ACSIJ
WWW.ACSIJ.ORG

An Adaptive K-random Walks Method for Peer-to-Peer Networks

Mahdi Ghorbani
Electrical, Computer and IT Engineering Dept., Qazvin Branch, Islamic Azad University
Qazvin, Iran
mahdigh13@yahoo.com

Abstract
Designing an intelligent search method in peer-to-peer networks will significantly affect efficiency of the network taking into account sending a search query to nodes which have more probably stored the desired object. Machine learning techniques such as learning automaton can be used as an appropriate tool for this purpose. This paper tries to present a search method based on the learning automaton for the peer-to-peer networks, in which each node is selected according to values stored in its memory for sending the search queries rather than being selected randomly. The probable values are stored in tables and they indicate history of the node in previous searches for finding the desired object. For evaluation, simulation is used to demonstrate that the proposed algorithm outperforms K-random walk method which randomly sends the search queries to the nodes.
Keywords: *Peer-to-peer, Search, K-random Walk, Learning Automata Theory.*

1. Introduction

The main characteristic of peer-to-peer networks is that the nodes can be connected to the network or leave it at any time. In other words, there is no central control on behavior of the nodes in the network, with all the nodes acting similarly in terms of performance level. One of the important issues in these networks is searching stored objects in the nodes. Finding an object will be exposed to some challenges with respect to how the nodes are located in the network. The peer-to-peer networks are divided into structured and unstructured regarding their structures. In the former, location of the nodes is predefined by distributed hash tables (DHT) so finding an object is done readily using hash functions. On the other hand, the latter, distribute their contents in a completely random fashion and the nodes have no information about the network status and location of the stored objects in other nodes [1-3]. Therefore, it is not simple to find an object in this structure and search mechanisms must be utilized here. It is very important to design a search method for these networks and it can further affect efficient of the network considerably.

Based on information of the nodes from the contents, the search techniques are classified as informed and blind [4-5]. In informed methods, the nodes store a number of metadata from their neighborhood. By this kind of information, the nodes would be informed about the network status as well as the location of contents for the other nodes. In blind search methods, the nodes have no information about location of the objects, so they employ flooding algorithms to direct their search queries. One of these methods is random walk [7-8]. Once a search for finding an object is unsuccessful, random walk selects some of the neighbors randomly and sends the search queries to them through a flooding algorithm until the search time is terminated. Each of these two search methods (informed and blind) has its own advantages and disadvantages which can noticeably affect some network criteria such as search success, average response time, average number of objects found in the search process and network load.

According to previous studies [6-9,13-18], using artificial intelligence methods such as reinforcement learning techniques [11-12], has improved search performance to a large extent. By these methods, each node in the networks learns to select which if the neighbor nodes for sending the search query.

This paper proposes a search algorithm based on learning automaton [16], where the nodes select their neighbors intelligently and not randomly for sending the queries. Each node will store a number in its history table based on the feedback received from the neighbor nodes. During the next levels of search, the nodes which have assigned themselves a greater number in the current object search will probably contain that object. The suggested algorithm compares the search success, the number of discovered objects per each query and also the amount of overload due to messages generated for each search queries with those obtained from K-random walk [8-10]. Oversim [22] software has been used for the purpose of simulation and

the obtained results show advantage of the proposed algorithm.
The following text is organized as follows. A short history of the relevant works conducted so far with the proposed protocol in presented in Section 2. In section 3, the learning automata briefly explained as the main learning strategy in the developed algorithm. The proposed algorithm is then introduced in Section 4, while the results of simulation are provided in Section 5. Finally, some conclusions are made in Section 6.

