Distributed Learning Automata-based Algorithm for Finding K-Clique in Complex Social Networks

Mohammad Mehdi Daliri Khomami
Department of Computer Engineering, Amirkabir University of Technology, Tehran, Iran
m.daliri@aut.ac.ir

Alireza Rezvanian
Department of Computer Engineering
University of Science and Culture, Tehran, Iran
rezvanian@usc.ac.ir

Ali Mohammad Saghiri
Department of Computer Engineering, Amirkabir University of Technology, Tehran, Iran
a_m_saghiri@aut.ac.ir

Mohammad Reza Meybodi
Department of Computer Engineering, Amirkabir University of Technology, Tehran, Iran
mmeybodi@aut.ac.ir

Abstract— **Maximal clique finding is a fundamental problem in graph theory and has been broadly investigated. However, maximal clique finding is time-consuming due to its nature and always returns tremendous cliques with large overlap nodes. In this paper, we have focused on the relaxed version of the clique called k-clique, which follows up the subset of vertices with size k such that each pair in this subset has an edge. The k-clique problem has several applications in different domains, such as motif detection, finding anomalies in large graphs, and community structure discovery. In this paper, an algorithm based on learning automata is proposed for finding k-clique called (KC-LA) to apply communities in complex social networks. In (KC-LA), a network of learning automata is considering to the underlying networks. Then, select the proper action from a set of allowable actions, the reward and penalty guide KC-LA to detect the k-clique. Also, we applied the k-clique in terms of finding communities in complex social networks. To show the algorithm's effectiveness, several experiments have been designed on real graphs and synthetic graphs, and the results demonstrate the high efficiency and effectiveness of the algorithm.**

Keywords—Complex Social Network, K-Clique, Clustering, Learning Automaton.

I. INTRODUCTION

Maximal clique finding is a crucial graph problem noticed in many domains of real applications such as protein-protein interactions, regular motif, link spam detection, and complex social systems. In a graph, a clique looking for a subset of nodes such that each node is connected pairwisely. A clique with maximum cardinality is called the maximum clique. Maximum clique finding has been extensively noticed in the literature, especially for finding communities in complex social networks [1]–[3]. Community detection in networks points out to clustering the nodes into clusters such that the connections within clusters are dense, but between them are sparse. [4]. The maximum clique finding is NP-hard, due to the clique's size can exponentially grow by the number of nodes in the graph [5].

The complexity of finding maximum clique as a possible solution is k-clique, following the clique with size k instead of finding the largest clique among all cliques. Like the clique problem, the k-clique is applied for finding dense subgraphs or communities in social networks. For these reasons, some theoretical and empirical work on detecting k-cliques in social networks [5]. In general, the algorithms proposed for k-clique are classified into three main categories: deterministic, heuristic, and approximation algorithms. The deterministic algorithms generally used the branch and bound technique to solve, but these algorithms are not applicable in real because of high computational complexity. Approximation algorithms provide a bound for optimality of the solutions, and the heuristic algorithm produces a feasible solution in a reasonable time. Since clique structure reflects a strict kind of community structure in networks, some researchers attempted to utilize the concept's clique for community detection [6].

In [7], Adamcsek et al. proposed a fast and scalable C-Finder algorithm to detect communities in the networks. Kumpula et al. [8] have run the clique percolation algorithm sequentially to enhance community detection. However, their algorithm improves poor results on networks on networks due to the existence of overlapping structures. In [9] defined the k-clique community first, and then the founded K-clique is studied as the communities. Saito et al. [10] introduced a new concept called k-dense communities, which selected a subset of densely connected nodes and proposed an algorithm for finding dense communities in the networks. Duan et al. [11] applied k-clique as k-clustering with the combination of DFS search in the context of dynamic social networks to reveal dynamic communities. They also proved theoretically that the algorithm is superior compared to spectral clustering in terms of time complexity. Hao et al. [12] proposed an intelligence computational technique FCA to find cliques and, with the aid of cliques, detect communities in social networks. The FCA utilizes formal context to create a modified adjacency matrix initially. Then, by introducing a new notion called k-equiconcept, they proved that the k-clique detection problem is equivalent to finding the k-equiconcepts. Besides, their algorithm is applied to find communities in the networks.

