International Journal of Computer & Information Technologies (IJOCIT)
www.ijocit.ir & www.ijocit.org
ISSN = 2345-3877

Presenting New Solutions for Discovering Source in the Grid Networks based on the Genetic Algorithms and Tree Structure

Bagher Alboghobeish[1], Mohammad Reza Noorei Meher[2] , Mashallah Abbassi Dezfooli[3]
Computer engineering, Ahvaz Branch, Islamic Azad University, Ahvaz, Iran[1,2,3]
Email: Bagher.albo@gmail.com

Abstract:- Grid is a network that allows you to create a great source using source systems connected to the Internet that already have accepted the terms of cooperation with network. Grid computing is a hardware and software structure that provides access to other unused computing capacity of other computers at a low cost in case of their approval. Grid network faces with many challenges, including discovering new source for requests. In this study, in order to find a source, firstly the total covered region is divided into smaller square regions and the nodes are placed in one region according to their geographic coordinates. Then, a value is assigned to each node based on the parameters of the computing power of node, node density (number of neighbors of nodes), physical location of node and duration of the presence and serving in the network; higher values represent the higher quality of the node. After that, the node that has the most value in each region is selected as the father and forms a tree in the region during the algorithm which will be explained in the following section. Then, different regions will be connected to each other and make the original tree. Finally, we look for the source utilizing the genetic algorithms built on the tree.

Keywords: discovering source, grid network, tree structure, genetic algorithm, and optimization.

1. Introduction

Grid is a vast network with high computational power that also has the ability to connect to the Internet. Grid does not consist of specific homogeneous computers but consists of a collection of computers distributed over the Internet or multiple Internets which are connected to each other non-exclusively through communication protocol via a grid management system [1].

In general, grid aims to share sources in a dynamic and heterogeneous environment. These sources become available with different policies. This sharing is used for computing purposes for scientific purposes and many cases where these sources can be a variety of sources including CPU, hard disk, software and sensors [2].

Today, with the development of grid network environments and the increase of the number of sources and geographic distribution of sources to find algorithms, discovering a source that can perform the required sources of a user in a short time with little traffic is needed although recently developed methods have fixed many deficiencies of previous methods such as single points of failure and heavy traffic [3].

There are many sources in the grid environment. These sources may be geographically distributed and locating them may be difficult. Thus, finding sources to meet the needs of users is a very important job. Traditional mechanisms of discovering centralized sources suffer from the single points of failure and heavy traffic of data; as a result, these mechanisms are not accountable for a lot of sources [12].

As noted, the most important challenge in the grid network is discovering source with the lowest cost. In addition, another important challenge in this context, which is more reflected on the focused algorithms, is the cost of modernization and development of the structure. If we have a weak structure, restructuring costs will be greater than the procedure of finding source. Another challenge is the grid dynamic network environment which, due to the many changes in its use, makes the common algorithms practically impossible, but our approach to this challenge is the use of intelligent algorithms. Ultimately, another challenge in this context is the cost-effectiveness of the entire network because in order to be able to use the computers at the Internet level, cost should to be paid to users according to the law and upon concluding the contract, and subsequently to implement the requests of users or institutions, payment should be received from them; dependent upon the payment, the quality of services is different [1].

In order to meet the following challenges, a plan is proposed based on tree structure and genetic algorithms.

In this study, by organizing nodes in a tree structure, it has been tried to offer a systematic structure in order to find appropriate sources. Moreover, in order to spend the least cost to build the tree structure as well as updating it, nodes with greater stability will be put at higher levels of the tree and this process is done by using mathematical formulas and using parameters such as the background of the node, the number of operational neighbors, the capacity of each node, and the ability of each node to adapt to the environmental conditions. In addition, using genetic algorithms, it is tried to make the best child in each node to send information and find the route. In this method, we choose the best child to find the route and a suitable source by producing different generations of children and evaluating them on the basis of various parameters and this procedure continues until a suitable source is found for the issued request. Furthermore, by dividing sources in terms of quality and cost and organizing them in several parallel trees, different appeals will be guided to the appropriate group depending on the payment which would be more productive for us.

2. Discovering the Source

Discovering source in static environments is easier than dynamic environments like grid because in static environments after running the algorithm at the moment of starting the act and finding the appropriate source, this source always remains at disposal, and whenever needed, we can use it without hesitation. But this is quite different in dynamic environments and the terms of the network and its sources may change at any moment. As a result, intelligent algorithms should be used because they find and provide the appropriate source by learning at any time. In addition, discovering a particular source in traditional computing environments was not that hard because the number of shared sources was small and all sources were under the control of a center. In grid systems, there are several major factors that turn the discovering issue to a very difficult issue. These factors include: a large number of sources, distributed ownership, heterogeneous sources, supply failure etc. [1].

Efficient source discovery can discover the source according to user requests by making the least traffic on the environment and without recourse to unnecessary and additional nodes in a short time.

