JACET Journal of Advances in Computer Engineering and Technology, 3(1) 2017

Dynamic Replication based on Firefly Algorithm in data Grid

Mehdi Sadeghzadeh[1]

Received (2016-06-19)
Accepted (2017-01-16)

Abstract - **In data grid, using reservation is accepted to provide scheduling and service quality. Users need to have an access to the stored data in geographical environment, which can be solved by using replication, and an action taken to reach certainty. As a result, users are directed toward the nearest version to access information. The most important point is to know in which sites and distributed system the produced versions are located. By selecting a suitable place for versions, the versions having performance, efficiency and lower access time are used. In this study, an efficient method is presented to select the best place for those versions created in data grid by using the users' firefly algorithm which is compared with two algorithms. Results show that firefly algorithm has better performance than others.**

Index Terms - **data grid, data replication, firefly algorithm.**

1- Department of Computer Engineering, Mahshar Branch, Islamic Azad University, Mahshahr, IRAN.(sadegh_1999@yahoo.com)

I. INTRODUCTION

GRID technology which its first steps taken from 1996 for the next generation Internet, is the result of cooperative efforts of America state universities, scientific centers as well as individual companies. This phenomenon with the best and fast search possibilities, and very high speed and strong possibilities for all types of scientific research is considered as a competent alternative for the internet. Internet network provides the processed information to the people but grid gives raw information, computing power, sensors and laboratory systems and it converts internet from a static environment to a programmable and dynamic environment. On the other hand, grid technology, in contrast to the web which is a service for exchanging information in internet, is a software service for sharing computing power and data storage space between computers connected to the internet and its final target is creating a global broad computing and information.

Grid computing, for the first time, was presented about the distributed resource and solved complicated problems in dynamic environment for scientific problems. [1] Grid is a heterogeneous distributed environment. Grid computing is a software and hardware structure providing cheap comprehensive and stable availability to the existing computing resources in the network based on the descriptions presented later by different people for grid. Resource subscription problem and problem solving are considered in grid. The most important problem proposed here is resource subscription. It is worth mentioning that resource subscription does not mean direct file exchange but the possibility of easy availability to the network computers and using computing power and other feasibilities by which the intended subscriptions are considered. [2]

In this study, the data grid is investigated and a new method is proposed for file management and placing version for the purpose of increasing efficiency and confidence in data grid.

II. LITERATURE REVIEW

As far as Grid is concerned, in addition to users, jobs also need different data accessibility. Nay work could be the applicant of several files. Certainly, if the requested work was in local place, i.e. in the site, the response time could be zero. Otherwise, the requested file must be transferred from the site to the requested place. In [3] three algorithms introduced for the problem of data placement:

1) Greedy Algorithm: In this method, a version selected until the optimal solution is found. In the first step, each site is evaluated individually and selected by the less cost (TC_1). Then, the second version is selected by comparing to the first version and finally (TC_2) it becomes minimal. The process is continued until j was found in such a way that ($TC_{j-1}<TC_j<TC_{j+1}$). In this case, the final cost equals to TC_j and j is the number of versions.

2) Centralized Area Algorithm. In comparison to the previous method, this algorithm is different in selecting the best candidate of any step by the greedy method. In this method, the existing traffic of each site is calculated in its site neighborhood. Then, the selected sites are determined by the most numbers of requests and its final cost is calculated (TC_1). Then, it is operated as greedy algorithm.

3) Output Density Maximum Algorithm: In this method, the sites are selected by maximum density and in any stage, the final cost is calculated. This operation is repeated until a special condition is reached.

In [4] without considering the storage resource cost of different networks such as (loop, tree, etc) optimal algorithm and efficiency are presented for data placement and in [5] the dynamic method is presented based on availability weight of any version. Chang prioritized any version according to the weight and the files used more previously, are applied more now. There are two types of propagation in Grid including static and dynamic type.

