Stability Analysis in Dynamic Social Networks

Wayne Wu
Department of Computer Science
Bard College
Annandale-on-Hudson, NY 12504-5000
wayne8798@gmail.com

Yu Zhang
Department of Computer Science
Trinity University
San Antonio, TX 78212
yzhang@trinity.edu

ABSTRACT
In this paper, we address the problem that how could the decentralized local interactions of autonomous agents generate social norms. Different from the existing work in this area, we focus on dynamic social networks that agents can freely change their connections based on their individual interests. We propose a new social norm rule called Highest Weighted Neighborhood (HWN) that agents can dynamically choose their neighbors to maximize their own utility through all previous interactions between the agents and these neighbors. Comparing with the traditional models that networks usually are static or agents choose their neighbors randomly, our model is able to handle dynamic interactions between rational selfish agents. We prove that in the 2-action pure coordination games, our system will stabilize in a clustering state and at that time all relationships in the network are rewarded the optimal payoff. Our preliminary experiments verify the theory.

Keywords
Social Network, Dynamic Network, Pattern Analysis.

1. INTRODUCTION
Simulation is becoming a convenient and efficient approach to study social behaviour problems [2]. With the help of modern computers, researchers are able to simulate social experiments that originally may take months to finish within hours. In many other cases, simulation allows researchers to study social behaviors in such a huge scale that is almost impossible to implement and observe in real life.

Multi-Agent Systems is one of the most powerful and widely used tools for social simulation [4]. Here agents are used to model social entities such as people, groups and towns. One purpose of these models is to study "generative social science" [6], i.e. how could the decentralized local interactions of autonomous agents generate social norms? However, the existing network models, such as random network, small-world network [13], and scale free network [9], all share a critical drawback. They are all static networks and do not allow agents to change the network structure. Here a network will be called *static* if edges are never created or removed after the generation of the graph. A *dynamic network* is one in which the edges are created and removed as the network evolves. Static networks may well model social behaviors in a relative stable environment such as a community where people barely move around and always keep the same relationships to others, but usually they will fail to model many dynamic networks such as behaviors in social network service (SNS) where agents frequently change their relationships to others. Examples include the friendship networks of high school students [7], the network of citations between scientific papers [10], links between web pages on the World Wide Web [1] and network of human sexual contact [8].

Zhang and Leezer previously proposed a social interaction rule called Highest Rewarding Neighborhood (HRN) in order to give agents the ability to update their neighborhood in a social network [14]. Adopting the HRN rule, one can easily transform a classic static network into a dynamic network with few restrictions on how agents observe their neighbors and how agents make the decision about when to keep an existing connection or disconnect it so they can connect to a new neighbor. However, we found some properties of the HRN rule are still not very realistic. In a dynamic network, agents need a certain function to evaluate their neighbors, and after the evaluation the agents also need to negotiate with their neighbors about whether to keep the connection or not. In the original HRN rule, agents keep track of the average reward they receive from all relationships and also the average reward they receive from every single relationship. The agents will decide whether to keep a connection or not through the comparison between these two average reward. However, when calculate the average reward, the agent evaluate all the past rewards equally. Here we argue that in real life recent rewards usually affect people's decision more than the rewards that they received long time ago. Also, people's future action may be more consistent with their recent actions than with their past decisions.

For the above reasons, we propose an extension to the HRN rule. The new rule is called Highest Weighted Reward (HWR). The HWR rule weighted recent rewards from a relation more than that of the long-time-ago rewards. Agents' decision about whether to keep a relationship or not is based on if the weighted rewards is greater than the weighted average reward earned from every relationship.

The paper is structured as the following. Section 2 introduces the HRN rule (from which the HWR rule is

extended) and other existing rules that lead to social norms emerge from a social network. Section 3 defines the HWR rule. Section 4 describes the environment of our model. Section 5 introduces the idea of how to prove that a system reach a clustering state. Section 6 is the proof that in the 2-action pure coordination games, our system will stabilize in a clustering state and at that time all relationships in the network are rewarded the optimal payoff. Section 7 presents our preliminary experiments showing that our system indeed stabilizes in a clustering state. Section 8 concludes the paper.

2. RELATED WORK AND BACKGROUND

Common theme researchers have found while studying the ways in which a society of agents act towards a problem is that the agents within that society tends to converge onto a single solution. However, different individual decision making algorithms render different lengths of time for the society of agents to converge to a mutual solution. There are two categories of these strategy selection rules. The first category is the study of how social norms emerge in static networks. The second category is the study of social norms in evolutionary networks or dynamic networks. Example rules of the first category include the *Highest Cumulative Reward* (HCR), also called highest current reward, and the *General Simple Majority* (GSM) rule. Research in the second category has just started. One example model is the *Highest Neighborhood Award* (HNA) rule. Next, we introduce then one by one.

