Network Topology Impact on Influence Spreading

Sasho Gramatikov, Dimitar Trajanov, Ljupco Kocarev, Aksenti Grnarov
Dept. of Computer Sciences
Faculty of Electrical Engineering and Information Techonology
University Ss. Cytil and Methodious Skopje
Skopje, R. Macedonia
{saso.gramatikov, mite, lkocarev, grnarov}@feit.ukim.edu.mk

***Abstract* — Networks composed of large number of nodes interacting in structured ways such as power grids, communication networks, social networks or market networks are critical port of the world's infrastructure. Many times minor changes of the sate in some of the nodes can spread rapidly and cause major effects in the network, so understanding the behavior of the complex network due to changes of the state is of a great interest. In this paper we present influence spreading in networks driven by the influence model. We also show how the topology and connectivity of a network affect the spread of influence.**

***Keywords* — influence, complex networks, standard deviation.**

I. Introduction

Networks containing large number of nodes connected in a specific way are common topic of research, because inner changes caused by some events make the network a dynamic system with certain behavior specific to network topology and variety of parameters of the network. The nature of the network dynamic is versatile and what is nowadays most commonly studied is spreading of failures of nodes, spreading of information, influence, or spreading of computer and natural viruses. Once an event is introduced in the network, nodes react to it, possibly changing their state, and according to their influence level in the network, they try to spread their current state to the nodes they are connected to.

For example, let us take failure of a router as an event in internet network. The event causes increased rerouted traffic in the neighbor routers, causing possible changes in their state. After the failure, each node determines its current state according to the influence it gets from the neighbor nodes. However, the state of each router does not only depend on the offered traffic from the neighbors, it also depends on the possibility of hardware failures or power loss. Similar example are social networks. When an idea is lunched, people make decision whether to accept the idea or not. Since in social network people are connected with weighted links defined according to influence power of each individual, every person gets certain amount of influence from the people it is connected to. Thus, both the attitudes of the environment and the attitude of each individual in a network are important for making certain decision. Clearly, nodes which have high power of influence are not very likely to change their own decision under influence of weak nodes, which on the other side, are susceptible to easy changes due to outer influence.

No matter what is chosen to be an event in the network, there are several possible models of spreading the change in the networks. In [1] two basic diffusion models for spreading influence are used: linear threshold model and independent cascade model, offering algorithms for maximizing the spread. [2] is presenting the spread of computer viruses in a computer network. In [3] the world-wide web is used as a complex network for spreading ideas in the blogosphere. The concept of spreading in complex networks is even popular in the field of marketing by promotion of new products and spreading their popularity to the consumers [4]-[6].

The concept that we would like to introduce in this paper is spreading the influence in a network using the influence model defined in [9]. In this model, nodes are presented with Markov chain. The state of each chain (node) not only depends on the state of its neighbors, but depends on its own current state. We consider networks with nodes that can have two different states.

The goal of this paper is to determine the system state regarding the average number of failed/influenced nodes, as well as its standard deviation for a different configuration of the local Markov chain, different network topologies and different weight calculation algorithms.

In the following text, we first give (section II) a short description of the general influence model and propose structure of local Markov chains for a heterogeneous network where sites are can be interpreted as network routers or individuals in social networks. Afterwards in section III we give different algorithms for calculating the weight of links in such networks. Since we want to analyze the behavior of different real network topologies, in section IV we give a brief overview of the most common topologies of complex network. Results of the behavior analysis of different complex networks, obtained by simulations are presented in section V. Eventually the conclusions of the work in this paper are given in section VI.

II. Influence Model

The influence model is suggested in [9] as a model of random, dynamical interactions on networks. We refer the reader to [9] for a full account of the model and its properties; here we give a brief description of the model.

In the influence model the network is observed at two levels: the network level and the local level. At network level each node is treated as one active entry and is called site. Each site can be in different state, defined at the local level. Looking at local level, each site is presented by a local Markov chain. Each node of the Markov chain is called state and represents the state of the site.

