Stochastic Graph as a Model for Social Networks

Alireza Rezvanian, Mohammad Reza Meybodi

Soft computing laboratory, Computer Engineering and Information Technology Department
Amirkabir University of Technology (Tehran Polytechnic), Hafez Ave., 424, Tehran, Iran

Abstract

Social networks are usually modeled and represented as deterministic graphs with a set of nodes as users and edges as connection between users of networks. Due to the uncertain and dynamic nature of user behavior and human activities in social networks, their structural and behavioral parameters are time varying parameters and for this reason using deterministic graphs for modeling and analysis of behavior of users may not be appropriate. In this paper, we propose that stochastic graphs, in which weights associated with edges are random variables, may be a better candidate as a graph model for social network analysis. Thus, we first propose generalization of some network measures for stochastic graphs and then propose six learning automata based algorithms for calculating these measures under the situation that the probability distribution functions of the edge weights of the graph are unknown. Simulations on different synthetic stochastic graphs for calculating the network measures using the proposed algorithms show that in order to obtain good estimates for the network measures, the required number of samples taken from edges of the graph is significantly lower than that of standard sampling method aims to analysis of human behavior in online social networks.

Keywords: complex social networks, social network analysis, user behavior, stochastic graphs, network measures.

1. Introduction

In recent years, online social networks such as *Facebook* and *Twitter* have provided simple facilities for online users to generate and share a variety of information about users' daily life, activities, events, news and more information about their real worlds which results in the online users have become the main features of online social networks and studying how users behave and interact with their friends in online social networks play a significant role for analysis of online social networks. Online social networks similar to many real world networks are usually modeled and represented as deterministic graphs with a set of nodes as users and edges as connection between users of networks. In social networks analysis, when networks are modeled as weighted or unweight deterministic graphs, one can study, characterize and analyze the statistical characteristics and the dynamical behaviors of the network and its user behaviors by some network measures (Costa, Rodrigues, Travieso, & Boas, 2007). Popular network

measures such as degree, betweenness, closeness and clustering coefficient are originally defined for deterministic graphs. These network measures not only used in characterization of networks but also used as a part of some algorithms such as Girvan-Newman community detection algorithm using high betweenness edges (Girvan & Newman, 2002), overlapping community detection using nodes' closeness (Badie, Aleahmad, Asadpour, & Rahgozar, 2013) and finding the outstanding nodes as a set of seed nodes for maximization of the spread of influence in social network aims to introduce and promote a new products, services, innovations or technologies by ranking important humans (Li, Wu, Wang, & Luo, 2014).

Online social networks are intrinsically non-deterministic and their structures or behaviors such as online activities of users have unpredictable, uncertain and time varying nature and for this reason modeling those using deterministic graph models with fixed weights are too restrictive to solve most of the real network problems. For example in online social networks the online activities of users such as friendship behaviors, liking a comment on a given post in *Facebook* and frequency of taking a comment on a wall post vary over time with unknown probabilities (Jin, Chen, Wang, Hui, & Vasilakos, 2013). Analyzing online social networks with deterministic graph models cannot take into consideration the continuum of activities of the users occurring over time. Even modeling social networks with weighted graphs in which the edge weights are assumed to be fixed weights considers only a snapshot of a real-world network and it is not valid when the activities and behavior of users on networks vary with time. Moreover, in analyzing online social networks not only understanding the structure and topology of the network is important but the degree of association among the users in network is also important for analysis of user behaviors in online social networks.

According to the aforementioned points, it seems that stochastic graphs in which weights associated with the edges are random variables is a better candidate as a graph model for real-world network applications with time varying nature. By choosing a stochastic graph as a graph model, every feature, measures and concept of the graph such as path (Beigy & Meybodi, 2006), cover (A. Rezvanian & Meybodi, 2015b), clique (A. Rezvanian & Meybodi, 2015a), and spanning tree (Akbari Torkestani & Meybodi, 2012) should be treated as stochastic features. For example, choosing stochastic graph as the graph model of an online social network and defining community structure in terms of clique, and the associations among the humans within the community as random variables, the concept of stochastic clique may be used to study community structure properties.

In this paper, after a brief overview of recent studies for user behavior and human activities in online social network, we first redefine some network measures for stochastic graphs and then design six algorithms for calculating them under the situation that the probability distribution functions of the weights associated with the edges are unknown. The proposed algorithms for calculating network measures by taking samples from the edges of the stochastic graph try to estimate the distribution of the network measures. The process of sampling from the edges of the graph is guided by the aid of learning automata in such a way that the number of samples needed to be taken from the edges of the stochastic graph for estimating the network measures to be reduced as much as possible. In the proposed algorithms, the guided sampling process implemented by learning automata aims to take more samples from the promising region of the graph, the regions that reflects higher rate of changes (e.g., higher rate of user activities), instead of walking around and taking unnecessary samples from non-promising region of the graph.

In order to study the performance of the proposed algorithms for calculating network measures in stochastic graphs, several experimental studies on different synthetic stochastic graphs are conducted. Experimental results show that in order to obtain good estimates for the network measures the number of samples needed to be taken from the edges of the graph by the proposed guided sampling algorithms is significantly lower than the required number of samples needed to be taken from the edges when standard sampling method is used aims to analysis of human activities and user behaviors in online social networks.

The rest of this paper is organized as follows. Section 2 dedicated to material and methods including brief introduction to some of existing network measures for deterministic networks, an overview of recent works about distribution of user behaviors in online social network studies and a brief description about learning automata theory. In section 3, the proposed network measures for stochastic graphs are described. The proposed algorithms for calculating the proposed network measures in stochastic graphs are described in section 4 and section 5 presents the simulation results. In section 6, discusses about this study and results. Finally, section 6 concludes the paper.

2. Material and methods

In this section, to provide the necessary background, we present a brief description of some network measures for deterministic unweighted and weighted networks. We also briefly review the studies performed by the researchers about the distributions of user behaviors and activities of users in online social networks. At the end of this section, learning automata and variable action set learning automata are introduced.

2.1. Network measures for deterministic networks

Network measures and calculating them play a significant role in social network analysis (Borgatti, 2005). Popular network measures such as degree, betweenness, closeness and clustering coefficient not only used for evaluating the node importance in actual complex network studies but also used as a part of some algorithms such as Girvan-Newman community detection algorithm using betweenness (Girvan & Newman, 2002), overlapping community detection using node closeness (Badie et al., 2013). In this section some of well-known network measures for deterministic networks are introduced.

2.1.1. Degree

Degree as a basic network measure has been widely used in many studies and degree of node v_i defined in binary network (also called unweighted network) as follows

$$k_i = \sum_{j \neq i} a_{ij} \tag{1}$$

where j is the index of all other nodes of graph and a_{ij} is 1 if node v_i is adjacent to node v_j, and 0 otherwise. In other words, degree of node v_i is the number of nodes that directly connected to node v_i. Degree centrality is useful in the context of finding the single human which gets affected by the diffusion of any information in the network. It follows from the fact that the human with high degree centrality has the chance of getting affected from many numbers of sources (Freeman, 1979).

2.1.2. Strength

Node strength of node v_i is defined as the sum of adjacent edge weights for weighted network as follows

$$s_i = \sum_{j \neq i} w_{ij} \tag{2}$$

where w_{ij} is greater than 0 if node v_i is adjacent to node v_j and its value indicates the weight of edge between node v_i and node v_j. Similar to degree, a human with high strength centrality is known as popular human with high strength links to other humans; however a human with high strength may not consist of necessarily the maximum number of friends. Strength centrality is useful in the context of finding a human which gets affected by the amount of spreading of any information in the network (Freeman, 1979).

2.1.3. Closeness

Closeness of a node is the inverse sum of shortest paths to all other nodes from that node and defined for binary network with n nodes as follows

$$c_i = \frac{1}{\sum_{j \neq i} d_{ij}} \tag{3}$$

where d_{ij} is the length of shortest path between node v_i and node v_j. Closeness can be regarded as a measure of how long it will take to spread information from that node to all other nodes sequentially. Since, the spread of information can be modeled by the use of shortest paths, in applications such as spread of information, a human with high closeness centrality can be considered as the central point because that human can spread the information faster than other human with lower closeness centrality (Freeman, 1979).

2.1.4. Betweenness

Betweenness of a node is the number of shortest paths from all nodes to all other nodes of graph that pass through that node. Betweenness of node v_i is defined for binary network as follows

$$b_i = \frac{g_{st}(i)}{g_{st}} \tag{4}$$

where g_{st} is the number of shortest paths between all pair of nodes of source node v_s and destination node v_t in a binary network and $g_{st}(i)$ is the number of those paths that pass through node v_i. Since, a node with high betweenness centrality will typically be the one which acts as a bridge between many pairs of nodes, the betweenness introduced as a measure for quantifying the control of a human on the communication between other humans in a social network. In this conception, humans that have a high probability to occur on a randomly chosen shortest path between two randomly chosen humans have a high betweenness. Betweenness can be regarded as a measure of how to control Information flows over communication links. Also, Information flows could be dominated by humans with high betweenness centrality in a network (Freeman, 1979; Newman, 2005).

2.1.5. Clustering coefficient

The clustering coefficient for node v_i is defined as a proportion of the number of all edges among its neighbors over the possible number of edges between them. The local clustering coefficient for a node v_i in binary network can be defined as follows

$$CC(i) = \frac{1}{k_i(k_i - 1)} \sum_{v_j, v_h \in N_i} a_{ij}.a_{ih}.a_{jh} \quad (5)$$

where k_i is the degree of node v_i, n is the total number of nodes of graph, N_i is the set of nodes v_j adjacent to node v_i and a_{ij} is 1 if node v_i is adjacent to node v_j, and 0 otherwise. The clustering coefficient of a node is a measure of the connectivity among the neighborhood of the node (transitivity of a node). A human with high clustering coefficient indicates the high tendency of that human to form cluster with other humans. In other words, the clustering coefficient measures transitivity of a network. And also, most of the friends of a human with high clustering coefficient can collaborate with one another even if the focal human is removed from the network.

2.2. User behavior in online social networks

Since, generating and sharing information through social networks are done by users; the online users have become the main features of online social networks. In general, activities of users in most online social networks can be included doing the social relationship with their friends, answering an online questionnaire, joining in online communities, participating in an online discussion, posting and sharing new favorite contents (e.g., photos, music, videos, scientific papers, services, and etc.), visiting a user profile and its contents, taking comment or like on a post. Studying how users behave and interact with their friends in online social networks play a significant role in social network analysis for several reasons (Buccafurri, Lax, Nicolazzo, & Nocera, 2015; Shen, Brdiczka, & Ruan, 2013; Zhu, Su, & Kong, 2015). In some fundamental studies, researchers can generate synthetic graphs similar to real networks in order to further simulations and investigation on a number of phenomena using a computer generated graph models in order to exploit the structures and dynamics of networks and their user behaviors in a systematic manner. In viral marketing, companies can exploit user behavior models and then spread or promote their new products or services to predict the user interactions and process of adoption of an idea or spread them. In some knowledge based researches, understanding the patterns of user participations to generate new contents and propagate information on the networks via users is very imperative in order to detect active users from spamming users, attract new users and keep some users, predict the trends of topics in user communities, and perform efficient content management in their basic system requirements. Modeling user behavior and human activities of online social networks in quantitative manner is also important for understanding user interactions and is a challenging task due to the variety of user interactions on the structure of online social networks and abundance of data from some online social networks. In recent years, several research works regarding user behavior modeling, characterizing the distribution of user behaviors and user activities and statistical observations from online users have been reported in the literatures, some of which are pointed out in the rest of this section.

Golder et al. (Golder, Wilkinson, & Huberman, 2007) analyzed the interactions among *Facebook* users. Their observations discovered strong daily and weekly patterns and found that the communication of the users is clustered mostly based on the universities. They reported that the interactions occurred almost exclusively among friends and only small percentages of friends interact with other friends which

results in the distribution of number of messages sent by a user followed a power-law distribution. *Leskovec et al.* (Leskovec, McGlohon, Faloutsos, Glance, & Hurst, 2007) analyzed about 45 thousands blogs and 2.2 million postings, they reported that the popularity of posts drops with a Power-law, instead of exponentially and the size of the cascades distribution follows a perfect Zipfian distribution. *Leskovec et al.* (Leskovec, Backstrom, Kumar, & Tomkins, 2008), also focused directly on the microscopic node behavior of four large online social network datasets (*Flickr*, *Delicious*, *Yahoo Answers*, and *LinkedIn*). They observed that user's lifetime follows exponential distribution, time gap between link creations follows a Power-law with exponential cutoff and user activity is uniform over its lifetime or decreases with time exponentially. *Nazir et al.* (Nazir, Raza, & Chuah, 2008) presented three applications for *Facebook* to measure the behavior of the users. The authors detected communities, the members of the communities frequently interact with each other and several kinds of human activities were monitored. Based on the measurements, they showed that the popularity of the applications follows a power-law distribution with an exponential decay.

Guo et al. (Guo, Tan, Chen, Zhang, & Zhao, 2009) analyzed the patterns of user content generation in online social networks. They identified three groups of users. The first group is users with distinct posting behaviors including steadily posting inactively posting, and the second group is users with occasional posting in the network. They found that the overall user lifetime does not follow the exponential distribution and also the distribution of different users' posting activities does not follow Power-law distributions rather it follows the stretched exponential distribution similar to the reference rank distribution of objects in Internet media systems, including viral video social networks such as *YouTube*. *Kwak et al.* (Kwak, Lee, Park, & Moon, 2010) crawled the entire *Twitter* site and obtained user profiles, social relations, trending topics and tweets. In their analysis on follower-following topology, they found that the distribution of followers is not power-law; rather the distribution of the users in a Retweet tree follows power-law distribution. *Gyarmati et al.* (Gyarmati & Trinh, 2010) analyzed the degree distribution of nodes for online sessions of users in *Bebo*, *MySpace*, *Netlog*, and *Tagged* based on the crawled public part of users' profile pages, which contained online status information of the users. They revealed that the time spent online by users follows a Weibull distribution that a fraction of users tend to lose interest surprisingly fast soon after subscribing to the service. They claimed that the distribution of session times of users and the number of sessions can be modeled with power-law distributions.