Foster and Ranganathan [6] proposed six distinct replica strategies : (No Replica, Best Client, Cascading Replication, Plain Caching, Caching plus Cascading Replica and Fast Spread) for multi-tier Data Grid. The results of simulations indicate that different access pattern needs different replica strategies. Cascading and Fast Spread performed the best in the simulations. Also, the authors combined different scheduling and replication strategies.

Rahman et al. [7] proposed an algorithm for replica selection by using a simple techniques called K-Nearest Neighbor (KNN). The KNN rule choose the best replica for a file by using previous file transfer logs. They also suggested a predictive way to estimate the transfer time between sites and decreased the prediction error as reported by using Neural Network techniques. Accordingly, one site can request the replica from a site which has minimum transfer time.

In [8] the authors presented a data replication strategy that has a provable theoretical performance guarantee and can be implemented in a distributed and practical manner. They also proposed a distributed caching strategy, which can be easily adopted in a distributed system such as Data Grids. The key point of their distributed strategy is that when there are several replicas, each Grid site keeps track of its closest replica site. This can dramatically enhance Data Grid performance because transferring large-sized files is time and bandwidth consuming [9]. The results of simulation demonstrated that the distributed replication algorithm significantly outperforms a popular existing replication strategy under various network parameters.

Tang et al. [10] presented Simple Bottom-Up (SBU) and Aggregate Bottom-Up (ABU) strategies to improve the average data access response time for a multi-tier data grid. The main idea of the two strategies is to store a replica to nodes close to its requesting clients when the file's access rate is higher than a pre-defined threshold. SBU uses the file access history for each node, but ABU aggregates the file access history for a system. With ABU. A node transmits aggregated historical access records to its top tiers, and the top tiers do the same until these records reach the root. The results show that ABU improves job response time and bandwidth consumption better than those of SBU because its aggregation capability.

Andronikou et al. [11] proposed a set of interoperable new data replication strategies that take into account the infrastructural constraints as well as the 'importance' of the data. The presented system is scalable and the strategies can be easily implemented on a Grid environment to provide fast execution. The proposed QOS-aware dynamic replication strategy determines the number of replicas required based on data request, content importance and requested QOS. It also places of the new replicas within the Grid environment according to the network bandwidth and the overhead that the replication technique presents. It can handle the dynamicity of the Grid system by increasing or decreasing the set of data replicas based on the number and the geography of the data demands.

Lee et al. [12] presented an adaptive data replication strategy for a star-topology Data Grid, called the Popular File Replicate First algorithm (PFRF). It periodically computes file access popularity to track the changes of users' access behaviours, and then replicates popular files to suitable clusters/sites to adapt to the variation. They considered several types of files access behaviors, including Zipf-like, geometric, and uniform distributions, to evaluate PFRF. The simulation results demonstrate that PFRF can reduce average job turnaround time and bandwidth consumption.

Saadat et al. [13] presented a new dynamic data replication strategy which is called Pre-fetching based Dynamic Data Replication Algorithm in Data Grids (PDDRA). PDDRA predicts future requires of Grid sites and pre-replicates them before needs are requested. This prediction is done based on the past file access history of the Grid sites. So when a Grid site requests a set of files, it will get them locally. The simulation results show that this strategy improves in terms of job execution time, effective network usage, number of replications, hit ratio and percentage of storage filled.

Taheri et al. [14] proposed a new Bee Colony based optimization strategy, called Job Data Scheduling using Bee Colony (IDS-BC). IDS-BC has two collaborating operations to efficiently schedule jobs onto computational elements and replicate data sets on storage elements in a system so that the two independent, and in many cases conflicting, objectives (i.e. make-span and transfer time of all data files) of such heterogeneous systems are concurrently decreased. Three tailor-made test Grids varying from small to large are applied to evaluate the performance of JDS-BC and compare it with other strategies. Results showed that JDS-BC's superiority under different operating scenarios. JDS-BC also presented a balanced decision making behavior, where it occasionally relaxes one of its objectives (e.g. transfer time) to obtain more from optimizing the other one (e.g. , make span).