3. Research background

- **The Yuan-Shun D and Xiao-Long method**

At the first stage, the requests reached by different nodes are arranged in descending order based on the time that source is needed. Sources assigned to tasks, which are not within the working scope, are fined so that the penalty coefficient is equal to 0.01. However, if a work is not fined and selects a source, that its time is smaller or equal to the previous source that it chose, we must reward it so that it continues this trend; the reward ratio is 0.02 [12].

- **The Karaoglanoglu Method**

The grid environment is considered as a combination of routers and sources. Each router in the environment is in charge of several sources. Famous algorithms in the field of Re-routing Tables, each router in the grid environment has a table called routing table. Routing table size is equal to a variety of sources on the grid. Numbers written in each of the columns of this table represent the minimum number of steps measured from the router to all available sources in the environment [5].

- **The Lee method**

In this way, each of the nodes in the environment is a cluster head. Discovering a source in this method is done in two states: between the cluster and within the cluster. In order to discover the source within the states, several routing tables called RDVT are used. In these tables, the distance from the source to the node is kept. To discover the source between the clusters, the requests of discovering a source are sent in the form of random steps or torrential steps [11].

- **The Konstantinos I. Karaoglanoglou Method**

In this method, the nodes are arranged in the form of a tree. Sources are evaluated qualitatively and divided into the categories of good, bad and average. Each node has a string of bits that indicate the child in which the source is placed or the source is not generally its subclass. This method, inspired by the Huffman code method, traverses the tree [12].

4. The Suggested Algorithm

In order to discover the source, firstly the total covered region is divided into smaller square regions and the nodes are placed in one region according to their geographic coordinates. Then, a value is assigned to each node based on the parameters and formulas which will be explained in the following; higher values represent the higher quality of the node. After that, the node that has the most value in each region is selected as the father and forms a tree in the region during the algorithm which will be explained in the following. Then, different regions will be connected to each other and make the original tree.

In the tree, every father knows what sources his children have and knows what sources are

available in their descendants but does not know the exact position of them; the method of routing will be explained in the following.

The most important step in the genetic algorithm is producing the fitness function; for this purpose, we use parameters of the computing power of node, node density (number of neighbor nodes), physical location of the node, and the history of presence and servicing in the network.

4.1. Computing power of node (PON)

One of the most important parameters in the selection of high quality node in the grid network is the computing power of node. Certainly, as the power of the node increases, it gives the network a high quality and more quick service.

1) PON =α1*RP + β1*MIPS

In the above formula, the MIPS of the number of executed instructions is in per unit of time and is calculated from the following formula.

2) MIPS = Instructions/S

In the first formula, RP represents a rated power of a node.

4.2. The History of Presence and Servicing of Node in the Network (PSN[1])

In order to use the tree structure, the nodes which are more stable in the network should be placed in the upper parts of the tree because due to greater stability, these nodes go out of the network less than other nodes. To calculate the parameters, the history of the presence or absence and the quality of the presence of a node in the network are used.

3) $PSN = \alpha_2 * {}^{t}/_{T} + \beta_2 * {}^{wt}/_{T}$

t: The standby time that the node is ready to give service, *T*: The total time of monitoring, *wt*: Average waiting time for the requests in queue of the node in order to implement the remaining work, α_2 : constant coefficient to control the impact of the standby that the node is ready to give service, β_2: constant coefficient to control the impact of the average waiting time at the time unit.

4.3. The total density of nodes (SHDS[2])

When the number of the neighboring nodes of a node outnumber, the selection of this node as a top node results in the coverage of more nodes, and subsequently the tree height is lower. As the height of the formed tree is less, less time is wasted in finding a suitable source in general. The number of the neighboring nodes of a given node represents the density of a node.

[1] Power Services Network

[2] Sum of the cluster Heads Distance

There is a simple method for discovering the density of each node in which each node sends a packet with TTL equal to 1 in the form of broadcasting, and each node that received packets, returns another packet called Ack. Thus, each node can notice the number of its neighbors. A node, which is placed at the center of the network, has better access to nodes that are several steps away from it, and for inclusion in the network backbone is a more appropriate option.

4.5. Central location in the Intended Geographical Region (SHC[3])

A node, which is placed at the center of the network, has better access to nodes that are several steps away from it, and for inclusion in the network backbone is a more appropriate option. On the other hand, when multiple nodes are adjacent to each other and have the same value in terms of the density parameter, the central node is the best choice for the higher-level nodes.