Ding et al. (Ding et al., 2010) studied behaviors of users in the Bulletin board system social network by analyzing the read and reply data and found that the distributions of discussion sizes and user participation levels follows a power-law distribution. *Galuba et al.* (Galuba, Aberer, Chakraborty, Despotovic, & Kellerer, 2010) analyzed the information propagation in *Twitter* and investigated the propagation of URLs through posts. They found that posting frequencies followed a power-low distribution across users and URLs. Also the authors based on the obtained datasets proposed a model that predicts the propagation of information. *Falck-Ytter et al.* (Falck-Ytter & Øverby, 2012) compared the number of followers who followed popular contents in *Twitter* and *YouTube*. Their observations show that Zipf's law was not suitable to describe followers in *Twitter*. In contrast, it was suitable to describe *YouTube* viewers. *Zhong et al.* (Zhong, Fan, Wang, Xiao, & Li, 2012) studied user behavior and interests in different online social networks and observed that they influenced one another. Besides, only small fraction of users actually participated in many activities in online social networks which resulted in a distribution of users with a long-tail property. *Morales et al.* (Morales, Losada, & Benito, 2012) analyzed the effects of user behavior on social structure emergence using the information flow on *Twitter* and

found that community structure was formed inside the network. In addition, the distribution of the number of posts by users was shown to fit an exponentially truncated Power-law.

Yan et al. (Yan, Wu, & Zheng, 2013) explored the behavior of users in posting microblogs systems and realized that the interval time distribution of people posting did not fit a normal distribution, rather it was a power-law distribution. *Liu et al.* (Liu, Nazir, Joung, & Chuah, 2013) found that the number of daily active users using some applications in *Facebook* follows a power-law distribution. *Feng et al.* (Feng, Cong, Chen, & Yu, 2013) explored pin distribution (visual bookmarks of images) and board distributions as well as number of comments, likes, repins in *Pinterest* and discovered that those characteristics follows a Power-law. *Vongsingthong et al.* (Vongsingthong, Boonkrong, Kubek, & Unger, 2015) tracked the activities of some users in a *Facebook* through a questionnaire. Their observation on the user activities such as posting photos and videos, visiting popular pages and friend profiles, playing online game, and questioning on products follows a Power-law distribution. *Bild et al.* (Bild, Liu, Dick, Mao, & Wallach, 2015), studied empirically aggregate user behavior using an exhaustive observations on *Twitter* with a focus on quantitative descriptions. They found that the lifetime tweet distribution is a type-II discrete Weibull stemming from a Power-law hazard function, that the tweet rate distribution, although asymptotically Power-law, exhibits a lognormal cutoff over finite sample intervals, and that the inter-tweet interval distribution is a Power-law with exponential cutoff.

The summary of studies for type of user behaviors/activities of online social networks and their respective distributions are given in table 1.

Table 1. Summary of studies for user behaviors and their respective distributions.

References	Network	Type of user activity or behavior	Distribution/property
(Kwak et al., 2010)	Twitter	The number of friends	Power-law distribution
(Bild et al., 2015)	Twitter	Tweet rates, Retweet rates	Power-law distribution
(Guo et al., 2009)	Answers	Overall user lifetime, different users' posting activities	Stretched exponential distribution
(Golder et al., 2007)	Facebook	Number of messages sent by a user	Power-law distribution
(Leskovec et al., 2008)	Delicious, Flickr, Answers, LinkedIn	User's lifetime	Exponential distribution
(Nazir et al., 2008)	Facebook	Popularity of the social applications among users	Power-law distribution with an exponential decay
(Nazir et al., 2008)	Facebook	Number of unique users that visit the application least once during a given day	Power-law distribution with an exponential cutoff
(Cha, Kwak, Rodriguez, Ahn, & Moon, 2007)	YouTube, Daum	The popularity distribution of user generated video	Power-law distribution with an exponential cutoff
(Gyarmati & Trinh, 2010)	Bebo, MySpace, Netlog, Tagged	Time spent online by users	Weibull distribution
(Gyarmati & Trinh, 2010)	Bebo, MySpace, Netlog, Tagged	Session times of users	Power-law distribution
		The number of sessions	
(Ding et al., 2010)	TianYa BBS	Discussion sizes and user participation levels	Power-law distribution
(Falck-Ytter & Øverby, 2012)	YouTube	YouTube viewers	Zipf's law distribution

(Morales et al., 2012)	Twitter	The number of posts by users	Exponentially truncated Power-law distribution
(Yan et al., 2013)	Sina	Interval time distribution of people posting	Power-law distribution
(Liu et al., 2013)	Facebook	The number of daily active users using applications	Power-law distribution
(Feng et al., 2013)	Pinterest	Pin distribution, visual bookmarks of images, board distributions, number of comments, likes and repins	Power-law distribution
(Vongsingthong et al., 2015)	Facebook	Posting photos and videos, visiting popular pages and friend profiles, Playing online game, and questioning on products	Power-law distribution
(Galuba et al., 2010)	Twitter	Posting frequencies across users and urls	Power-law distribution
(Bild et al., 2015)	Twitter	Lifetime tweet	Type-II discrete Weibull distribution
(Bild et al., 2015)	Twitter	Tweet rate	Power-law distribution
(Bild et al., 2015)	Twitter	Inter-tweet interval	Power-law distribution with exponential cutoff

2.3. Learning automata theory

The first learning automata models were developed in mathematical psychology (Burke, Estes, & Hellyer, 1954; Estes, 1950) and have become a topic of interest to computer engineers in view of its role in machine intelligence (Narendra & Thathachar, 1989). A learning automaton (LA) refers to an abstract model which randomly chooses an action out of its finite set of allowed actions and performs it on an unknown random environment. The action is chosen at random based on a probability distribution kept over the action-set and at each instant the chosen action is served as the input to the random environment. Environment then evaluates the chosen action and responds to the automata with a reinforcement signal. Based on chosen action, and received signal, the automaton updates its internal state and chooses its next action. Generally, the goal of a learning automaton is to find the optimal action from the action-set so that the average penalty received from the environment is minimized.

Learning automata can be classified into two main classes: fixed and variable structure learning automata (VSLA) (Narendra & Thathachar, 1989) which is used in this paper. A VSLA is represented by a 6-tuple $\langle \beta, \phi, \alpha, P, G, T \rangle$, where β denotes the set of inputs, ϕ is a set of internal states, α is the action sets as output, P is the state probability vector governing the choice of state as each instant k, G is the output mapping, and T is the learning algorithm (also known as learning scheme). The learning algorithm T refers to a recurrence relation which is used to update the action probability vector. Let $\alpha_i(k) \in \alpha$ be the action that is chosen by a learning automaton and $p(k)$ is the probability vector defined over the set of action at instant k. At each instant k, the action probability vector $p(k)$ is updated by the linear learning algorithm given in equation (6) if the chosen action $\alpha_i(k)$ is rewarded by the random environment (β=0) and it is updated according to equation (7) if the chosen action is penalized (β=1).

$$p_j(k+1) = \begin{cases} p_j(k) + a\left[1 - p_j(k)\right] & j = i \\ (1-a)p_j(k) & \forall j \neq i \end{cases} \tag{6}$$

$$p_j(k+1) = \begin{cases} (1-b)p_j(k) & j = i \\ \left(\frac{b}{r-1}\right) + (1-b)p_j(k) & \forall j \neq i \end{cases} \quad (7)$$

where a denotes reward parameter which determines the amount of increases of the action probability values, b is the penalty parameter determining the amount of decrease of the action probabilities values and r is the number of actions that learning automaton can take. If $a=b$, the learning algorithm is called linear reward-penalty (L_{RP}) algorithm; if $b=\varepsilon a$ where $0<\varepsilon<1$, then the learning algorithm is called linear reward-ε-penalty ($L_{R\varepsilon P}$) algorithm; and finally if $b=0$, the learning algorithm is called linear reward-Inaction (L_{RI}) algorithm which is perhaps the earliest scheme considered in mathematical psychology.

In the recent years, learning automata have been successfully applied to a wide variety of applications such as optimization (Mahdaviani, Kordestani, Rezvanian, & Meybodi, 2015), image processing (Mofrad, Sadeghi, Rezvanian, & Meybodi, 2015), graph problems (Mousavian, Rezvanian, & Meybodi, 2013, 2014), social network analysis (M Soleimani-Pouri, Rezvanian, & Meybodi, 2012; Mohammad Soleimani-Pouri, Rezvanian, & Meybodi, 2014), sampling social networks (Alireza Rezvanian & Meybodi, 2016; Alireza Rezvanian, Rahmati, & Meybodi, 2014), and community detection in complex networks (Khomami, Rezvanian, & Meybodi, 2016).

3. Proposed network measures for stochastic graphs

In this section, we first define stochastic graphs and then define network measures: strength, closeness, betweenness and clustering coefficient for stochastic graphs.

Definition 1. A *stochastic graph* G can be described by a triple $G=\langle V, E, W\rangle$, where $V=\{v_1, v_2, \ldots, v_n\}$ is the set of nodes, $E=\{e_{i,j}\}\subseteq V\times V$ is the edge-set, and W is a matrix in which w_{ij} is a random variable associated to edge e_{ij} between node v_i and node v_j if such an edge exists.

Definition 2. The *strength* of a node v_i in a stochastic graph is a random variable defined by following equation

$$ES_i = \sum_{j\neq i} \overline{w}_{ij} \quad (8)$$

where $\overline{w}_{ij}$ is the random variable associated with edge e_{ij} outgoing from node v_i.

Before we give definitions for closeness and betweenness in stochastic graphs we need to define the concept of shortest path in stochastic graphs.

Definition 3. In a stochastic graph, a *path* π_i with weight of w_{π_i} and length of n_i from source node v_s to destination node v_d can be defined as an ordering $\{\pi_1^i, \pi_2^i, \ldots, \pi_{n_i}^i\}\subset V$ of nodes in such a way that $\pi_1^i = v_s$ and $\pi_{n_i}^i = v_s$ are source and destination nodes, respectively and edge $e(\pi_j^i, \pi_{j+1}^i)\in E$ for $1\leq j\leq n_i$, where π_j^i is the j^{th} node in path π_i.

Definition 4. Let in a stochastic graph, there are r distinct paths $\Pi_{s,t}=\{\pi_1, \pi_2, \ldots \pi_r\}$ between source node v_s and destination node v_d, the *shortest path* between source node v_s and destination node v_d is defined as a path with minimum expected weight $w_{\pi^*} = \min_{\pi\in\Pi}\{w_\pi\}$ and called *stochastic shortest path*.

Definition 5. *Closeness* of node v_i in a stochastic graph is a random variable defined by the following equation

$$EC_i = \frac{1}{\sum_{j \neq i} ED_{ij}} \tag{9}$$

where ED_{ij} is a random variable associated with the weight of shortest path between node v_i and node v_j with minimum expected weight.

Definition 6. *Betweenness* of node v_i in a stochastic graph is a random variable EB_i defined by equation (10)

$$EB_i = \frac{Eg_{st}(i)}{Eg_{st}} \tag{10}$$

where $Eg_{st}(i)$ is the number of stochastic shortest path between node v_s and node v_t that passes through node v_i and Eg_{st} is the number of all stochastic shortest paths between node v_s and node v_t.

Definition 7. *Clustering coefficient* of a node v_i in a stochastic graph is a random variable and can be defined as follows

$$ECC_i = \frac{1}{(k_i - 1).ES_i} \sum_{j \neq i} \sum_{k \neq i,j} \frac{(w_{ij} + w_{ik})}{2} a_{ij}.a_{ik}.a_{jk} \tag{11}$$

where k_i is the number of nodes adjacent to node v_i, and a_{ij} is 1 if node v_i is adjacent to node v_j, and 0 otherwise. ES_i is the strength of node v_i and w_{ij} is the random variable associated with weight of edge e_{ij}.

4. Proposed algorithms for calculating network measures in stochastic graphs

In the previous section, we defined some network measures for stochastic graphs. In this section, several algorithms based on learning automata are proposed for calculating network measures in stochastic graphs under the situation that the probability distribution functions of the weights associated with the edges of graph are unknown. The proposed algorithms for calculating network measures by taking samples from the edges of the stochastic graph try to estimate the distribution of the network measures. The process of sampling from the edges of the graph is guided by the aid of learning automata in such a way that the number of samples needed to be taken from the edges of the stochastic graph for estimating the network measures to be reduced as much as possible. In the proposed algorithms, the guided sampling process implemented by learning automata aims to take more samples from the promising region of the graph, the regions that reflects higher rate of activities, instead of walking around and taking unnecessary samples from non-promising region of the graph. The algorithm after initialization phase iterates the updating phase until one of the stopping conditions is met. The details of initialization phase, updating phase, and stopping conditions are given below.

Initialization Phase:

Let $G\langle V, E, W\rangle$ be the input stochastic graph, where $V=\{v_1, v_2, \ldots, v_n\}$ is the set of nodes, $E=\{e_{i,j}\}\subseteq V\times V$ is the edge-set, and $W=\{w_{i,j}\}$ is the set of random variables each of which is associated with an edge weight of the input stochastic graph. It is assumed that the weight of each edge is a positive random variable with unknown probability distribution function. The proposed algorithm uses a network of learning automata isomorphic to the input graph and tries to estimate weights of edges by sampling from the edges of the graph in such a way that the number of samples taken from the edges of graph be reduced. Learning automaton A_i residing in node v_i has two actions as $\alpha_i = \{\alpha^1, \alpha^2\}$ where

action α^1 is "take sample from the edges of chosen node v" and action α^2 is "do not take sample from the edges of chosen node v". Let $p(v_i) = \{p_i^1, p_i^2\}$ be the action probability vector of learning automaton A_i and initialized equally $p_i^1 = p_i^2 = 1/2$ for all $v_i \in V$. The process of taking samples from the edges on which a node is incident is governed by the learning automaton assigned to that node. At the beginning of the algorithm, the weight of each edge is initialized with some random samples in order to provide a coarse estimate of the weight of that edge.

Updating Phase: Updating the network measure and action probability vectors

In this phase, every learning automaton in the network chooses an action according to its action probability vector in parallel manner. Then a sample from every edge incident on a node whose corresponding learning automaton has chosen the action of taking sample (α^1) will be taken. Calculation of the new estimates for the means and standard deviations of the unknown distributions of the edge weights are then performed. Using these new estimates, the network measure is calculated for the given network and then the difference between the calculated network measure and the network measure obtained in the previous iteration is calculated using DDQC (equation (18)). Based on this difference, the probability vectors of learning automata assigned to the nodes of the input graph are updated according to the learning algorithm. That is, if this difference is lower than or equal to a given error ε_p, then action α^1 of all learning automata are rewarded proportional to the amount of improvement and penalized otherwise.

