Journal of Mobile Multimedia, Vol. 5, No. 4 (2009) 333–350

ANALYSIS OF TEMPORAL EVOLUTION OF SOCIAL NETWORKS

KHALED MAHDI[a]
Chemical Engineering Department, Kuwait University, Kuwait
Al-Khaldiya, Kuwait City, Kuwait
k_mahdi@kuc01.kuniv.edu.kw

MAYTHAM SAFAR HISHAM FARAHAT
Computer Engineering Department, College of Engineering and Petroleum
Kuwait University, PO Box 5969, Safat 13060, Kuwait
maytham.safar@ku.edu.kw hishamfarahat@gmail.com

Received March 1, 2009
Revised July 28, 2009

The article presents an analysis of dynamic social network where words, nodes and edges appear and disappear through time. We study a popular virtual social network in the internet, known as Paltalk. We analyze the exact and approximated cyclic entropy variation with time with the purpose to establish the robustness of the network. In addition, we study the effect of weighed links on the analysis of such graphs.

Keywords: Cyclic Entropy, Cycles, Graphs, Directed, Undirected
Communicated by: D. Taniar

1 Introduction

Networking is the study concerned with analyzing, monitoring, designing or maintaining structures that consist of nodes, links and data transferred through the links to reflect the relationships. Such data is transferred in different formats including and not limited to text, voice, video or signals. One important type of networks is communication networks, for instance phone lines, mobiles; mailing system, e-mails, chatting and the web. In principle, each network has its unique definition of a node, a link and a relationship that dictates the transferred data. For example, in social networks, the nodes represent social actors (e.g. Human beings, animals, etc), the links represent the social relationship and the data can take several forms and formats (e.g. text, media, signals, etc). By understanding the characteristics of the links among the actors, we might predict the behavior of the social actor involved; the main motivation of social network studies.

Social networks analysts have been studying the human sociality for a very long time, and have tried to answer questions: How do people communicate? What are the rules that really control these communications? Half a century ago, it was almost impossible to provide qualitative or quantitative assessment to obtain accurate answers to these questions. Nevertheless, since the technological advancement in the communication field and the creation of the Internet and mobiles, the ability to provide insight has grown tremendously. The necessary

[a]Chemical Engineering Department, College of Engineering and Kuwait University, PO Box 5969, Safat 13060, Kuwait

data surrounding the communications among people can easily be collected from the virtual communities in the Internet, social platforms indifferent to spatial or temporal limitations. Examples of well known social networks websites of different scopes, users and services are Facebook , Hi5, Myspace, LinkedIn and Flicker.

With the increasing population of the world, the importance of modeling social networks increases. For example, the work in [1] presents an enhanced chat-bot architecture (EHeBby) in which a humoristic conversational agent is capable to generate, propose, tell and answer to the user: humoristic expressions, riddles, and jokes. It is based on traditional rule-based knowledge representation, i.e. the standard AIML KB, which exploits both WordNet and the CMU pronouncing dictionary. It builds an LSA-inspired semantic space in which the sentences are stored in the standard AIML KB. This network of expressions can be modeled as a special social network and the user behavior can be predicted by analyzing such a network.

Numerous applications of social networks modeling exist in the literature. In [2] they tried to use a mail inbox as a source to develop a social network and assess the network with the objective of fighting spam messages. In [3] they tried to use data stored in banks, phone records, vehicle sales, surveillance reports and registration records to create a social network and to analyze this network to fight criminal organizations. Some other researches tried to use social networks to represent web-communities to analyze the World Wide Web. Other applications includeing data modeling, compression methods, indexing and query operators were suggested in [4, 5, 6] respectively to analyze social network. Most studies characterize social networks using degree distribution, clustering coefficient, average length and average degree [7]. Recently [8, 9] proposed to compute a statistical mechanic properties, the cyclic entropy of the network as a measure of the degree of network robustness, such property is based on counting the number of cycles of certain length existing in the network.

The scope of this work is to analyze a dynamic social network, that is a network with no restrictions in changing the relations between the actors. In other words, nodes and edges appear and disappear with time. An example of such network is a popular virtual social network in the internet known as Paltalk. It is an online chatting program that enables its members to text and voice chat with each other in different rooms. Paltalk as a room-based chatting network is more dynamic than other virtual friends networks (e.g. Facebook , My Space), The dynamicity comes from the movement of the members between the rooms. A user can chat with a group of users today, and chat with totally different users the next day. In this work, we construct the network from data obtained from Palktalk social network. Then, we analyze the cyclic entropy variation with time with the purpose to establish the robustness of the network.

The problem of finding and counting circuits in large graphs has been of interest to researchers lately due to its challenging complexity; has been known to be an NP-Complete problem. Exhaustive enumeration, even by smart algorithms proposed in earlier research, is restricted to small graphs as the number of circuits grow exponentially with the size of the graph [9]. Therefore, it is believed that it is unlikely to find a precise and an efficient algorithm for counting circuits. An alternative is to approximate the number of circuits in a graph. In this work we illustrate a method of approximation that has been suggested to estimate, in polynomial time, the number of circuits in a graph as a function of their lengths. The algorithm is based on a work done in [10].

Furthermore, in this work, we study the behavior of social networks when taking into consideration the weights of links between the nodes. Those weights represent the degree of interaction between users or rooms in the social network. Various studies were conducted in the literature, that used weights of the edges to define the strength of the relation between the nodes. In [11], they used the idea of weighted networks and computed the clustering coefficient and weighted degree. This was used to study the differences between normal and tumour networks.

