International Journal of Control and Automation
Vol.9, No.1 (2016), pp.163-176
http://dx.doi.org/10.14257/ijca.2016.9.1.15

# A Model for Generating Random Networks with Clustering Coefficient Corresponding to Real-World Network Graphs

Natarajan Meghanathan

*Jackson State University*
*natarajan.meghanathan@jsums.edu*

***Abstract***

*In this paper, we propose that when a random network graph model that prefers to close two-hop chains and transform them to triangles as part of link formation. We hypothesize that such a model would generate random network graphs with high clustering coefficients and a larger variation in node degree, compared to that of the well-known Erdos-Renyi (ER) model. We refer to the proposed model as two-hop neighbor preference (THNP)-model that prefers to pair a node with any of its two-hop neighbors rather than to an arbitrary node. The probability of link formation is still governed by the formulation used to generate links in the ER model. We observe the THNP-model to generate random network graphs wherein the clustering coefficient of a node decreases with increase in node degree (resembling closely to several of the real-world network graphs), and the graphs still exhibit a Poisson-style distribution for the node degree and path length.*

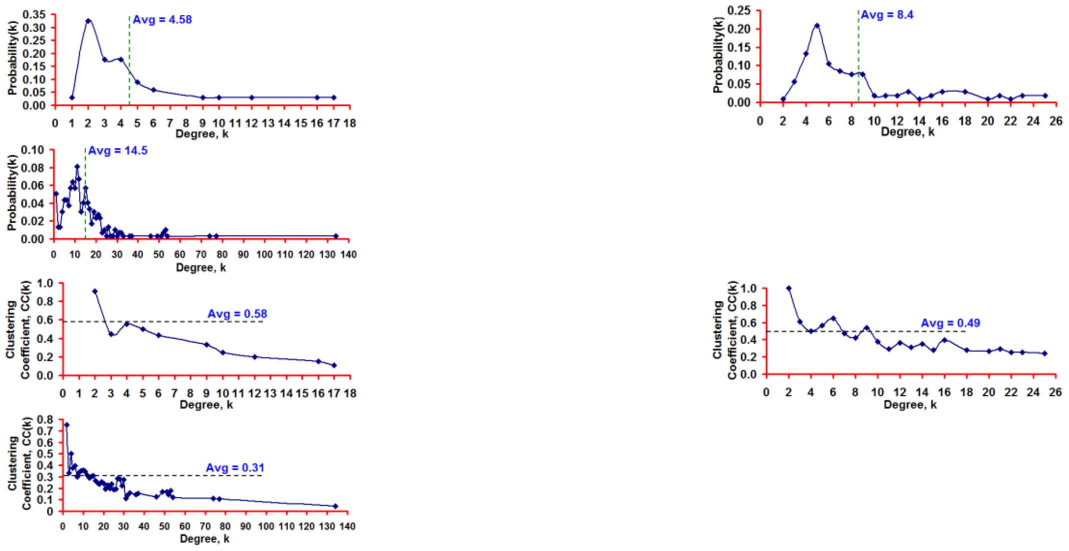
***Keywords**: Two-Hop Neighbors, Random Network Graphs, Clustering Coefficient, Node Degree Distribution, Complex Network Analysis.*

## 1. Introduction

With the growth of social networks and the availability of data about several real-world networks (like biological networks, citation networks, etc), there has been a significant interest to analyze these complex networks from a graph theoretic point of view. A crucial step in analyzing a complex real-world network is to extract the degree distribution of the vertices in the network and compare it with well-known distributions (e.g., Poisson, Gaussian, Exponential, Power-law, etc) of networks [1] generated from theoretical models using the same parameters (like the average node degree, standard deviation of node degree, etc) as that of the real-world network. Once the equivalent theoretical graph model for a real-world network is identified, one can effectively analyze the various characteristics of the real-world network based on the parameters and analytics metrics of the graph model identified.

