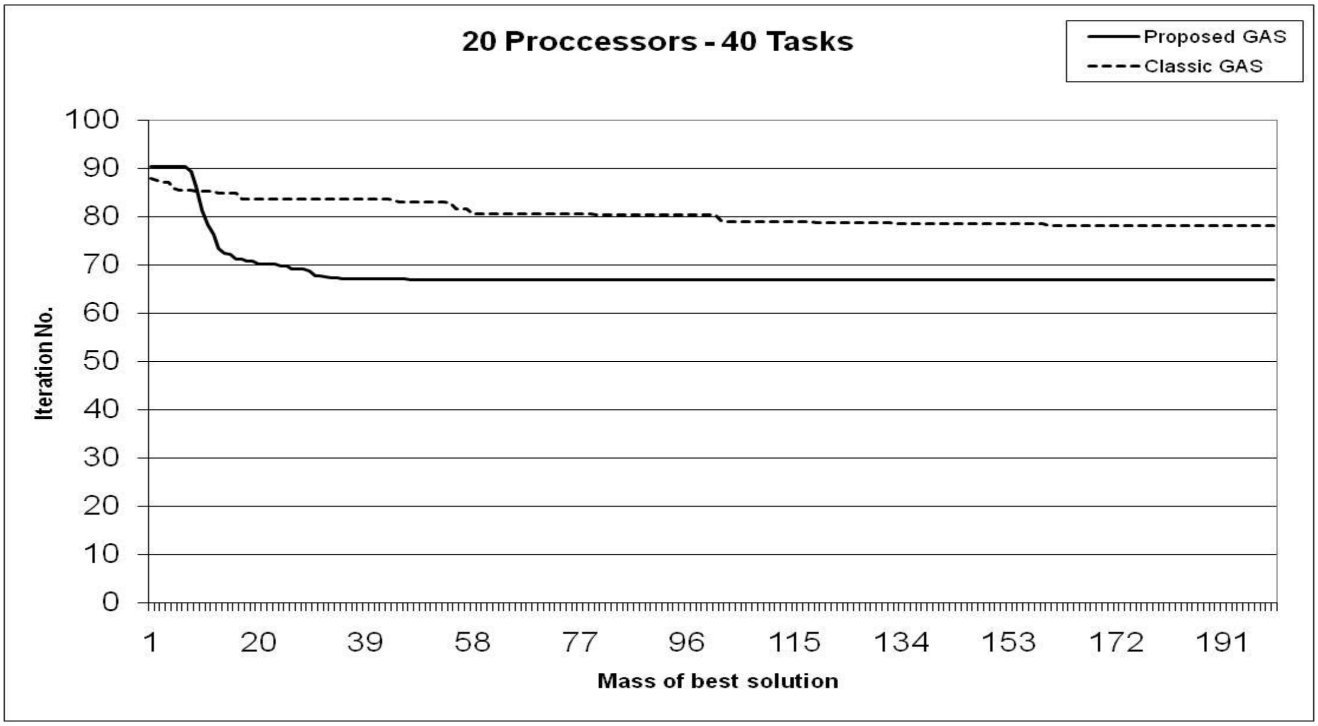

Figure 3. Comparison of two algorithms running on the problem with 20 processors and 40 jobs