Mansouri and Dastghaibyfard [15] presented a Dynamic Hierarchical Replication (DHR) strategy that store replica in suitable sites where the particular file has been accessed most, instead of storing file in many sites. It also decreases access latency by selecting the best replica when different sites hold replicas. The proposed replica selection strategy chooses the best replica location for the users' running jobs by considering the replica requests that waiting in the storage and data transfer time. The simulation results show, it has less job execution time in comparison with other strategies especially when the Grid sites have comparatively small storage size.

Mansouri and Dastghaibyfard [16] presented a Modified Dynamic Hierarchical Replication (MDHR) replaces replicas based on the last time the replica was requested, number of access, and size of replica. MDHR selects the best replica location from among the many replicas based on response time that can be determined by considering the data transfer time, the storage access latency, the replica requests that waiting in the storage queue, the distance between nodes and CPU process capability. Simulation results utilizing the OptorSim show MDHR achieves better performance overall than other strategies in terms of job execution time, effective network usage and storage usage.

The purpose of static propagation is load balance optimization and certain ability which is propagated in different places during data initiation and does not change by load amount alteration. Grid environment is a dynamic one and static method is not optimal in this environment while it is used due to its algorithm facility.

In dynamic method is more efficient than static method, as the existing resources are considered as a current mode while it has not been used due to the complexity of foundation. In this case, the decision made for data propagation operations is based on factors such as file size, network delay, system certain liability, network band

and network delays and in [6] a combination of greedy and genetic methods have been used.

III. FIRFLY ALGORITHM

Firefly Algorithm is a type of algorithm obtained from nature and collective smart algorithm which is presented by yang (2008). This algorithm is a modern technique based on collective behaviors inspired from firefly collective intelligence, which is a type of artificial intelligence method for the social behaviors in the nature based on collective behaviors in neutralized and self-organized foundations. Fireflies generate rhythmic and short beams. Optical patterns of each firefly are differentiated from other patterns. Fireflies use these beams for two main reasons including pair attraction process and attracting hunt. Moreover, these beams are used as a protection mechanism for fireflies. Rhythmic beams and rate of radiation and interval rate between light signals lead to the absorption of two genders. Any particle of a firefly in multidimensional search space is updated by absorbing dynamically based on the knowledge of firefly and its neighbors.

Firefly optimization algorithm could be stated as follows: [17]

- All fireflies are single- gender and the factor of pairs' attractiveness is not related to their gender.
- Each firefly attracts all fireflies and is attracted by all fireflies.
- Attractiveness is related to their glow. Therefore, for any pair of firefly, a worm with less light attracted toward a worm with more light. Attractiveness power is related to their beam and the light intensity is decreased by increasing the distance between two fireflies. If a firefly is not brighter than others, their movement performed randomly.
- Brighter firefly moves randomly (all fireflies could not attract them).
- Firefly brightness is determined by objective function value. The problem of light intensity could be determined easily by target function.
- Firefly particles are randomly distributed in search space based on the above principles, and two main parts exist in firefly algorithm, attracting firefly and movement toward the attracted firefly.

General form of firefly algorithm is shown in the Figure 1.

As it is evident from the figure, at first, primary coordination, light intensity rate and the distance between fireflies are determined in the search area. The search procedure in firefly algorithm is as follows: any firefly compared with others individually. If any firefly has less light than the compared one, it moves toward a firefly with more light (the problem of finding maximum point) and this process leads to the centralization of the particles around a particle with more light. Also, if there is a particle with more light in the next generation of algorithm, the particles move toward other particles again with more light. The search stages must be generated related to the maximum number of generation.

```
Firefly algorithm
Initialize algorithm parameters:
MaxGen: the maximum number of generations
Objective function of f () ,where $X = (x_1, x_2, \dots, x_d)^T$
Generate initial population of fireflies or
$X_i, i = 1,2,\dots,n$
Define light intensity of $I_i$ at $X_i$ via $f(X_i)$
While (t<MaxGen)
For i=1 to n (all n fireflies);
For j=1 to n (all n fireflies)
If ($I_j > I_i$), move firefly i towards j; end if
Attractiveness varies with distance r via Exp[$\gamma r^2$]
Evaluate new solutions and update light intensity;
End for j;
End for i;
Rank the fireflies and find the current best;
End while;
Post process results and visualization;
End procedure.
```