The results of the analysis of certain real-world networks indicate that the degree distribution of the vertices in these networks does not closely match with any of the well-known distributions. For example, we observe (from Figure 1) that the degree distribution of some sample real-world networks studied in this paper is a Poisson-curve (characteristic of random networks) [2], but with a long tail (characteristic of scale-free networks) or sometimes even with a long head. If these real-world networks are modeled as a random network, then one cannot capture the presence of a few, but appreciable number of vertices that have a significantly larger degree than the rest, as well as cannot capture the decreasing nature of the clustering coefficient of the vertices with increase in node degree (the clustering coefficient of the vertices in a random network is independent of node degree and is simply equal to the probability of a link between any two nodes) [3]. On the other hand, if these real-world networks are modeled as a scale-free network

[4], then one would wrongly capture the degree distribution of vertices with lower degree (while scale-free networks are expected to have a larger number of vertices with lower degree, the real-world networks shown in Figure 1 have only fewer vertices with lower degree). Thus, the motivation of the observations from Figure 1 is that we need a random network graph model that captures the presence of smaller, but appreciable percentage of vertices with both lower degree as well as higher degree (i.e., larger variation in node degree), and also at the same time capture the complex inverse relationship between clustering coefficient and node degree.



Karate Club Network US Politics Book Network C. Elegans Neural Network
(34 nodes, 78 edges) (105 nodes, 441 edges) (297 nodes, 2148 edges)

**Figure 1. Degree Distribution and Clustering Coefficient vs. Degree for Real-World Networks**

In this paper, we propose that whenever we are looking for a link to be setup for a node *u*, we give preference to the two-hop neighbors of node *u*, rather than to an arbitrary node that may or may not be a two-hop neighbor of node *u* before the establishment of the link. We hypothesize that the proposed approach could lead to the closure of several two-link chains as triangles - contributing to a larger clustering coefficient for the vertices as well as contribute to a larger variation in node degree among the vertices, compared to the well-known Erdos-Renyi (ER) model [5] for random networks. The ER model has a characteristic of generating random network graphs with lower variation in node degree (the degree of a majority of the vertices lies close to the average degree) and lower clustering coefficient (the clustering coefficient for any vertex in an ER random graph is close to the probability of a link between any two nodes). According to the proposed two-hop neighborhood preference (THNP) approach, if we are to setup a new link for a node *u* and decide whether the node is to be paired with a randomly chosen candidate node *x* (with which node *u* does not yet have a link), we consider node *x* along with the two-hop neighbors of node *u* (i.e., there exists two-hop chains, say *u*-*w*-*v* involving links *u*-*w* and *w*-*v*) and randomly choose one among the nodes (from the expanded candidate list) with a probability ($p_{link}$) as the neighbor node. The $p_{link}$ value used in the THNP model is the same as the $p_{link}$ value used in the ER model and is obtained by dividing the average node degree of the real-world network by number of nodes - 1. Unlike the ER-model based random network graphs, the THNP-model based random network graphs exhibit an inverse relationship for the clustering coefficient vs. node degree as well as a larger spectral radius ratio for node degree (i.e., larger variation in node degree) [6]. We evaluate the relative proximity (similarity) of the graphs generated according to the THNP model and the ER model with that of the real-world network graphs by computing the rooted sum of the mean square difference values for the average node degree, average clustering coefficient, average path length and the spectral radius ratio for node degree. We observe the THNP-model based random network graphs to be relatively more similar to the real-world network graphs considered, compared to that of the ER-model based random network graphs.

The rest of the paper is organized as follows: Section 2 presents the proposed THNP model for generating random network graphs with a larger clustering coefficient and higher variation in node degree. Section 3 reviews the ER-model, presents the real-world network graphs considered in this paper and illustrates a comparative analysis of the ER-model and the THNP-model with that of the real-world network graphs with respect to the analytics metrics mentioned above. Section 4 discusses related work. Section 5 concludes the paper. Throughout the paper, we use the terms 'node' and 'vertex', 'link' and 'edge', as well as 'network' and 'graph' interchangeably. They mean the same.