2. Background

Search strategies in the peer-to-peer networks are categorized into blind and informed search methods considering the amount of information available for the nodes from the network status. In the blind methods, each node directs the query to all its neighbors and the search is terminated once "success" or "failure" is occurred, or when TTL is over. The blind methods waste a lot of network bandwidth and cause a large overload [6].
In K-random walk method, the queries are sent just to "k" randomly selected neighbors instead of all neighbors. Complexity of the generated messages is small well, but the amount of success is variable due to random selection of the neighbors for direction of the search queries. On the other hand, random selection of the neighbor nodes can negatively affect response time of the search queries.
In the informed method, each node stores some information about its neighbors and network status as well. Adaptive probabilistic search (APS) [9,10], SALA [17], LARW [18] are some examples of the informed methods which utilize experiences of the previous neighbors for searching the next levels.
Adaptive probabilistic search (APS) woks based on using "k" independent steps and random direction of the queries. The probabilistic selection instead of random selection is accounted for an advantage of these techniques. Each intermediate node directs the queries toward its neighbor which has stored a probabilistic value in its local index. Values of the indices are updates by receiving feedback from the steps. APS enhanced reliability of the search and optimizes consumption of the bandwidth.
In [17] a novel self-adaptive learning automata based search algorithm (SALA) is introduced. In this method, each node uses a learning automata algorithm to select the suitable neighbors. By applying three tables comprising of the previous experiences of nodes, the neighbors train and increase their chance to participate in search. Although SALA is an adaptive search to the network size but the learning rate is low because of updating three tables for each iteration.
LARW [18] is a new version of k-random walks algorithm which utilizes a new form of learning automata algorithm is called KSALA [16]. KASLA helps the nodes decide to select the suitable neighbors for search. Decision is according to the probability values of neighbors for participating in search in previous iterations. The performance of search is good enough but at the first steps of search, the performance is low because of slow learning rate of nodes.

3. Overview of Learning Automata Theory

Learning automaton [19-21] is a machine which can perform many finite actions. Each selected action is evaluated by a probabilistic environment and the result of evaluation is given to the automaton in the form of positive or negative signals. Thereby, the automaton is affected by this response in selection of its next action. The final goal is that the automaton learns to choose the best action among the available ones. This could be an action which maximizes the probability to receive reward from the environment.
The environment can be represented by a triple like $E \equiv \{a, b, c\}$, in which $a \equiv \{a_1, a_2, ..., a_r\}$ is the set of inputs, $b \equiv \{b_1, b_2, ..., b_m\}$ is the set of outputs, and $c \equiv \{c_1, c_2, ..., c_r\}$ is the set of penalty probabilities. The probability for an undesirable result of c_i action equals a_i. In static environment, the values of c_i remains unchanged, though these values change over time in non-static environment. The learning automata are divided into two groups with constant and variable structures. The following will introduce the learning automaton with variable structure.
The learning automaton with variable structure can be shown by a quadruplet like $\{a, b, p, T\}$, where $a \equiv \{a_1, a_2, ..., a_r\}$ and $b \equiv \{b_1, b_2, ..., b_m\}$ represent the sets of actions and inputs of the automaton, respectively, while $p = \{p_1, p_2, ..., p_r\}$ gives the vector of selection probability for each of the actions and $p(n+1) = T[a(n), b(n), p(n)]$ denotes the learning algorithm. The following algorithm is an example of the linear learning algorithms. Assume that the action a_i is selected at the n^{th} level.
Favorable response:

$$p_i(n+1) = p_i(n) + a[1 - p_i(n)]$$
$$p_j(n+1) = (1-a)\, p_j(n)$$

Unfavorable response:

$$p_i(n+1) = (1-b)\, p_i(n)$$
$$p_j(n+1) = (b/r - 1) + (1-b)\, p_j(n)$$

In Equations (1) and (2), a is the reward parameter and b is the penalty parameter. Three following states can be considered according to the values of a and b. When a and b are equal, the algorithm is called L_{RP}; when b is much smaller than a, the algorithm is called $L_{R\varepsilon P}$; and when b is zero the algorithm is called L_{RI}.

4. Proposed Search Algorithm

In this paper, the learning automaton has been used for training the nodes in the network for selection of the desired neighbor nodes. At first, the data structure of the proposed algorithm is explained and then, the proposed algorithm is introduced.

4.1 Data Structure of Search Algorithm

Two tables are used in the developed search algorithm for learning the network status and the location of objects. Each row of these tables indicates an existing object in the network, whereas each column of these tables shows one neighbor of that node. For each neighbor node, there is a value called LA-value which denotes the extent of its history for finding the objects in the previous searches. The table called Query-LA-table is a table involving neighbor nodes which have experienced search before. During search and when a query is going to be sent, the values in this table are used. Selection of the neighbors is done randomly based on the learning algorithm. However, as much big as their LA-values are, the probability of their selection will be greater.
Figure 1 shows a sample of these tables.