Fig. 1. Firefly algorithm

IV. PROPOSED ALGORITHM

Suppose that Grid environment is taken place with the following elements and parameters:

- **M:** The site with computing elements and separated storing in an election network with point to point topology.
- ***S_i*;** th rate of site(i) storing capacity ($1 \le i \le M$)
- ***C(i,j)*** the cost of relation between Site *i* to site *j*
- **N** The file with the names $o_1, o_2, \dots, o_N$ and volumes $O_1, O_2, \dots, O_K$ and ($1 \le K \le N$)

R_{ik}: number of site i requests for reading the file k.

W_{ik}: number of site i requests for writing file k.

P_k: For any file, there are several versions in the sites and main version of the file O_k has been shown with P_k.

RS_k: Per any version O_k, the information is by RS_k.

In order to compatibility among versions, and due to the change of O_k versions, alterations are sent to P_k and then are operated from P_k toward other versions.

S_{ik} : The closest site is costly consisted of O_k version toward site i

Suppose that Grid environment was an M×N Matrix shape

A) $X_{ik} = 1$, if main version existed in site i.

B) $X_{ik} = 0$, if main version didn't exist in site i.

In order to find an appropriate and optimal method in relation to the costs, the calculation of cost function is very important.

The main purpose here is to reach the least final cost as follows:

$$\text{Minimize TC(X)} \qquad (1)$$

Here, there are two limitations for the Matrix X.

The total volume of the files and the existing versions in site i should not be more than site i capacity.

$$\sum_{K=1}^{N} X_{iK} O_K \le S_i \text{ for all } 1 \le i \le M \qquad (2)$$

Therefore, the cost of reading and writing regarding to RS are as follows. [18]

$$X_{pK} = 1 \text{ for all } 1 \le K \le N \qquad (3)$$

$$R(RS) = \sum_{i=1}^{M}\sum_{k=1}^{N} R_{ik} = \sum_{i=1}^{M}\sum_{k=1}^{N} \Upsilon_{ik} o_k c(i, R_{ik}) \quad (4)$$

$$W(RS) = \sum_{i=1}^{M}\sum_{k=1}^{N} W_{ik} = \sum_{i=1}^{M}\sum_{k=1}^{N} W_{ik} O_{ik} \left[C(i, P_k) + \sum_{i=k=1}^{N} C(P_{ik}, j) \right] \quad (5)$$

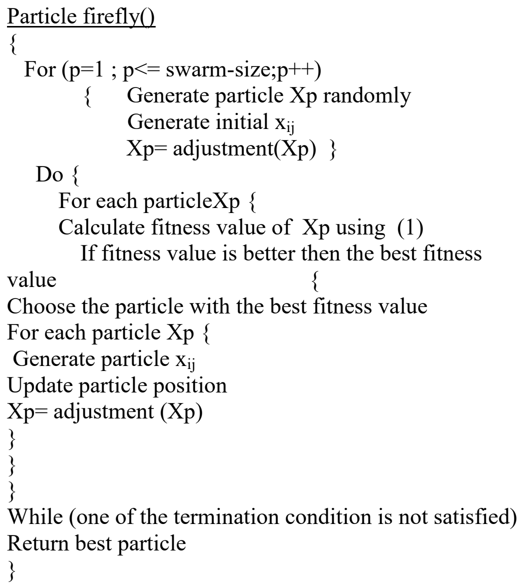
Equation (4) indicates the final costs of reading all files in Grid theatrically.

It should be noted that site S_{ik} responds to those reading applications having the least cost. In this study, the closest node was considered.