## 2. Two-Hop Neighbor Preference (THNP) Model

Let there be a total of *N* nodes among which we need to generate a random network with a larger clustering coefficient and higher variation in node degree. Let $p_{link}$ be the probability for a link between any node pair considered. Initially, all nodes are considered to be isolated in the network. We consider a set *S* of all possible node pairs (a network of *N* nodes has *N*(*N*-1)/2 node pairs). The network generation proceeds in iteration. In each iteration, we pick a node pair *u*-*x* from the set *S* and randomly pick one among the two vertices (*u* or *x*) as the node for which we attempt to setup a link during that iteration. For the sake of discussion, we assume (without loss of generality that between the two vertices *u* or *x*) that vertex *u* gets selected as the vertex for which we attempt to setup a link. Let *C* be the list of candidate vertices with which we could setup a link for *u*. We add vertex *x* as the first vertex to the list *C*. We then consider all the two-hop neighbors *v* of *u* (i.e., connected through a chain *u*-*w*-*v* where *w* is the direct neighbor of *u*) that are not directly connected to vertex *u* and add them to the list *C*. We randomly pick a vertex (say, vertex *v*) from the list *C*. We generate a random number *randNum* in the range [0...1]. If $randNum \le p_{link}$, we setup a link between *u* and *v*; otherwise not. If a link between *u* and *v* gets setup, we remove the pair *u*-*v* from the set *S*. Irrespective of the outcome of the iteration, we remove the pair *u*-*x* from the set *S* and continue with the next iteration. We repeat the above steps in each iteration until the set *S* becomes empty. Figure 2 presents the pseudo code for the THNP-algorithm.

-----------------------------------------------------------------------------------------------------------
-----

**Input:** Number of Nodes, *N*; Probability of a Link, $p_{link}$
**Output:** Set of Links, *E*
**Auxiliary Variables:** Set of Node Pairs, *S*; Candidate List of Vertices, *C*; *Candidate Vertex*
**Initialization:** $E = \phi$
**Begin** ***THNP Algorithm***

```
1   for every vertex $u = 1$ to $N$ do
2       for every vertex $v = u+1$ to $N$ do
3           $S = S \cup \{(u, v)\}$
```

```
4               end for
5       end for
6       while ($S \neq \phi$) do
7               Select a pair ($u$, $x$) randomly from $S$
8               $S = S - \{(u, x)\}$
9               $C = \phi$
10              Generate a random number $r$ in the range [0...1]
11              if $r \leq 0.5$ then
12                      $C = C \cup \{x\}$
13                      $Candidate\ Vertex = u$
14              else
15                      $C = C \cup \{u\}$
16                      $Candidate\ Vertex = x$
17              end if
18              for every vertex $w \in$ Neighbor($Candidate\ Vertex$) do
19                      for every vertex $v \in$ Neighbor($w$) do
20                              if $v \notin$ Neighbor($Candidate\ Vertex$) then
21                                      $C = C \cup \{v\}$
22                              end if
23                      end for
24              end for
25              Pick a vertex $v$ randomly from the list $C$
26              $C = C - \{v\}$
27              Generate a random number $randNum$ in the range [0...1]
28              if $randNum \leq p_{link}$ then
29                      $E = E \cup \{(Candidate\ Vertex - v)\}$
30                      $S = S - \{(Candidate\ Vertex, v)\}$
31              end if
32      end While
33      return $E$
End THNP Algorithm
-----------------------------------------------------------------------------------------------------------
```