The quantitative measure of the influence that each node has on the neighbors, is defined at the network level with the directed graph $\Gamma(D')$ with nodes from 1 to n. D is nxn stochastic matrix called network influence matrix, containing information about the nodes interconnectivity. The entry d_{ij} has a non zero value only if node i is connected to node j. The magnitude of d_{ij} defines the amount of influence node i exerts on node j. In order to get a network where the influence that each node receives from its neighboring nodes equals one, the graph is defined through the transposed form of D.

On the local level each node is represented with another directed graph $\Gamma(A)$. A is again stochastic matrix with size $m \times m$, called local-state transition matrix, where m is the number of different states that a node can take. The graph defines a Markov chain, where each entry a_{ij} is the transition probability from state i to state j.

At any discrete moment k, the node i has status defined with the vector $s_i[k]=[0...010...0]'$. This vector has only one 1 at position equal to the status of the node. The status of the network, expressed with one vector will be:

$$S[k] = s_1[k] s_2[k] ... s_n[k] \; ' \qquad (1)$$

The probability that node i will have certain status in time k is defined with the vector $p_{\mathrm{i}}[\mathrm{k}]=[p_{\mathrm{i}}(0)\; p_{\mathrm{i}}(1) ... p_{\mathrm{i}}(m)]'$ where $p_{\mathrm{i}}(m)$ is the probability that the Markov chain of node i is in state m. The status probability of the whole network, expressed with one vector will be:

$$P[k] = p_1[k]\, p_2[k] ... p_n[k] \; ' \qquad (2)$$

The evolution of the state at every next time step $k+1$ is related to the probability of the current time step and defined with the equations:

$$S'[k+1] = \mathrm{MultiRealize}(P'[k+1]) \qquad (3)$$

$$P'[k+1] = S'[k]H \qquad (4)$$

The *MultiRealize* operation is equivalent to n independent flipping of a coin. Each outcome of the flipping determines the sate of a node.

H is influence matrix defined as *Kronecker* product of the network matrix D' and the transition matrices of each local chain A_{ij}

$$H = D' \otimes \{A_{ij}\} = \begin{bmatrix} d_{11}A_{11} & ... & d_{n1}A_{n1} \\ & ... & \\ d_{1n}A_{1n} & ... & d_{nn}A_{nn} \end{bmatrix} \qquad (5)$$

H is not a stochastic matrix because its row sum is not one, but it still has some properties of stochastic matrix: it is nonnegative and has 1 as a dominant eigenvalue [9]. A_{ij} represents a transition matrix of a Markov chain and can be any matrix which satisfies the condition $A_{ij} 1_{m_i} = 1_{m_i}$, where m is the number of states of the local Markov chain. Each submatrix $d_{ij}A_{ij}$ of H contains the influence that every single state of node i exerts on node j. That influence can be decomposed in two parts. The first part A_{ij} represents the dynamics of states which influence the state of node j, whereas the second part d_{ij} is a connection specific value which determines the amount of that dynamics that will be used for deciding the state of node i.

According to the value of A_{ij}, the influence model can be homogenous or heterogeneous. In the homogeneous model each node has the same structure of local Markov chain and therefore $A_{ij} = A$. In the heterogeneous model, nodes have different structure of the local Markov chain.

In reality networks are heterogeneous and have local Markov chains which are different for every single node. For example, from a functional aspect of view, in the internet network routers have local Markov chains with two states, on or off. The transition probabilities of each router are different and depend on many factors like, traffic load, maintenance, environmental factors etc. It is the same case in the social networks: individuals have different probabilities of changing their own attitude under no influence. In order to make a model where nodes will not have randomly distributed local Markov chains, we define the structure of A_i by a simple rule: the importance of a node. The main idea comes from the fact that in practice, well connected nodes are of great importance, and therefore are better protected and maintained rather then nodes that are connected with just a few neighbors. Nevertheless, there is a possibility that even those nodes fail because well connected nodes are subdued to a larger demand of service by the neighbor nodes. Therefore better connected nodes are assigned smaller probabilities for failure rather than weakly connected nodes.