In recent years, several learning automata-based algorithms are presented to solve the graph problem. Some successful algorithms using learning automata have been reported in the literature: positive influence dominating set, independent set, vertex cover, and community detection.

This paper provides a new distributed learning automata-based algorithm to solve k-clique called (KC-LA) in social networks. In the KC-LA algorithm, a network of LAs is mapped to a given graph. Then, by collaboration between LAs and learning mechanisms for each LA, the algorithm can guide solutions.

The rest of the paper is organized as follows: In section 2, learning automata and distributed learning automata are introduced briefly. The proposed algorithm based on distributed learning automata for finding k-clique is discussed in section 3. The performance of the algorithm is evaluated

978-0-7381-4309-5/20/$31.00 ©2020 IEEE

through the simulation in section 4. Furthermore, section 5 concludes the paper.

II. LEARNING AUTOMATA

Learning automaton (LA) is a reinforcement learning model. This model can find an appropriate action from an action set during repeated interaction with a random environment [13]. LAs can be classified into two classes: Variable Structure Learning Automata (VSLAs) and fixed structure Learning Automata (FSLAs). Figure 1 shows the relationship between the LA and its random environment.

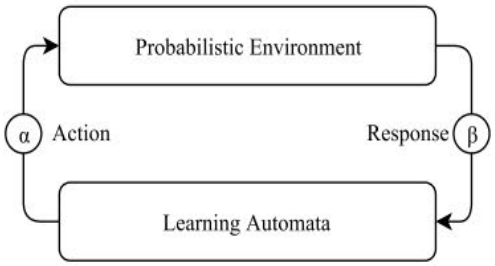

Fig. 1. The relationship between the LA and its probabilistic environment

VSLAs can be described with quadruple $\{\alpha, \beta, P, T\}$ where

- $\alpha = \{\alpha_1, \dots, \alpha_r\}$ denotes the finite set of actions
- $\beta = \{\beta_1, \dots, \beta_m\}$ denotes the set of input values that can be taken by reinforcement signal
- $P = \{p_1, \dots, p_r\}$ denotes the probability vector of the action set.
- T is a learning algorithm where $P(n+1) = T[\alpha(n), \beta(n), p(n)]$.

Equations (1) and (2) illustrate changes in the probability vector in each step. LA updates its action probability vector based on equation (1) for favorable responses as:

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

And equation (2) for unfavorable ones:

$$p_i(n+1) = (1-a)p_i(n)$$
$$p_j(n+1) = \left(\frac{b}{r-1}\right) + (1-b)p_j(n) \qquad (2)$$

Where r denotes the number of actions, the learning rates a and b denotes the reward and penalty parameters. If $a = b$ learning algorithm is called linear reward penalty (L_{R-p}), if $b \ll a$ given learning algorithm is called linear reward-ε penalty ($L_{R-\varepsilon p}$), and finally, if $b = 0$ it is called linear reward inaction (L_{R-I}) [14], [15].

A. Distributed Learning Automata

A learning automaton is a simple agent for doing simple things. The full potential of learning automata is realized when they are interconnected to cooperate. A distributed learning automaton is a network of learning automata that cooperate to solve a particular problem. In this work, just a learning automaton is activated. The number of actions performed by automata is equal to the number of learning automata connected to it. By choosing an action of automata in this network, the connected automata on the other side will be activated. The model of DLA denotes a graph in each vertex is an automaton [16].

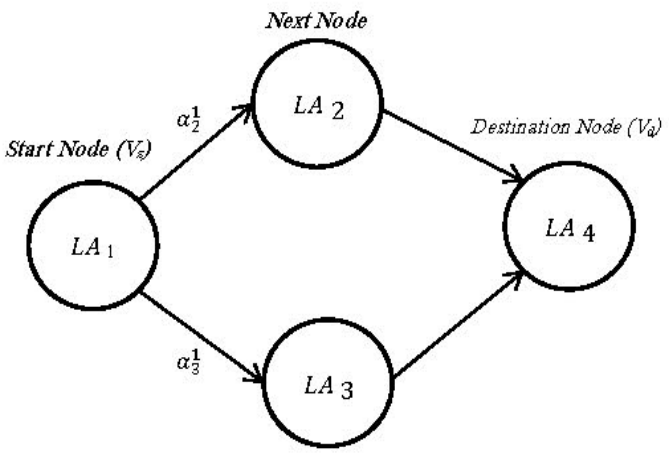

Fig. 2. Component of distributed learning automata.