**Figure 2. Pseudo Code for the Two-Hop Neighbor Preference-based Random Network Generation Model**

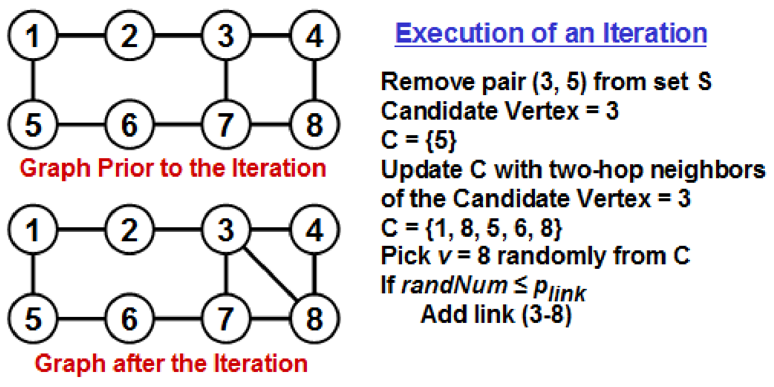
With regards to the time-complexity: lines 1-5 are executed in $\Theta(N^2)$ time, where $N$ is the number of nodes in the network. The while loop from lines 6-32 runs for a total of $N(N\text{-}1)/2 = \Theta(N^2)$ iterations - one iteration for each node pair. For each such iteration, lines 7-17 and 25-31 can be executed in $\Theta(1)$ time; lines 18-24 involve a traversal of the two-hop neighborhood of a vertex. If $K = 2L/N$ is the average degree of a node [3] in the final network with a total of $L$ links, each execution of lines 18-24 takes on average $\Theta(K^2)$ time (as essentially the neighborhood of each of the $K$ neighbors of a node is explored as part of the search for two-hop neighbors). Hence, lines 18-24 could be considered to be of complexity $\Theta(L^2/N^2)$ for each of the $\Theta(N^2)$ iterations. Thus, each iteration of the while loop from lines 7-30 could be considered to take $\Theta(L^2/N^2)$ time and for a total of $\Theta(N^2)$ iterations, the time-complexity for lines 6-32 is $\Theta(L^2)$. Hence, the overall time-complexity of the algorithm to generate a random graph under the THNP model is $\Theta(N^2)$ - for lines 1-5 + $\Theta(L^2)$ - for lines 6-32 = $\Theta(N^2 + L^2)$.

## 2.1. Relevance to Social Networks

As new links get added, the number of two-hop neighbors of a node increases and there are high chances that a node *u* gets linked with one of its two-hop neighbors rather than to a vertex not in its two-hop neighborhood. In social networks, if a user intends to pair with someone in the network, the user is more likely to prefer pairing with someone who is known to his/her neighbors rather than pairing with an unknown stranger. Several social media networks, including Facebook, have been designed to suggest friends of friends as possible candidates to send friend requests. Thus, the two-hop neighbor preference model is very well applicable to model friendship-based interactions in social networks. Note that if more neighbors of the *Candidate Vertex* have a particular vertex as their two-hop neighbor, then the vertex has greater chances of becoming a neighbor of the *Candidate Vertex*. This is in tandem with how links are formed in social networks like Facebook. If a user *A* gets a Friend Request from two users *B* and *C* and is in a situation of accepting just one of these Friend Requests, then user *A* is more likely to accept the Friend Request from the user who has relatively larger number of common friends.