For these types of networks, each node i has the same dynamics of influence towards every node it is connected to, and therefore we assume that the local Markov chain for a single node has the same structure in the influence matrix H i.e. $A_{ij} = A_i, \forall j \in \{1,...,n\}$. Although A_i can be of any size, we consider that each node is a two-state Markov chain, with transition matrix A_i.

Let A_i is defined as follows:

$$A_i = \begin{bmatrix} 1-p_i & p_i \\ q_i & 1-q_i \end{bmatrix} \qquad (6)$$

where $1-p_i$ is the probability that once in normal state, node i will remain normal, while p_i is the probability of failure. Seemingly, $1-q_i$ is the probability that a failed node will remain failed, while q_i is the probability that the node will be repaired. Let $p_{\min}$ and $p_{\max}$ are the minimum and maximum values that can be assigned to any p_i in the network. Let $d(i)$ is the degree of node i and $d_{\min}$ and $d_{\max}$ are the minimum and maximum degree of the network. The probability p_i is defined as:

$$p_i = p_{\max} - \frac{p_{\max} - p_{\min}}{d_{\max} - d_{\min}} (d(i) - d_{\min}) \qquad (7)$$

According to (7) each node gets portion of failure probability inversely proportional to the degree. As far as the probability q is concerned, we assume that it has the same value for every node. That means that every failed node is recovered with the same probability.

III. WEIGHT CALCULATION TECHNIQUES

The amount of influence that a link possesses depends on many factors. In the following text we present four different techniques for weight calculation, which take into consideration different aspects of network topology.

A. *Equal weight distribution algorithm*

The simplest solution for assigning weights to incoming links of a node in a network is to take a model where each neighbor of a node implies the same amount of influence. Thus, the influence that node *A* imposes on node *B* decreases as the number of incoming links of *B* increases. Except the influence from the neighbor nodes, each node has the ability of self-influence of a same amount as the influence from other nodes. This results with adding a self-loop in the network graph. Assuming that the total influence a node can get from the neighbors and from itself is:

$$\sum_{i \in V} d_{ij} = 1 \qquad (8)$$

and assuming that the j is connected to m neighboring nodes, then the weight of each link that leads to the node of interest will be

$$d_{ij} = \frac{1}{m+1} \qquad (9)$$

This solution does not give the best results in practice, because the influence each node has on others depends on the number of nodes it is connected to. If we take a simple example in computer networks consisting of few stations generating high traffic flow towards the router they are connected to, joining of another station with very little traffic flow towards the router equally reduces their influence. This means that in real networks of any type, not all the nodes have the same importance and influence to the nodes they are connected to.

B. *Node betweenness algorithm*

In order to get a model closer to the real networks, we assign weights to the links according to the importance of the nodes. For that reason we use the betweenness from the theory of graphs [7], as parameter for the weight calculation algorithm.

Betweenness is a measure of the importance of a node in a network, and is calculated as the fraction of shortest paths between node pairs that pass through the node. Betweenness is, in some sense, a measure of the influence a node has over the flow of information through the network. Let *G* is a graph given with set of nodes *V* and set of edges *E*. Let *s* and *t* are two nodes of the graph. σ_{st} is the number of paths that pass from *s* to *t*. Let $\sigma_{st}(v)$ is the number of shortest paths that pass through the node *v*. The central betweenness of node *v* is:

$$C(v) = \sum_{s \neq v \neq t \in V} \frac{\sigma_{st}(v)}{\sigma_{st}} \qquad (3)$$

For example, let us consider a simple network shown with the directed graph on Fig. 1.

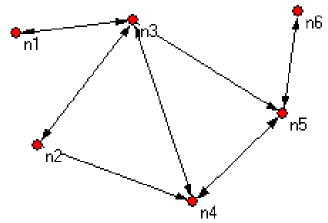

Fig. 1. Simple network

Fig.2 shows importance of each node according to the number of shortest paths that pass thorough it.