In equation (5), the writing cost is consisted of two stages including writing in P_k and the cost of publication from P_k to all sites including O_k versions.

$$TC(RS) = R(RS) + W(RS) \qquad (6)$$

Figure 2 proposed the algorithm initiated by an initial population in which any of these particles represents a solution. Finally, creating a solution is generated until a specified time is expired. During the generation of any particle, the superiority has been found by those particles in any generation obtained from the best places by using their personal experience and collective intelligence. Finally, any particle has revealed its place by any load. Here, in order to satisfy the constraints, the adjustment function was used. Adjustment algorithm code can be observed in Figure 3. Finally, the best particle is selected as the best solution.



```
Particle firefly()
{
  For (p=1 ; p<= swarm-size;p++)
         {   Generate particle Xp randomly
             Generate initial x_ij
             Xp= adjustment(Xp)  }
    Do {
      For each particleXp {
      Calculate fitness value of  Xp using  (1)
         If fitness value is better then the best fitness
value                              {
Choose the particle with the best fitness value
For each particle Xp {
 Generate particle x_ij
Update particle position
Xp= adjustment (Xp)
}
}
}
While (one of the termination condition is not satisfied)
Return best particle
}
```

Fig. 2. Proposed algorithm.

```
Particle adjustment (particle Xp)
  For each file
  j=(((p(k)-1)*N)+k);
    Xpj=1;
  }
s=1;
for each site i { if constraint no satisfy
   j=(((i-1)*N)+1);
   g=i*N;
   while j<=g
      if((a(j)==1)&&(i~=p(j-(i*N)+N)))
         calculate Δci(j-iN+N)
         diff1(s)=ch;
         diff2(s)=j;
         s=s+1;
      end
   end
end
sort(diff1)
   while sum<=s(h)
      j=diff2(s1);
      x(j)=0;
      s1=s1+1;
   end
end
```

Fig. 3. Adjustment Algorithm.

V. SIMULATION

The method of showing ability of an algorithm is evaluated according to other algorithms.

This process is performed by the aid of simulation. In this study, MATLAB software was used for simulation. Our algorithm was compared by other similar ones by using propagation method such as pso and simulated annealing. The simulated Data have 10 to 150 nodes. The network structure is E-shaped and node cost is between 1to 10 indicating the interval among the packets for reaching the objective. The least and most value of files is 2k byte, and 7 megabyte, respectively. The numbers of various process is between 50 to 450. The initial site is considered randomly and the file volume is considered to be 25% to 90%.

Figure 4 Results of the mentioned algorithm simulations in networks shown with different sizes.

Figure 5 indicates the middle rate of efficiency for mentioned algorithm in different modes with various parameters.

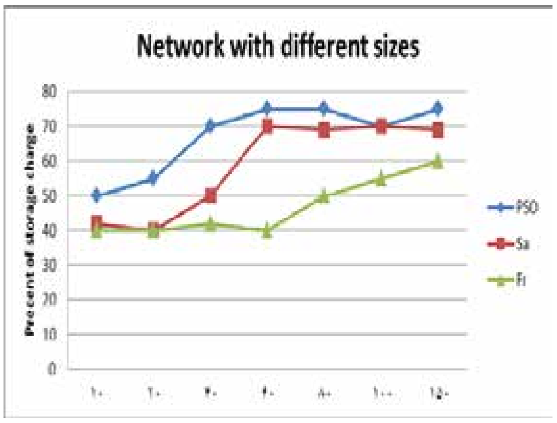


Fig. 4. The rate of final cost optimization with different network size.

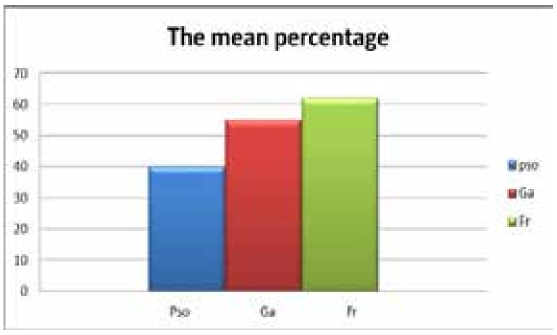


Fig. 5. The rate of algorithm final efficiency in different networks.