Shoham and Tennenholtz propose the *Highest Cumulative Reward* rule [11]. For each update period, an agent would switch to a new action if and only if the total payoff obtained from that action in the latest time step iteration is greater than the payoff obtained from the currently chose action in the same time period. They then implemented this HCR rule to agents, simulating test cases and modifying variables (memory update frequency, memory restart frequency, and both) to experiment agent on an agent's effectiveness in learning about their environment. For our experiment, we ran the same tests using Jason's HRN rule and Q-learning to compare results with Shoham and Tennenholtz.

Jordi Delgado proposes the *Generalized Simple Majority* rule [5]. Agents will change to an alternative strategy if so far they have observed more instances of it on other agents than their present strategy. This rule generalizes simple majority since as the limit of $\beta \rightarrow \infty$, we can note that the change of state occurs as soon as more than half of the agents are already in that alternate state.

$$f_\beta(k_{\bar{S}}) = \frac{1}{1 + e^{2\beta(2k_{\bar{S}}/k-1)}}.$$

Generalized Simple Majority Definition [5]

The second category of research into generating agents' social behavior is the study of evolutionary networks or dynamic networks, such as [3, 12]. Here researchers investigate the ways in which populations of agents may converge onto a particular strategy. While the research is merited, assumptions are often made that make the experiments unrealistic. For example agents often have no control over whom they play with. Also, agents don't employ selfish reward maximizing decision making but instead often imitate their neighbors. Lastly, agents are often able to see the actions and rewards of their neighbors, which is unrealistic in many social settings.

Zhang and Leezer propose the Highest Rewarding Neighborhood (HRN) rule [14]. The HRN rule allows agents to learn from the environent and compete in networks. Different from those agents alike in work [3] which can observe their neighbors' action and imitate it, the HRN agents employ selfish reward maximizing decision making and can learn from the environment. Under the HRN rule, cooperative behavior emerges even though agents are selfish and attempt to only maximize their own utility. This is because agents are able to break unrewarding relationships and therefore are able to maintain mutually beneficial neighborhoods. This leads to a Pareto-optimum social convention.

This paper proposes the Highest Weighted Reward (HWR) rule, which is extended from the HRN rule. In HRN rule, the agent values its neighbors based on 14their performances equally throughout the history. However, this leads to the fact that the agent will focus too much on the history of the neighbor and fail to respond promptly to the neighbor's latest performance. What is worse is that, after the agent has stored a very long history of one neighbor, the agent's evaluation of this neighbor will be almost entirely based on the past history and can hardly be changed in a short time. Apparently, this is not a rational behavior we want the agent to have. Therefore, we introduce the HWR rule, which introduces a time discount factor that helps the agents to focus more on the recent history.

3. Highest Weighted Reward Rule

Definition Highest Weighted Reward Rule. According to the HWR rule, an agent will only maintain a relationship iff the weighted average reward earned from that relationship is no less than a specified percentage of the weighted average reward earned from every relationship.

In the HWR rule, we calculate the average reward in the following way:

1. For each neighbor, we have a variable T called *TotalReward* in order to store the history interaction of the agent. In each turn, *TotalReward* is updated as the following:

$$T_t = c \times T_{t-1} + R_t. \quad (1)$$

Where T_t is the *Total Reward* at time t, T_{t-1} is the past *Total Reward*, c is a time discount factor and R is the current reward.

2. In a similar way, we keep a variable *GT* called *General Total Reward* to store the rewards the agents got from all agents in the history. In each turn, we update the *General Total Reward* as the following:

$$GT_t = c \times GT_{t-1} + \frac{T_t}{n} \quad (2)$$

where GT_t is the *General Total Reward* at time *t*, GT_{t-1} is the past *General Total Reward*, T_t is the *Total Reward* in this turn, and *n* is the number of neighbors. Therefore, $\frac{T_t}{n}$ is the average reward the agent gets from each interaction.