2 Entropy of Cycles (Loops)

Entropy of a network is related to the probability of finding the network in a given state. For a system of moving molecules, the state is obviously the positions and the momentum of each molecule at a given instant. For a system of magnets, the state is defined through the direction of the magnets (e.g. north or south). In social networks, there are several choices that define the state of the network; one is the number of social links associated with a social actor, known as the degreeof a node. This definition is commonly used by almost all researchers. Characterization of social network through the degree leads to different non-universal forms of distribution. For instance, random network has a Poisson distribution of the degree. Small-World network has generalized binomial distribution. Scale-Free network has a power law distribution form. There is no universal distribution reported. Here we propose a universal distribution form that is applicable for all social networks by using a different definition of the state of a network. We define the state as the degree of loops or cycles existing in the social network. Then we find the distribution function of such loops. Philosophically, it answers the fundamental question, in a social network, what is the probability that an actor will receive the same information dispatched by him again? From this definition, we clearly assigned the entropy of non-cyclic social network, purely hierarchical, to infinity. Clearly a non-cyclic social network has very high entropy and hence no robustness based on our definition and it is in close if not in the state of network equilibrium.

Loops were one of the major concerns in social network field. In [12], they mentioned that the loops (cycles) can be considered as the major aspect that can separate the graph to sub-graphs or components. A cyclic component is a group of intersecting cycles. They intersect with each other by lines or points. Other researches proved that there is a strong relation between the structural balance of a social network and the loops in the network. Balancing in social network is the collection of rules that defines the normal relations between the clients. In [13] and [14] they modeled some social networks as signed directed graphs (graphs with a sign associated with each arc) and they claimed that a network is balanced if it contains only semi-cycles (a cycle that contains arcs in different directions) and even number of negative arcs. Also they have found that the cycle or semi-cycle is cluster able if it does not contain exactly one negative link. Some networks must be acyclic. That is, they do not contain any cycles. Such networks are called hierarchies. So, some researches were interested in detecting the invalid relation by counting the cycles in the graph as in [13]. This process, gives each level in the hierarchy a number, a ranking process.

Our previous work presented in [8] defines the relation between network robustness and the entropy. They defined the entropy as "the degree of disorder in the system". From statistical mechanics, the entropy can be calculated from a given probability distribution $P(k)$ of the

system in state k:

$$S = -\sum_{k} P(k) ln P(k). \tag{1}$$

We stated that a robust network (unconstrained and dynamic network) has low entropy, and a static rigid network has large entropy. We proposed a new concept of computing the entropy, the network cycles' distribution is used instead of the well known degree based entropy. The cyclic entropy characterizes the network more accurately than the degree entropy.

3 Degree of Cycles Distribution

Based on the definition of entropy in Eq.(1), we need to evaluate from real data the distribution function that characterizes a social network. As stated earlier, entropy of a network is related to the probability of finding the network in a given state. For a system of moving molecules, the state is obviously the positions and the momentum of each molecule at a given instant. For a system of magnets, the state is defined through the magnets directed north or south. In social networks, there are several choices that define the state of the network; one is the number of social links associated with a social actor, known as degree. This definition is commonly used by almost all researchers.

Characterization of social network through the degree leads to different non-universal forms of distribution. For instance, random networks have a Poisson distribution of the degree. Small-World networks have generalized binomial distribution. Scale-Free networks have a power law distribution form. There is no universality class reported. Here we propose a universal distribution form that is applicable for all social networks by using a different definition of the state of network. We define the state as the degree of loops or cycles existing in the social network. Then we find the distribution function of such loops.

4 PaltalkScene

PaltalkScene is an online chatting program that enables members to interact with each other using text, audio or video [15]. Interaction can be between two members or within a large group of members gathered in one chat room. Chat rooms are categorized into different regions or hobbies. For example: Europe, America, Middle East, Sport, Music, ... etc. PaltalkScene (previously called Just Paltalk) claims 4 million members worldwide. On average 50,000 members and 4,000 rooms are online at time. Accessing rooms are limited to four rooms at a time; this means members can text chat in more than one room simultaneously.

Based on Social network definition, Paltalk is considered as a social network where its actors are Paltalk's members and relations are pairs of two members who visited the same room at the same time (i.e. there exists a relation between members x and y if and only if x and y were in the same room at a given time) . Moreover the relations between members in this social network are dynamic; since they appear and disappear frequently (i.e. members can visit and leave rooms at any time they decide).

5 Network Modeling

5.1 Data extraction

To study and analyze the Paltalk social network, we have observed the network for 20 days focusing on the Middle East rooms only. Every day at 10 PM (Kuwait local time), a program

starts to extract the data from the network. The extraction is divided into two phases:

1 Data Collection: During this phase a program runs to visit all the rooms under these subcategories:

   - Kuwait
   - Business & Technology
   - Egypt
   - Friends and friendship
   - Government and Politics
   - Iraq
   - Lebanon
   - Music
   - Saudi Arabia
   - UAE

   The program is actually an AU script programmed using Autoit V3 scripting language which is designed to automate windows GUI. The script passes over all the rooms, and for each room a file is created that contains all the members who are online and visiting this room. This phase takes around 40 minutes to finish.
2 Data Storing: Another Java program is designed to read and parse the files created by the AU script, and store it in a MySQL database. A new ID is created for each new member discovered during the previous pass. The same goes for new rooms.

To get a better view of the network, the extraction process is done four times for each day. This gives an indication on how long each member stays in the room.

5.2 Social network graph representation

The analysis of a network system needs the network to be modeled mathematically as a graph, [16]. The graph theory has been used to analyze and compute the robustness of the Paltalk social network. Here we proposed two models, Fine grained and coarse grained. The fine grained model is more like the real Paltalk network because it contains all the details of the network, the members are the nodes of the graph. The coarse grained model is an abstract model that eliminates members and their details and just deals with rooms (i.e. the nodes represent rooms not members). Coarse grained model is simpler and easier to analyze and study than the fine grained, since the number of rooms is much less than the number of members.

5.2.1 Fine grained model

Let $G(V, E)$ be an undirected graph representation of the network, where V is the set of members that have visited the previously mentioned subcategories during our observation to the network. And E is the set of edges, where e is represented as the tuple (u, v, c, r) that means that users u and v have been at the same room r, c times in this day. c needs to be there because as we mentioned earlier in the data extraction process that we pass over all the chosen rooms four times, that means c is between 1 and 4. To illustrate how the Paltalk can be modeled as a graph, assume the situation stated in Table 1 (for simplicity, we assume only two passes over three rooms are done not and four passes).

