7th Iranian Joint Congress on Fuzzy and Intelligent Systems
18th conference on Fuzzy Systems and 16th conference on Intelligent Systems
29 - 31 Jan 2019
University of Bojnord

# An Adaptive Topology Management Algorithm in P2P Networks Based on Learning Automata

**Mahdi Ghorbani***†, Faculty of Computer and Information Technology Engineering, Qazvin Branch, Islamic Azad University, Qazvin,
**Mohammad Reza Meybodi**, Dept. of Computer Engineering, Amirkabir University of Technology, Tehran, Iran
**Ali Mohammad Saghiri**, School of Computer Science, Institute for Research in Fundamental Sciences (IPM), Tehran, Iran

**Abstract:** The topological structure of peer-to-peer networks is one of the topics of interest in these types of networks. Using the concept of community, as a technique for putting together the peers with similar interests, has largely contributed to the topological structure of peer-to-peer networks. A community is created when one or more numbers of a peer claim a similar interest about a common subject. Discovering a community and proposing it to a peer regardless of the priority of the keywords in the vector of interests leads to surplus connections and higher traffic in the network. Hence, in this paper it is tried to, unlike previous studies and using learning automata, prioritize the interests of a peer in the vector of interest and the peer chooses more accurate communities for the sake of its own durability. Using this method, one node could choose those communities suggested to it with members with more similar interest. By implementing the interest-based searching method on the network obtained through the proposed algorithm, the search overhead and the success rate are investigated and the obtained results proved our claim.

**Keywords:** Structuring, Community, Peer-to-peer networks, Learning automata.

## 1 Introduction

One of the topological structuring methods in peer-to-peer networks is the interest-based classification method [2,5,10,11]. The vector of interests is a set of keywords obtained from the users' inquiries. This vector leads to the classification of the users and consequently the creation of the network. In this method, due to the implicit sharing of the users' search results, the response time to the inquiries decreases [8]. One of the fundamental challenges in the topological structuring of this type of networks is not paying attention to the prioritization of the keywords in the vector of interest. This leads to the incorrect classifications. The incorrect classification leads to surplus connections and high traffic in the network. Different methods are studied for the structuring of the networks with a focus on clustering the classification of the nodes. A number of these methods are applied to peer-to-peer networks as well. In what follows, some of these methods are introduced: In [4], Caronni proposes a dynamic classification of the network nodes with

*Corresponding Author
†mahdigh13@yahoo.com

variable sizes. These groups are generated randomly by the group manager who is responsible for the receipt and processing of the incoming and outgoing inquiries from the nodes. The parameters of the group are published through in network using a central service provider and particular nodes use multicasting and broadcasting methods for sending the messages in the network. In [7], Lupu and Keoh considered an ad hoc network tool as a community. This approach specifies doctrines, regulations and authorizations on the specifications of a community by which the users should act in the community. Hong proposes a dynamic structuring and classification technique in which the nodes are collected based on their displacement correlations. These groups are generated dynamically or they are separated and the members of the group could enter or exit the network randomly [6,9]. In [8], Khambatti proposes the creation of the communities using the classification based on the users' vector of interest. In this method, a network is divided into different groups. Each group has a specific vector of interest and the nodes with similar interest are placed in a community of nodes. The nodes implicitly enter or exit the network without the need for the group manager and could find their membership effective in their community. The creation of these communities, affect the global traffic of the network.

The organization of the paper is as follows: In section 2, the learning automata is described as the main learning strategy in the proposed algorithm. The proposed algorithm and the simulations results are presented in sections 3 and 4, respectively. In the final section, the conclusion is provided.

## 2 Overview of Learning Automata Theory

Learning automata [1,11], is a machine that can learn appropriate action from a finite set of actions in an unknown environment. The environment evaluates the selected action. The outcome of evaluation is given to the automata in a reward or punishment signal. The answer of the environment will affects on the learning automata to select the next action. The goal of the learning automaton is to learn to choose the best action from its own actions. The best action is the one that maximizes the probability of receiving a reward signal from the environment. The relationship between the learning automata and the random environment is shown in figure 1.

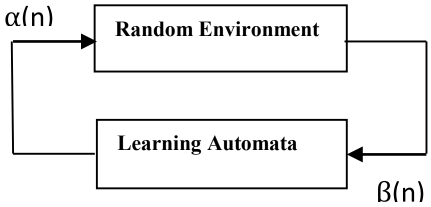

Fig 1. The relationship between the learning automata and its random environment [9].