SHC is the Sum of the cluster Heads Centrality that can be calculated by the following formula.

$$a = \left| x_n - {}^{(x_2 + x_1)}/_{2} \right|$$
$$b = \left| y_n - {}^{(y_2 + y_1)}/_{2} \right|$$

(4)

$$SHC = \sqrt[2]{a^2 + b^2}$$

(5)

n **:** The intended node

x_n: The coordinate of the intended node in relation to the horizon

y_n: The coordinate of the intended node in relation to the vertical line

x_1 : Coordinates of the top corner and the left side of the square of the intended region in which the *n* node is located in relation to the horizon

x_2 : Coordinates of the bottom corner and the right side of the square of the intended region in which the *n* node is located in relation to the horizon

y_1: Coordinates of the top corner and the left side of the square of the intended region in which the *n* node is located in relation to the vertical line

y_2 : Coordinates of the bottom corner and the right side of the square of the intended region in which the *n* node is located in relation to the vertical line

4.6 Normalization of the Parameters

[3] Sum of the cluster Heads Centrality

At this stage, we have parameters that their greatness is different from each other. For example, PON is according to GHz that, for instance, its lower value can be 0.5 * 109 and, on the other hand, a good value for PSN is ultimately equal to 1. Now, if we add these two parameters, the PSN value becomes completely fade in the greatness of PON and this amount is not at all acceptable to us.

On the other hand, if we want to calculate the sum of parameters in order to obtain the entire information of nodes, the resulting answer may mislead us. For example, in some conditions, it is possible that we have values like 1, 50, and 100 for each of the parameters; therefore, if we use the sum for the overall evaluation of the parameters, the sum is equal to 151. If, in other condition, we choose three other parameters which are respectively 49, 50, and 52, the sum is again equal to 151. But it is evident that the first category is not at all a suitable choice.

Therefore, we use standard deviation to adjust the parameter so that the well-being of a parameter does not justify the deficiency of another parameter, which can be calculated by the following formula:

$$6)\ x_{new} = \frac{x_{old} - \mu}{\sigma}$$

In the above formula, σ is the standard deviation of data and μ the mean of the data. Moreover, we place the calculated values of PON, SHDS, PSN, and SHC instead of x_{old}, and x_{new} is the new amount of each of the mentioned parameters.

In this way, all four parameters are placed in the normal and acceptable range. We must combine these four parameters to arrive at a single value. Now, in order to obtain an amount to determine the value of the node, the following formula is used:

$$7)\ F = \omega_1 e^{PSN} + \omega_2 e^{-SHC} + \omega_3 e^{SHDS} + \omega_4 e^{PON}$$

Now, with determining a ranked formula and implementing it for the individual nodes, each group has a number which shows the fitness of that node.

In general, this phase of the algorithm can be explained with the following pseudo code:

Algorithm 1 calculate node value

Input: all node N and monitor environment E
Output: value of nodes float
PON=0,PSN=0,SHDS=0,SHC=0
While all N is visited do
PON= Calculate PON From (1)
PSN = Calculate PSN From (3)
SHC = Calculate SHC From (5)
Int count=0
While All neighbour of I visited do

```
Send Hello Packet to selected neighbour
        If(Receive Ack)
            Count++
          endWhile
        SHDS=count
 For All in (PON, PSN,SHDS,SHC)
   Var =Calculate Variance
   Dev= Calculate StandardDeviation
            EndFor
   F= Calculate SHC From (7)
          AssignTo I
          EndWhile
```

(8. Pseudo code)

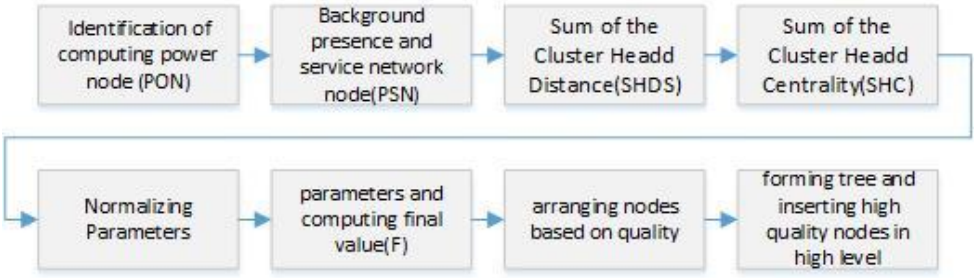

Figure 1: Total Diagram of Forming Tree

other and the root that has the most value is selected as the father of roots.

4.8. Way of Searching in the Tree

Requests for sources may reach any node at any height of the tree. At first, that node controls whether it has the source in its sources or not; if it has the source in its sources, it directly allocates it to the applicant node.

Otherwise, because each node knows what sources his children have, it examines whether there is a source according to the request in its children and grandchildren; if it is not available, it sends the request to its father because this father has more children and grandchildren. Like before, father also searches in its own sources and if it has the suitable source, it gives it to the applicant, and if it is not available, it sends the request to its father. If a node notices that there is a child in its descendants that has the requested source, it provides the chromosome node which is in fact a way to achieve the goal.

4.7. The Construction Process of the Tree

Because in this algorithm, we are trying to reduce the cost resulting from the network structuring and also its further settings, as mentioned before, nodes with high capacity are placed at higher levels of the tree so that the possibility of resetting is reduced.