Table 1. . Example of small data gathered during two passes.

Pass # 1	Pass # 2
Users (1, 2, 3, 4) are in room 1	Users (1, 2, 3) are in room 1
Users (2, 5, 6) are in room 2	Users (5, 6, 4) are in room 2
Users (1, 3, 7, 8) are in room 3	Users (7, 8, 4) are in room 3

An example of a fine grained model is shown in Figure 1, where the nodes represent the members. The labels on the edges represent the rooms that the two members visited and how many times they have visited the same room. The same example is shown in Figure 2 but with normalized weights. The weights are normalized by 3, which is the max number of links in the example of Figure 1. An example of a cycle is <1,2,5,4,3,8,1>, which has a length of 6 (un-weighted measure), and a weight of 2.667.

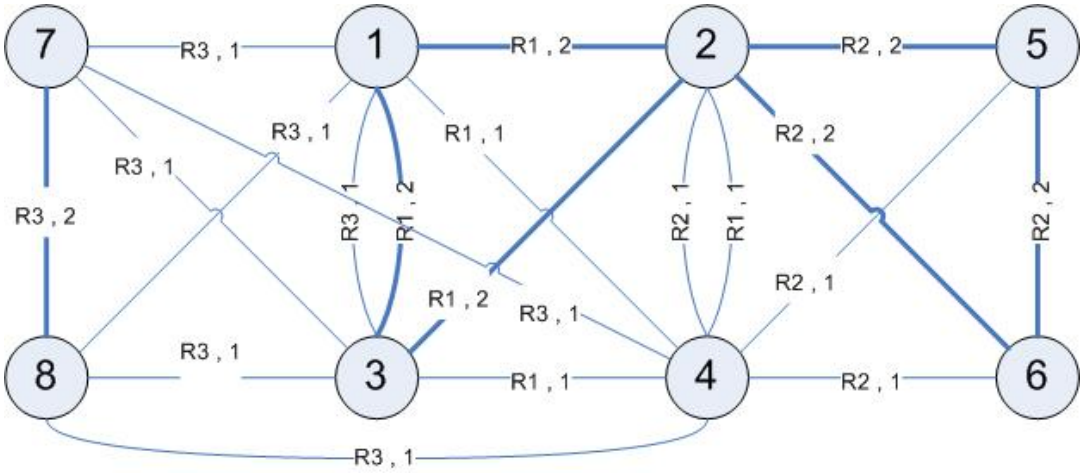

Fig. 1. Fine Grained Model of a Social Network with Users and Rooms.

5.2.2 Coarse grained model

Let $G(V, E)$ be an undirected graph, where V is the set of rooms that the extraction program have passed over during our observation of the network. And E is the set of edges, where e is represented as the tuple (u, v, n), where u and v are rooms visited by the same member, and n is the number of different users that visited the rooms u and v in the same day. In other words, there is an edge between two rooms if a member has visited both rooms in the same day.

An example of a coarse grained model is shown in Figures 3 (un-weighted) and 4 (weighted), that are based on the examples in Figures 1 and 2. The nodes represent the room, and the labels on the edges represent the number of users visited those rooms at the same time (or weight). An example of a cycle is $< R_1, R_2, R_3, R_1 >$, which has a length of 4 (un-weighted measure), and a weight of 2.

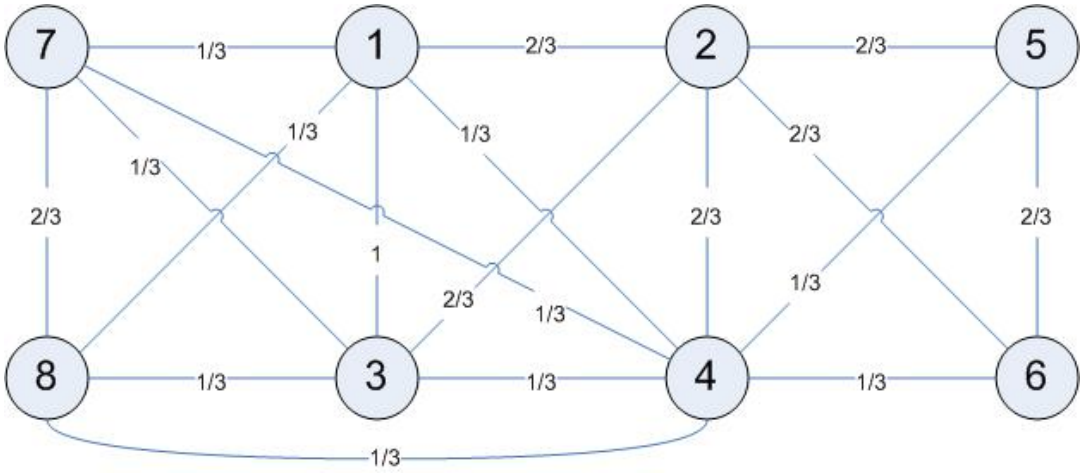

Fig. 2. Fine Grained Model example with weights.

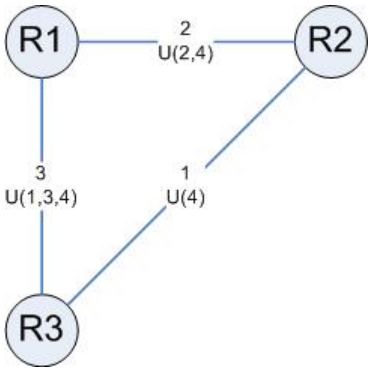

Fig. 3. Coarse Grained Model example.

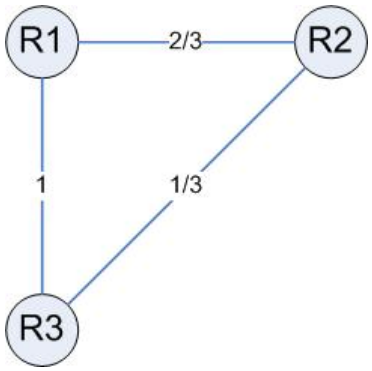

Fig. 4. Grained Model example with weights.