The connection between LA_i and LA_j in the networks means, selecting action α_j^i cause activation of LA_j by LA_i.

III. PROBLEM DEFINITION AND PROPOSED ALGORITHM

This section provides the preliminaries related to KC-LA for finding the k-clique problem. Before presenting the KC-LA algorithm, we first define some necessary definitions and notions used in this section.

A. Social networs, Clique and K-clique

Definition 1: A social graph G is defined by a couple: $G = (V, E)$ where $V = \{v_1, \dots, v_n\}$is a set of nodes and edge set $E = \{uv | u, v \in V\}$ denotes the connection between entities.

Definition 2: A clique is a subset $S \subset V$ such that any pair of nodes $v_i, v_j \in S$, there exist an edge $(v_i, v_j) \in E$.

Definition 3: k-clique is defined as a subset $S \subset V$ and $|S| = k$ such that any pair of nodes $v_i, v_j \in S$, there exist an edge $(v_i, v_j) \in E$.

B. *KC-LA based for k-Clique*

This section represents the detail of the KC-LA algorithm for finding k-clique in a social graph. For this goal, the input graph G and parameter k is given as input for the KC-LA algorithm. The algorithm returns the k-clique of the graph.

Initially, a network of LA is mapped by assigning automata to each vertex. The set of actions taken by each LA corresponds to the edges connected to the vertex of the graph. Let LA_i is selected and activated automata and $\{LA_j | j \in N(LA_i)\}$ indicate the set of action which is taken by the LA_i and $N(LA_i)$ is the neighborhood nodes of LA.

Let it is supposed that all learning automata which are mapped to the network are disabled initially. At first, to starting the algorithm, a learning automaton is selected randomly and activated. The selected automata added to the k-clique set and select one of its actions. Let the set of activated automata kept the candidate nodes as k-clique, which is called K-Set.

Let the selected action by automaton LA_i be LA_j, the algorithm checks whether adding learning automaton LA_j to K-Set then LA_j is added; otherwise, the algorithm rescaled the action set to select and formed the proper actions. After action selection, the activated automata disables the activated

automaton belonging to K-Set. The process of action selection and checking the selected action proceeds until there are no automata for activation.

When all learning automata are disabled, the cardinality of K-Set is evaluated with the cardinality of the optimal set, which is set empty initially. If the cardinality of K-Set is greater than the optimal set, the selected action by the learning automata is reworded; else, the selected action is penalized. Moreover, if the K-Set's cardinality is greater than the k-clique size, the algorithm greedily removes the nodes to reach the clique with size k.

One of the essential parts of the learning automata-based algorithm is how to select an action properly. Let $\alpha = \{\alpha_1, \alpha_2, \dots, \alpha_r\}$ be the action set of automaton LA_i and $p = \{p_1, p_2, \dots, p_r\}$ corresponds to the probability of selecting each action from the action set. Let suppose that the activated automata select an action from the available action set and $\eta = \{\eta_1, \eta_2, \dots, \eta_k\} \subseteq \alpha$ indicate the set of allowable action which is taken by the automata. A common choice to select action is formulated in equation 3.

$$\bar{q}_i = \frac{q_i}{\sum_{j=1..k} q_j} \tag{3}$$

where $\bar{q} = \{\bar{q}_1, \bar{q}_2, \dots, \bar{q}_k\}$ is the probability of selecting each action. However, the selection may not properly guide the learning automata for finding the solution. Another policy used in this paper is to modify action probability instead of using the actual action probability vector described. Let $\{v_1, v_2, \dots, v_k\}$ indicates a possible action that can be selected by η. The modified probability action vector $\overline{mq} = \{\overline{mq}_1, \overline{mq}_2, \dots, \overline{mq}_k\}$ is defined as:

$$\overline{mq}_i = \frac{\left(\frac{\bar{q}_i}{n-1-deg_G(v_i)}\right)^{\beta}}{\sum_{j=1,\dots,k}\left(\frac{\bar{q}_j}{n-1-deg_G(v_j)}\right)^{\beta}} \tag{4}$$

where $deg_G(v_i)$ indicate as the degree of v_i and $\beta > 1$ is a normalized factor. These strategies, by considering the degree of nodes and selecting nodes with high degree and high probability, the chance of choosing these nodes is increasing, and the chance of choosing nodes with a lower degree decreases. The pseudo-code of the algorithm is directed in figure 3.