Stopping Phase:

Updating phase is repeated until one of the following stopping conditions is met.

1. The number of iterations k exceeds a given number K_{max},
2. The difference between the calculated measure in two consecutive iterations becomes lower than a specified error E_{min}, or
3. The average of entropy of probability vector of learning automata reaches a predefined value T_{min}.

In the proposed algorithm, the entropy is used for measuring learning process. The information entropy for a learning automaton with r actions can be defined as follows (Mousavian et al., 2013):

$$H = -\sum_i^r p_i . Log(p_i) \tag{12}$$

where p_i is the probability of choosing i^{th} action of a learning automaton. The entropy for a learning automaton has maximum value of one when all the actions have equal probabilities of choosing and has minimum value of zero when the action probability vector is a unit vector. The pseudo-code of the proposed algorithm 1 is shown in Figure 1.

Algorithm 1: The proposed algorithm for calculating network measures in stochastic graphs

Input: Stochastic Graph $G\langle V, E, W\rangle$, Network measure M_Θ, Thresholds K_{max}, T_{min}, E_{min}, ε_p
Output: Calculated network measures $\hat{\theta}_1, \hat{\theta}_2, \ldots, \hat{\theta}_n$ where $\hat{\theta}_i$ is the calculated network measure for node v_i
// That is $\hat{\theta}_i$ is the expected of ES_i (Eq. (8)) if the given network measure M_Θ is the strength of the network,
// $\hat{\theta}_i$ is the expected of EC_i (Eq. (9)) if the given network measure M_Θ is the closeness of the network,
// $\hat{\theta}_i$ is the expected of EB_i (Eq. (10)) if the given network measure M_Θ is the betweenness of the network, and
// $\hat{\theta}_i$ is the expected of ECC_i (Eq. (11)) if the given network measure M_Θ is the clustering coefficient of the network.
Initialization:
Create a network of learning automata isomorphic to the graph G by assigning an automaton A_i to each node v_i;
Let $\alpha_i = \{\alpha^1, \alpha^2\}$ be the set of actions for learning automata A_i;
Let $P = \{p_1, p_2, \ldots, p_n\}$ be the set of action probabilities in which $p_i = \{p_i^1, p_i^2\}$ and initialized equally $p_i^1 = p_i^2 = 1/2$ for all $v_i \in V$;
Let k be the iteration number and initially set to 1;

Let $\bar{w}_{i,j}$ be the estimate for mean of edge weight of all samples taken from edge e_{ij} and initialized with some random samples;
Let $\hat{\theta}_1^k, \hat{\theta}_2^k, \ldots, \hat{\theta}_n^k$ be the calculated network measure for the nodes at iteration k;
Let G^k be the network with mean edge weight at iteration k;
Begin
Let $H_i(k)$ be the entropy of learning automaton A_i at iteration k and calculated based on equation (12);
Repeat
For all learning automata do in parallel
Learning automaton A_i chooses an action according to its action probability vector;
If ($\alpha_i = \alpha^1$) **then**
Take a sample from every edge on which node v_i is incident;
Construct G^k by calculating the new weight for every edge on which node v_i is incident;
Else
No action;
End if
End for
If (M_Θ is strength)
Calculate new estimates for the strengths of the nodes $\hat{\theta}_1^k, \hat{\theta}_2^k, \ldots, \hat{\theta}_n^k$ using equation (8);
Else If (M_Θ is closeness)
Calculate new estimates for the closeness $\hat{\theta}_1^k, \hat{\theta}_2^k, \ldots, \hat{\theta}_n^k$ using equation (9);
Else If (M_Θ is betweenness)
Calculate new estimates for the betweenness $\hat{\theta}_1^k, \hat{\theta}_2^k, \ldots, \hat{\theta}_n^k$ using equation (10);
Else If (M_Θ is clustering coefficient)
Calculate new estimates for the clustering coefficient $\hat{\theta}_1^k, \hat{\theta}_2^k, \ldots, \hat{\theta}_n^k$ using equation (11);
End if
Calculate $DDQC^k(G^k, G^{k-1})$ using equation (18); // Calculate DDQC between consequent estimated network measures
If ($k > 1$) **Then**
Calculate $\Delta^k = DDQC^k - DDQC^{k-1}$; // Calculate the change rate between iterations k and k-1;
If ($|\Delta^k| \geq \varepsilon_p$) **Then**
Reward the chosen actions of all learning automata;
Else
Penalize the chosen actions of all learning automata;
End If
End If
$H(k) \leftarrow \frac{1}{n}\sum_{i=1}^{n} H_i(k)$;
$k \leftarrow k + 1$;
Until ($k < K_{max}$ **OR** $H(k) > T_{min}$ **OR** $|\Delta^k| > E_{min}$)
End Algorithm

Figure 1. Pseudo-code for algorithm 1 for calculating network measures in stochastic graphs.

4.1. Improvements

In the remaining part of this section some improvements of the proposed algorithm (Algorithm 1) are described. These modifications try to improve either the speed or the efficiency of the algorithm or both of them.

4.1.1. Improvement 1

The performance of any learning automata based algorithm is very sensitive to learning rate of the learning algorithm. Large value for learning rate results in increasing the speed of convergence and decreasing the accuracy of the algorithm while small value for learning rate results in increasing the accuracy and decreasing the speed of convergence of the algorithm. In Algorithm 1, all learning automata use a same learning rate proportional to the amount of improvement. It means that the algorithm in each iteration treats all nodes equally during the execution of the algorithm. For this improvement, we use $a_i(k) = c_k . d_i^j(k)$ as the learning parameter of automaton A_i at iteration k where $c_k = \Delta^k / Max(\Delta^k)$ is a

scaled change rate between 0 and 1 at iteration k, $d_i^j(k)$ is an estimation of reward probability for action α^j calculated by the algorithm at iteration k. $d_i^j(k)$ is calculated by dividing the number of times that action α^j is chosen up to time k by automaton A_i and received reward by the number of times that automaton A_i is activated. In this way, the edges which were chosen more frequently will be more probable to be chosen in the future. As in the experiment, such a strategy will lead to higher rate of convergence and fewer samples from the graph. Algorithm 1 in which the learning rate is adapted according to this improvement is called *Algorithm 2*.

4.1.2. Improvement 2

For this improvement, we use the information entropy of a learning automaton (defined by equation (12) for adaptation of learning rate of that learning automaton. The learning rate of learning automaton A_i at iteration k is calculated as $a_i(k) = c_k.H_i(k)$ where $c_k = \Delta^k / Max(\Delta^k)$ is a scaled change rate between 0 and 1 at iteration k and $H_i(k)$ is the information entropy of learning automaton A_i which reflect the uncertainty of A_i about their actions. In this way, the nodes whose estimated edge weights have more changes will be more probable to be chosen in the future for the sampling purpose. The results of experiment 5 show that such a strategy will lead to a higher rate of convergence and also fewer samples taken from the graph. Algorithm 1 in which the learning rate is adapted using the information entropy is called *Algorithm 3*.

4.1.3. Improvement 3

In algorithm 1, we may reach a point that the changes in probability action vectors of some of the learning automata in two consecutive iterations are negligible which in this case we may turn off these learning automata in order to prevent taking more samples from the edges which may not lead us to a much better estimation of the edges weights and consequently better estimate of network measures. Algorithm 1 in which the above improvement is applied is called *Algorithm 4*. Experiment 5 shows that algorithm 4 comparing to algorithm 1 produces higher convergence speed for all test graphs.

4.1.4. Improvement 4

This improvement is obtained by equipping each node with two learning automata, the first learning automata as in algorithm 1 decides whether or not a node to be chosen and the second learning automata (with number of actions equal to the number of nodes adjacent to that node) decides which edge on which the node is incident must be sampled. Experimentation shows that the utilization of another automaton in each node will lead to a higher rate of convergence and also taking fewer samples from the edges of the graph. Algorithm 1 in which the above improvement is used is called *Algorithm 5*. Experiment 2 shows that algorithm 5 produces fewer numbers of samples and higher accuracy taken from the graph as compared to the case where the second learning automata are absent.

4.1.5. Improvement 5

This improvement is obtained by combing improvements 2, 3 and 4. The resulting algorithm is called *Algorithm 6*. The pseudo-code of this algorithm is given in Figure 2.

Algorithm 6: The proposed algorithm for calculating network measures in stochastic graphs
Input: Stochastic Graph $G\langle V, E, W\rangle$, Network measure $M_{\odot}$, Thresholds K_{max}, T_{min}, E_{min}, ε_h, ε_p **Output:** Calculated network measures $\hat{\theta}_1, \hat{\theta}_2, \ldots, \hat{\theta}_n$ where $\hat{\theta}_i$ is the calculated network measure for node v_i // That is $\hat{\theta}_i$ is the expected of ES_i (Eq. (8)) if the given network measure $M_{\odot}$ is the strength of the network,

// $\hat{\theta}_i$ is the expected of EC_i (Eq. (9)) if the given network measure M_Θ is the closeness of the network,
// $\hat{\theta}_i$ is the expected of EB_i (Eq. (10)) if the given network measure M_Θ is the betweenness of the network, and
// $\hat{\theta}_i$ is the expected of ECC_i (Eq. (11)) if the given network measure M_Θ is the clustering coefficient of the network.

Initialization:

Create the first network of learning automata A_1 isomorphic to the graph G by assigning an automaton A_1^i to each node v_i;
Create the second network of learning automata A_2 isomorphic to graph G by assigning an automaton A_2^i to each node v_i;
Let $\alpha_1=\{\alpha_{1,1}, \alpha_{1,2}, \ldots, \alpha_{1,n}\}$ be the set of actions for the first network of learning automata and $\alpha_{1,i} = \{\alpha_{1,i}^1, \alpha_{1,i}^2\}$ is the set of actions for learning automaton A_1^i;
Let $\alpha_2=\{\alpha_{2,1}, \alpha_{2,2}, \ldots, \alpha_{2,n}\}$ be the set of actions for the second network of learning automata and $\alpha_{2,i} = \{\alpha_{2,i}^1, \alpha_{2,i}^2, \ldots, \alpha_{2,i}^{ri}\}$ is the set of actions for learning automaton A_2^i with $r_i=|v_i|$ actions;
Let $P_1=\{p_{1,1}, p_{1,2}, \ldots, p_{1,n}\}$ be the set of action probabilities of the first network of learning automata in which $p_{1,i}=\{p_{1,i}^1, p_{1,i}^2\}$ and initialized equally $p_{1,i}^1 = p_{1,i}^2=1/2$ for all $v_i \in V$;
Let $P_2=\{p_{2,1}, p_{2,2}, \ldots, p_{2,n}\}$ be the set of action probabilities of the second network of learning automata in which $p_{2,i}=\{p_{2,i}^1, p_{2,i}^2, \ldots, p_{2,i}^{ri}\}$ and initialized equally $p_{2,i}^1 = p_{2,i}^2 = \cdots = p_{2,i}^{ri}=1/r_i$ for all $v_i \in V$;
Let S_i be the state of learning automaton A_1^i and initially set to active for all learning automata A_1;
Let k be the iteration number and initially set to 1;
Let $\overline{w}_{i,j}$ be the estimate for mean of edge weights of all samples taken from edge e_{ij} and initialized with some random samples;
Let $\hat{\theta}_1^k, \hat{\theta}_2^k, \ldots, \hat{\theta}_n^k$ be the calculated network measure for the nodes at iteration k;
Let G^k be the network with mean edge weight at iteration k;

Begin

Let $H_i(k)$ be the information entropy of learning automaton A_1^i at iteration k and calculated based on equation (12);
Repeat
For all active first learning automata do in parallel
Each learning automaton A_1^i randomly choose an action according to its action probability vector;
If ($\alpha_{1,i}=\alpha_{1,i}^1$) **then**
Learning automaton A_2^i randomly chooses an action according to its action probability vector;
Let the chosen action be $\alpha_{2,i}^j$;
Take a sample from edge $e(v_i, v_j)$;
Construct G^k by calculating the new estimate for weight of edge $e(v_i, v_j)$;
Else
No action;
End if
If ($k > 1$ **AND** $H_i(k)>T_{min}$ **AND** $|H_i(k)-H_i(k\text{-}1)|<\varepsilon_h$)
$S_i \leftarrow 0$; // turn off learning automaton A_1^i
End if
End for
If (M_Θ is strength)
Calculate new estimates for the strengths of the nodes $\hat{\theta}_1^k, \hat{\theta}_2^k, \ldots, \hat{\theta}_n^k$ using equation (8);
Else If (M_Θ is closeness)
Calculate new estimates for the closeness $\hat{\theta}_1^k, \hat{\theta}_2^k, \ldots, \hat{\theta}_n^k$ using equation (9);
Else If (M_Θ is betweenness)
Calculate new estimates for the betweenness $\hat{\theta}_1^k, \hat{\theta}_2^k, \ldots, \hat{\theta}_n^k$ using equation (10);
Else If (M_Θ is clustering coefficient)
Calculate new estimates for the clustering coefficient $\hat{\theta}_1^k, \hat{\theta}_2^k, \ldots, \hat{\theta}_n^k$ using equation (11);
End if
Calculate $DDQC^k(G^k, G^{k-1})$ using equation (18); // Calculate DDQC between consequent estimated network measures
If ($k > 1$) **Then**
Calculate $\Delta^k \leftarrow DDQC^k - DDQC^{k-1}$; // Calculate the change rate between iterations k and k-1
Calculate $c_k \leftarrow \frac{\Delta^k}{Max(\Delta^k)}$; // normalized coefficient
Calculate $H(k) \leftarrow \frac{1}{n}\sum_{i=1}^{n} H_i(P)$; // information entropy
$a_i(k) \leftarrow c_k.H_i(k)$; // update learning parameter
If ($|\Delta^k| \geq \varepsilon_p$) **Then**
Reward the chosen actions of both learning automata according to learning algorithm;
Else
Penalize the chosen actions of both learning automata according to learning algorithm;
End If

```
    End If
      k ← k + 1;
  Until (k < K_max OR H(k) > T_min OR |Δ^k| > E_min)
End Algorithm
```

Figure 2. Pseudo-code for algorithm 6 for calculating network measures in stochastic graphs.