At first, nodes are arranged in each region in descending order according to their value which was calculated in the previous step. Then, the best node is selected as the root of the trees in that region.

To find the children, a random number is selected based on the following formula:

$$r = \{rand(a, b) \mid a = \min \& b = max\}$$

In this formula, *a* is the minimum chosen amount and *b* is the maximum chosen amount and *rand* is the function generating random values.

After producing this number, we select the *r* of the remaining nodes which have the highest value and consider them as the children of the intended root. Now, we start from the leftmost child, and again according to the above formula, we make a random number which shows the number of the children of that node.

After constructing regional trees for all the regions, the neighboring roots connect to each

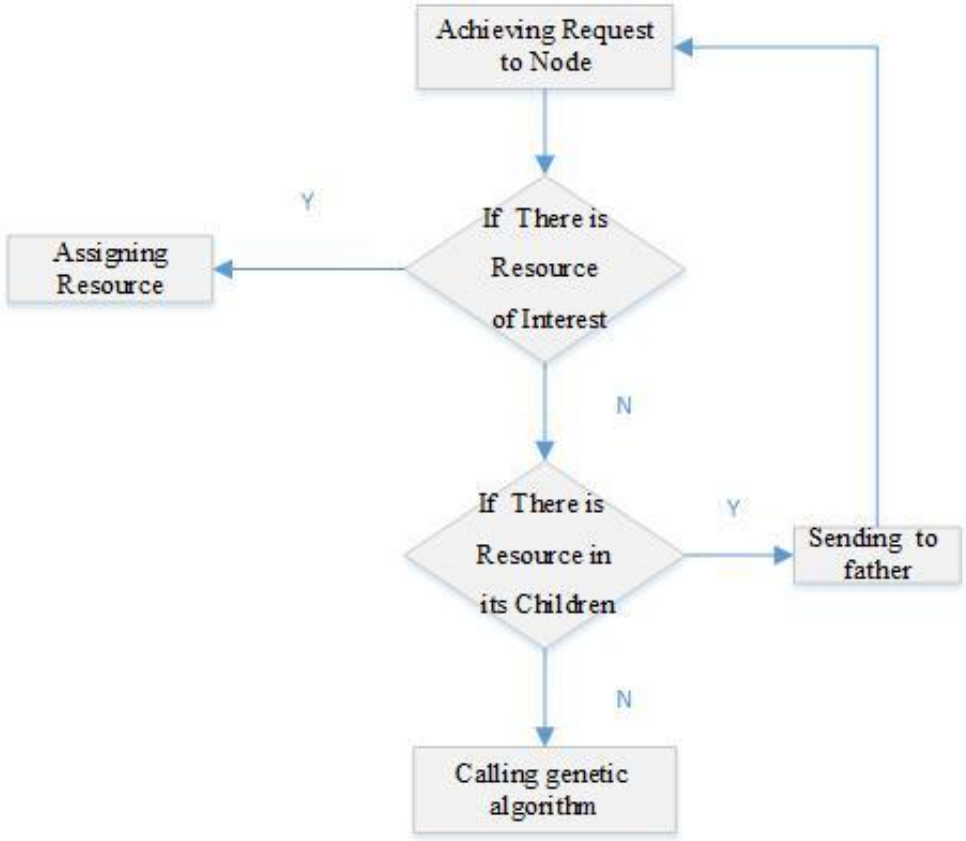

Figure 2: The Overall Diagram of Searching in the Tree

Figure 3: The Chromosome Structure

4.9. Chromosome Structure in Algorithm

Given that each node knows how much its height is in the tree, it builds a chromosome according to its height which has genes which are equal to the height of the node itself.

Each gene represents a child that the request should be sent to it.

In fact, the structure of a chromosome for a node which is located in the height of 3 is in the following figure:

This chromosome means that the request is sent to the second child at first and later to the fifth child and later to the fourth child until it complete its work.

4.10. The Initial Population

In most cases, the initial population is made of chromosomes randomly. We also use the random population.

The law of natural selection is that only species that have the best properties can descend from a population.

The initial population of chromosomes depends on the number of bits of chromosomes, and as the number of these bits increases, the initial population should also increase. For example, if the number of bits is equal to 8, we should consider fewer chromosomes than the time the number of bits is equal to 16.

4.10.1. Example

If node number 1 wants to build the initial population, because the height of this node is equal to 3, it considers the maximum number of genes equal to 3 inside each chromosome and selects a random value between 1 and 3 and equalizes the number of genes to it.

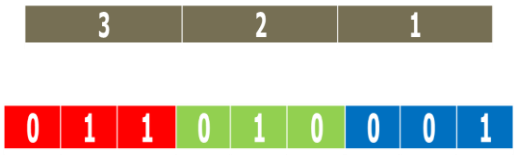

Figure 4: Chromosome initial

This chromosome represents the transmission to node 4, 9 and 18, and finally searches for the intended source in the node 18.