## 2.2. Example to Illustrate the Execution of the THNP Model

Figure 3 presents an example to illustrate the execution of an iteration of the THNP model to setup a link. Let a pair (3, 5) be removed from the set *S* and considered for the possibility of setting up a link for *Candidate Vertex* = 3 (chosen randomly among vertices 3 and 5). Since vertex 3 is chosen as the *Candidate Vertex*, the other vertex (vertex 5) is included to the list *C* of vertices that could be paired with the *Candidate Vertex*. We now look for the two-hop neighbors of the *Candidate Vertex* (vertex 3) and include them to the list *C*. Note that a two-hop neighbor vertex could be included multiple times to the list *C*, with one entry per neighbor. We observe vertex 8 having two entries in the list *C* because of it being the neighbor of vertices 4 and 7. The other two-hop neighbors (vertices 1 and 6) have only one entry in the list *C* as they are the neighbor of only one of the neighbors of the *Candidate Vertex* 3. We randomly pick a vertex from the list *C* = {1, 8, 5, 6, 8}. As is observed in the simulations, vertices that have multiple entries in the list *C* have a relatively larger chance of being selected (increasing the number of closed triangles and hence increasing the average clustering coefficient of the entire network), we observe vertex 8 to be selected as the vertex with which the *Candidate Vertex* 3 will set up a link.



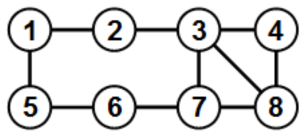
Execution of an Iteration

Remove pair (3, 5) from set S
Candidate Vertex = 3
C = {5}
Update C with two-hop neighbors of the Candidate Vertex = 3
C = {1, 8, 5, 6, 8}
Pick $v$ = 8 randomly from C
If $randNum \leq p_{link}$
Add link (3-8)

**Figure 3. Example to Illustrate the Execution of an Iteration of the THNP Model**

## 2.3. Impact of the Preference for More Commonly Occurring Two-Hop Neighbors

Figure 4 illustrates the impact on the average clustering coefficient and average path length with regards to preference for link formation to a more commonly occurring two-hop neighbor (vertex 8) vis-a-vis preference to a random two-hop neighbor (vertex 6). We notice that the setting up of link 3-8 closes two chains 3-4-8 and 3-7-8 into triangles (contributing to a larger average clustering coefficient of 0.21), but could increase the number of hops from vertices 4, 5 and 6 to the rest of the vertices (contributing to a slightly larger path length of 1.96); on the other hand, setting up of link 3-6 closes only one chain 3-7-6 (contributing to a lower average clustering coefficient of 0.10), but relatively decreases the number of hops on the shortest paths between a majority of the node pairs (an average path length of 1.89). We thus notice that giving preference to the more commonly occurring two-hop neighbor could even double the average clustering coefficient at the expense of a very modest increase (less than 5% increase) in the average path length.



Distance Matrix

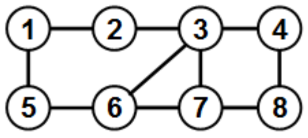
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|
| **1** | 0 | 1 | 2 | 3 | 1 | 2 | 3 | 3 |
| **2** | 1 | 0 | 1 | 2 | 2 | 3 | 2 | 2 |
| **3** | 2 | 1 | 0 | 1 | 3 | 2 | 1 | 1 |
| **4** | 3 | 2 | 1 | 0 | 4 | 3 | 2 | 1 |
| **5** | 1 | 2 | 3 | 4 | 0 | 1 | 2 | 3 |
| **6** | 2 | 3 | 2 | 3 | 1 | 0 | 1 | 2 |
| **7** | 3 | 2 | 1 | 2 | 2 | 1 | 0 | 1 |
| **8** | 3 | 2 | 1 | 1 | 3 | 2 | 1 | 0 |

Avg. Path Length: **1.96**

Clustering Coefficient

| Vertex | CC |
|---|---|
| 1 | 0 |
| 2 | 0 |
| 3 | 2/6 |
| 4 | 1 |
| 5 | 0 |
| 6 | 0 |
| 7 | 1/3 |
| 8 | 0 |
| Avg. CC | **0.21** |