3. The decision-making process for choosing bad neighbors is this. We first calculate the average reward for every agent. The *Average Reward*, denoted by *AR*, is calculated in the following way: suppose the agent has been playing with the neighbor for *n* turns,

$$AR_t = \frac{T_t}{1+c+c^2+\cdots+c^n} = \frac{T_t}{1-c^n} \times (1-c) \quad (3)$$

Notice that when $n \to \infty$, $c^n \to 0$. Therefore, when $n \to \infty$,

$$AR_t = T_t \times (1-c) \quad (4)$$

4. Calculate the *Average Total Reward*, denoted by *ATR*. Similar to step 3,

$$ATR_t = \frac{GT_t}{1+c+c^2+\cdots+c^n} = \frac{GT_t}{1-c^n} \times (1-c) \quad (5)$$

Notice that when $n \to \infty$, $c^n \to 0$. Therefore, when $n \to \infty$,

$$ATR_t = GT_t \times (1-c) \quad (6)$$

5. Finally to decide if a neighbor is bad so the agent will disconnect with it, we compare the ratio of *Average Reward* vs *Average Total Reward* with the threshold θ. The rule is as the following.

$$\begin{cases} if\ \frac{AR_t}{ATR_t} > \theta, & keep\ the\ neighbor \\ Otherwise, & disconnect\ bad\ neighbor \end{cases} \quad (7)$$

In this way, the agent can evaluate its neighbor with an emphasis on recent history. For example, if we set *c* to be 0.95, then the reward we get 100 turns ago will be weighted very lightly as this: $0.95^{100} \times 0.05 = 0.03\%$.

Next, we analyze the bound on *c*. Basically, the bound on *c* is a empirical value and varies on different payoff matrix for the pure coordination game. Here we analyze a special case. Assume we are playing the pure cooperation game with reward 1 for cooperation and -1 for defect. If we set *c* too low, the last reward the agent got from the neighbor may dominate the whole history. The following illustrate an example of this.

$$1-c-c^2-\ldots-c^n > 0$$

$$\Rightarrow 1 > c+c^2+\ldots+c^n$$

$$\Rightarrow 1 > c\frac{1+c^n}{1-c} \quad (8)$$

When $n \to \infty$, $c^n \to 0$, then

$$\Rightarrow 1 > \frac{c}{1-c}$$

$$\Rightarrow c < 0.5 \quad (9)$$

Based on this, when $c < 0.5$, the latest reward has a huge influence over the whole history. For this reason, we set a bound on *c* be $c \in (0.5, 1]$. Usually, *c* will take a value between 0.9 and 1, in order to take consideration of a relatively long history.

Notice that when $c = 1$, the HWR rule is exactly the HRN rule proposed by Zhang and Leezer [14]. So HWR is a generalized rule of HRN.

The above definitions are given under the assumption that the agent has infinite memory, which means the agent can remember all past interactions and their relative payoffs. However, this may not always be the case in real world. Therefore, sometimes we set a memory limit *L*. In that case, the agent will only remember the interactions in the last *L* turns. But the decision is made in a similar way.

4. ENVIRONMENT

The environment of our model has the following properties. These are also what our proof (see Section 6) and experiments (see Section 7) are based on.

- All trials run in a random network.
- The number of connections in the network remains the same throughout the trial.
- Every time when a connection is broken, both agents have a 50% chance to gain the right to connect to a new neighbor. But only one of them will eventually make a new neighbor. This restriction guarantees the number of connections remains the same.
- All agents have a limit memory size of k.
- All agents adopt the ideal learning rule, which means the agent will always choose the action that the majority of its neighbors used in the last turn. If there is same number of neighbors adopting different actions, the agent will not change its current action.
- The agents can only see local information. They do not know the payoff matrix and the identities of their neighbors.
- Our domain is the two-action Pure Coordination Game. It is a simple game in which agents receive a reward in the event they choose the same strategy and a penalty in the event they choose different strategies. Table 1 shows the payoff matrix for the Pure Coordination Game we use.

Table 1 Payoff Matrix for the Pure Coordination Game

	Cooperate	Defect
Cooperate	1, 1	0,0
Defect	0,0	1,1

Notice that there are two equal Nash Equilibrium in the game: (Cooperate, Cooperate) and (Defect Defect). Notice that the optimal strategy depends on the strategy of an agent's neighbor. However, when this game is played with multiple neighbors, the optimal strategy is the strategy adopted by the majority of an agent's neighbors.