5.2.3 Directed graph

To simplify the graph, we proposed another model that uses the time of the room's visits to create another graph. The previous graph models definitions apply here but with changing the set of edges E, here E is the set of edges where e is the ordered tuple (u, v) where u and v are rooms visited by the same member if and only if the member had visited room u before room v.

6 Cycles Computation

To analyze our social network, we will only consider cycles as the main parameter that characterizes our graphs. A simple cycle is a sequence of nodes (path) $< v_0, v_1, .., v_k >$ if $k > 3, v_0 = v_k$ and $v_1, v_2, .., v_k$ are distinct. Two cycles are distinct if a one is not a cyclic permutation of the other [17].

In this work, we will compute the cycles distribution using exact and approximate algorithms that are based on the works in [17] and [10], respectively. In both algorithms, we assume that we have as an input a directed graph $G(V, E)$ where V is a set of vertices and E is a set of ordered pairs called edges, which is represented as an adjacency list AG. We assume that those graphs have: 1) No self-loops, i.e. no edges of type (v, v), 2) No multi-edges , i.e. no two edges has the same source and destination, 3) G is strongly connected graph, and 4) The vertices are numbered with IDs from 1 to n.

6.1 Exact cycles distribution algorithm

With this algorithm we are interested in finding the number of cycles for each possible cycle length. We developed a java program to find all the simple cycles in a graph that is based on an algorithm created by Johnson [17]. The algorithm is based on backtracking technique. It starts with node s, which is the vertex with the least ID, and begins to enumerate all cycles that passes through s. This is done by building a simple path starting from s using a stack to save the vertices. Whenever s is encountered again, a cycle is created and printed. Addition to that any vertex is currently in a path (stored in the stack) is being blocked so it cannot be added again to the stack. When a node is finished (the algorithm passes through all of its edges), it is being popped from the stack and unblocked for future use. After enumerating all the cycles with s is a common node, the algorithm removes s from the graph G and starts the process with the second least vertex. These steps are repeated until G has two nodes. Since our graph is undirected, the equation

$$C_{un} = \frac{C_{d\sigma} - e}{2}. \tag{2}$$

is used to convert from directed cycles to undirected cycles. Where $C_{d\sigma}$ is the total number of cycles in the directed version of the original undirected graph (i.e. replace each undirected edge with two directed edges in opposite directions). C_{un} is the total number of cycles in the undirected graph and e is the total number of edges in the graph.

The pseudo code for this algorithm is shown in Algorithm 1.

We summarize the algorithm with these steps:

1 Assign IDs for all the nodes in the graph.
2 Choose node S as the node with least ID.

Alg. 1 CircuitFinding(G).

Input: $G = (V, E)$ Graph
Integer List arrays $A_k[n], B[n]$, Boolean array *blocked*$[n]$, Integer s

```
1   begin
2       empty stack
3       s ← 1
4       while (s < n) do
5           Ak ← adjacency structure of strong component K with least
            vertex in subgraph of G induced by {s, s+1, ..., n}
6           if (A_k ≠ φ) then
7               s ← least vertex in V_k
8               for (i ∈ V_k) do
9                   blocked(i) ← false
10                  B(i) ← φ
11              end
12              dummy ← CIRCUIT(s)
13              s ← s + 1
14          else s ← n
15      end
16  end
```

Procedure **CIRCUIT(Integer** v**)**

```
1   begin
2       f ← false
3       stack v
4       blocked[v] ← true
5       for (w ∈ A_k[v]) do
6           if (w = s) then
7               output circuit composed of stack followed by s
8               f ← true
9           end
10          else if !blocked(w) then
11              if CIRCUIT(w) then f ← true
12      end
13      if f then UNBLOCK(v)
14      else for (w ∈ A_k[v]) do
15          if v ∉ B[w] then put v on B[w]
16      unstack v
17      return f
18  end
```

Procedure **UNBLOCK(Integer** u**)**

```
1   begin
2       blocked (u)← false
3       for (w ∈ B[u]) do
4           delete w from B[u]
5           if blocked(w) then UNBLOCK(w)
6       end
7   end
```

3 Initialize a path by making S is the root of the path.
4 Start a depth first traversal using S as the root, for each new unblocked node add it to the current path.
5 If S is found again, then a new cycle is found and displayed.
6 When a node is finished, it set to unblocked so that it can be used in other circuits.
7 After finishing all the nodes, remove S from the graph and start again from step 2.
8 If S is the last node in the graph, end the algorithm.

Before finding the cycles, a simplification process is applied to the graph that removes nodes and edges that cannot be a part of any cycle. This speeds up the Johnson algorithm. This process can be summarized with these steps:

1 Multi edges between the same nodes are considered as one edge.
2 Remove nodes with degree less than 2 because these nodes are leaves that will never form a cycle.
3 Remove cut-edges, which are edges that if removed from a graph, the graph will be divided into two or more subgraphs.

We noticed that steps from 2 to 7 are similar and repeated for each node, the only difference is that the graph has one less node than its previous. Using this we can parallelize the algorithm by creating $N - 1$ threads, each thread is responsible of finding cycles originating from its root.

The complexity of Johnson's algorithm is $O((n + e)c)$ where n is the number of nodes, e is the number of edges and c is the number of circuits in the graph. To have a sense of how the cycles computation problem is an NP-complete, Table 2 summarizes the results of the algorithm running on complete graphs which has the worst case running times.

Table 2. . Excection time of running the exact algorithm.

Number of nodes	Number of cycles	Time (msec)
5	37	313
8	8018	906
11	5488059	31671
13	710771275	3188079

6.2 Approximation cycles distribution algorithm

The approximation algorithm is based on the work in [10]. It estimates the number of circuits (cycles) in an undirected graph and provides analytical results for the typical entropy of circuits in sparse random graphs.

The approximation in this algorithm is based on a statistical mechanics approach. It uses a Bethe approximation technique [18] and iterations of the Belief Propagation equations and an approximation method to approximate the statistical mechanics model and find the cycles distribution. Two methods can be used as approximation algorithms, Monte Carlo simulation or Bethe approximation. Bethe is used here because of the well-known correspondence between both Bethe and Belief Propagation.