5. Experimental Results

In this section, to study the performance of the proposed algorithms, several computer simulations are conducted on synthetic stochastic graphs which are generated based on well-known random graph with their weights of edges being random variables. We use well-known computer generated network model of *Barabási-Albert* model (*BA* model) as a scale-free network with heavy-tailed degree distribution (Barabási & Albert, 1999) which is utilized widely in literature. We set network parameters of BA model as $N \in \{1000, 2000, 5000, 10000\}$ and $m_0=m=5$. Other synthetic network as a small world network is *Watts–Strogatz* model (WS model) (Watts & Strogatz, 1998) which reflect a common property of many real networks such as short average path length with $N \in \{1000, 2000, 5000, 10000\}$, $k=4$ and $p=0.2$. Also, we use well-known random network of *Erdös–Rényi* model (*ER* model) (Erdos & Rényi, 1960) which is utilized widely in literature as $N \in \{1000, 2000, 5000, 10000\}$ and $p=0.2$. The edge weights for these graphs are random variables with exponential distributions whose mean are chosen randomly from set $\mu \in \{0.8, 1, 1.2, 1.5\}$. It is noted that the chosen mean values is adopted from an empirical observations by *Bild et al* (Bild et al., 2015).

5.1. Distance measures

In order to evaluate the accuracy of estimations by the proposed algorithms, we used several distance function between real parameters and estimated parameters. In this paper, we use Kolmogorov-Smirnov distance statistic, skew divergence distance, Pearson's correlation coefficient and normalized L_1 distance as distance measures to compare distribution properties of the proposed algorithms for estimating network parameters. These distance measures as evaluating criteria are described below.

5.1.1. Kolmogorov-Smirnov distance statistic

Kolmogorov-Smirnov (KS) Statistic (Jalali, Rezvanian, & Meybodi, 2016) is one of the statistical test methods commonly used for assessment the distance between two cumulative distribution functions (CDFs). *KS* measures acceptability between original distribution and estimated distribution. The result of this test is a value between 0 and 1 that as closer as it is to zero, both distributions will have a greater similarity; and as closer as it is to unit, the two distributions will show a greater discrepancy. This measure has been defined as

$$KS(P,Q) = \underset{x}{\text{MAX}}|P(x) - Q(x)| \tag{13}$$

where P and Q are two CDFs of original and estimated data, respectively, and x represents the range of the random variable. So it is computed as the maximum vertical distance between the two distributions.

5.1.2. Skew divergence distance

Skew divergence (Jalali et al., 2016) can also be used for assessment the distance between two Probability Distribution Functions (PDF) and defined as follows

$$SD(P,Q,\alpha) = D[\alpha P + (1-\alpha)Q || \alpha Q + (1-\alpha)P] \tag{14}$$

where D is the *Kullback-Leibler* (KL) *divergence*, which measures the similarity between two PDFs P and Q that do not have continuous support over the full range of values and α=0.99. The KL *divergence* is defined as follows

$$KL(P(x)||Q(x)) = \sum_x P(x) log \frac{P(x)}{Q(x)} \tag{15}$$

5.1.3. Pearson's correlation coefficient

One of the good indices to measure the similarity between the estimated parameter values and the original parameters values is Pearson's correlation coefficient (PCC) (Luo, Li, Wu, & Zhang, 2015) and calculated as follows

$$PCC(P,Q) = \frac{n\sum_i p_i q_i - (\sum_i p_i)(\sum_i q_i)}{\sqrt{n\sum_i p_i^2 - (\sum_i p_i)^2}\sqrt{n\sum_i q_i^2 - (\sum_i q_i)^2}} \tag{16}$$

where p_i and q_i are the values of the original parameter P and estimated parameter Q in the original data and obtained samples data respectively and also n is the number of elements for each parameters. If the result of PCC is much closer to 1, then the similarity of real and estimated parameters is much better.

5.1.4. Normalized L_1 distance

In some cases, for evaluation, the distance between two positive m-dimensional real vectors P and Q are measured where P is the original vector and Q is the estimated vector. Normalized L_1 distance (Jalali, Rezvanian, & Meybodi, 2015) is defined as follows

$$L_1(\mathrm{P},\mathrm{Q}) = \frac{1}{n}\sum_i \frac{|\mathrm{p}_i - q_i|}{\mathrm{p}_i} \tag{17}$$

5.1.5. Degree Distribution Quantification and Comparison (DDQC)

In this paper, in the proposed algorithm, DDQC measures the difference between two calculated network measures in the consecutive iterations. Recently, DDQC, as a new method for quantification and comparison of network degree distributions is presented (Aliakbary, Habibi, & Movaghar, 2015). This method is an improved version of *Janssen*'s method (Janssen, Hurshman, & Kalyaniwalla, 2012) for feature extraction from the degree distribution. The DDQC can be applied for any network measure distribution. In DDQC, the given degree distribution divided into eight regions based on the following description and the sum of parameter distribution in each region is calculated as distribution percentiles. The absolute difference between these extracted values for real and estimated distributions in the original data and obtained samples data is considered as the distance between two distributions as given following equation.

$$DDQC(G,G') = \sum_{i=1}^{8} |Q_i(G) - Q_i(G')| \tag{18}$$

where $Q(G)=\langle IDP(I(i))\rangle_{i=1,\ldots,8}$ is the set of values for eight regions of graph G and $IDP(I)$ is the interval distribution probability and can be calculated as given below

$$IDP(I) = Pr(left(I) \leq D \leq right(I)) \tag{19}$$

where *Left(R)* and *Right(R)* is the minimum and maximum values for each interval I. The interval I can be divided in two regions as following

$$I(i) = \begin{cases} \left[left\left(R\left(\left\lceil \frac{i}{2} \right\rceil\right)\right), left\left(R\left(\left\lceil \frac{i}{2} \right\rceil\right)\right) + \frac{\left|R\left(\left\lceil \frac{i}{2} \right\rceil\right)\right|}{2} \right] & i \text{ is odd} \\ \left[left\left(R\left(\left\lceil \frac{i}{2} \right\rceil\right)\right) + \frac{\left|R\left(\left\lceil \frac{i}{2} \right\rceil\right)\right|}{2}, right\left(R\left(\left\lceil \frac{i}{2} \right\rceil\right)\right) \right] & i \text{ is even} \end{cases} \tag{20}$$

where $|R(r)| = \max\big(right\big(R(r)\big) - left\big(R(r)\big), 0\big)$ is the range length of values.

(21)

The range of values is divided in four regions based on mean and standard deviation of a network measure.

$$R(r) = \begin{cases} [D_{min}, \mu - \sigma] & r = 1 \\ [\mu - \sigma, \mu] & r = 1 \\ [\mu, \mu + \sigma] & r = 1 \\ [\mu + \sigma, D_{max}] & r = 1 \end{cases} \tag{22}$$

where $\mu = \sum_{D_{min}}^{D_{max}} x \times Pr(x)$ and $\sigma^2 = \sum_{D_{min}}^{D_{max}} Pr(x) \times (x - \mu)^2$ are the mean and variance of the probability distribution function *Pr(x)* respectively, which obtained from a network measure.

5.2. Experimental settings

The stopping criteria for the proposed algorithms is either the number of iteration k reached $K_{max}=50\times n$ iterations where n is the number of nodes for each instance graph or the value of average information entropy $H(k)$ is lower than $T_{min}=0.10$ or the difference between the calculated measure in two consecutive iteration Δ^k becomes lower than $\varepsilon_h=0.001$. The edge weights for synthetic stochastic graphs are random variables with exponential distributions whose mean are selected randomly from set $\mu \in \{0.8, 1, 1.2, 1.5\}$. For all algorithms, the reinforcement scheme used for updating the action probability vector of learning automata is $L_{R\varepsilon p}$ with a=0.05, b=0.01 and a=0.01, b=0.005 for the first and the second set of learning automata, respectively. For algorithm 4, a learning automation is disabled when its information entropy reaches 0.05. The reported results are the averages taken over 30 runs.

In this paper, we assume that the structural and behavioral parameters of online social networks are time varying parameters which can be observed by the network. Using the observed parameters, the network can calculate different measurements such as strength, closeness and betweenness in an ongoing fashion. The algorithm proposed in section 4 can be used by the network as a means for observing the time varying parameters of the network for the purpose of network's measurements computations. The aim of this algorithm is to collect information from the network in order to find good estimates for the network's measurements using fewer numbers of samples than that of standard sampling methods. The first experiment that follows gives a hint how one can use the approach proposed in the paper to analyze a

social network modeled as a stochastic graph. The remaining experiments (experiments 2 through 5) study the performance of the algorithms proposed for estimation of the network's measurements.

Experiment I

Using the distributions of the measures for the nodes one can analyze the user behavior and human activities of an online social network using some statistical tests. The following experiments elaborate on this. In this experiment, we select a set of six randomly nodes from synthetic stochastic BA graphs and then ranked them according to their strength, closeness, betweenness and clustering coefficient. The distributions of betweenness, clustering coefficient, closeness and strength for these randomly selected nodes are given in figure 3 through figure 6, respectively. To obtain the statistical ranking, we use multi-comparison statistical tests using Friedman (García, Molina, Lozano, & Herrera, 2009) with 95% significance interval (α=0.05). The average ranking for multi-comparison statistical tests using Friedman's test are presented in Table 2 and the results of *p*-values of the comparing nodes in terms of different network measures are given in table 3 through table 6. The *p*-value computed by Friedman's test is 2.34E-06, which is below the significance interval of 95% (α=0.05). Thus, a significant difference exists among the observed results. In applications such as rumor spreading, disease propagation, and protecting critical regions from intended attacks we need to study the problem of maximization of spread of influence which involves ranking the set of nodes in the network according to their strengths or closeness centralities (Li et al., 2014).

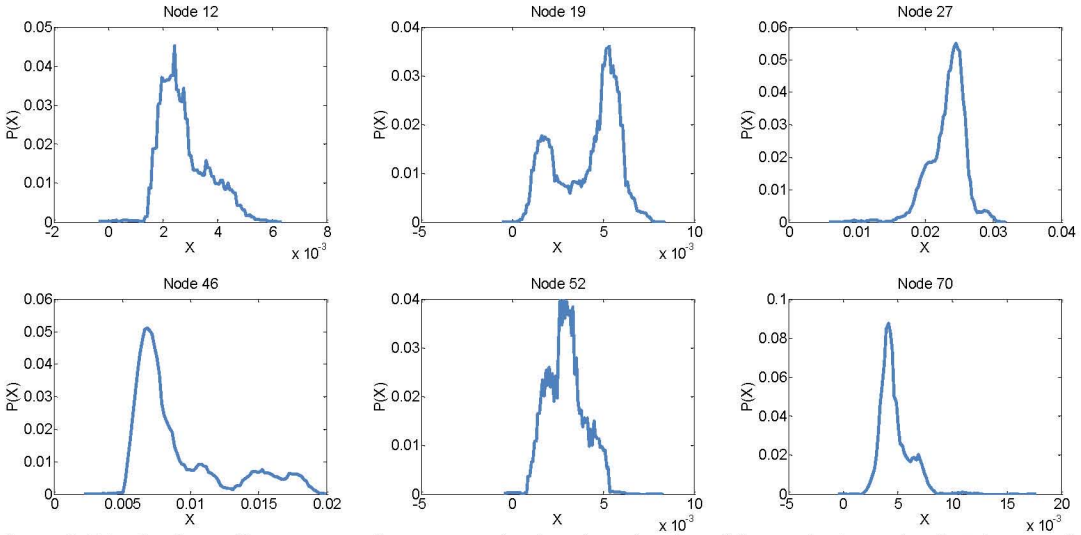


Figure 3. Distributions of betweenness for some randomly selected nodes of the synthetic stochastic BA network.

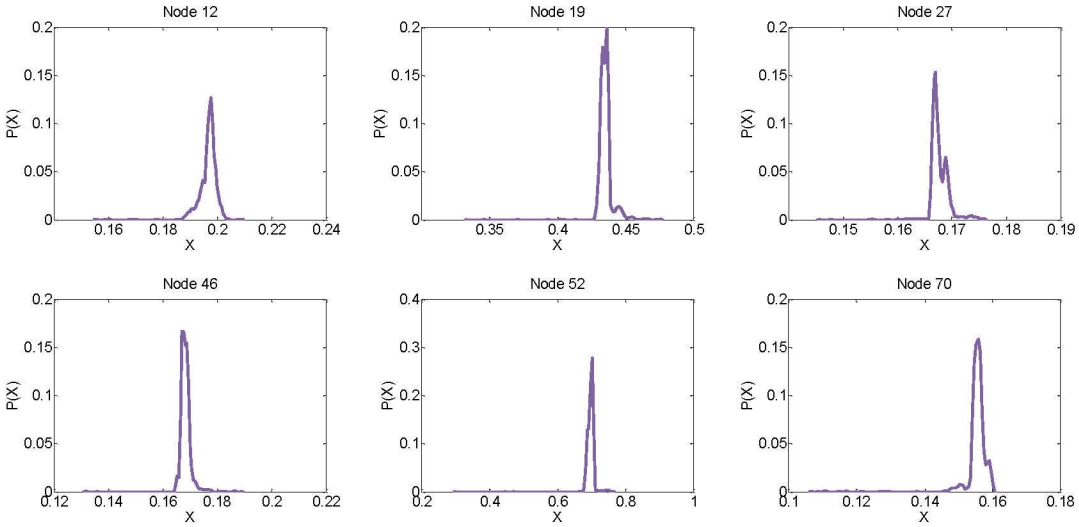


Figure 4. Distributions of clustering coefficient for some randomly selected nodes of the synthetic stochastic BA network.