Environment can be shown by E≡ $\{\alpha, \beta, c\}$ in which $\alpha \equiv \{\alpha_1, \alpha_2,.,\alpha_r\}$ is the Set of inputs, $\beta \equiv \{\beta_1, \beta_2,\ldots\ldots, \beta_m\}$ Set of outputs of the environment and $c \equiv \{c_1, c_2, \ldots., c_r\}$ is the set of possibilities of receiving punishment signals. When $\beta_i$ has two values, the environment is called the P-type. In this environment, $\beta_i(n) = 1$ is assumed as a punishment signal or failure and $\beta_i(n) = 0$, as a reward signal or success. In an environment of Q, $\beta_i$ can discretely get one of the limited values in the interval [0,1]. In the S Model, $\beta_i$ is a random variable between zero and one $\beta_i(n) \epsilon [0,1]$. In S model with r action, if the action is selected in n-th iteration step and the environment's answer to that is the vector of automata probabilities will be updated according to the following equation.

$$p_j(n+1) = \begin{cases} p_j(n) + a[1 - p_j(n)] & j = i \\ (1-a)p_j(n) & \forall j, \quad j \neq i \end{cases} \tag{1}$$

$$p_j(n+1) = \begin{cases} (1-b)p_j(n) & j = i \\ (\dfrac{b}{r-1}) + (1-b)p_j(n) & \forall j, \quad j \neq i \end{cases} \tag{2}$$

If $a = b$, the recurrence (1) and (2) are called linear reward penalty ($L_{RP}$) algorithm, if $a \gg b$ the given equations are called linear reward-ε penalty ($L_{R\varepsilon P}$), and finally if $b = 0$, they are called linear reward-Inaction ($L_{RI}$).

## 3 Proposed Algorithm

In [8], Khambatti introduces communities for the membership of each peer per each time the algorithm is executed. After the execution of the community discovery algorithm, each peer should add the newly proposed community to its communities proposed in the previous executions. Since in this method, the corresponding peer is not able to decide accurately on the membership in the proposed community, surplus links are created and the traffic of the network increases. Therefore, using learning automata, the structure of each peer is changed to make better decision for a more proper community.

In this paper, S-model environment is used and $L_{RP}$ is the learning automata. First, the structure of the proposed algorithm is described and then, the algorithm will be explained. As mentioned before, in the learning automata, the possible values which are updated given the feedback by the environment, are very important for making decisions.

In order to employ the learning automata in the proposed algorithm, two simple definitions are required. Assume the current peer for which communities are proposed is denoted by CN in equation 3 and the vector of probabilities for each peer is represented according to equation 4.

$C = \{C_1, C_2, \ldots | C_i$ *is* $i_{th}$ *recommanded community for CN,* $i=1,2,\ldots\}$ (3)

$P_{vector} = \{P_{1i}, P_{2i}, \ldots, P_{ri} | r$ *is the number of actions for* $C_i$ $, \sum P_{ki}=1\}$ (4)

In the proposed algorithm, each peer uses a learning automata which store a table of the proposed communities in its column denoted by AC. In each row of this table, the communities ID and the corresponding vector of probabilities ($P_{vector}$) are stored. As shown in figure 2, the vector of probabilities associated with each community has two possible values of $P_{1i}$ and $P_{2i}$, defined as follows:

$P_{1i}$ is the possible value for the membership of the corresponding peer in the $i_{th}$ community which is based on the previous stages of communities discovery, while $P_{2i}$ is the possible value for the peer not being a member of the $i_{th}$ community in the previous stages of the community discovery.

At the beginning of the implementation of the algorithm, the values of these probability vectors are considered to be 0.5. Each time the community discovery algorithm runs, a row might be added to the table.

|  | *AC* |
|---|---|
| $C_1$ | $\{P_{11}, P_{21}\}$ |
| $C_2$ | $\{P_{12}, P_{22}\}$ |
| $C_3$ | $\{P_{13}, P_{23}\}$ |
| ⋮ |  |
| $C_r$ | $\{P_{1r}, P_{2r}\}$ |

Fig 2. A simple format of AC

Reward and penalty for each community is updated through equations 5 and 6.

$$\beta=0 \begin{cases} p_1(n+1) = p_1(n) + a\,.\,(1 - p_1(n)) \\ p_2(n+1) = p_2(n)\,.\,(1-a) \end{cases} \quad (5)$$

$$\beta=1 \begin{cases} p_1(n+1) = p_1(n) \\ p_2(n+1) = p_2(n) \end{cases} \quad (6)$$

In order to train the peers, being or not being a member of a community is of great importance considering the existing records in the peers table as well as the activity of the proposed community. The activity of the community means whether the community still exists in the peers table.

In the proposed algorithm, one could find out about the membership of the peer in a community by examining the probability associated with each community in the peers table. As stated before, in the first execution of the community discovery algorithm, the values of both components of the probability vector is 0.5 for each community. After running the algorithm, the value of the first component is evaluated and if values higher than 0.5 are observed, this means that the corresponding peer belongs to that community. Otherwise, the peer does not belong to that community.