VI. CONCLUSION

Some experts in this field believe that Grid technology is regarded as the second chance of internet and has developed very fast and every days, new issues are proposed or has expanded the existing discussions. Data generation in a distributed system is a method which guarantees efficiency optimization of systems. In Data generation, we need number and place of determining versions. In this study, by using cost-based method, a new algorithm was presented. The supposed algorithm presents more optimized results for investigated algorithms.

REFERENCES

[1] Foster I., Kesselman C, 2004, The Grid Blueprint for a new computing infrastructure. Morgan Kaufman.

[2] Tanenbaum A S., Van Steen M., 2007, Distributed systems Principles and Paradigms. Prentice Hall.

[3] Yang m., Fei Z. , 2003, A Model for Replica placement in Content Distribution Networks for Multimedia Application, IEEE international conference on communication, Vol. 1, pp. 557-561.

[4] Wolfson o., Milo, A, 1991, The Multicast Policy and Its Replicated Data Placement, ACM Transaction Database systems, Vol. 16, No.1, pp. 181-205.

[5] Chang R, Chang H. , 2008, A dynamic weighted data replication strategy using access-weights in data grids. Journal of Supercomputing.45: pp. 277-295.

[6] Ranganathan K. and Foster I, 2001, Identifying dynamic replication strategies for a high-performance data grid, Proceedings of International Workshop on Grid Computing, pp. 75–86.

[7] Rahman R. M., Barker K. and Alhajj R., 2008, Replica selection strategies in Data Grid, Journal of parallel and distributed computing, Vol. 68, pp. 1561-1574 .

[8] Nukarapu D. T. , Tang B., Wang L., Lu S., 2011, Data replication in data intensive scientific applications with performance guarantee, IEEE Transactions on parallel and distributed system, Vol. 22, No. 8, pp. 1299-1306.

[9] Chevenak A., Schuler R, Ripeanu M., Amer M. A., Bharathi S. , Foster I. and Kesselman C., 2009, The Globus replica location service: design and experience, IEEE Transaction on parallel and distributed systems, Vol. 20, pp. 1260-1272.

[10] Tang M., Lee B. S. , Yao C. K., and Tang X. Y., 2005, Dynamic replication algorithm for the multi-tier Data Grid, Future generation computer systems, Vol. 21, No. 5, pp. 775-790.

[11] Andronikou V. , Mamouras K., Tserpes K., Kyriazis D. and Varvarigou T., 2012, Dynamic Qos-aware data replication in Grid environments based on data importance, Future generation computer systems. Vol. 28, No.3, pp. 544-553.

[12] Lee M. C., Leu F. Y., and Chen Y., PFRF, 2012, An adaptive data replication algorithm based on startopology Data Grids, Future generation computer systems, Vol. 28, No. 7, pp. 1045-1057.

[13] Saadat N., Rahmani A. M., 2012, PDDRA: A new pre-fetching based dynamic data replication algorithm in Data Grids, Future generation computer systems, Vol. 28, No.4, pp. 666-681.

[14] Taheri J., Lee Y. C., Zomaya A. Y. and Siegel H. J., 2013, A Bee Colony base doptimization approach for simultaneous job scheduling and data replication in Grid environments, Computers & Operations Research, Vol.40, No.6, pp. 1564-1578.

[15] Mansouri N. and Dastghaibyfard H., 2012, A dyamic replica management strategy in Data Grid, Journal of network and computer applications, Vol.35, No.4, pp. 1297-1303.

[16] Mansouri N. and Dastghaibyfard H., 2014, Improving Data grids performance by using modified dynamic hierarchical replication strategy, Iranian journal of electrical &electronic engineering, Vol.10, No.1, pp. 27-37.

[17] Yang, x.s., 2009, firefly algorithm for multimodal optimization, in: stochastic Algorithm foundations and applications, SAGA, lecture notes in computer science.

[18] Manghui Tu, Member, Ieee, Peng Li, I-Ling Yen, Member, Ieee, Bhavani Thuraisingham, Fellow, Ieee, And Latifur Khan, Member, IEEE, 2010 , Secure Data Objects Replication In Data Grid , IEEE Transactions On Dependable And Secure Computing, Vol. 7, No. 1.