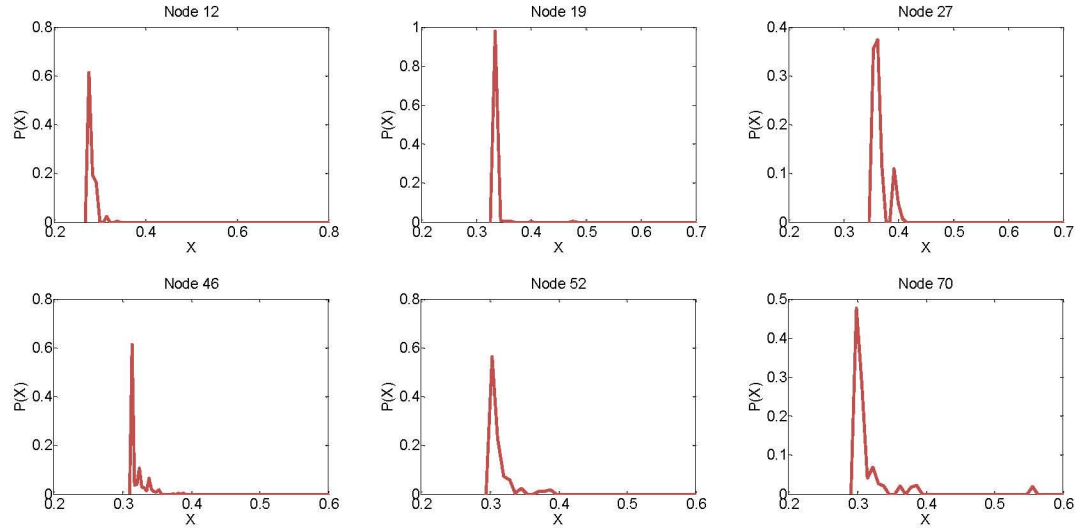


Figure 5. Distributions of closeness for some randomly selected nodes of the synthetic stochastic BA network.

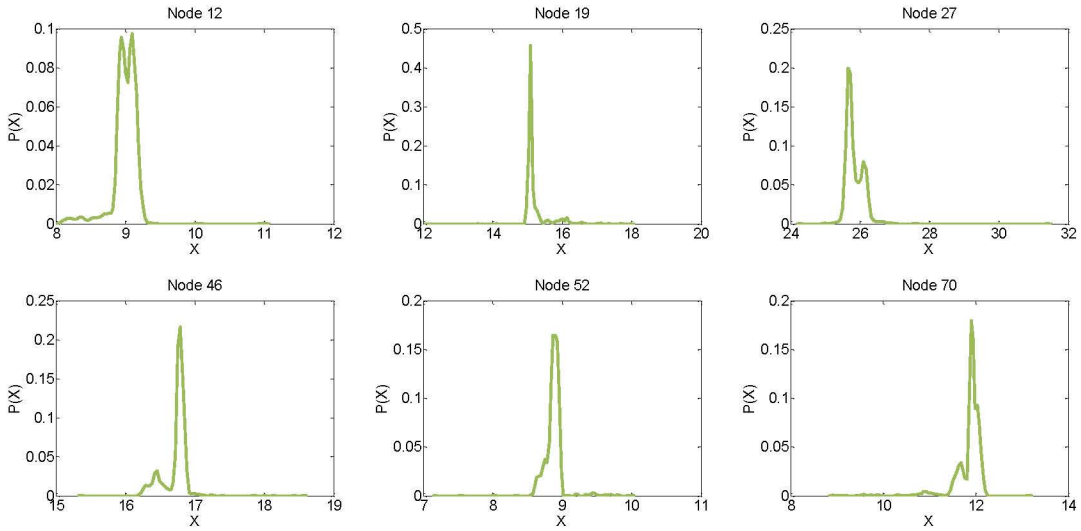


Figure 6. Distributions of strength for some randomly selected nodes of the synthetic stochastic BA network.

Table 2. Average ranking of Friedman's test for randomly selected nodes of the synthetic stochastic BA network.

	Betweenness		Clustering coefficient		Closeness		Strength	
Randomly selected nodes	Average Ranking	Ranking	Average Ranking	Ranking	Average Ranking	Ranking	Average Ranking	Ranking
Node 12	5.98	6	1.99	2	2.18	2	2.98	3
Node 19	1.49	1	4.64	5	1.02	1	1.00	1
Node 27	2.70	2	4.34	4	2.90	3	2.01	2
Node 46	4.83	5	1.01	1	4.03	4	5.84	6
Node 52	2.97	3	6.00	6	4.86	5	4.01	4
Node 70	3.01	4	3.01	3	5.99	6	5.15	5

Table 3. *p*-values of the comparing nodes on the synthetic stochastic BA networks in terms of betweenness of nodes.

i	Hypothesis	Unadjusted *p*-value	*Holm p*-value	*Shaffer p*-value
15	Node 12 vs. Node 19	0.00E+00	3.33E-03	3.33E-03
14	Node 19 vs. Node 46	0.00E+00	3.57E-03	5.00E-03
13	Node 12 vs. Node 27	0.00E+00	3.85E-03	5.00E-03
12	Node 12 vs. Node 52	0.00E+00	4.17E-03	5.00E-03
11	Node 12 vs. Node 70	0.00E+00	4.55E-03	5.00E-03
10	Node 27 vs. Node 46	0.00E+00	5.00E-03	5.00E-03
9	Node 46 vs. Node 52	0.00E+00	5.56E-03	7.14E-03
8	Node 46 vs. Node 70	0.00E+00	6.25E-03	7.14E-03
7	Node 19 vs. Node 70	0.00E+00	7.14E-03	7.14E-03
6	Node 19 vs. Node 52	0.00E+00	8.33E-03	8.33E-03
5	Node 19 vs. Node 27	0.00E+00	1.00E-02	1.25E-02
4	Node 12 vs. Node 46	0.00E+00	1.25E-02	1.25E-02
3	Node 27 vs. Node 70	1.12E-02	1.67E-02	1.67E-02
2	Node 27 vs. Node 52	2.20E-02	2.50E-02	2.50E-02
1	Node 52 vs. Node 70	8.06E-01	5.00E-02	2.50E-02

Note: Holm's procedure rejects those hypotheses that have an unadjusted p-value ≤ 5.00E-02; Shaffer's procedure rejects those hypotheses that have an unadjusted p-value ≤3.33E-03.

Table 4. *p*-values of the comparing nodes on the synthetic stochastic BA networks in terms of clustering coefficient of nodes.

i	hypothesis	Unadjusted *p*-value	*Holm p*-value	*Shaffer p*-value
15	Node 46 vs. Node 52	0.00E+00	3.33E-03	3.33E-03
14	Node 12 vs. Node 52	0.00E+00	3.57E-03	5.00E-03
13	Node 19 vs. Node 46	0.00E+00	3.85E-03	5.00E-03
12	Node 27 vs. Node 46	0.00E+00	4.17E-03	5.00E-03
11	Node 52 vs. Node 70	0.00E+00	4.55E-03	5.00E-03
10	Node 12 vs. Node 19	0.00E+00	5.00E-03	5.00E-03
9	Node 12 vs. Node 27	0.00E+00	5.56E-03	7.14E-03
8	Node 46 vs. Node 70	0.00E+00	6.25E-03	7.14E-03
7	Node 27 vs. Node 52	0.00E+00	7.14E-03	7.14E-03
6	Node 19 vs. Node 70	0.00E+00	8.33E-03	8.33E-03
5	Node 19 vs. Node 52	0.00E+00	1.00E-02	1.25E-02
4	Node 27 vs. Node 70	0.00E+00	1.25E-02	1.25E-02
3	Node 12 vs. Node 70	0.00E+00	1.67E-02	1.67E-02
2	Node 12 vs. Node 46	0.00E+00	2.50E-02	2.50E-02
1	Node 19 vs. Node 27	9.24E-03	5.00E-02	5.00E-02

Note: Holm's procedure rejects those hypotheses that have an unadjusted p-value ≤ 3.33E-03; Shaffer's procedure rejects those hypotheses that have an unadjusted p-value ≤3.33E-03.

Table 5. *p*-values of the comparing nodes on the synthetic stochastic BA networks in terms of closeness of nodes.

i	Hypothesis	Unadjusted *p*-value	*Holm p*-value	*Shaffer p*-value
15	Node 19 vs. Node 70	0.00E+00	3.33E-03	3.33E-03
14	Node 19 vs. Node 52	0.00E+00	3.57E-03	5.00E-03
13	Node 12 vs. Node 70	0.00E+00	3.85E-03	5.00E-03
12	Node 27 vs. Node 70	0.00E+00	4.17E-03	5.00E-03
11	Node 19 vs. Node 46	0.00E+00	4.55E-03	5.00E-03
10	Node 12 vs. Node 52	0.00E+00	5.00E-03	5.00E-03
9	Node 27 vs. Node 52	0.00E+00	5.56E-03	7.14E-03
8	Node 46 vs. Node 70	0.00E+00	6.25E-03	7.14E-03
7	Node 19 vs. Node 27	0.00E+00	7.14E-03	7.14E-03
6	Node 12 vs. Node 46	0.00E+00	8.33E-03	8.33E-03
5	Node 12 vs. Node 19	0.00E+00	1.00E-02	1.25E-02
4	Node 27 vs. Node 46	0.00E+00	1.25E-02	1.25E-02
3	Node 52 vs. Node 70	0.00E+00	1.67E-02	1.67E-02
2	Node 46 vs. Node 52	0.00E+00	2.50E-02	2.50E-02
1	Node 12 vs. Node 27	0.00E+00	5.00E-02	5.00E-02

Note: Holm's procedure rejects those hypotheses that have an unadjusted p-value ≤ 3.33E-03; Shaffer's procedure rejects those hypotheses that have an unadjusted p-value ≤3.33E-03.

Table 6. *p*-values of the comparing nodes on the synthetic stochastic BA networks in terms of strength of nodes.

i	Hypothesis	Unadjusted *p*-value	*Holm p*-value	*Shaffer p*-value
15	Node 19 vs. Node 46	0.00E+00	3.33E-03	3.33E-03
14	Node 19 vs. Node 70	0.00E+00	3.57E-03	5.00E-03
13	Node 27 vs. Node 46	0.00E+00	3.85E-03	5.00E-03
12	Node 27 vs. Node 70	0.00E+00	4.17E-03	5.00E-03
11	Node 19 vs. Node 52	0.00E+00	4.55E-03	5.00E-03
10	Node 12 vs. Node 46	0.00E+00	5.00E-03	5.00E-03
9	Node 12 vs. Node 70	0.00E+00	5.56E-03	7.14E-03
8	Node 27 vs. Node 52	0.00E+00	6.25E-03	7.14E-03
7	Node 12 vs. Node 19	0.00E+00	7.14E-03	7.14E-03
6	Node 46 vs. Node 52	0.00E+00	8.33E-03	8.33E-03
5	Node 52 vs. Node 70	0.00E+00	1.00E-02	1.25E-02
4	Node 12 vs. Node 52	0.00E+00	1.25E-02	1.25E-02
3	Node 19 vs. Node 27	0.00E+00	1.67E-02	1.67E-02
2	Node 12 vs. Node 27	0.00E+00	2.50E-02	2.50E-02
1	Node 46 vs. Node 70	0.00E+00	5.00E-02	5.00E-02

Note: Holm's procedure rejects those hypotheses that have an unadjusted p-value ≤ 3.33E-03; Shaffer's procedure rejects those hypotheses that have an unadjusted p-value ≤3.33E-03.

Experiment II

In this experiment, we compare the number of samples required by the proposed algorithms in order to reach certain accuracy with the number of samples needed by the standard sampling method (SSM) to reach the same accuracy. In order to have a fair comparison, both the proposed algorithm and SSM must use the same confidence level 1-ε. The proposed algorithm may reach the confidence level of 1-ε by a proper choice of learning parameter a. According to (A. Rezvanian & Meybodi, 2015a, 2015b), such learning parameter can be estimated using $ax/(e^{ax}-1) = max_{i\neq j}(d_j/d_i)$ where d_i is the reward probability of action α_i. Based on the SSM, to obtain a certain confidence level 1-ε for a subset of edges in stochastic graph, we need to build a confidence level 1-ε_i for each edge e_i such that $\varepsilon = \sum_{i=1}^{m} \varepsilon_i/m$. In this experiment, same confidence level 1-ε_0 is assumed for all edges of the stochastic graph. The minimum required number of samples for each edge of graph for SSM is calculated subject to $p[|\bar{x}_n - \mu| < \delta] \geq 1-\varepsilon$, where δ=0.01.

For this experimentation, error rate ε varies from 0.3 to 0.05 with increment 0.05 (confidence levels 0.7 to 0.95) for the proposed algorithms. Different synthetic stochastic graphs with size from 1000 to 10000 are used. The results of this experimentation are averages taken over all the test networks with respect to edge sampling rate (ESR) which is defined as the average number of samples that is needed to be taken from each edge of stochastic graphs to reach a certain confidence level (CL). The average and standard deviation of ESR are given in Table 7 through Table 9 for synthetic stochastic ER, BA and WS graphs, respectively. In order to show the significant statistical difference among the proposed algorithms, a non-parametric statistical test called Wilcoxon rank sum test is conducted for independent samples (García et al., 2009; Wilcoxon, 1945) at the 0.05 significance level. In tables 7-9, the test result regarding corresponding Algorithm versus Algorithm 6 is shown as "†", "§" and "≈" when corresponding algorithm is better than, worse than, and similar to that of Algorithm 6, respectively. From the results, several conclusions can be made: 1) For all the proposed algorithms, the average number of samples taken from each edge of graph is fewer than the number of samples taken using SSM for all synthetic stochastic graphs and different confidence levels; 2) From the results, we may observe that Algorithm 6 requires the lowest number of samples from each edge as compared to algorithms 1 through 4 for all synthetic stochastic graphs and different confidence levels.

Table 7. Average results [± standard deviation (std)] for SSM and the proposed algorithms for synthetic stochastic ER graphs with different error rates in terms of ESR.

Methods \ CL	0.7	0.75	0.8	0.85	0.9	0.95
SSM	24.54 ± 0.77§	30.00 ± 1.08§	36.98 ± 1.34§	46.42 ± 1.69§	60.26 ± 2.20§	85.15 ± 3.13§
Algorithm 1	9.49 ± 2.82§	12.66 ± 3.62§	17.46 ± 3.06§	24.50 ± 5.27§	31.37 ± 6.08§	52.38 ± 7.49§
Algorithm 2	9.72 ± 1.43§	12.19 ± 1.62§	14.40 ± 2.24§	19.78 ± 4.33§	26.50 ± 2.57§	48.03 ± 9.23§
Algorithm 3	8.95 ± 1.21§	11.22 ± 0.95§	13.93 ± 0.79§	18.24 ± 3.67§	24.74 ± 6.24§	43.44 ± 6.22§
Algorithm 4	10.21 ± 3.95§	12.52 ± 4.01§	16.84 ± 5.60§	24.35 ± 6.85§	32.36 ± 10.04§	50.77 ± 8.31§
Algorithm 5	7.72 ± 1.47≈	8.95 ± 1.55≈	12.12 ± 0.75§	15.55 ± 2.81≈	21.43 ± 2.57≈	40.43 ± 6.82≈
Algorithm 6	7.49 ± 1.38	9.39 ± 1.42	11.78 ± 0.54	14.56 ± 1.87	20.82 ± 2.68	42.72 ± 8.69

Note: "†" and "§" indicate a 0.05 level of significance by Wilcoxon rank sum test. "†", "§" and "≈" denote that the performance of the corresponding algorithm is better than, worse than, and similar to that of Algorithm 6, respectively.