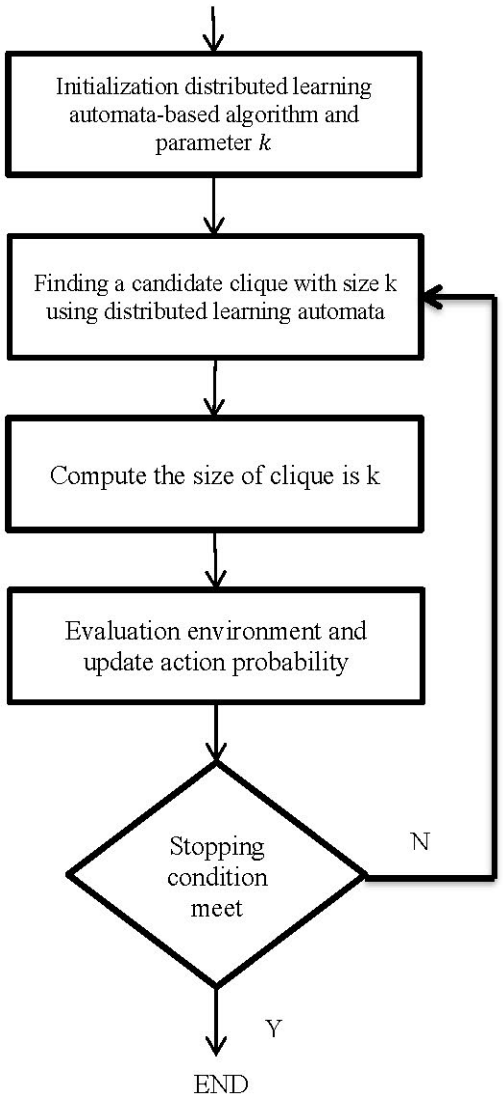

Fig. 3. The flowchart of the proposed algorithm

IV. Simulation Results

For evaluation of the proposed algorithm, experiments are applied to some popular social networks. The description of popular social networks [17], [18] is listed in Table 1. We note that we have used the L_{R-I} learning algorithm with a 0.02 learning rate for the experiments.

TABLE I. Specification of the datasets for the experiments

	Data sets	Nodes	Edges
I	*Karate*	*34*	*78*
II	*Dolphins*	*62*	*159*
III	*Jazz*	*198*	*2700*
IV	*Yeast*	*1500*	*1900*

The topology of Zachary's karate club and dolphin network is presented in figures 4 and 5, the most popular social network datasets.

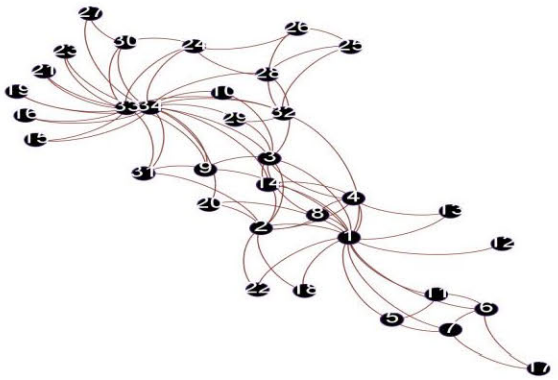

Fig. 4. Zachary's Karate Club social networks

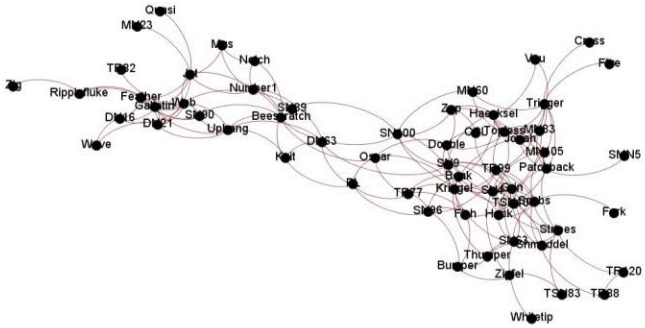

Fig. 5. Bottlenose Dolphins social networks

We compare and run our algorithm on a computer with 2.5 GHz Intel Core I 5 CPU and 4G memory running on Windows 7 Operating system. The experimental result is compared with existing works CPM [7], GN [19], and CDPM [20], respectively. Also, we have used the k-Clique problem in the application of community detection for detecting real communities.