First of all, the graph is reduced; all leaf nodes (nodes with degree 1 or 0) are removed from the graph. Each edge of the graph is initialized with a random positive value $y^{(0)}$. Each edge is

iterated from its initial value until convergence reaching to a fixed value of y^*. Convergence is determined according to some accuracy level. In this algorithm, the convergence is considered to be satisfied when $|y^{T+1} - y^T| \leq 0.001$. The value y represents the probability that the edge is present in a cycle c. The y value can be calculated using the following equation:

$$y_{i\to j}^{T+1} = \frac{u \sum\limits_{m\in\beta_{i-j}} y_{m\to i}^{T}}{1 + 0.5u^2 \sum\limits_{\substack{m,n\in\beta_{i-j}\\ m\neq n}} y_{m\to i}^{T} y_{n\to i}^{T}} \tag{3}$$

where u is a positive real value. Then from all y's two values are calculated; C_L and L

$$L = \sum_{(i,j)\in E} \frac{u\, y_{i\to j}^{*}\, y_{j\to i}^{*}}{1 + u\, y_{i\to j}^{*}\, y_{j\to i}^{*}} \tag{4}$$

$$R = \frac{1}{N}\sum_{i\in V} ln\left(1 + 0.5u^2 \sum_{\substack{m,n\in\beta_{i-j}\\ m\neq n}} y_{m\to i}^{*}\, y_{n\to i}^{*}\right) - \frac{1}{N}\sum_{(i,j)\in E} ln\left(1 + u\, y_{i\to j}^{*}\, y_{j\to i}^{*}\right) - \frac{L\, ln(u)}{N} \tag{5}$$

$$C_L = e^{RN} \tag{6}$$

where

- β_i is the set of neighbors of node i.
- β_{i-j} is the set of neighbors of i except the neighbor j.
- N is the number of nodes in the graph.
- C_L is the number of cycles of size L.

Refer to [10] for further details on the above equations.

The procedure explained above is repeated starting from an initial value of $u = u_0$ to $u = u_{max}$. Where u_0 and u_{max} are greater than 0. At each iteration step, a new distribution point (L, C_L) is produced. The iteration step for u is 0.0001 at the early stages of the algorithm. This value is not fixed. It will be changed when $L_{new} - L_{old} < 0.001$ (i.e. the progress in L is slow). If this condition is satisfied, u will be increased by 10%. As noticed from the equations above, the output at each step (L, C_L) depends on u. At specific stages of the iteration (when u gets large), many iterations are wasted giving nearly the same point. To avoid this condition, a jump in u is made. This algorithm yields a plot of (L, C_L) points. To extract the needed distribution points (3 to n), we use interpolation. Interpolation equations needs at least two points to find the third one, this point should be surrounded with the other two nodes. For example, P1(5.9876, 453) and P2(6.0124, 490) and we need to find the number of cycles for $L = 6$ then interpolation leads to a value of 471.5. The pseudo code for this algorithm is shown in Algorithm 2. This algorithm has a running time that is growing polynomially with the graph size and logarithmically with the required accuracy. Since the algorithm uses the adjacency matrix to represent the graph, the space complexity is $O(2n^2)$.

Alg. 2 ApproximationAlgo(G).

Input: $G = (V, E)$ Undirected graph
Output: $A[1..n]$ Array of size n which contains the cycle count of each length

```
1     begin
2         A[1] ← A[2] ← 0
3         Reduce G by removing all leaves nodes
4         U ← 0
5         Points ← null set of points
6         while (L < n)
7         begin
8             y ←random number between 0 and 10 ( do for all edges)
9             while (|y_new − y| < 0.0001 for all edges)
10            begin
11                Calculate y_new using equation 3 for all edges twice
12                y ← y_new
13            end
14            L ←equation 4
15            C_L ← equation 6
16            Add (L, C_L) to points
17            u ← u + 0.0001
18            if ( |L_new − L_old| < 0.001)
19                u ← u ∗ 1.1
20        end
21        for (i = 3 → n)
22        begin
23            Find two points that surround i(L_p < i < L_{p+1})
24            Calculate C_i using interpolation
25            A[i] ← C_i
26        end
27    end
```

7 Experiments and Results

We conducted three sets of experiments to characterize social networks using cycles. In the first set, we used a grid computing machine and exact cycles computation algorithm. While in the second set of experiments we employed the approximated cycles computation algorithm. For both of those sets of experiments we used un-weighted graphs. For the third set of experiments we used weighted graphs and exact cycles computation algorithms.

7.1 Using exact cycles distribution algorithm

We have used a grid of 30 Mac Pro machines watch with two 2.66 dual core Intel processors, which gives a total of 319.2 GHZ processor power. One extra machine is used to control the grid and distribute the tasks to the machines (called clients). The controller follows these steps: 1- read the network from the database, 2-Create the graph based on the chosen model, 3- divide the algorithm into threads and 4-submit each thread as a task to the clients. XGrid has been used as the distributed computing protocol which is developed by Apple and preinstalled on Mac OS X Leopard .The code is written to ne compiled and run under Java 5. With this setup, each thread (mentioned above) can be executed on a single core, which gives us a parallelism of 120 threads at a time.

As mentioned above, the algorithms that are being used to compute the number of cycles in a graph have a very high time complexity, especially for graphs with large number of nodes, edges and hence cycles which is our case. Fine grained network is very large to be computed in a finite time. The Coarse grained network is smaller but still needs more hardware resources than we have. We took small partitions of the network by considering rooms with one specific subcategory (e.g. Kuwait and Saudi Arabia rooms).