5. PROOF STRUCTURE

Researchers have proved under most circumstances a static network will reach convergence when agents adopt the Generalized Simple Majority rule [5]. However, during our research we found that certain networks will never converge under the GSM rule. The simplest one may be referred as the "traffic light" network, which is a network containing only two agents. At the beginning, the two agents are connected to each other, but they use different actions. Since for each agent the only neighbor they have is the other agent, they both will change their action in the next turn based on the GSM rule. As the consequence, the two agents will keep repeating this process and flipping between two actions. Thus the network can never reach convergence. Even though this is a special case of static network, dynamic networks could have the same problem. For example, if our network is initialized as a complete network (theoretically there is an extremely low probability that it could happen), then the network will remain as static since it is impossible to add more connections between any pair of agents (because all agents are connected to each other). Thus, the special case we just pointed out can be treated as an extremely rare but valid case for our model.

Due to the existence of this kind of the special case, not all networks will reach a stable clustering state. Therefore, inspired by Shoham's work [11], we proved that as time step approaches infinity, the probability that a network reaches the stable clustering state will approaches to 1. In Section 6, we show our proof in great details. In order to make our proof easier to understand, here we first explain the basic idea behind the proof. The idea is very similar to the logic of the classic Monkey and Shakespeare problem. Let a monkey randomly type on a keyboard. Given infinite time, the monkey will eventually finish a work of Shakespeare. The idea is: given a specific process that can happen with a very low probability and infinite time, the probability that this process can approach to 1 eventually.

The proof contains three major parts:

- A starting state that should be valid at any time step throughout the trial;
- An ending state, which is also the goal state of the proof;
- A specific process that will lead the system from the starting state to the ending state. In our proof, the starting state is a normal random state of the system. The goal state is the stable clustering state. We have also described the specific process in great detail in our proof.

6. PROOF

6.1 Definition

We have the following definitions.

- A_i denotes the i^{th} agent in the network.
- a_i denotes the action choice for agent A_i. In coordination games, a_i has binary values representing different action choice.
- R_{ij} denotes the link between agent A_i and agent A_j.
- N_i denotes agent A_i's neighborhood. If $A_j \in N_i$, then there exists a link R_{ij} between the two agents.
- C_i denotes agent A_i's coordinating neighbors. $C_i \subseteq N_i$. If $A_j \in C_i$, then $a_j = a_i$.
- $\overline{Ci}$ denotes agent A_i's non-coordinating neighbors. $\overline{Ci} \subseteq N_i$ and $\overline{Ci} \cap C_i = \phi$. If $A_j \in \overline{Ci}$, then $a_j \neq a_i$.
- We call agent A_i a *majority coordinating neighbor* if $|C_i| \geq |\overline{Ci}|$; we call agent A_i a *majority non-coordinating neighbor* if $|C_i| < |\overline{Ci}|$.

Notice that any agent in the network must fall into one of the above two categories.

- We say a network is in a *stable clustering state* if for any agent A_i in the network, $|\overline{Ci}| = 0$.
- We say a group of agents D form a *closed cluster* if for any agent $A_i \in D$, $|\overline{Ci}| = 0$, and for any $A_j \in C_i$, $A_j \in D$. Also, for any pair of agent A_i and A_j, $a_i = a_j$.
- We say an agent has a *free connection* when the agent broke one old connection and gains the right to connect to a new neighbor.
- We call an agent a *free action agent* when it is possible for the agent to become either a majority coordinating neighbor or a majority non-coordinating neighbor by connecting to different new neighbors.
- We define *whipping* as the following process: every time when a free action agent looks for new neighbors, it only connects to the agent who uses different action from it. In this way, the agent will keep flipping between the two possible actions. Since the agent's neighbors may have high tolerance on the relationships between them and the agent, even when they play different actions the connection won't be broke immediately. Therefore, we let the agent's neighbor go through the whipping process too. Eventually, we let the agent's neighbors keep flipping between the two possible actions in the way that each time the agent and its neighbors use different action. In this way, the reward for these connections will be constantly 0. After at most k turns, the agent will lose all of its neighbors and becomes isolated.

6.2 Theorem and Proof

We have the following assumption:

Assumption. In a two-action coordination game, if the network has not reached a stable clustering state yet, then

there will always be connections broken between agents playing different actions. And there always exists agents adopting different actions with free connections.

Theorem 1. If in a network all agents are majority coordinating neighbors, then there it is possible that the network will reach stable clustering state in k turns.

Proof for Theorem 1. If we only allow the agent to connect to same action agents, then after k turns the agents will only keep connection with the neighbors who adopt the same action as them. In this way, for any agent A_i we see $|\overline{Ci}| = 0$, and the network reaches the stable clustering state. □

Theorem 2. If a network has at least one closed cluster, then it is possible for the network to reach a stable clustering state within $g(n)$ turns.