	LA-values to neighbours			
Keywords	N_1	N_2	….	N_m
Object 1	V_{11}	V_{12}	….	V_{1m}
Object 2	V_{22}	V_{22}	….	V_{2m}
.....	….	....	….	….
Object n	V_{n1}	V_{n2}	….	V_{nm}

Fig. 1 A sample Query-LA-table with *n* objects and *m* keywords.

The second table which is used in this algorithm is Neighbor-LA-table and contains those set of neighbor nodes which have not experienced a search with the current query before. The values in this table are indicative of the performance for each of these neighbors obtained in the previous searches. Figure 2 shows a sample Neighbor-LA table.

N_1	N_2	….	N_m
V_{11}	V_{12}	….	V_{1m}

Fig. 2 A sample Neighbor-LA-table with *m* keywords.

4.2 How the Search Algorithm Works?

As mentioned before, two tables are used to learn the network status and the location of objects. These tables are empty at first. A row is added to them by sending each query with all the neighbors being initialized by a uniform probability distribution. The values of these tables are then updated based on the results obtained from the search.
The neighbor nodes with a higher search performance are rewarded and their probability is increased. Value of this reward is determined by distance of the object discovered from its applier or in other words, number of the steps passed for reaching the desired object. The nodes which do not participate in the search process will neither receive reward nor penalized. The values in these tables are used in the next searches to select neighbors for sending intelligent queries.
In sending a query to a node a hit is returned when the search is successful, otherwise the search continues by sending queries to a random number of the neighbor nodes with the highest LA-value in Query-LA-table. The search would resume in Neighbor-LA-table by choosing a random number of neighbors once the searched object is not found in this table. The search is terminated when the desired object is found or when TTL is over. At last, the existing values in the tables are updated in accordance with the feedback they get from their environment.
Figure 3 depicts pseudo code of the search algorithm.

```
// Query Keyword – Q; Query Source – S; Total
number of walkers K//

1. User submits a query
2. Search source node for Q
3. If Q is not in S
    3.1 Search for Q in the Query-LA-table
    3.2 If Q is found
         Select K walkers from Query-LA-table
         with automata algorithm
        Generate K query messages
        Search starts with K walkers
       Else
```

```
        Select the walkers from Neighbor-LA-
        table
4. If a hit occurs, sent back result on the
   reverse path.
5. All node on the path, update appropriate LA-
   table.
```

Fig. 3 Pseudo code of the proposed search algorithm.

5. Simulations

In this section, hypotheses of simulation are stated first and result of the simulation are investigated then.
Some network criteria such as overload due to the messages generated in the network are compared in addition to search quality criteria like success of the search and average number of the objects found per one query with the so-called K-random walk method.

5.1 Hypothesis of Simulation

Oversim simulation software is utilized for simulating the proposed algorithm. The network used here is a random graph [4] including 1000 nodes, where minimum output degree of each node is 6. In this network 200 different objects are randomly distributed between the nodes with each node containing a number of objects according to the memory capacity considered for it. Maximum time needed to deliver packages through the network (TTL) is taken 6. Dynamism of the network is addressed using failure rate of the nodes which is expected to be 30% in average. In other words, among all the existing nodes in the network 30% of them are either inactive or leaving the network. Maximum number of walkers is 15 in the simulations. A standard learning automaton is used with L_{RP} algorithm, in which the initial values of and are both considered 0.5.

5.2 Evaluation of Proposed Search Algorithm

Performance of the proposed search algorithm is evaluated by implementing different simulations. The method developed here is compared with K-random walk method.

5.2.1 Success Rate

When a query is being sent to one node or several nodes, a hit message will be generated once the desired object is found, which indicates success of the search. The search is called successful when at least one hit message is received. Average number of these successes can be calculated as well. Figure 4 has compared success of the search between the proposed algorithm and K-random walk method. As shown in the diagram, success of the search is significantly higher in the suggested method due to intelligent selection of the neighbors unlike the random selection in K-random walk.