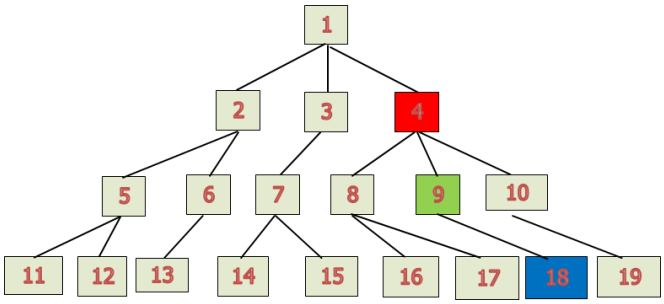

Figure13: Searching for the source

4.11. Chromosome Selection

The roulette method was used to select the chromosomes. For each chromosome in the current population, it is possible that the probability of each chromosome represents the chromosome chance of selecting that chromosome to transfer to the genetic pool.

After determining the possibility of selecting chromosomes, a random number is generated between 0 and 1.

The first chromosome is selected that its cumulative probability is greater than or equal to the generated number.

Chromosome selection formula: $P_k = \frac{f_k}{\sum_{i=1}^{n} f_i}$

We have 4 chromosomes and want to choose 2 chromosomes from them.

۳	۲	۱
۲	1	1
۳	۱	۱
۱	۱	۲

It is supposed that the calculated values for these genes are respectively equal to 3.2, 4.1, 1.6, and 2.1.

$$3.2 + 4.1 + 1.6 + 2.1 = 11$$

$$P_1 = \frac{3.2}{11} = 0.29 \quad P_2 = \frac{4.1}{11} = 0.37 \quad P_3 = \frac{1.6}{11} = 0.15 \quad P_4 = \frac{2.1}{11} = 0.19$$

Table 1: Frequency table of chromosome

Name	Probability	Percentage	The cumulative percentage
P_1	3.2	0.29	0.29
P_2	4.1	0.37	0.66
P_3	1.6	0.15	0.81
P_4	2.1	0.19	1.00

value

Now, we choose a random number between 0 and 1, for example 0.59. Because this number is in the second interval, P_2 is selected as the first chromosome. Then, for the second chromosome, we choose another random number between 0 and 1, for example 0.96. Because this number is in the last interval, P_4 is selected as the second chromosome.

4.11.1. Combination

Crossover operator uses chromosomes and combines them together. Chromosomal integration has different ways that we use the following three ways: 1. Single-point integration, 2. Two-point integration, 3. Integration by mask.

4.12.1. Single-Point Integration

In this method, we choose two chromosomes from among the chromosomes, and then randomly choose a point of chromosome and replace all the genes which are after this point in two chromosomes. We consider the probability of the selection of this method equal to 40%.

4.12.2. Two-Point Integration

In this method, we choose two chromosomes from the chromosomes of the middle population, and then randomly choose two points of chromosome and replace all the genes which are between these two points in the chromosomes. We consider the probability of the selection of this method equal to 50%.

4.12.3. Integration by Mask

In this method, we use a mask for integration; that is, we create the array which has elements equal to the number of genes. The elements of this array can only accept zero or one amounts. Mask array elements are randomly initialized. Now, using this mask, we combine two chromosomes. The zero value in the mask array indicates that the gene should be selected from the first chromosomes. The one value in the mask array also indicates that the gene should be selected from the second chromosome. We consider the probability of the selection of this method equal to 8%.

Firstly, we create a random number between 1 and 100. If the number is between 1 and 40, we use the single-point integration; if it is between

41 and 90, we use two-point integration; if it is between 91 and 98, we use mask integration and finally if it is 99 and 100, we use mutation.

4.13. The Function Giving Value to Chromosomes

If there were transmission to child in a chromosome and the number of children of that father was less than the intended numbers, the value of that chromosome would be considered zero. If, in the route of chromosome movement, we reach a node whose children do not have the intended source, the value of that node will be equal to zero and we do not continue it anymore. And if none of the above conditions are met, we calculate the value of chromosome using the following formula.

$$V = \omega_5 dis + \omega_8 {}^{RS}/_{FS}$$

V represents the value of each chromosome, *dis* represents the number of steps from the applicant node to the node in which the source is found, *RS* represents the power of requested source by the applicant node, *FS* represents the power of found source in the destination node.

Therefore, by generating a random initial population to find the source, we are trying to create a new generation of chromosomes using reproduction and mutation operations and remove the chromosomes with low fitness and replace them with new chromosomes; the provision of ending this process is achieving a specific fitness or certain number of repeating algorithms; otherwise, the process should be repeated again. Therefore, we represented an intelligent algorithm to find a suitable source in the grid network using the tree structure and genetic algorithm which, considering the proposed method for ranking nodes and also the use of various parameters, gives us a good answer to find the source.