A. Evaluation Measure

The F-measure is used to evaluate how well an algorithm can detect the k-clique from a complex social network. F-measure is calculated as follows:

$$F - measure = \frac{2 \times recall \times precision}{recall + precision} \quad (4)$$

where *recall* indicate the fraction of nodes pair pairs belonging to the same k-clique, which are also in the same community, and precision is the fraction of node pairs in the same community, which are also present in the same k-community.

B. Experiments

The experiment's goal is to find the k-clique for different parameters of k, which is varying from 2 to 7 with a step length of one. To perform this experiment, we plot the different values of k versus the algorithm's running time. The obtained result is depicted in figure 6. From the result, we may conclude that the proposed algorithm detects the clique with different size properly. Moreover, the algorithm's execution time is very low for small networks, and other networks are negligible.

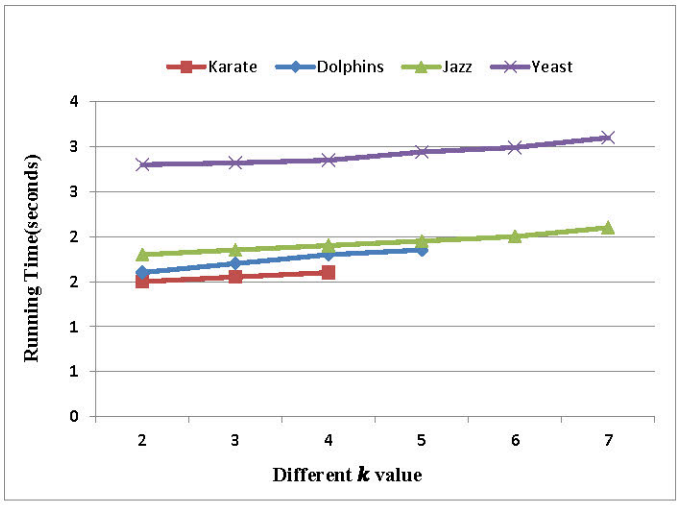

Fig. 6. Consuming Time for different networks

This experiment is conducted to evaluate the proposed algorithm's performance in terms of F-measure value compared to other algorithms. As shown in figure 7, we may conclude that the DLA-based algorithm outperforms other algorithms in all networks due to searching for finding communities. In other words, the algorithms, with the aid of learning mechanisms and interaction with the external environments, try to find different cliques as communities. Therefore the algorithm can reveal network cliques as communities.

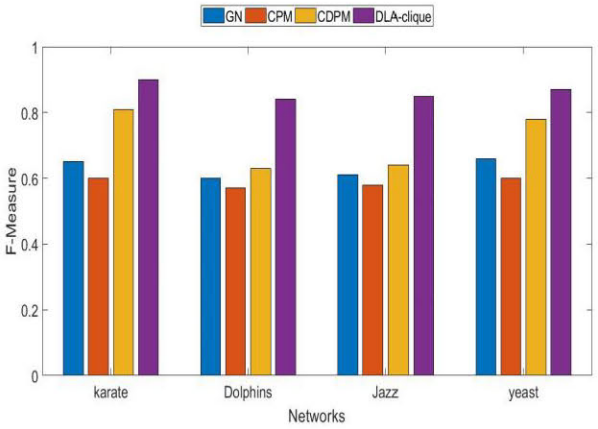

Fig. 7. Comparing the different algorithms in terms of F-measure value for evaluation of the performance on four data sets.

V. Conclusion

In this study, an algorithm called KC-LA based on distributed learning automata for finding the k-clique problem is proposed. In KC-LA, the network of learning automata is mapped to the network. Then, the algorithm selects a node randomly, and the corresponding LA is activated. The activated LA is select as an action and added to the candidate clique set. This procedure is continued until there is no activated automaton or the clique's size reaches k. To show the effectiveness of the algorithm, several experiments have been conducted on well-known datasets. Simulation result on popular social networks datasets shows the proposed algorithm's effectiveness in terms of evaluation measure.