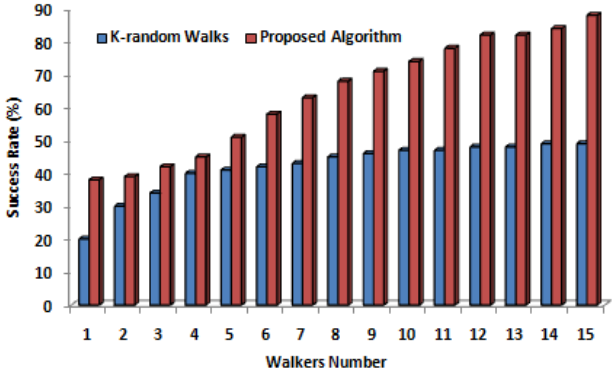

Fig. 4 Comparing success rate for two algorithms.

5.2.2 Number of Discovered Objects

Figure 5 shows the number of objects found for each query. The greater this number is accuracy, speed and resistance of the algorithm is higher. Resistance of the algorithm is improved because whenever a node containing several objects fails in the network, other samples of that objects exist in the network. This can enhance resistance against node failures. Figure 5 demonstrates that the proposed algorithm has an excellent performance in finding the objects in the network, whereas random selection of the neighbors in K-random walk method leads to an increased generation of misses.

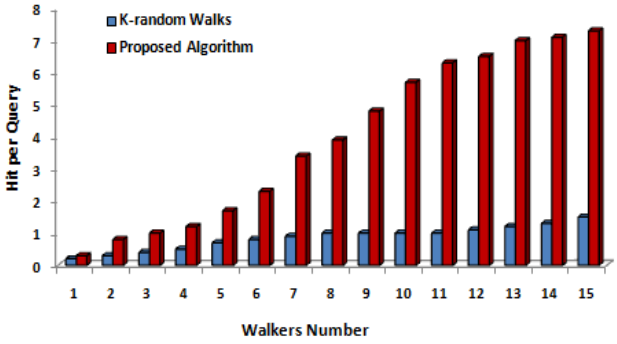

Fig. 5 Comparing number of objects found for two algorithms.

5.2.3 Overhead from Generated Messages

Figure 6 gives a comparison between the proposed algorithm and K-random walk algorithm taking into account the overload generated from messages of each query. It can be observed in this diagram that the proposed

algorithm provides a much smaller average number of messages generated for each query as compared with that of K-random walk, because no additional message and query is sent for this purpose into the network. The obtained overload from generation of the messages is very significant in K-random walk method since the queries are randomly sent.

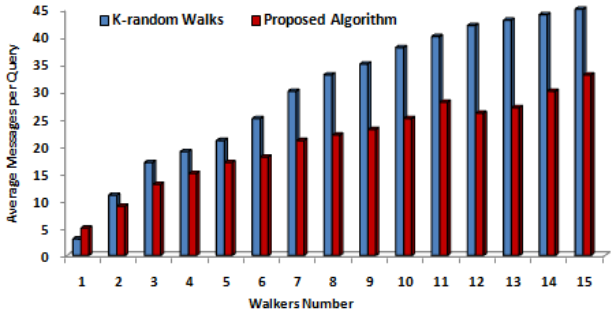

Fig. 6 Comparing overhead from generated messages for two algorithms.

4. Conclusions

In the proposed method, each node stores some tables which keep histories of the neighbors in previous searches. These values are probabilistic and are updated considering the hit/miss message received from the neighbor nodes using learning automaton. The suggested search algorithm was compared with K-random walk method. The obtained results reveal that due to using the history tables for each node, those with a greater probability to contain the desired object are selected and thus, the rate of success is increased. On the other hand, intelligence of the proposed algorithm leads to send a smaller number of queries and as a result, fewer messages are generated within the network which reduces the overload. Meanwhile, number of the hops visited for reaching the chosen nodes is decreased. Therefore, the proposed algorithm outperforms K-random walk method in terms of success, network overload and average number of discovered objects per each query.

References

[1] S. Androutesellis, and D. Spinellis, "A survey of peer-to-peer content distribution technologies," ACM Computing Surveys, December 2004, vol. 36, no. 4, pp. 335-371.

[2] E. K. Lua, J. Crowcroft, M. Pias, R. Sharma, and S. Lim, "A survey and comparison of peer-to-peer overlay network scheme," IEEE Communication Survey and Tutorial, March 2004.

[3] A. Saghiri, and A. Bagheri, "Enhance Your Search Engine Functionality with Peer-to-Peer Systems," Proc. Of 2nd Int. Conf. on Computing and Automation Engeering, 2010, pp. 583-586.