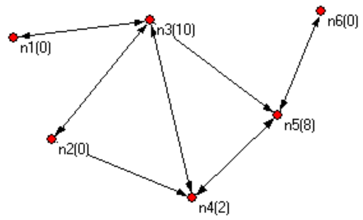

Fig. 2. Simple network with assigned node betweenness

Once we have the betweenness of each node, we aim to assign weight to incoming links according to the value of node betweenness of their originating nodes. Let P_i is a subset of nodes from V that have outgoing links directed to node *i*. Let *C(i)* is the betweenness of node *i*. We assign weight to each incoming link proportional to the originating node betweenness. We first sum the betweenness of all nodes that belong to the set P_i, including the betweenness of node *i*, and then divide the betweenness of each node by the sum. Each link that originates from node $j \in P_i$ is assigned magnitude which is fraction of the total incoming influence, proportional to the fraction of its betweenness in the total sum of betweenness. Taking in consideration that the incoming influence of each node equals 1 (8), the expression for determining the weight of the influence that *j* has on node *i* will be calculated as

$$d_{ji} = \frac{C(i)}{C(i) + \sum_{k \in P_j} C(k)} \qquad (10)$$

C. *Edge betweenness algorithm*

Just like node betweenness denotes the importance of the nodes, the edge betweenness, in the similar way assigns values to links according to their importance. It is calculated as a number of shortest paths that pass through the edge. Let $\delta_{st}(e)$ is the number of shortest paths from *s* to *t* that pass through the edge *e* and δ_{st} is the total number of paths from *s* to *t*. The edge betweenness of edge *e* is:

$$C(e) = \sum_{s \neq t \in V} \frac{\delta_{st}(e)}{\delta_{st}} \quad (11)$$

The weight that is assigned to links leading to a certain node is calculated from the edge betweenness divided by the sum of all incoming links, thus providing that the sum of the incoming influence of a node is one.

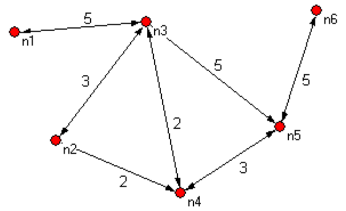

Fig. 3. Simple network with assigned edge betweenness

Fig. 3 shows normalized values of edge centrality calculated for the simple directed graph on Fig. 1. The links with highest values have the highest importance.

Let Q_i is a subset of edges from E that have direction towards node *i*. Let *C(e)* is the betweenness of edge *e*. Since the incoming weights of a node must satisfy (8) the weigh of edge *e* leading from node *i* to *j* will be calculated as

$$d_e = \frac{C(e)}{\sum_{k \in Q_j} C(k)} \quad (12)$$

D. Degree algorithm

The degree weight calculation algorithm, assigns weight to incoming links of a node according to the degree of nodes the links are originating from. Let P_i is the same subset of nodes from V that have outgoing links to node *i* and let *D(i)* is the degree of node *i.* Like in the previous techniques, we first sum the degree of all neighbor nodes of *i* and then divide the degree of each node by the sum. This technique assigns high values of weight for links that are originating from well connected nodes. The weight of the link between nodes *i* and *j* is

$$d_{ji} = \frac{D(i)}{D(i) + \sum_{k \in P_j} D(k)} \quad (13)$$

IV. Complex Networks

Since one of our main goals is to analyze the behavior of different real network topologies, for that purpose we use different types of complex networks. Complex networks have certain properties that make them different from aspect of topology. The difference comes from the way nodes are connected among each other. According to the inter-link dependences several types of network topologies are defined.

A. Random Networks

The simplest and most straightforward realizations of complex networks are random networks. These networks are characterized by nodes that are connected randomly connected to each other, with certain probability *p* [10]. For networks with large number of links, the average number of links per node is the same and the degree distributed follows the Poisson distribution. This fact shows that the probability that a node will have large deviation from the average value is exponentially small. These networks are pioneers in complex network theory, because most of the large scale networks found in reality (WWW, internet, cellular, power, neural networks) were considered to be random. However, it was later discovered that, the real networks have different topology dependences.

B. Geographic Random Networks

A special case of random networks are geographically random networks. These networks are characterized by nodes that are randomly distributed in the space, and are connected only to the nodes in their proximity. These networks always have one giant cluster component that contains most of the nodes. A typical example of random geographic network is wireless ad hoc network where each wireless station is connected to the stations that are within its range of coverage.