Proof for Theorem 2. We prove this theorem by case analysis. First we check whether there exists any majority non-coordinating agent. If there doesn't exist, then by Theorem 1 we know it's possible for the network to reach stable clustering state in k turns. If there exists at least one majority non-coordinating agent, we select one and check whether this agent is a free action agent. If not, we pull free connections from non-coordinating agents to make the agent a free action agent. Then we let all of its free connections connect to a closed cluster when they try to link new neighbors. If there is no closed cluster at the moment, we can change the current agent into a closed cluster through the whipping process. In this way, the original majority non-coordinating agent forms a closed cluster by itself. When the majority non-coordinating agent tries to connect to the closed cluster, if the agent has more free connections than the agents in the closed network, we can let the majority non-coordinating agent go through the similar process we describe above, except this time we will make the agent keep one free connection at last and use that connection to connect to the closed cluster. In this way, the agent becomes part of the closed cluster. We repeatedly run the above steps until there is no more majority non-coordinating agent left in the network. Then by *Theorem 1*, the network is possible to reach stable clustering state in k turns. □

Theorem 3. Given a pure coordination game, placing no constraints on the initial choices of action by all agents, and assuming that all agents employ the HWR rule, then the following holds:

- For every $\varepsilon > 0$ there exists a bounded number M such that if the system runs for M iterations then the probability that a stable clustering state will be reached is greater than $1 - \varepsilon$.
- Once the stable clustering state is reached, it will never be left.
- If a stable clustering state is reached then the all agents are guaranteed to receive optimal payoff from all connections.

Proof for Theorem 3. Similar to the proof for *Theorem 2*, we check whether there exists any majority non-coordinating agent first. If there doesn't exist any, then by *Theorem 1* we know that there is a probability $p = 1/f(n)$ the network will reach a stable clustering state in the next turn. If there is at least one majority non-coordinating agent, then we repeat the processes described above in order to make the agent become a closed cluster or part of an existing closed cluster. For each agent, there is a probability $p = 1/g(n)$ that this process can happen in $h(n)$ turns. We repeat the above process until there is no majority non-coordinating agent or the network reaches stable clustering state.

As a result, if the system runs for $M = x \times [n \times g(n) \times h(n) + f(n)]$ iterations then the probability that a stable clustering state will not be reached is at most e^{-x}. Taking $x > -\log(\varepsilon)$ yields the desired result.

For the second part of *Theorem 3*, since there only exist connections between agents who play the same actions when the stable clustering state is reached, every connection will be optimal and no agent will try to break any connection. Therefore the stable clustering state will never be left.

For the third part of *Theorem 3*, as mentioned above, since all connections are optimal, all agents are also guaranteed to receive optimal payoff from all connections. □

7. PRELIMINARY EXPERIMENT

To demonstrate the theorems, we have run several preliminary experiments and indeed observed the desired results. Agents are connected by a random network and play the two-action pure coordination game defined in Table 1. The time discount c=0.95. The memory limit L=10. The experiments are run in a relative small scale with only 300 agents. Each result is the average of 30 trials. The system is initialized with 50% "cooperate" agents and 50% "defect" agents. Throughout the trial, we keep track of four attributes:

- *Total Neighbor Lost*: this is the number of the connections broken in each turn.
- *Number of Cooperation*: this is the number representing the size of cooperation camps.
- *Number of Defect*: this is the number representing the size of defect camps.
- *Number of Perfect Agents*: here we call an agent a perfect agent when all of its neighbors play the same actions as it does. An agent with no neighbor can also be viewed as a perfect agent.

Figure 1 shows the change of the Total Neighbor Lost (y-axis) along time steps (x-axis).

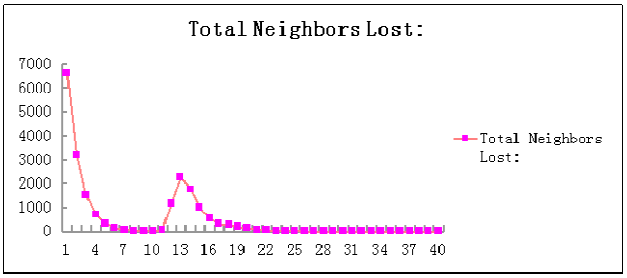

Figure 1 The Total Neighbor Lost

We notice that after about 20 steps, the total neighbor lost begins to approach 0, which means the network enters a stable state. In order to further understand how the stable state is reached and what pattern we can find in the stable state we need to analysis the changes of the other three attributes.