[4] A. Saghiri, and A. Bagheri, "An adaptive architecture for personalized search engine in ubiquitous environment with peer-to-peer systems," Proc. Int. Conf. on Information and multimedia technology, 2009, pp. 107-111.

[5] D. Tsoumakos, and N. Roussopoulis, "Analysis and comparison of p2p search methods," 1st Int. Conf. Scalabale Information Systems, 2006, Article no. 25.

[6] S. M. Thampi, C. K. Sekaran, "Survey of search and replication schemes in unstructured p2p networks," Network Protocols and Algorithms, 2010, vol 2, no. 1, pp. 93-131.

[7] R. Dorrigiv, A. L'opez-Ortiz and P. Pralat, "Search algorithms for unstructured peer-to-peer networks," Proc. Of 32nd IEEE Conference on Local Computer Networks, 2007, pp. 343-349.

[8] C. Gkantsidis, M. Mihail, and A. Saberi, "Random walks in peer-to-peer networks," In INFOCOM 2004, Hong Kong, , 2004, vol. 1, pp. 120-130.

[9] D. Tsoumakos and N. Roussopoulos, "Adaptive probabilistic search for peer-to-peer networks," 3rd Int. Conf. P2P Computing ,2003, pp. 102-109.

[10] D. Tsoumakos ,and N. Rossopoulos, "Probabilistic knowledge discovery and management for p2p networks," P2P Journal, 2003, pp. 134-141.

[11] R. S. Sutton, A. G. Barto, Reinforcement learning: introduction, Proc. of the MIT Press, 1996.

[12] E. Mance and S. H. Stephanie, Reinforcement learning: A tutorial, Proc. of the Wright Laboratory, 1996.

[13] S. M. Thampi, and C. K. Sekaran, "Collaborative load-balancing scheme for improving search performance in unstructured p2p networks," Proc. Of the 1st Int. Conf. Contemporary Computing, August 2008, pp. 161-169.

[14] S. M. Thampi, and C. K. Sekaran, "An efficient distributed seach technique for unstructured peer-to-peer networks," Int. Jou. Computer and Network Security, vol. 8, no. 1, January 2008, pp. 128-135.

[15] F. Torabmostaedi, and M. R. Meybodi, "An intelligent search algorithm for peer to-peer networks", Int. Conf. Contemporary Issues in Computer and Information Sciences, June 2011, pp. 495-500.

[16] M. Ghorbani, A. M. Saghiri, and M. R. Meybodi, "A novel learning-based search algorithm for unstructured peer-to-peer networks," Technical Journal of Engineering and Applied Sciences, January 2013, vol. 3, no. 2, pp. 145-149.

[17] M. Ghorbani, M. R. Meybodi, and A. M. Saghiri, ''A novel self-adaptive search algorithm for unstructured peer-to-peer Networks utilizing learning automata," Proc. of the 3rd Joint Conference of Robotics & AI and the 5th RoboCup IranOpen International Symposium, April 2013, pp.42-47.

[18] M. Ghorbani, M. R. Meybodi, and A. M. Saghiri, "A new version of k-random walks algorithm in peer-to-peer networks utilizing learning asutomata," Proc. of the 5th Conference on Information and Knowledge Technology, May 2013.

[19] C. J. C. H. Watkins, P. Dayan, Machine learning, vol. 8, Springer, 1992, pp. 279-292.

[20] K. Najim, and A. S. Poznyak, "Learning automata: theory and application," Proc. of the Tarrytown, 1994, New York, Elsevier Science Publishing Ltd.

[21] K. S. Narenda, and M. Thathachar, Learning Automata: an introduction,"New York, Prince-Hall, 1989.

[22] I. Baumgart, and B. Heep, Oversim community site, [Online], Available: http://www.oversim.org/wiki

Mahdi Ghorbani He received his B.Sc. degree in computer science from Shahid Bahonar University of Kerman and M.Sc. degree in computer networks from Qazvin Azad University (QIAU) respectively in 2008 and 2013. He works as the network administrator and the researcher in the state organization of deeds and properties in Iran. His main research area includes peer-to-peer networks, learning systems and complex systems. He has published more than 10 papers in different international journals and conferences such as IEEE Xplore and Springer publications.