C. Small-world Networks

According to the link structure, small world networks stand between random and lattice connected network. They are generated by randomly replacing fraction of links from d-dimensional lattice structure [11]. If the fraction equals zero, than the network is lattice, and if the fraction is one, than the network is random network. For fraction between the extreme values, we get a small- world network. The name of these networks comes from the property that the average shortest distance between two nodes increases logarithmically with the number of nodes. Therefore the wider the network, it is easier to connect two distant nodes with just a few links. Thus, although the network is large, at the same time it is small because any node is reachable in average a few steps.

D. Scale-free Networks

Small world networks are composed of highly connected clusters, in which everybody knows everybody from the cluster, and very few of them provide connectivity to the rest of the world by setting links with other clusters. However, some of the real networks like the world-wide web, networks of scientific citations etc. have additional properties which classify them as a subtype of small world networks. These networks are called scale-free, meaning that they have distribution of connectivity that decays with power low. The number of nodes with exactly k links follows a power law, each with a unique degree exponent. These networks are characterized by presence of nodes called hubs, with large number of links. These nodes are dominant in the structure of all scale-free networks, making each node from the network easily reachable from any point [11]-[12].

V. RESULTS

The most important parameters which are studied for defining the behavior of different networks are mean value and standard deviation of failed nodes. These parameters for homogeneous networks are given in analytical form in [9], however, the analytical methods for calculation of mean value and standard deviation are not applicable for heterogeneous influence model. Therefore we use simulations to determine the behavior of different network topologies and weight calculation algorithms.

We are simulating the behavior of four different network topologies: scale-free, small-world, random and geographic random, each having 250 nodes. We used network generators with input parameters adjusted to values that will enable generating networks with nodes connected to an average of 6 other nodes. All the networks are connected to one giant cluster. The minimum probability that a node will fail is $p_{\min} = 0.59$ and the maximum probability is $p_{\max} = 0.99$. The simulations are executed for 500 time steps, and repeated for 50 different networks.

In all the simulations we plot the dependence of network behavior on the structure of the local Markov chain. Since the probability of failure *p* is calculated for each node according to (7), we plot the network behavior only against the recovery probability *q*. At the beginning of time *k=0*, all the nodes are in normal state, and very quickly the network reaches the mean value, deviating around it. Our goal will be finding the mean value and the standard deviation of node failures.

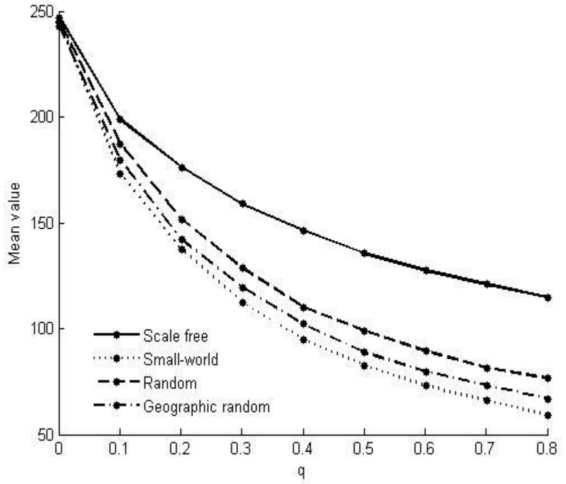

Fig. 4. Dependence of network topologies and local Markov chain structure on mean value of failed nodes for equal weight algorithm

On Fig. 4 the dependence of mean value of the four different topologies on recovery probability q is shown. The links weight is calculated according to the equal weight calculating algorithm mentioned above. It is clear that mean value strongly depends on network topology, having the highest values for scale-free networks. The other topologies have similar dependence. Random network is the second most influent topology on mean value, leaving behind geographic random and small world network.