In order to reorganize the communities, the peer broadcasts a message to a randomly selected number of the community members which it has stored in its table. Each of these communities which still exist in the network sends an acknowledgment. Otherwise, a message implying its nonexistence in the communities of the network is issued. Figure 3 illustrates the pseudo code associated with the proposed algorithm for the prioritization of the vector of interests of a peer.

```
--------------------------------------------------------------------
/*execute for each recommended group*/
    1. The new group is added to peer's history set and
       create a LA with two actions
    2. All the groups' LAs make decision whether the
       peer is a member or not
            If the p1-value is >= 0.5 then
                        IsMember
                    Else
                        IsNotMember
    3. The peer broadcast a message to a random
       selected member of each group
            If ack is received then
                        IsAlive
                    Else
                        IsNotAlive
--------------------------------------------------------------------
```

Fig 3. Proposed Algorithm

After identifying the membership parameters and the activity of the proposed community, the value of the error of the membership of the peer in the communities is reduced for the next executions given the generated probability values. Figure 4 illustrates the pseudo code associated with how the reward and penalty are assigned to the peer.

```
--------------------------------------------------------------------
for all groups in peer's history, P-vector is updated as
follows:

  If IsAlive and IsMember           then  eq (5)   //reward
  If IsAlive and IsNotMember        then  eq (6)   //penalty
  If IsNotAlive and IsMember        then  eq (6)   //penalty
  If IsNotAlive and IsNotMember     then  eq (5)   //reward
--------------------------------------------------------------------
```

Fig 4. Updating $P_{vectors}$ according to feedbacks

## 4 Simulation

Oversim simulator [11] is used for the simulation of the proposed community discovery algorithm. It is worth noting that in the drawn diagrams, the proposed algorithm called LATS and the algorithm to be compared called khambatti_alg. are pointed out. Table 1 briefly shows the simulation parameters with default values.

| Parameters | Default Values |
|---|---|
| Network Topology | Random Graph |
| Network Type | Unstructured |
| Num. of Nodes | 1000 |
| Max. Num. of Attributes | 20 |
| Outlink Threshold | 40 % |
| Num. of Objects | 100 |

Table 1. Simulation Parameters

### 4.1 Experimental Evaluation

In this section, the efficiency of the proposed algorithm is examined through different experiments. First, two methods of community discovery and the amount of communities selected by each peer to become a member of them are compared and then, CBS method is used to examine the generated overhead and success rate for the search in the discovered communities through the two above mentioned methods.

### 4.1.1 Community Discovery

As shown in figure 5, the community discovery percentage through LATS increases after its execution compared to the other method. This increase in the community discovery is due to the more accurate selection of the community by the peer given the calculations of the probabilities in the previous stages of the execution of the algorithm. The community discovery rate was 82% with LATS, while this value was 62% for khambatti_alg.

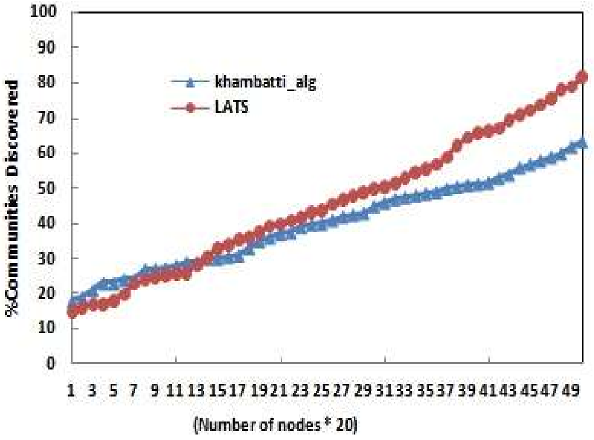

Fig 5. Rate of Communities Discovered

### 4.1.2 Overhead

The overhead in this experiment is based on the calculation of the generated messages during the implementation of CBS searching algorithm. By running CBS algorithm on the generated networks, the generated overhead after running the algorithm for 10 times is illustrated in figure 6. This diagram shows the number of generated messages for a maximum number of 50 inquiries for each peer. As observed, the overhead due to the execution of the searching algorithm on the generated network is much lower for LATS compared to that of khambatti_alg since the communities in which the nodes become a member of in the proposed method are more likely to respond to the searching inquiries. This is due to the prioritization of the vector of interests in the community discovery.