Table 8. Average results [± standard deviation (std)] for SSM and the proposed algorithms for synthetic stochastic BA graphs with different error rates in terms of ESR.

Methods \ CL	0.7	0.75	0.8	0.85	0.9	0.95
SSM	24.66 ± 3.39§	30.16 ± 2.28§	37.19 ± 2.82§	46.65 ± 3.55§	60.59 ± 4.65§	85.62 ± 6.59§
Algorithm 1	9.27 ± 1.74§	12.83 ± 1.70§	15.04 ± 1.78§	19.19 ± 1.94§	27.18 ± 2.57§	43.45 ± 4.39§
Algorithm 2	9.77 ± 1.19§	12.48 ± 1.36§	15.55 ± 2.24§	19.56 ± 2.15§	24.98 ± 2.53§	42.62 ± 5.86§
Algorithm 3	8.92 ± 1.32§	11.98 ± 2.09§	15.36 ± 4.43§	21.19 ± 7.70§	23.50 ± 2.23§	38.61 ± 2.81§
Algorithm 4	10.09 ± 0.49§	11.91 ± 1.19§	14.30 ± 1.49§	17.75 ± 1.94§	24.95 ± 3.56§	40.59 ± 6.18§
Algorithm 5	7.51 ± 1.53≈	8.75 ± 1.13≈	11.46 ± 1.87§	16.04 ± 1.25§	21.72 ± 3.28≈	35.22 ± 6.89§
Algorithm 6	7.29 ± 1.52	8.31 ± 1.81	10.51 ± 1.84	14.88 ± 1.63	21.30 ± 5.76	30.86 ± 3.03

Note: "†" and "§" indicate a 0.05 level of significance by Wilcoxon rank sum test. "†", "§" and "≈" denote that the performance of the corresponding algorithm is better than, worse than, and similar to that of Algorithm 6, respectively.

Table 9. Average results [± standard deviation (std)] for SSM and the proposed algorithms for synthetic stochastic WS graphs with different error rates in terms of ESR.

Methods \ CL	0.7	0.75	0.8	0.85	0.9	0.95
SSM	25.58 ± 0.80§	31.29 ± 1.10§	38.58 ± 1.36§	48.42 ± 1.73§	62.90 ± 2.24§	88.89 ± 3.19§
Algorithm 1	10.92 ± 1.42§	13.43 ± 2.17§	15.87 ± 2.68§	18.84 ± 2.15§	28.85 ± 6.78§	47.61 ± 8.80§
Algorithm 2	9.65 ± 1.08§	12.21 ± 2.63§	16.25 ± 3.01§	18.72 ± 3.01§	26.92 ± 5.63§	43.57 ± 9.17§
Algorithm 3	9.89 ± 1.45§	11.21 ± 1.20§	15.62 ± 4.47§	19.34 ± 5.45§	25.07 ± 4.68§	45.07 ± 6.14§
Algorithm 4	9.17 ± 0.73§	10.95 ± 1.19§	14.82 ± 1.52§	17.82 ± 2.48§	26.04 ± 4.99§	45.27 ± 6.83§
Algorithm 5	8.73 ± 1.10§	9.88 ± 1.35≈	12.54 ± 1.02§	16.17 ± 1.62≈	21.28 ± 3.46≈	41.83 ± 4.62§
Algorithm 6	8.04 ± 1.21	9.17 ± 1.39	11.22 ± 1.08	15.00 ± 1.62	19.54 ± 2.94	38.86 ± 4.22

Note: "†" and "§" indicate a 0.05 level of significance by Wilcoxon rank sum test. "†", "§" and "≈" denote that the performance of the corresponding algorithm is better than, worse than, and similar to that of Algorithm 6, respectively.

Moreover, a multi-comparison statistical tests using Friedman and Iman–Davenport with 95% confidence level (α=0.05) among the proposed algorithms are also conducted (García et al., 2009) for all synthetic stochastic graphs. As a statistical analysis, Friedman's test is first applied to obtain rankings. To obtain the adjusted *p*-values for each comparison between the control algorithm (the best-performing one) and the other algorithms, Nemenyi, Holm and Shaffer tests are conducted as post-hoc methods (if significant differences are detected). The rankings obtained by Friedman's test are presented in Table 10. The *p*-value computed by Friedman's test is 4.64E-11, which is below the significance interval of 95% (α=0.05). Thus, a significant difference exists among the observed results. Post-hoc methods (Nemenyi, Holm and Shaffer tests) are also performed to obtain the adjusted *p*-values. Table 11 shows the adjusted *p*-values of the Nemenyi, Holm and Shaffer tests. According to the results of statistical significance in Table 10, one can conclude that Algorithm 6 outperforms the other proposed algorithms in terms of average number of samples taken from each edge.

Table 10. Average ranking of Friedman's test of comparison algorithms on the stochastic test networks.

Test results	Algorithm 1	Algorithm 2	Algorithm 3	Algorithm 4	Algorithm 5	Algorithm 6
Average ranking	5.38	4.61	3.77	4.22	1.88	1.11
Ranking	6	5	3	4	2	1

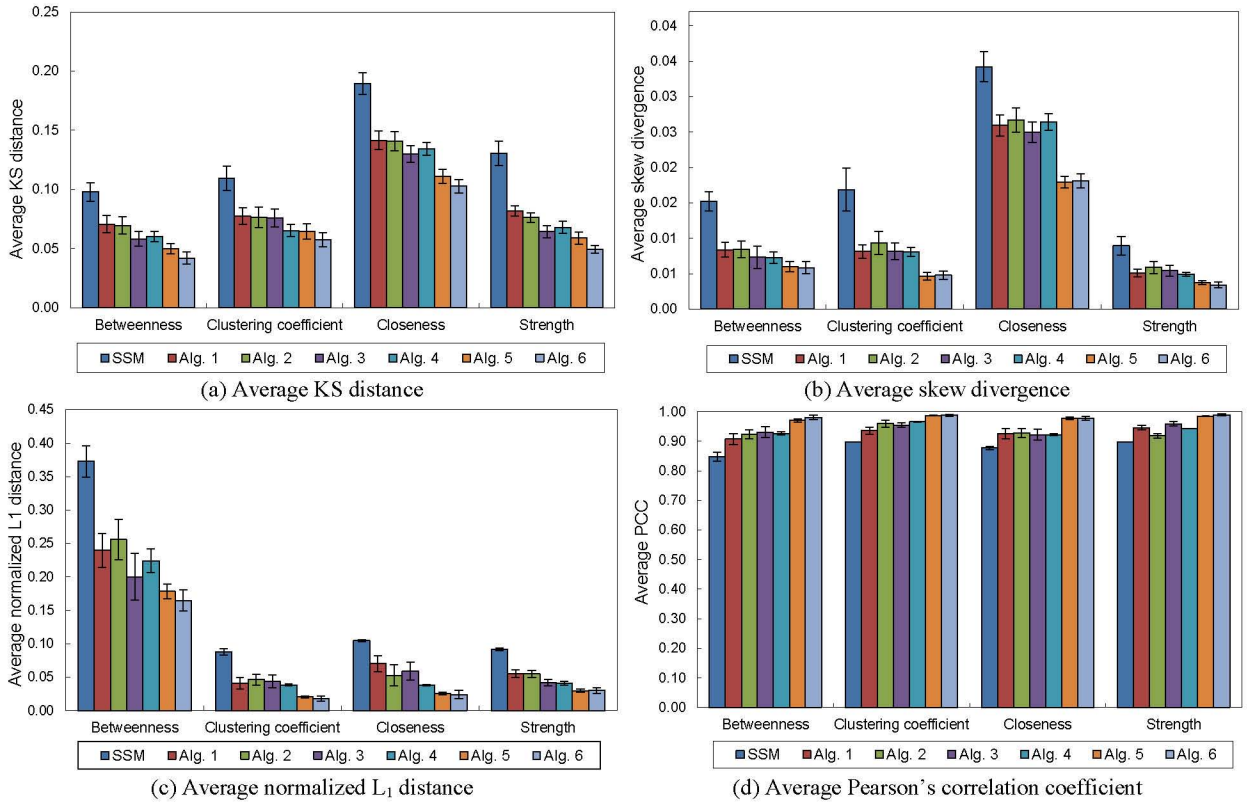
Table 11. *p*-values of the comparing algorithms on the stochastic test networks.

i	Hypothesis	Unadjusted *p*-value	*Nemenyi p*-value	*Holm p*-value	*Shaffer p*-value
15	Algorithm 1 vs. Algorithm 6	0.00E+00	0.00E+00	0.00E+00	0.00E+00
14	Algorithm 1 vs. Algorithm 5	0.00E+00	0.00E+00	0.00E+00	0.00E+00
13	Algorithm 2 vs. Algorithm 6	0.00E+00	0.00E+00	0.00E+00	0.00E+00
12	Algorithm 4 vs. Algorithm 6	1.00E-06	9.00E-06	7.00E-06	6.00E-06
11	Algorithm 2 vs. Algorithm 5	1.30E-05	1.90E-04	1.40E-04	1.27E-04
10	Algorithm 3 vs. Algorithm 6	1.90E-05	2.85E-04	1.90E-04	1.90E-04
9	Algorithm 4 vs. Algorithm 5	1.83E-04	2.74E-03	1.65E-03	1.28E-03
8	Algorithm 3 vs. Algorithm 5	2.45E-03	3.68E-02	1.96E-02	1.72E-02
7	Algorithm 1 vs. Algorithm 3	9.78E-03	1.47E-01	6.85E-02	6.85E-02
6	Algorithm 1 vs. Algorithm 4	6.14E-02	9.21E-01	3.68E-01	3.68E-01
5	Algorithm 2 vs. Algorithm 3	1.81E-01	2.72E+00	9.07E-01	7.26E-01
4	Algorithm 5 vs. Algorithm 6	2.12E-01	3.18E+00	9.07E-01	8.49E-01
3	Algorithm 1 vs. Algorithm 2	2.12E-01	3.18E+00	9.07E-01	8.49E-01
2	Algorithm 3 vs. Algorithm 4	4.76E-01	7.14E+00	9.52E-01	9.52E-01
1	Algorithm 2 vs. Algorithm 4	5.33E-01	7.99E+00	9.52E-01	9.52E-01

Note: Nemenyi's procedure rejects those hypotheses that have an unadjusted p-value ≤ 6.67E-03; Holm's procedure rejects those hypotheses that have an unadjusted p-value ≤ 1.67E-02; Shaffer's procedure rejects those hypotheses that have an unadjusted p-value ≤ 6.67E-03.

Experiment III

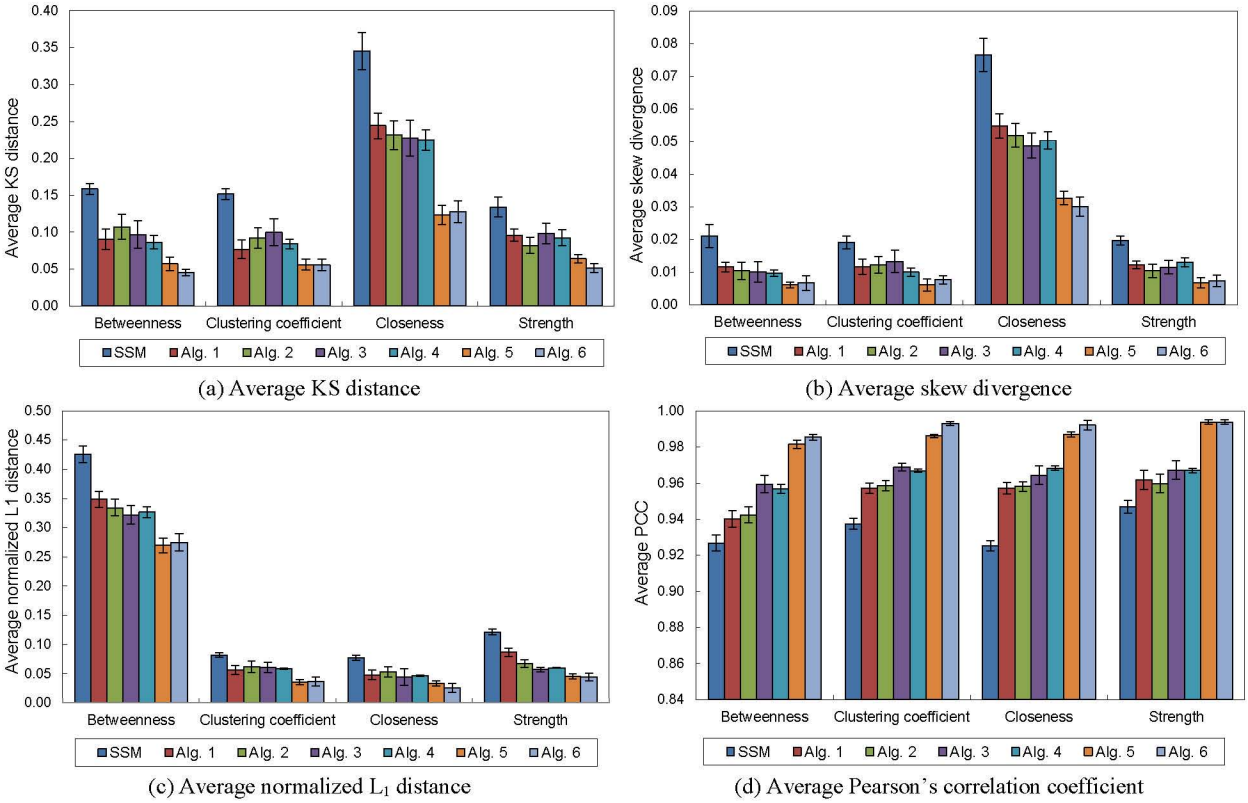
This experiment is conducted to study the performance of the proposed algorithms (Alg. 1, Alg. 2, Alg. 3, Alg. 4, Alg. 5 and Alg. 6) and standard sampling method (SSM) for sampling from stochastic graphs in terms of different distance metrics. For this experimentation, different synthetic stochastic graphs (BA, WS and ER) with size from 1000 to 10000 are used. The results of this experimentation are averages taken over all the test networks with respect to different distance metrics: Kolmogorov-Smirnov (KS) distance, skew divergence (SD), normalized L_1 distance (ND) and Pearson's correlation coefficient (PCC) for estimated measures including: betweenness, clustering coefficient, closeness and strength. The results of average and standard error with 95% confidence interval of mentioned metrics are given in Figure 7 through figure 9 for synthetic stochastic graphs. From the figures 7-9, Algorithm 6 outperforms other proposed algorithms and standard sampling method for all test graphs and standard sampling method achieves the low accuracy in terms of KS, SD, ND and PCC for mentioned metrics. For all test graphs, the results for betweenness, clustering coefficient and strength is more reliable than the results for closeness in terms of KS, SD and ND. Also, the results for betweenness in terms of ND are larger than the results for other metrics.