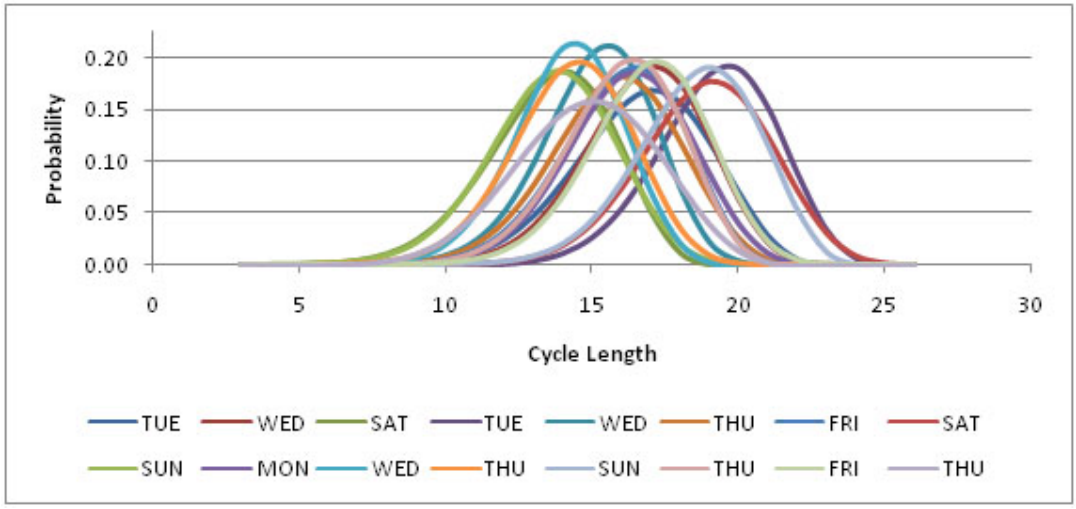

Fig. 5. Exact Probability distribution of Kuwaiti rooms.

Two main experiments on two different subcategories have been done. First experiment was concerned on the Kuwaiti rooms only which are around 30 different rooms during three weeks. The graphs are created based on the method we specified in the previous section for

undirected graphs. Figure 5 illustrate how the cycle probability distribution looks like; notice that all cycles are centered around 15 to 20 cycle length. Figure 6 draws the entropy evolution of the graphs during the two weeks; we can notice that graphs with the minimum entropy are on Wednesdays.

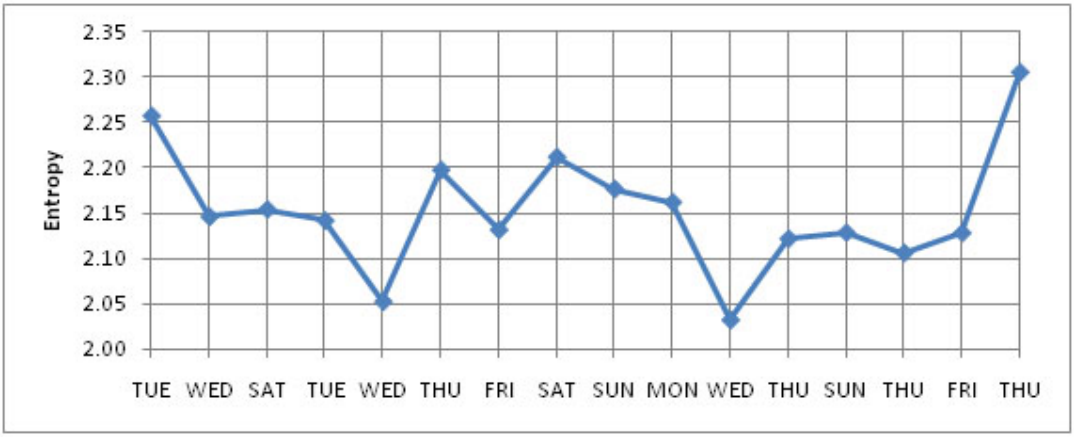

Fig. 6. Exact Cyclic entropy evolution of Kuwaiti Rooms.

The Saudi rooms were our target in the second experiment. For this experiment we choose the other way to model the network which is for directed graphs. Figures 7 and 8 summarize the results.

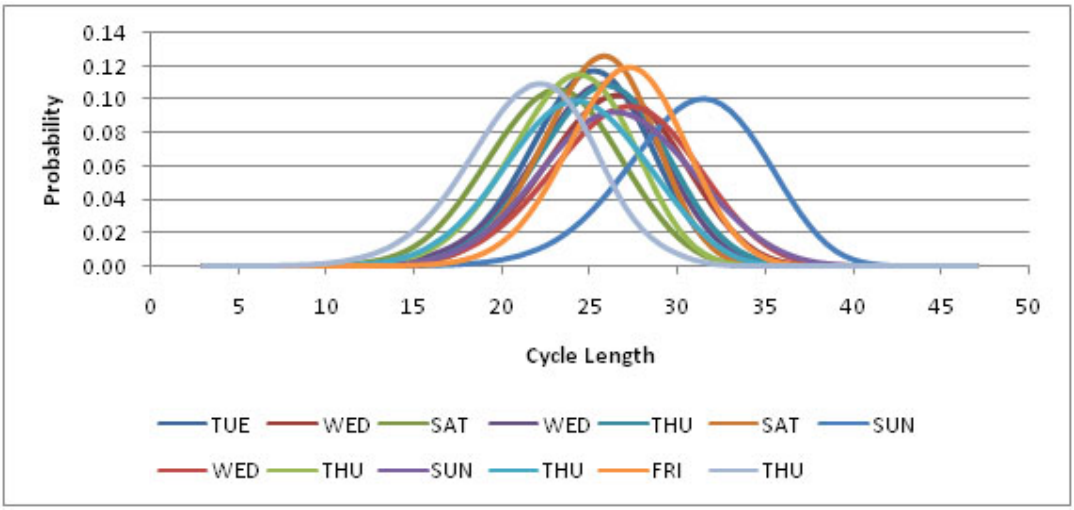

Fig. 7. Probability distribution of Saudi rooms.