Fig. 5 shows how the different weight calculation algorithms affect the network behavior of a scale-free network. Most intuitive behavior can be noted when the equal weight algorithm is used, because as recovery probability *q* increase, the number of failed nodes decreases evenly, unlike the node betweenness algorithm which introduces very high drop of failed nodes, especially for low values of recovery probability *q*. This is due to the fact that node betweenness is calculated by the importance of every node. Because scale-free networks have hubs with many links, their betweenness is high, so once a hub is repaired its influence is very rapidly spread through the network. From the figure we can conclude that the edge betweenness and the degree weight algorithm are very close in nature, when number of failed nodes is concerned.

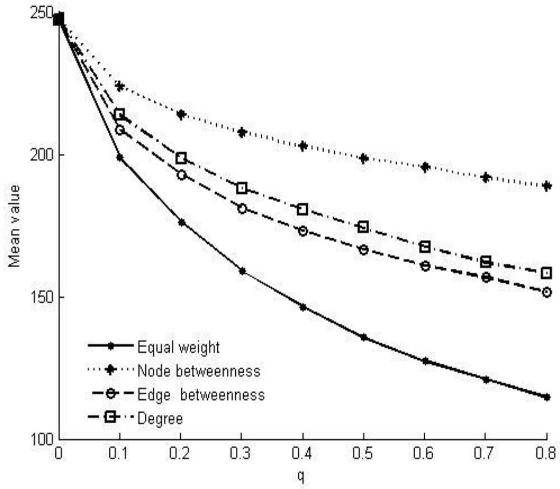

Fig. 5. Dependence of weight calculation algorithm on mean value of failed nodes for a scale-free network

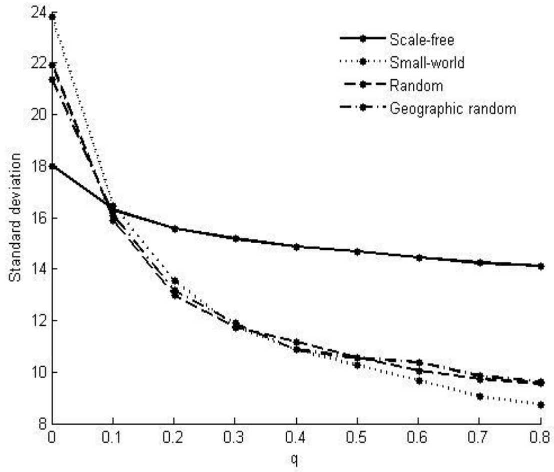

Fig. 6. Dependence of standard deviation for different topologies on recovery probability q for node betweenness algorithm

On Fig. 6 the dependence of standard deviation on network topology is shown. Again the scale-free networks are the networks which reach highest values of standard deviation. They are almost resistant to changes of value of the recovery probability *q*. The other types of networks have approximately the same values for the standard deviation. Although they are lower then the standard deviation of scale-free networks, for low values of recovery probability *q*, they reach higher values of standard deviation than the scale-free networks.

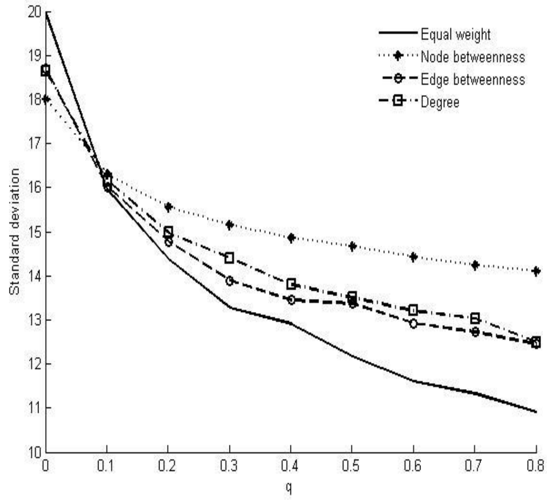

Fig. 7. Standard deviation for different weight calculation algorithms for scale-free networks

Fig. 7 presents the dependence of weight calculation algorithm on standard deviation for a scale-free network. Similarly like the mean value, the standard deviation is most sensitive to recovery probability *q* for equal weight distribution algorithm, and most resistant to changes of *q* for node betweenness algorithm. In between stand the other weight calculation algorithms.