VI. References

[1] E. A. Akkoyunlu, "The enumeration of maximal cliques of large graphs," *SIAM Journal on Computing*, vol. 2, no. 1, pp. 1–6, 1973.

[2] M. Regneri, "Finding all cliques of an undirected graph," 2007.

[3] A. Rezvanian and M. R. Meybodi, "Finding maximum clique in stochastic graphs using distributed learning automata," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 23, no. 01, pp. 1–31, 2015.

[4] E. Tomita, A. Tanaka, and H. Takahashi, "The worst-case time complexity for generating all maximal cliques and computational experiments," *Theoretical computer science*, vol. 363, no. 1, pp. 28–42, 2006.

[5] L. Engebretsen and J. Holmerin, "Clique is hard to approximate within n 1-o (1)," in *International Colloquium on Automata, Languages, and Programming*, 2000, pp. 2–12.

[6] E. Gregori, L. Lenzini, and S. Mainardi, "Parallel k-clique community detection on large-scale networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 8, pp. 1651–1660, 2012.

[7] B. Adamcsek, G. Palla, I. J. Farkas, I. Derényi, and T. Vicsek, "CFinder: locating cliques and overlapping modules in biological networks," *Bioinformatics*, vol. 22, no. 8, pp. 1021–1023, 2006.

[8] J. M. Kumpula, M. Kivelä, K. Kaski, and J. Saramäki, "Sequential algorithm for fast clique percolation," *Physical Review E*, vol. 78, no. 2, p. 026109, 2008.

[9] G. Palla, I. Derényi, I. Farkas, and T. Vicsek, "Uncovering the overlapping community structure of complex networks in nature and society," *nature*, vol. 435, no. 7043, p. 814, 2005.

[10] K. Saito, T. Yamada, and K. Kazama, "The k-dense method to extract communities from complex networks," in *Mining complex data*, Springer, 2009, pp. 243–257.

[11] D. Duan, Y. Li, R. Li, and Z. Lu, "Incremental K-clique clustering in dynamic social networks," *Artificial Intelligence Review*, vol. 38, no. 2, pp. 129–147, 2012.

[12] F. Hao, G. Min, Z. Pei, D.-S. Park, and L. T. Yang, "$ K $-clique community detection in social networks based on formal concept analysis," *IEEE Systems Journal*, vol. 11, no. 1, pp. 250–259, 2015.

[13] K. S. Narendra and M. A. Thathachar, *Learning automata: an introduction*. Courier Corporation, 2012.

[14] A. Rezvanian, A. M. Saghiri, S. M. Vahidipour, M. Esnaashari, and M. R. Meybodi, "Learning Automata Theory," in *Recent Advances in Learning Automata*, Springer, 2018, pp. 3–19.

[15] R. Vafashoar, H. Morshedlou, A. Rezvanian, and M. R. Meybodi, *Cellular Learning Automata: Theory and Applications*. Springer, 2021.

[16] H. Beigy and M. R. Meybodi, "A New Distributed Learning Automata Based Algorithm For Solving Stochastic Shortest Path Problem.," in *JCIS*, 2002, pp. 339–343.

[17] M. M. Daliri Khomami, A. Rezvanian, A. M. Saghiri, and M. R. Meybodi, "SIG-CLA: A Significant Community Detection based on Cellular Learning Automata," in *2020 8th Iranian Joint Congress on Fuzzy and intelligent Systems (CFIS)*, 2020, pp. 039–044.

[18] M. M. Daliri Khomami, A. Rezvanian, A. M. Saghiri, and M. R. Meybodi, "Utilizing Cellular Learning Automata for Finding Communities in Weighted Networks," in *2020 6th International Conference on Web Research (ICWR)*, 2020, pp. 325–329.

[19] M. E. Newman and M. Girvan, "Finding and evaluating community structure in networks," *Physical review E*, vol. 69, no. 2, p. 026113, 2004.

[20] L. Wan, J. Liao, and X. Zhu, "Cdpm: Finding and evaluating community structure in social networks," in *International Conference on Advanced Data Mining and Applications*, 2008, pp. 620–627.