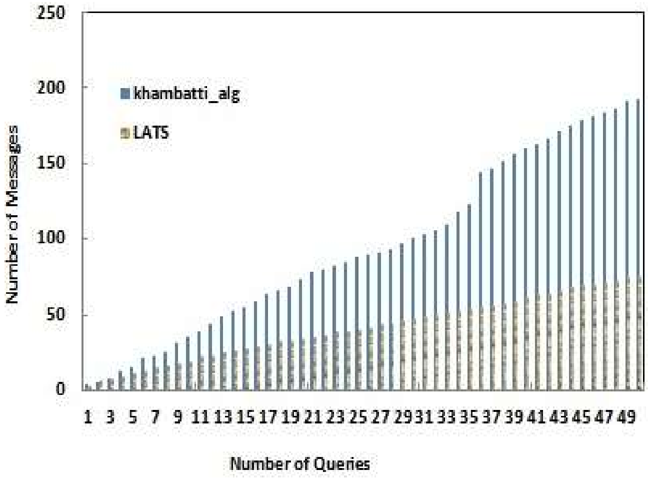

Fig 6. Overhead of CBS on two generated topologies

### 4.1.3 Success Rate

The success rate is illustrated in figure 7 after running the searching algorithm for 10 times. As observed, due to the smart selection of the communities by the corresponding peer through previous stages with higher accuracy and probability, the searching success rate in LATS is much higher than khambatti_alg. In LATS, the success rate shows an increasing growth after the tenth inquiry and in the last inquiry, the success value has reached 93.4% while this value reaches 69% in the 50th inquiry for khambatti_alg method.

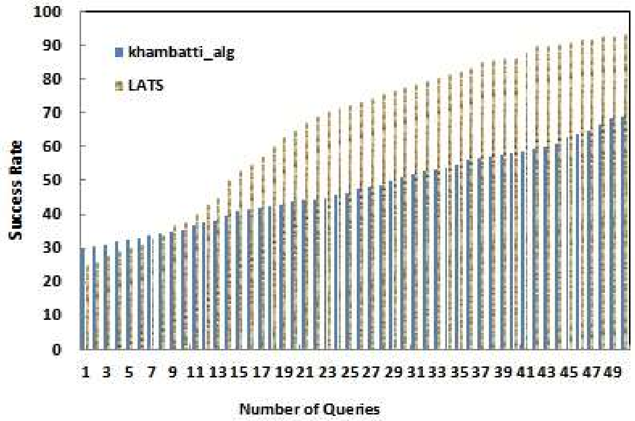

Fig 7. Success rate of CBS on two generated topologies

## 5 Conclusion

The concept of community in peer-to-peer networks is a method for collecting the peers based on their interests. Identifying which proposed community a peer tends to become a member of is one of the important aspects in the more accurate formation of the communities. In this paper, each peer has a set of communities which are previously proposed to it. Each peer uses the learning automata to make decision regarding its membership in the proposed community at different times i.e. the interests of a peer are prioritized in the vector of interest and a peer increases its membership sustainability given the feedbacks received by the proposed communities. The simulation results show that using the community based searching method, the search overhead and the search success rate obtained through our proposed method are much higher than those of the other method. This is due to the accuracy in the smart selection of the communities by the peer using the learning automata.

## References

[1] S. M. Abolhasani and M. M. Meybodi, LADIT: Learning Automata Based Protocol for Routing in Sensor Networks, 2th Conference on Sensor Networks, Yazd, (2008), 20-33.

[2] S. Ashraf Khan and L. N. Tokarchuk, Interst-based Self Oraganization in Group-Structured P2P Networks, IEEE international conference on Computer Networks, (37) (2009), 60-67.

[3] I. Baumgart, and B. Heep, Oversim community site, Available:

http://www.oversim.org/wiki

[4] G. Caronni, M. Waldvogel, D. Sun, and B. Plattner, Efficient security for Large and Dynamic Multicast Groups, IEEE 7th International Workshop on Enabling Technologies, (1998), 376-383.

[5] L. Festinger, Laboratory Experiments: The Role of Group Belongingness, *in J. G. Miller, ed., Experiments in Social Process, 1950.*

[6] X. Hong and M. Gerla, Dynamic Group Discovery and Routing in Ad Hoc Networks, Proceedings of the 1st Annual Mediterranean Ad Hoc Networking Workshop, Sardinia, Italy, (2002).

[7] S. L. Keoh and E. Lupu, Trust and the establishment of ad-hoc communities, 2nd Internal iTrustWorkshop on Trust Management in Dynamic Open Systems, London, UK , (2003).

[8] M. Khambatti, D. Ryu and P. Dasgupta, Structuring Peer-to-Peer Networks Using Interest-Based Communities, 2004, The Springer Link website.Available: http://www.springerlink.com/content/ja62jfgqu6kxa1xr/fulltext.pdf.

[9] A. Meissner and S. B. Musunoori, Group Integrity Management Support for Mobile Ad-Hoc Communities, Workshop proceedings Middleware for Pervasive and Ad-Hoc Computing, Rio de Janeiro, Brazil, (2003), 53-59.

[10] H. A. Murray, Explorations in personality, 1938.

[11] K. Najim and A. S. Poznyak, Learning automata: theory and application, in Proceeding of the Tarrytown, New York, Elsevier Science Publishing Ltd , (1997).