(a) Average KS distance (b) Average skew divergence

(c) Average normalized L_1 distance (d) Average Pearson's correlation coefficient

Figure 7. Comparing results of average KS distances, average skew divergence, average normalized L_1 distance and average Pearson's correlation coefficient over betweenness, clustering coefficient, closeness and strength for synthetic stochastic ER graphs.



(a) Average KS distance (b) Average skew divergence

(c) Average normalized L_1 distance (d) Average Pearson's correlation coefficient

Figure 8. Comparing results of average KS distance, average skew divergence, average normalized L1 distance and average Pearson's correlation coefficient over betweenness, clustering coefficient, closeness and strength for synthetic stochastic BA graphs.

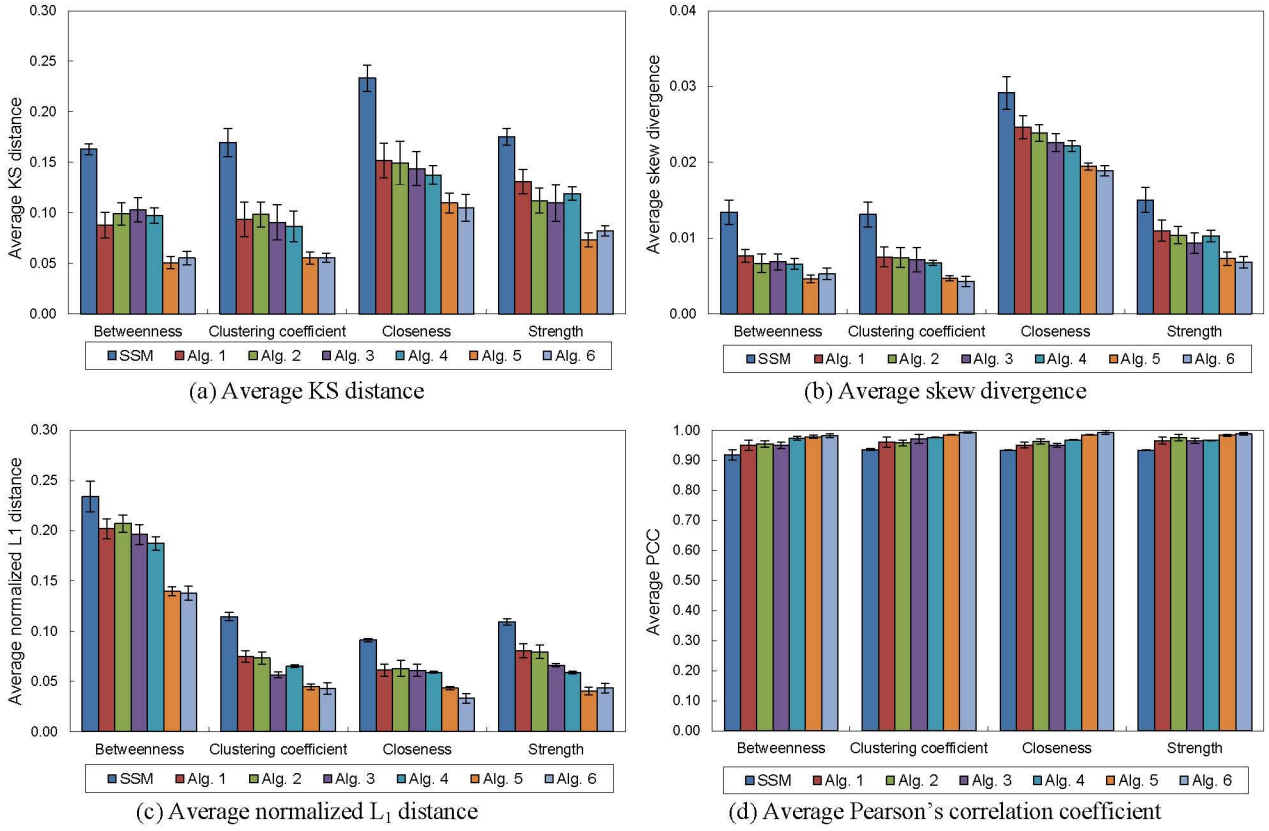
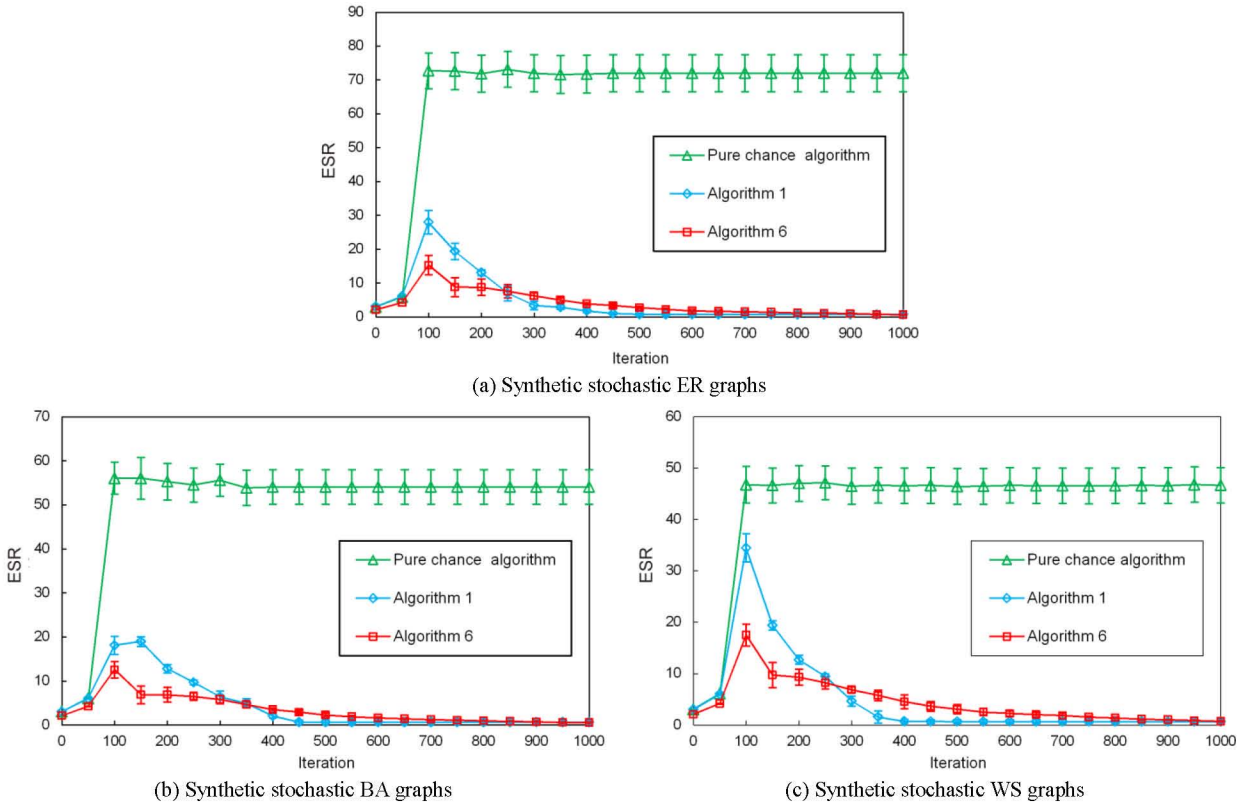


Figure 9. Comparing results of average KS distance, average skew divergence, average normalized L_1 distance and average Pearson's correlation coefficient over betweenness, clustering coefficient, closeness and strength for synthetic stochastic WS graphs.

Experiment IV

This experiment is conducted to compare the proposed algorithms (Algorithm 1 and algorithm 6) with the pure chance algorithm (algorithm 1 in which the leaning automaton residing in each node is replaced by a pure chance automaton) with respect to the number of samples taken from the edges of the graph. In pure chance automaton, the actions are chosen with equal probabilities. In order to perform this experiment, we plot average number of samples taking from each edge of graph (ESR) during the execution of the algorithms for calculating network measures versus iteration number for both proposed algorithms (Algorithm 1 and Algorithm 6) and pure chance algorithm. The results of this experiment are taken averages for each synthetic stochastic test graphs and are given in Figure 10. In Figure 10, the points along the curves show the average value and the error bars represent 95% confidence interval. As it is shown, as the algorithms proceeds, the number of samples taken from each edge (ESR) abundantly increases in initial iterations for all algorithms and then reduces and gradually approaches zero for Algorithm 1 and Algorithm 6, however ESR value for pure chance algorithm remains unchanged. This implies that for the proposed algorithms, the average number of samples taken from each edge of the graph gradually approaches zero. And also it is done with fewer numbers of samples taken from the graph as compared to the case where learning is absent that indicates the important impact of learning on the superiority of the proposed learning automaton based algorithms for calculating network measures in stochastic graphs. From the error bars shown in Figure 11, we conclude that the variances in the results decreases as the iteration number increases. According to the results for comparison among network

models, we can see that for Synthetic stochastic ER graphs, the average number of samples taking from each edge of graph is much more than that of both Synthetic stochastic BA and WS graphs. Similar result can be obtained for other algorithms.



(a) Synthetic stochastic ER graphs

(b) Synthetic stochastic BA graphs

(c) Synthetic stochastic WS graphs

Figure 10. The plot of average cumulative *ESR* versus iteration number for stochastic test graphs.

Experiment V

In this experiment, we study the convergence behavior of the proposed algorithms using information entropy (as defined in equation (12)) of probability vector of learning automata residing in the nodes of the graph responsible for deciding whether or not taking samples from the edges of graph. For this purpose, for each algorithm we plot the average information entropy taken over all leaning automata residents in the nodes versus iteration number as given in Figure 11 for three kinds of synthetic stochastic graphs as mentioned before (ER-2000, ER-5000, ER-10000; BA-2000, BA-5000, BA-10000; WS-2000, WS-5000, WS-10000). As it is shown, for every algorithm the average entropy decreases as the algorithm proceeds. Using the fact the average entropy decreases as the algorithm proceeds and also the result of previous experiment, we may conclude that the learning automaton of each node gradually converges to the action of "do not take sample from the edges of chosen node". We can also conclude that Algorithm 2 has the highest and Algorithm 6 has the lowest speed of convergence. The convergence behaviors of the proposed algorithms among the network models indicate that the difference between algorithms 5-6 and algorithms 1-4 for synthetic stochastic small world graphs is more obvious than other network models.

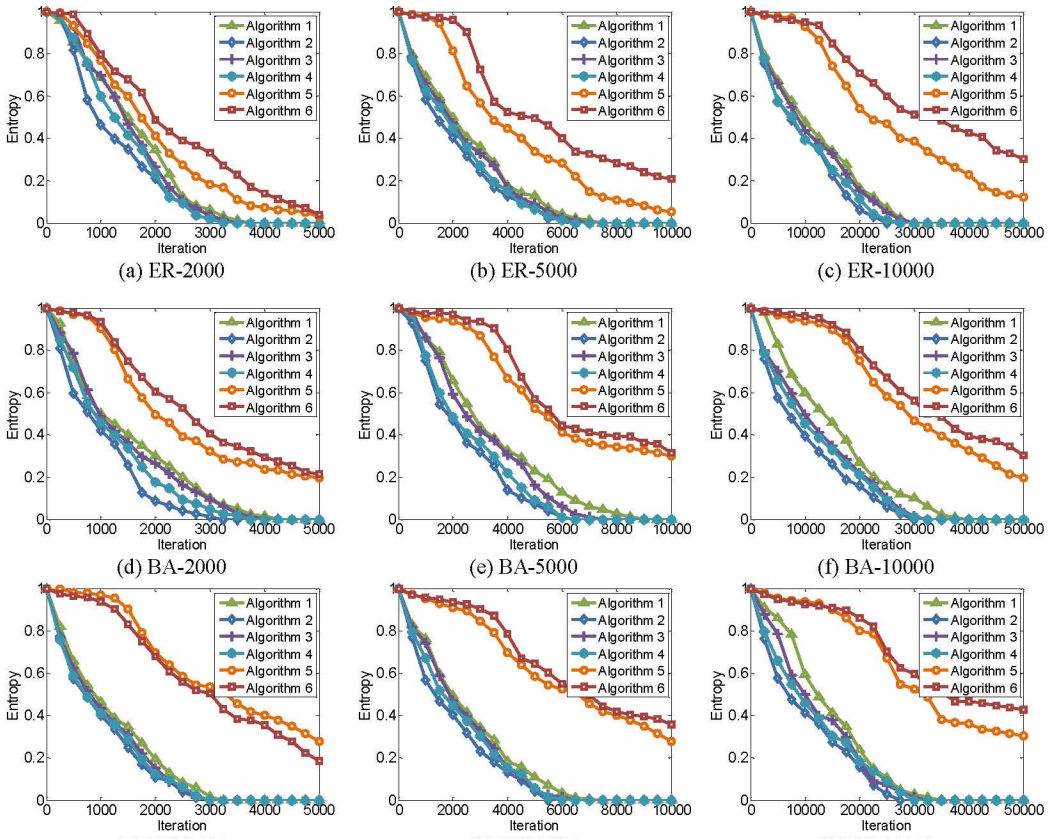


Figure 11. Comparison of convergence behaviors of the proposed algorithms with respect to the information entropy for synthetic stochastic graphs.