4.14. Experimental Results

++ OMNeT is an object-oriented simulation and is considered as the "discrete event" of software category and it is the abbreviation of Objective Modular Network and is based on C ++. To compare the performance of the proposed algorithm, we compare this algorithm with TinyLap, EAR and PGR algorithms, and the results of this comparison can be seen in the following diagrams.

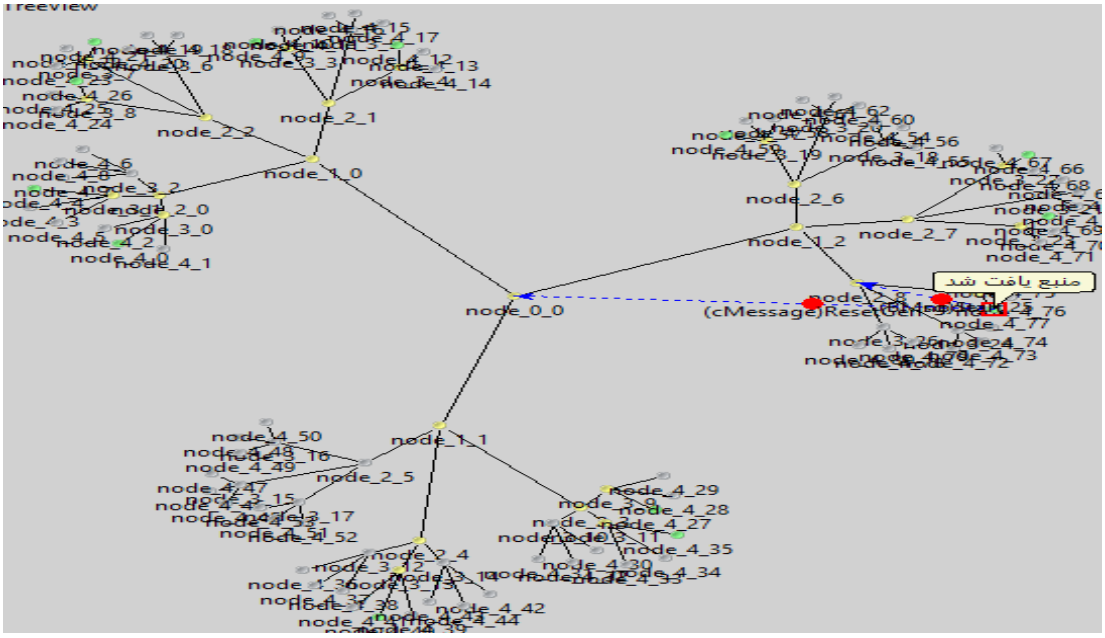

Figure 5: Connection of trees and status of nodes

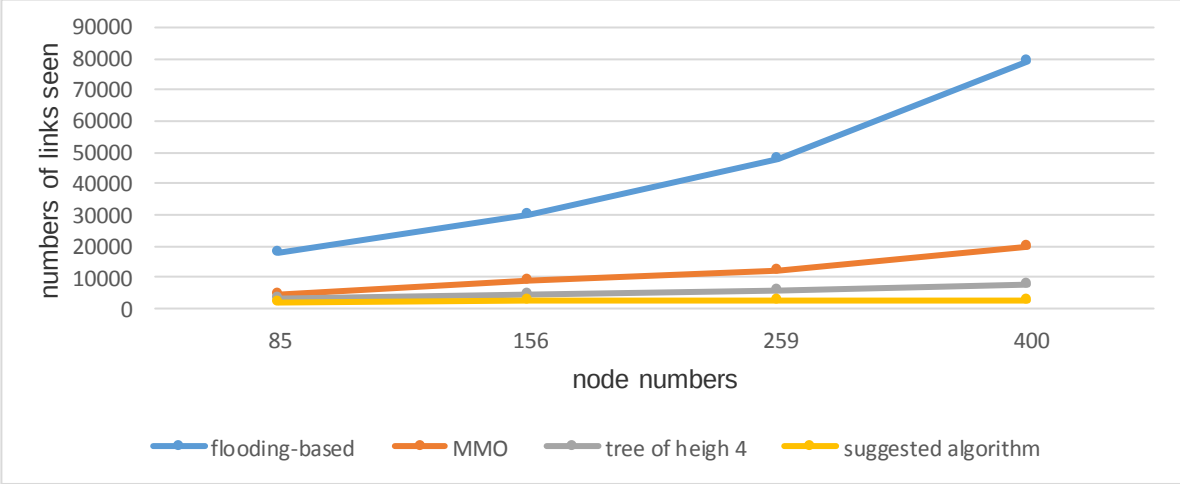

Figure 6: Diagram of seen links in 300 requests

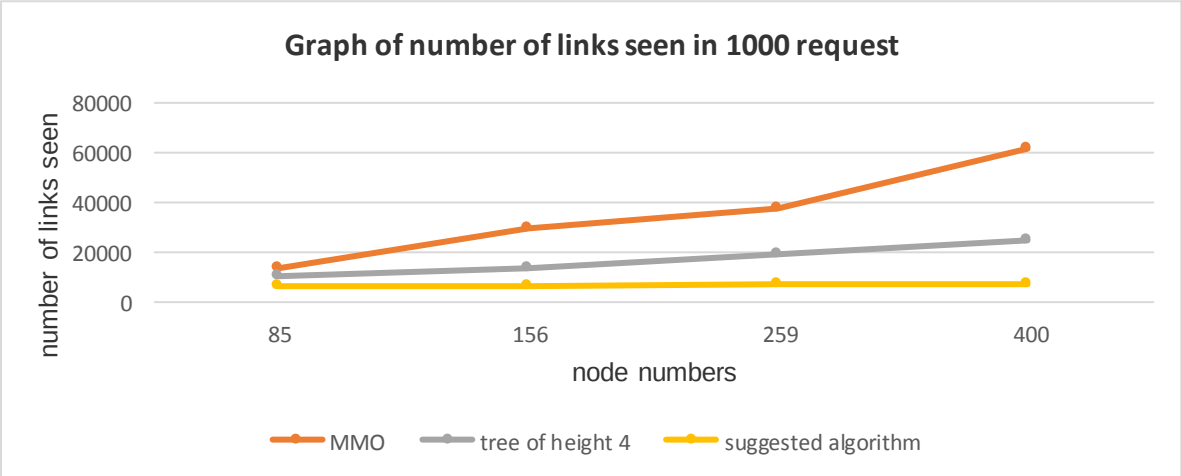

Figure 7: Diagram of seen links in 1000 requests

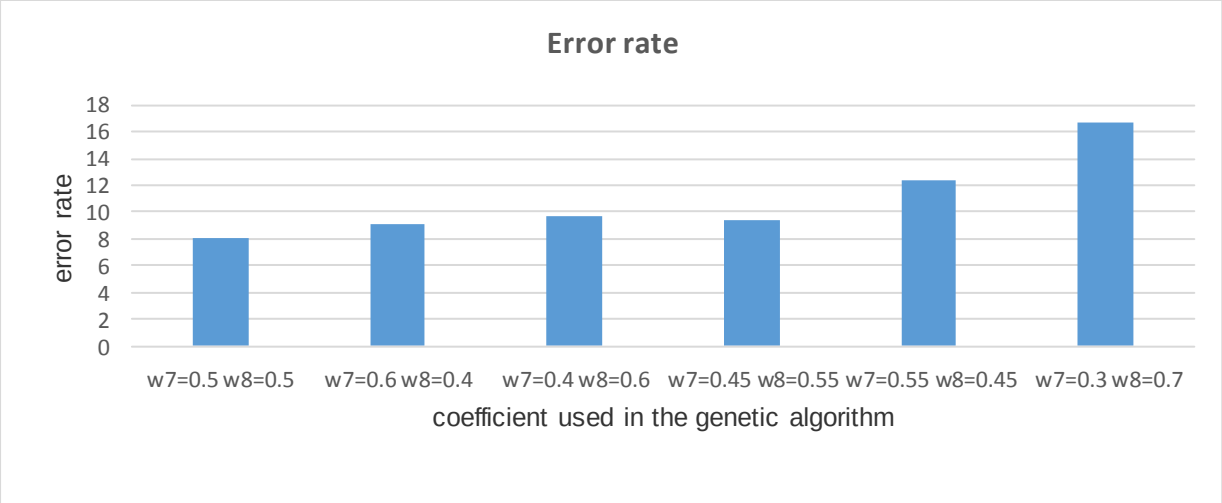

Figure 8: Diagram of error rate with the change of constant coefficients

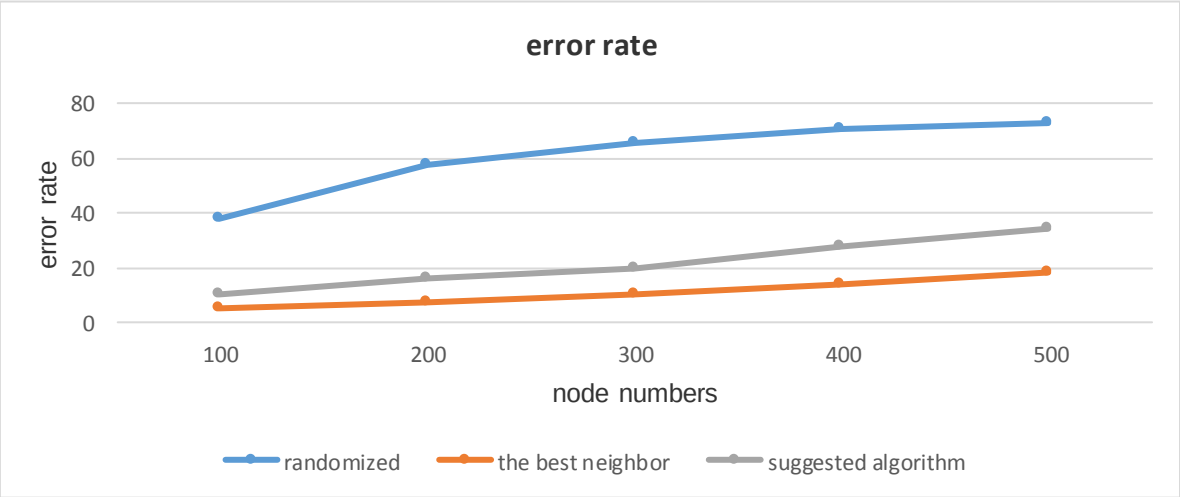

Figure 9: Graph of error rate

4.15. Future Research

In order to improve this thesis, the tree structure can be turned into a multi-row tree so that for offtree nodes, we consider that these nodes can supervise several levels directly regardless of the relationship between father and child, and when a request reach a node, with the probability of 0.1, for example, instead of looking for the source in the usual tree structure, it sends the request to the super group and this node finds the intended source out of the tree structure and sends the request to it and this probability can be amounts less than 0.05 and this result can be reached by simulation.