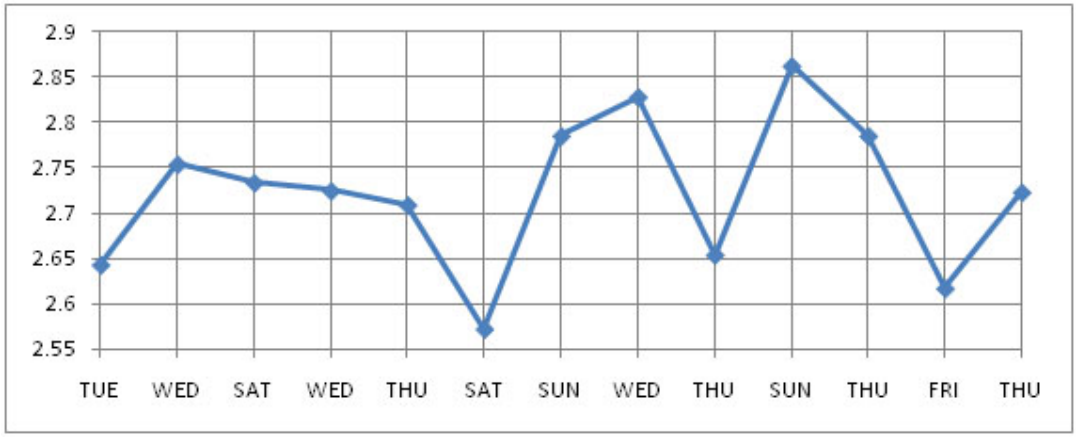

Fig. 8. Cyclic Entropy of Saudi Rooms.

7.2 Using approximate cycles distribution algorithm

For this set of experiments we have used a regular Intel Core Duo based laptop with a 2GB of RAM. For these experiments we used the same data collected from Kuwaiti rooms, which were around 30 different rooms during three weeks. We focused on one day with a graph of 26 nodes/rooms and 91 links (coarse grain model). Figure 9 illustrates the approximated and exact cycles distributions, while Figure 10 shows the approximated and exact cycles probability distributions.

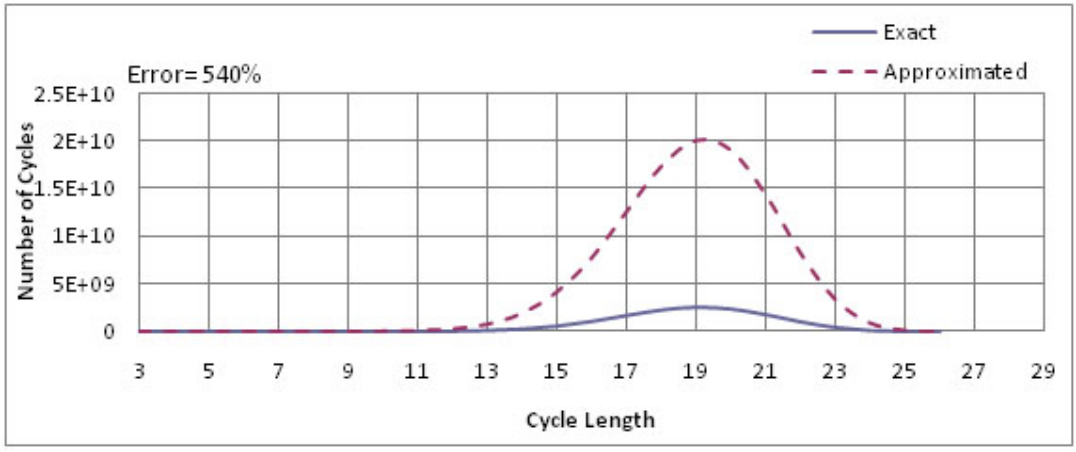

Fig. 9. Exact and approximated Cycles distribution of Kuwaiti rooms.

Figure 9 shows a big error value in computing the cycles distribution, however the approximated algorithm was able to capture the overall probability distribution of the cycles with a very low error as shown in Figure 10. The advantage of using the approximated algorithm was so obvious in reducing the computation time. While the exact algorithm took around 22

hours to compute the cycles distribution, the approximated algorithm finished in around 45 seconds (a speed up of almost 3 magnitudes of order). To get more accurate cycles distributions, you can run the approximated algorithm with a different set of parameters, but this would be on the expense of the running time.

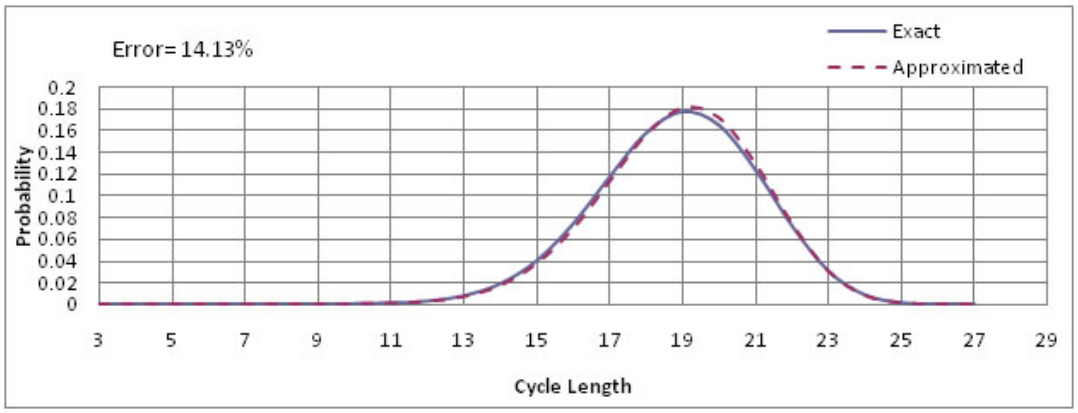

Fig. 10. Exact and approximated cycles probability distribution of Kuwaiti rooms.

7.3 Using weighted graphs

With this set of experiments we tried to show the effect of assigning weights on the graph links and check whether it would better mimic the actual behavior of the networks or not.

For these experiments we used the same data collected from Kuwaiti rooms, which were around 30 different rooms during three weeks. We focused on one day with a graph of 23 nodes/rooms and 65 links (coarse grain model). Figure 11 illustrates the exact weight and un-weighted cycles distributions.