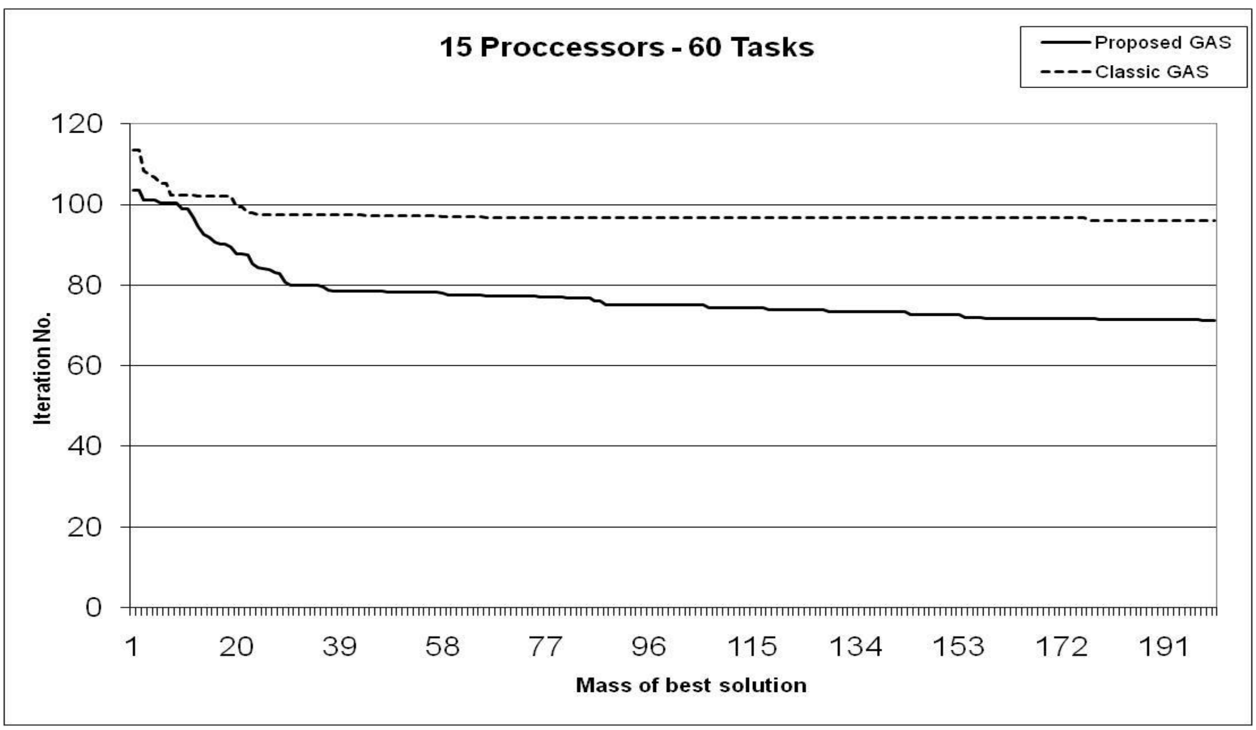

Figure 4. Comparison of two algorithms running on the problem with 15 processors and 60 jobs

7. Conclusion

Getting caught in the trap at local optimum is one of the problems all the heuristic algorithms face to solve optimization problems. Classic gravitational attraction search is facing this problem too. This algorithm also does not find the best solution at the proper time. In this paper, a solution is presented for these two problems. The results of solving different samples of static grid job scheduling problem show that the GASVM algorithm has been able to reduce the required time to solve the problem considerably. Additionally, canalizing and directing the move of particles in search space and intelligent delaying the convergence of solutions guarantee achieving optimal solutions.

Acknowledgment

This paper reflects the results of a research project with the issue in accordance with this paper sponsored by Islamic Azad University Mahshahr branch. Hence it is necessary to thank and appreciate this financial and moral supports.

References

[1] I. Foster and C. Kesselman (editors), *The Grid: Blueprint for a Future Computing Infrastructure, Morgan Kaufmann Publishers, USA, 1999.*

[2] Ruay-Shiung Chang_, Jih-Sheng Chang, Po-Sheng Lin , *An ant algorithm for balanced job scheduling in grids,ELSEVIER Future Generation Computer Systems 25 (2009) 20–27*.

[3] K.S .Tang, K.F.Man, S.Kwong and Q.He, " Genetic algorithms and their applications", *IEEE Signal Processing Magazine 13 (6), 1996.*

[4] S.Kirkpatrick, , C.D.Gelatto and M.P.Vecchi, "Optimization by simulated annealing", *Science 220 (4598), 671–680.1983.*

[5] J.D. Farmer, N.H.Packard and ,A.S.Perelson," The immune system,adaptation, and machine learning*", Physica D 22, 187–204. 1986.*

[6] M.Dorigo, V.Maniezzo, and A.Colorni,"The Ant System: optimization by a colony of cooperating agents.", *IEEE Transaction on systems, Man, and Cybernetics-part B, vol. 26, no.1, 1996, pp. 1-13.*

[7]. J.Kennedy and R.C.Eberhart, “Particle swarm optimization”, *Proceedings of IEEE International Conference on NeuralNetworks, vol. 4, 1995.*

[8] Barry Webster, Philip J. Bernhard, “a local search optimization algorithm based on natural principles of gravitation”, *Florida Institute of Technology Technical Report #cs-2003-10.*

[9] Javier Carretero, Fatos Xhafa, “USE OF GENETIC ALGORITHMS FOR SCHEDULING JOBS IN LARGE SCALE GRID APPLICATIONS”, *TECHNOLOGICAL AND ECONOMIC DEVELOPMENT OF ECONOMY . 2006, Vol XII, No 1, 11-17*

[10] Voudouris, chris, Edward Tsang, Guided Local Search. Technical Report CSM-247*, Department of Computer Science, University of Essex, UK, August 1995.*

[11] H.Saadi, G.Habibi, H.Mohammadi, "Designing a scheduler for grid computing using genetic algorithm", *Proceeding of 13th conference of Iranian Computer Association, March 2007*