Or we can classify the requests based on the amount of money that the request owners have paid and give priority to requests that their owners have paid more for calculation. We should enhance the priority of non- implemented requests over time so that the requests that their owners have paid less do not starve. As such, these requests are also finally being done, but probably a little later than other requests.

5. Conclusion

According to the results of the simulation and due to the low overhead of the algorithm, it is a suitable option to be used in the grid network.

In addition, the high ability of genetic algorithm should be used to find appropriate ways according to environmental changes, and also the node computing power, the history of presence etc. should be contributed.

By combining the methods of giving value to nodes and genetic algorithm, we could present an algorithm to discover the source in the grid

networks based on the tree structure which according to the results of simulation, it brings an acceptable power for us.

References

[1] Agustín C. Caminero, Robles-Gómez,A., Ros, S., Hernández, R., Tobarra, L., (2013) "P2P-based resource discovery in dynamic grids allowing multi-attribute and range queries, Parallel Computing", Vol. 39, No.10, pp. 615-637

[2] Khanli, L.M., Kargar, S.,(2011) "FRDT: Footprint Resource Discovery Tree for grids", Future Generation Computer Systems 27 PP 148–156

[3] Khanli, L.M., Mahan, F., Isazadeh,(2011) A.,"Active rule learning using decision tree for resource management in Grid computing, Future Generation Computer Systems", Volume 27, No 6, PP 703-710

[4] Konstantinos I. Karaoglanoglou, Helen D. Karatza, (2008) "Resource Discovery in a dynamical grid based on Re-routing Tables, Simulation Modelling Practice and Theory", Vol 16, No 6, PP 704-720