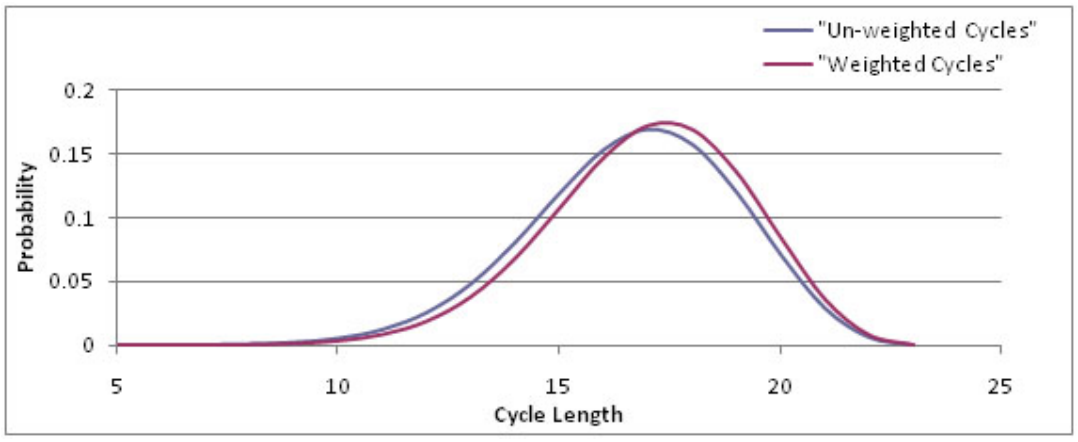

Fig. 11. Exact weighted and un-weighted cycles probability distribution of Kuwaiti rooms.

Figure 11 shows a minor change in the cycles probability distribution graphs for the weighted and un-weighted cycles. We would expect that more differences would appear for larger graphs with more nodes and links. This shows that the frequency of interactions between users or rooms has an affect on the way the information is being spread around the network.

8 Conclusion

Cyclic entropy can be used as a characteristic parameter of a social network and it can imply the behavior of the social network if calculated. Saudi rooms show higher entropy than Kuwaiti rooms but the difference between the minimum and the maximum entropy within two weeks of study are similar. Interestingly, as shown in Figure 6, the least entropy in Kuwaiti rooms occurs consistently on Wednesdays. However such consistency does not exist in the Saudi. From social perspective, Saudi's social actors are more involved with their friends when compared to their Kuwaitis counterparts on regular basis. The cyclic entropy analysis indicates that Saudi rooms tend to be at a higher equilibrium state than Kuwaiti rooms and exhibits natural tendency to be more robust. Using approximated algorithm for computing cyclic entropy enhances the performance of the computation, and in some cases we were able to have an almost 3 magnitudes improvement in the performance. Further studies should be conducted in the future to study the effect of the frequency of interactions between the users in the network and its entropy.

References

1. G. Pilato, A. Augello, G. Vassallo, S. Gaglio (2008), *EHeBby: An evocative humorist chat-bot*, Journal of Mobile Information Systems, Vol.4, No. 3, pp. 165-181.
2. P. Boykin and V. Roychowdhury (2005), *Leveraging Social Networks to Fight Spam*. In IEEE Computer Society Press, Computer, Vol.38, No. 4, pp. 61-68.
3. C. Hsinchun and X. Jennifer (2005), *Criminal Network Analysis and Visualization*. Communications of ACM , pp. 100-107.
4. A. Bagchi, A. Bandyopadhyay and S.K. Mitra (2006), *Design of a data model for social network applications*, Journal of Database Management.
5. T.C Bhanu, S. Mitra. A. Bagchi, A.K. Bandyopadhyay (2006), *Pre-processing and path normalization of a web graph used as a social network*, Special Issue on Web Information Retrieval of JDIM.
6. S. Mitra, A. Bagchi and A.K. Bandyopadhyay (2006), *Complex queries on web graph representing a social network*, 1st International Conference on Digital Information Management, Bangalore, India, pp 430-435.
7. R. Albert and A.-L. Barabasi (2002), *Statistical Mechanics of Complex Networks*, Reviews of Modern Physics, 74.
8. K. Mahdi, M. Safar, & I. Sorkhoh (2008), *Entropy of Robust Social Networks*, Proceedings of IADIS International e-Society Conference.
9. K. Mahdi, H. Farahat, M. Safar (2008), *Temporal Evolution of Social Networks in Paltalk*, Proceedings of the 10th International Conference on Information Integration and Web-based Applications & Services (iiWAS).
10. E. Marinari, R. Monasson, and G. Semerjian (2006), *An algorithm for counting circuits: application to real world and random graphs*, Europhysics Letters.
11. G. Kalna and D. J. Higham (2007), *A clustering coefficient for weighted networks, with application to gene expression data*, AI Communications, Vol.20, No. 4, pp. 263-271.

12. J. Scott (2000), *Social Network Analysis: a Handbook*, Sage Publications Ltd; Second Edition, U.S.A.
13. W. Nooy, A. Mrvar and V. Batagelj (2005), *Exploratory Social Network Analysis with Pajek (Structural Analysis in the Social Science)*, Cambridge University Press, England.
14. S. Wasserman and K. Faust (1994), *Social Network Analysis: Methods and Applications*, Cambridge University Press, England.
15. Retrieved August 1, 2008, from Paltalk: http://www.paltalk.com.
16. G. De Marco and L. Barolli (2007), *On Some Current Results of Graph Theory for Ad-hoc Networks*, Journal of Mobile Multimedia (JMM), Vol. 2, No. 4, pp. 15-33.
17. D. B. Johnson (1975), *Finding All the Elementary Circuits of a Directed Graph*, SIAM Journal on Computing , pp. 77-84.
18. P. De Los Rios, S. Lise and A. Pelizzola (2001), *Bethe approximation for self-interacting lattice trees*, Europhys. Lett., Vol.53, pp. 176-182.
19. T. H. Cormen, C. E. Leiserson, R. L. Rivest and C. Stein (2001), *Introduction to Algorithms (2nd ed.)*, London: The MIT Press.
20. D. Knoke and S. Yang (2008), *Social Network Analysis (2nd ed.)*. SAGE.