6. Discussion

This study argued about the important role of activities of online users as the main features of online social networks for generating and sharing information through social networks. In general, activities of users in online social networks such as interacting with their friends, answering comments of their friends, participating in online communities, posting and sharing new information contents on the page of their friends, visiting profile pages of their friends, taking comment or like on a post of their friends are considered as the weights associated with the edges of graph and due to nature of user activities in social network these weights may be uncertain, unpredictable and time-varying. Therefore in this study, we proposed that stochastic graphs, in which weights associated with the edges are random variables, may be a better candidate as a graph model for social network analysis. We assumed that the structural and behavioral parameters of online social networks which are time varying parameters can be observed by the network analyzers. Using the observed parameters, the network analyzers can calculate different measurements such as strength, closeness and betweenness in an ongoing fashion. For example, one may want to find a user which can influence a large number of users by spreading of influence in a social network for marketing goals or control the information flows over communication between other users for managing misinformation. This may be easily realized by calculating network centralities using one of the proposed algorithms as the network operates. The proposed algorithms for calculating network measures can be used by the network analyzers as a means for observing the time varying parameters of the network for the purpose of network's measurements computations."

In the simulation, it was showed that modeling social networks as stochastic graph models by applying the framework of learning automation can be very beneficial to reach good results with a certain confidence level. Since, in many fundamental studies, researchers generate synthetic graphs similar to real networks in order to further simulations and investigate a number of phenomena on a computer generated graph models in order to be able to exploit their structures and dynamics of a network in a systematic manner, in the simulation, it was used several synthetic stochastic networks such as *Barabási-Albert* model (Barabási & Albert, 1999) as a synthetic scale-free network, *Watts–Strogatz* model (Watts & Strogatz, 1998) as a synthetic small world network and *Erdös–Rényi* model (Erdos & Rényi, 1960) as a synthetic random network in which edge weights are assumed to be random variables with exponential distributions. Furthermore, it was indicated that how the proposed algorithms can collect information from the network in order to find good estimates for the network's measurements using fewer numbers of samples than that of standard sampling methods. The overall results confirm that using two set of learning automata can be achieve better results than that of one set of learning automata in terms of accuracy and convergence behavior.

Analyzing social networks and its user behavior with stochastic graph models can be applied for some applications. Such application can be in viral marketing in such a way that companies want to exploit user behavior models and then spread or promote their new products or services to predict the user interactions and process of adoption of an idea or spread them. Another example can be in domain of knowledge based systems by understanding the patterns of user participation to generate new contents, propagating information on the networks via users, detecting active users from spamming users, attracting new users and keeping some users, predicting the trends of topics in user communities, and performing efficient content management.

In summary, due to important role of modeling user behavior of online social networks in quantitative manner and variety of user interactions on the structure of social networks, analyzing social networks with stochastic graph models may be a better candidate as a graph model for analysis of social network and its user behaviors which can take into consideration the continuum of behavioral parameters of network occurring over time.

7. Conclusions

The conventional social network models solely consider either the existence of connections between users in the form of binary networks or consider fixed weights for the edges in the form of weighted networks. In this paper, we first proposed stochastic graph as a model for social networks and then redefined some of the social network measurements to be applicable to stochastic graphs. Also several algorithms based on learning automata were designed for finding good estimations of these measurements for the purpose of analysis. We believe that the approach proposed in this paper for modeling and analysis of social networks can provide a better way for studying online social networks such analysis of user behavior and online human activities. In the future, we would like to generalize our measures and algorithms for some applications of social network analysis such as network sampling and information diffusion.

References

Akbari Torkestani, J., & Meybodi, M. R. (2012). A learning automata-based heuristic algorithm for solving the minimum spanning tree problem in stochastic graphs. *The Journal of Supercomputing*, *59*(2), 1035–1054.

Aliakbary, S., Habibi, J., & Movaghar, A. (2015). Feature extraction from degree distribution for comparison and analysis of complex networks. *The Computer Journal*, *58*(9), 2079–2091.

Badie, R., Aleahmad, A., Asadpour, M., & Rahgozar, M. (2013). An efficient agent-based algorithm for overlapping community detection using nodes' closeness. *Physica A: Statistical Mechanics and Its Applications*, *392*(20), 5231–5247.

Barabási, A. L., & Albert, R. (1999). Emergence of scaling in random networks. *Science*, *286*(5439), 509–512.

Beigy, H., & Meybodi, M. R. (2006). Utilizing distributed learning automata to solve stochastic shortest path problems. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, *14*(5), 591–615.

Bild, D. R., Liu, Y., Dick, R. P., Mao, Z. M., & Wallach, D. S. (2015). Aggregate characterization of user behavior in Twitter and analysis of the retweet graph. *ACM Transactions on Internet Technology (TOIT)*, *15*(1), 4.

Borgatti, S. P. (2005). Centrality and network flow. *Social Networks*, *27*(1), 55–71.

Buccafurri, F., Lax, G., Nicolazzo, S., & Nocera, A. (2015). Comparing twitter and facebook user behavior: privacy and other aspects. *Computers in Human Behavior*, *52*, 87–95.

Burke, C. J., Estes, W. K., & Hellyer, S. (1954). Rate of verbal conditioning in relation to stimulus variability. *Journal of Experimental Psychology*, *48*(3), 153.

Cha, M., Kwak, H., Rodriguez, P., Ahn, Y.-Y., & Moon, S. (2007). I tube, you tube, everybody tubes: analyzing the world's largest user generated content video system. In *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement* (pp. 1–14). ACM.

Costa, L. F., Rodrigues, F. A., Travieso, G., & Boas, P. R. V. (2007). Characterization of complex networks: A survey of measurements. *Advances in Physics*, *56*(1), 167–242.

Ding, F., Cheng, H., Si, X.-M., Liu, Y., Xiong, F., & Shen, B. (2010). Read and reply behaviors in a BBS social network. In *2010 2nd International Conference on Advanced Computer Control (ICACC)* (Vol. 4, pp. 571–576). IEEE.

Erdos, P., & Rényi, A. (1960). On the evolution of random graphs. *Publications of the Mathematical Institute of the Hungarian Academy of Sciences*, *5*, 17–61.

Estes, W. K. (1950). Toward a statistical theory of learning. *Psychological Review*, *57*(2), 94.

Falck-Ytter, M., & Øverby, H. (2012). An Empirical Study of Valuation and User Behavior in Social Networking Services. In *World Telecommunications Congress (WTC)* (pp. 1–6). IEEE.

Feng, Z., Cong, F., Chen, K., & Yu, Y. (2013). An empirical study of user behaviors on pinterest social network. In *2013 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT),* (Vol. 1, pp. 402–409). IEEE.

Freeman, L. C. (1979). Centrality in social networks conceptual clarification. *Social Networks*, *1*(3), 215–239.

Galuba, W., Aberer, K., Chakraborty, D., Despotovic, Z., & Kellerer, W. (2010). Outtweeting the twitterers-predicting information cascades in microblogs. In *Proceedings of the 3rd conference on Online social networks* (Vol. 39, pp. 1–9).

García, S., Molina, D., Lozano, M., & Herrera, F. (2009). A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: a case study on the CEC'2005 special session on real parameter optimization. *Journal of Heuristics*, *15*(6), 617–644.

Girvan, M., & Newman, M. E. J. (2002). Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, *99*(12), 7821–7826.

Golder, S. A., Wilkinson, D. M., & Huberman, B. A. (2007). Rhythms of social interaction: Messaging within a massive online network. In *Communities and technologies 2007* (pp. 41–66). Springer.

Guo, L., Tan, E., Chen, S., Zhang, X., & Zhao, Y. E. (2009). Analyzing patterns of user content generation in online social networks. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 369–378). ACM.

Gyarmati, L., & Trinh, T. A. (2010). Measuring user behavior in online social networks. *IEEE Network*, *24*(5), 26–31.

Jalali, Z. S., Rezvanian, A., & Meybodi, M. R. (2015). A two-phase sampling algorithm for social networks. In *2015 2nd International Conference on Knowledge-Based Engineering and Innovation (KBEI)* (pp. 1165–1169). IEEE.

Jalali, Z. S., Rezvanian, A., & Meybodi, M. R. (2016). Social network sampling using spanning trees. *International Journal of Modern Physics C*, *27*(5), 1650052.

Janssen, J., Hurshman, M., & Kalyaniwalla, N. (2012). Model selection for social networks using graphlets. *Internet Mathematics*, *8*(4), 338–363.

Jin, L., Chen, Y., Wang, T., Hui, P., & Vasilakos, A. V. (2013). Understanding user behavior in online social networks: A survey. *IEEE Communications Magazine*, *51*(9), 144–150.

Khomami, M. M. D., Rezvanian, A., & Meybodi, M. R. (2016). Distributed learning automata-based algorithm for community detection in complex networks. *International Journal of Modern Physics B*, *30*, 1650042.

Kwak, H., Lee, C., Park, H., & Moon, S. (2010). What is Twitter, a social network or a news media? In *Proceedings of the 19th international conference on World wide web* (pp. 591–600). ACM.

Leskovec, J., Backstrom, L., Kumar, R., & Tomkins, A. (2008). Microscopic evolution of social networks. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 462–470). ACM.

Leskovec, J., McGlohon, M., Faloutsos, C., Glance, N. S., & Hurst, M. (2007). Patterns of Cascading behavior in large blog graphs. In *SDM* (Vol. 7, pp. 551–556). SIAM.

Li, Y., Wu, C., Wang, X., & Luo, P. (2014). A network-based and multi-parameter model for finding influential authors. *Journal of Informetrics*, *8*(3), 791–799.

Liu, H., Nazir, A., Joung, J., & Chuah, C.-N. (2013). Modeling/predicting the evolution trend of osn-based applications. In *Proceedings of the 22nd international conference on World Wide Web* (pp. 771–780). ACM.

Luo, P., Li, Y., Wu, C., & Zhang, G. (2015). Toward cost-efficient sampling methods. *International Journal of Modern Physics C*, *26*(5), 1550050.

Mahdaviani, M., Kordestani, J. K., Rezvanian, A., & Meybodi, M. R. (2015). LADE: Learning Automata Based Differential Evolution. *International Journal on Artificial Intelligence Tools*, *24*(6), 1550023.

Mofrad, M. H., Sadeghi, S., Rezvanian, A., & Meybodi, M. R. (2015). Cellular edge detection: Combining cellular automata and cellular learning automata. *AEU-International Journal of Electronics and Communications*, *69*(9), 1282–1290.

Morales, A. J., Losada, J. C., & Benito, R. M. (2012). Users structure and behavior on an online social network during a political protest. *Physica A: Statistical Mechanics and Its Applications*, *391*(21), 5244–5253.

Mousavian, A., Rezvanian, A., & Meybodi, M. R. (2013). Solving Minimum Vertex Cover Problem Using Learning Automata. In *proceedings of 13th iranian conference on fuzzy systems (IFSC 2013)* (pp. 1–5).

Mousavian, A., Rezvanian, A., & Meybodi, M. R. (2014). Cellular learning automata based algorithm for solving minimum vertex cover problem. In *2014 22nd Iranian Conference on Electrical Engineering (ICEE)* (pp. 996–1000). IEEE.

Narendra, K. S., & Thathachar, M. A. L. (1989). *Learning Automata: An Introduction*. Printice-Hall.

Nazir, A., Raza, S., & Chuah, C.-N. (2008). Unveiling facebook: a measurement study of social network based applications. In *Proceedings of the 8th ACM SIGCOMM conference on Internet measurement* (pp. 43–56). ACM.

Newman, M. E. J. (2005). A measure of betweenness centrality based on random walks. *Social Networks*, *27*(1), 39–54.

Rezvanian, A., & Meybodi, M. R. (2015a). Finding maximum clique in stochastic graphs using distributed learning automata. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, *23*(1), 1–31.

Rezvanian, A., & Meybodi, M. R. (2015b). Finding Minimum Vertex Covering in Stochastic Graphs: A Learning Automata Approach. *Cybernetics and Systems*, *46*(8), 698–727.

Rezvanian, A., & Meybodi, M. R. (2016). A new learning automata-based sampling algorithm for social networks. *International Journal of Communication Systems*, *in-press*(DOI: 10.1002/dac.3091), 1–21.

Rezvanian, A., Rahmati, M., & Meybodi, M. R. (2014). Sampling from complex networks using distributed learning automata. *Physica A: Statistical Mechanics and Its Applications*, *396*, 224–234.

Shen, J., Brdiczka, O., & Ruan, Y. (2013). A comparison study of user behavior on Facebook and Gmail. *Computers in Human Behavior*, *29*(6), 2650–2655.

Soleimani-Pouri, M., Rezvanian, A., & Meybodi, M. R. (2012). Solving maximum clique problem in stochastic graphs using learning automata. In *2012 Fourth International Conference on Computational Aspects of Social Networks (CASoN)* (pp. 115–119).

Soleimani-Pouri, M., Rezvanian, A., & Meybodi, M. R. (2014). Distributed Learning Automata based Algorithm for Solving Maximum Clique Problem in Stochastic Graphs. *International Journal of Computer Information Systems and Industrial Management Applications*, *6*, 484–493.

Vongsingthong, S., Boonkrong, S., Kubek, M., & Unger, H. (2015). On the Distributions of User Behaviors in Complex Online Social Networks. In *Recent Advances in Information and Communication Technology 2015* (pp. 237–246). Springer.

Watts, D. J., & Strogatz, S. H. (1998). Collective dynamics of "small-world" networks. *Nature*, *393*(6684), 440–442.

Wilcoxon, F. (1945). Individual comparisons by ranking methods. *Biometrics Bulletin*, *1*(6), 80–83.

Yan, Q., Wu, L., & Zheng, L. (2013). Social network based microblog user behavior analysis. *Physica A: Statistical Mechanics and Its Applications*, *392*(7), 1712–1723.

Zhong, E., Fan, W., Wang, J., Xiao, L., & Li, Y. (2012). Comsoc: adaptive transfer of user behaviors over composite social network. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 696–704). ACM.

Zhu, Z., Su, J., & Kong, L. (2015). Measuring influence in online social network based on the user-content bipartite graph. *Computers in Human Behavior*, *52*, 184–189.