[5] Jafari Navimipour, N., Rahmani, A., M., Habibizad Navin, A., Hosseinzadeh, M., (2014)"Resource discovery mechanisms in grid systems: A survey, Journal of Network and Computer Applications", Vol 41, PP 389-410

Systems based on QoS", Second International Conference on Computer and Electrical Engineering IEEE, Dubai. PP 418-425

[12] Juan L, (2009) "Grid resource discovery based on semantically linked virtual organizations", Elsevier, Vol 26, No 3. PP 361-373

[6] Karaoglanoglou, K., Helen D. Karatza, (2008) "Resource Discovery in a dynamical grid based on Re-routing Tables", Simulation Modelling Practice and Theory 16 PP 704–720

[7] Karaoglanoglou , K., Helen D. Karatza, (2008) "Resource Discovery in a dynamical grid based on Re-routing Tables", Simulation Modelling Practice and Theory 16 PP 704–720

[8] Ruay-Shiung Chang, Min-Shuo Hu, (2010) "A resource discovery tree using bitmap for grids", Future Generation Computer Systems 26 PP 29-37

[9] Juan Li, (2010) "Grid resource discovery based on semantically linked virtual organizations", Future Generation Computer Systems 26 PP 361-373

[10] Ali Sarhadi, Mohammad Reza Meybodi, (2009) New Algorithm to Resource Selection in Computational Grid Using Learning Automata, International Journal of Intelligent Information Technology Application, PP 204-208

[11] Esmaeelzadeh R.H., Rahmani A.M., Zamanifar K,(2009) "Resource Management in Semantics Grid

[13] Yuan-Shun D and Xiao-Long W, (2005)"Optimal resource allocation on grid systems for maximizing service reliability using a genetic algorithm", Elsevier, Vol 91, No 9, PP 1071-1082