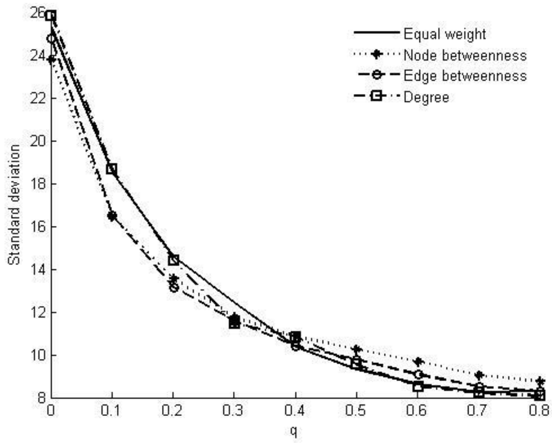

Fig. 8. Standard deviation for different weight calculation algorithms for small-world networks

Fig. 8 presents the dependence of weight calculation algorithm on standard deviation for a small-world network. It is clear that the standard deviation is almost the same for any of the algorithms for calculating the weight. The only parameter that is important in the standard deviation is the local Markov chain and the network topology. Although not presented on the figure, we can come to the similar results for all of the complex network topologies.

VI. Conclusion

From our simulations we can conclude that network topology has a major impact on the average number of failed/influenced nodes. The effect is mostly visible in the scale free network which have rather large number of such nodes for any values of the recovery probability, compared to the small-world, random and geographic random networks, which have similar behavior. We also conclude that besides the topology, another key concept for the network behavior is the weight that is assigned to the links. When we analyzed different weight calculation algorithms, we concluded that that a failure/influence is most rapidly spread when weights are calculated according to the node betweenness algorithm. Edge betweenness and degree algorithm give very similar results. The influence is spread most slowly when the equal weight distribution algorithm is used for assigning weights to the links.

Another conclusion from our work is that although the standard deviation depends on network topology, the scale of dependence is very low compared to the size of the network. For scale-free networks the recovery probability has hardly any affect on the standard deviation. For other types of networks, standard deviation depends on the recovery probability and is almost the same for all kinds of weight calculation algorithms.

Our future work will include analysis of influence propagation in complex networks, using other models for influence spreading like, SI, SIS and SIR.

References

[1] Garetto, M. Gong, W. Towsley, D: Modeling malware spreading dynamics, *INFOCOM 2003*

[2] Akshay Java, P. Kolari, T. Finin, T. Oates. Modeling the Spread of Influence on the Blogosphere

[3] J. Brown, P. Reinegen. Social ties and word-of-mouth referral behavior. Journal of Consumer Research 14:3(1987), 350-362.

[4] F. Bass. A new product growth model for consumer durables, Management Science 15(1969), 215-227.

[5] J. Goldenberg, B. Libai, E. Muller. Using Complex Systems Analysis to Advance Marketing Theory Development. *Academy of Marketing Science Review* 2001.

[6] D. Kempe, J. M. Kleinberg, and _E. Tardos. Maximizing the spread of influence through a social network. *ACM SIGKDD international conference on Knowledge discovery and data mining, 2003*

[7] Freeman, L. C. Centrality in social networks: *Conceptual clarification. Social Networks 1, 215–239, 1979*

[8] Reinhard Diestel, Graph Theory, *Springer-Verlag Heidelburh, New York,1995*

[9] C. Asavathiratham, S. Roy, B. Lesieutre, G. Verghese. The Influence Model. IEEE Control Systems, Dec. 2001.

[10] Michael Karonski and Adrzej Rucinski: The Origins of the Theory of Random Graphs, *The Mathematics of Paul Erdos, Berlin, Springer, 1997*

[11] L. A. N. Amaral, A. Scala, M. Barthe lemy, H. E. Stanley: Classes of small-world networks, *PNAS 97:11149-11152*

[12] A. L. Barabasi: Linked, *Penguin Group, London, May, 2003*

[13] Barabási, Albert-László "Scale-Free Networks". *Scientific American, 288:60-69, May 2003.*