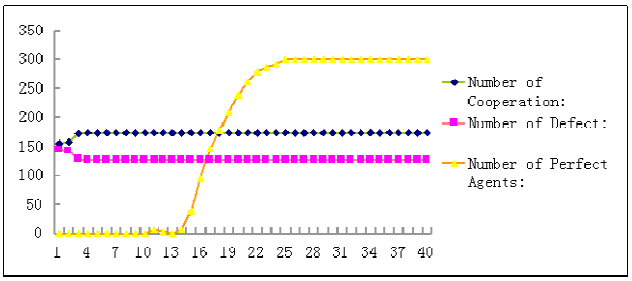

Figure 2 Pattern Analyses Regarding the Stable State

As shown in Figure 2, the Number of Cooperation and the Number of Defect start off almost with the same number because of the 50%-50% initialization condition. After the beginning of the simulation, quickly, the Number of Cooperation rises a little bit while the Number of Defect decreases. After a few turns, both numbers stay constant through the rest of the trial, even after when the system reaches a stable state (at around the 20th step). Therefore, we can conclude that the system eventually reaches the stable clustering state. This can be additionally verified by the Number of Perfect Agents. In the first few turns, the Number of Perfect Agents remains zero, since in the randomized network most agents have non-coordination agents (i.e. the agents that play different actions from its own action). As the Total Neighbor Lost falls, we see the Number of Perfect Agents begin to emerge in the network. When the system reaches the stable clustering state, the number of perfect agents also reaches 300, which means all agents only connect to coordination agents and every connection is beneficial for both players.

8. CONCLUSION

In this paper, we have presented the Highest Weighted Reward (HWR) rule to simulate dynamic interaction between agents. In the future, this research can be extended in two different directions. One is in the experiment. We will perform more experiments to study the speed of the stabilization in various conditions when the agents have different memory windows and update frequencies. Another extension is to allow new agents to join the network and old agents to leave the network. A new set of rules can be employed that will make existing models be able to simulate more dynamic environment.

9. ACKNOWLEDGMENTS

This work was supported in part by the U.S. National Science Foundation under Grants IIS 0755405 and CNS 0821585.

10. REFERENCES

[1] Albert, R., and Barb'asi A.L. "Statistical Mechanics of Complex Networks." *Modern Physics*, 2002: 47-97.

[2] Axelrod RM, "The Complexity of Cooperation: Agent-Based Models of Competition and Collaboration", Princeton University Press, 1997.

[3] Borenstein, Elhannan and Ruppin, Eytan, "Enhancing Autonomous Agents Evolution with Learning by Imitation." *Journal of Artificial Intelligence and Simulation of Behavior*, 1(4): 335-348, 2003.

[4] Davidsson, Paul, "Agent-Based Social Simulation: A Computer Science View", *Journal of Artificial Societies and Social Simulation*, 2002: 5(1).

[5] Delgado, Jordi. "Emergence of Social Convenetions in Complex Networks." *Artificial Intelligence*, 2002: 171-185.

[6] Epstein, Joshua, "Agent-Based Computational Models and Generative Scoail Science", *Complexity*, 1999: 4(5).

[7] Fararo, T.J., and M.H. Sunshine. "A Study of a Biased Friendship Net". Syracuse, NY: Syracuse Univ. Press, 1964.

[8] Liljeros, F., C.R. Edling, L.A.N. Amaral, H.E. Stanely, and Y. Aberg. "The Web of Human Sexual Contacts." *Nature*, 2001: 907-908.

[9] Newmann, M.E.J. "The Structure and Function of Complex Networks." *Society for Industrial and Applied Mathematics*, 2003.

[10] Redner, S. "How Popular is Your Paper? An Emprical Study of the Citation Distribution." *Eur. Phys.*, 1998: 131-134.

[11] Shoham, Y, and M Tennenholtz. "On the emergence of social conventions: Modeling, analysis and simulations." *AI*, 1997: 139-166.

[12] Zimmermann, Martin, and Victor Eguiluz. "Cooperation, Social Networks and the Emergence of Leadership in a Prisoners Dilemma with Adaptive Local Interactions." *Physical Review*, 2005.

[13] Watts, D.J. *Small Worlds.* Princeton: Princeton University Press, 1999.

[14] Zhang Y and Leezer J, Emergence of Social Norms in Complex Networks, Symposium on Social Computing Applications (SCA09), The 2009 IEEE International Conference on Social Computing (SocialCom-09), Vancouver, Canada, August 29-31, 2009, pp 549-555.