*Preference to More Commonly Occurring Two-Hop Neighbors*



Distance Matrix

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|
| **1** | 0 | 1 | 2 | 3 | 1 | 2 | 3 | 4 |
| **2** | 1 | 0 | 1 | 2 | 2 | 2 | 2 | 3 |
| **3** | 2 | 1 | 0 | 1 | 2 | 1 | 1 | 2 |
| **4** | 3 | 2 | 1 | 0 | 3 | 2 | 2 | 1 |
| **5** | 1 | 2 | 2 | 3 | 0 | 1 | 2 | 3 |
| **6** | 2 | 2 | 1 | 2 | 1 | 0 | 1 | 2 |
| **7** | 3 | 2 | 1 | 2 | 2 | 1 | 0 | 1 |
| **8** | 4 | 3 | 2 | 1 | 3 | 2 | 1 | 0 |

Avg. Path Length: **1.89**

Clustering Coefficient

| Vertex | CC |
|---|---|
| 1 | 0 |
| 2 | 0 |
| 3 | 1/6 |
| 4 | 0 |
| 5 | 0 |
| 6 | 1/3 |
| 7 | 1/3 |
| 8 | 0 |
| Avg. CC | **0.10** |

*Preference to a Random Two-Hop Neighbor*

**Figure 4. THNP Model - Impact of the Preference to More Commonly Occurring Two-Hop Neighbor over a Random Two-Hop Neighbor**

## 3. Simulations

In this section, we discuss the simulation results observed when the real-world networks are modeled based on the proposed THNP model vis-a-vis the well-known Erdos-Renyi (ER) model for random networks. We evaluate the proximity (similarity) of the theoretical graphs generated with the THNP and ER models with that of the graphs abstracting real-world networks using analytics metrics such as (i) Average Clustering Coefficient-*CC*; (ii) Average Path Length-*PL*; (iii) Average Degree-*K* and (iv) Spectral Radius Ratio for Node Degree-*SRK*. The proximity value is modeled as the square root of the sum of the squares of the relative differences with respect to the above analytics metrics for the theoretical graph (abbreviated as *ThG* - could refer to the THNP or ER model graph) and the real-world network graph (abbreviated as *RwG* in equation 1). According to this formulation, the targeted (desirable) proximity value is 0 - indicating that the theoretical graph model captures the fundamental characteristics of a real-world network graph as close as possible.

$$\text{Proximity Value} = \sqrt{\left(\frac{{}_{RwG}}{CC_{RwG}}\right)^2 + \left(\frac{PL_{ThG} - PL_{RwG}}{PL_{RwG}}\right)^2 + \left(\frac{K_{ThG} - K_{RwG}}{K_{RwG}}\right)^2 + \left(\frac{SRK_{ThG} - SRK_{RwG}}{SRK_{RwG}}\right)^2} \quad ..(1)$$

### 3.1. Erdos-Renyi Model

We simulate the generation of a random graph under the ER model as follows: The input parameters are the number of nodes $N$ and probability of link between any two nodes, $p_{link}$. We accumulate a set $S$ of all possible pairs of nodes. We proceed in iterations. In each iteration, we randomly remove a pair ($u$, $v$) from the set $S$ and generate a random number *randNum* in the range [0...1]. If $randNum \leq p_{link}$, we setup the link $u$-$v$; otherwise not.

```
------------------------------------------------------------------------------------------------
Input: Number of Nodes, $N$; Probability of a Link, $p_{link}$
Output: Set of Links, $E$
Auxiliary Variables: Set of Node Pairs, $S$
Initialization: $E = \phi$
Begin ER Algorithm
1       for every vertex $u$ = 1 to $N$ do
2               for every vertex $v$ = $u$+1 to $N$ do
3                       $S = S \cup \{(u, v)\}$
4               end for
5       end for
6       while ($S \neq \phi$) do
7               Select a pair ($u$, $v$) randomly from $S$
8               $S = S - \{(u, v)\}$
9               Generate a random number randNum in the range [0...1]
10              if $randNum \leq p_{link}$ then
11                      $E = E \cup \{(u - v)\}$
12              end if
13      end While
14      return $E$
End ER Algorithm
------------------------------------------------------------------------------------------------
```

**Figure 5. Pseudo Code for the Erdos-Renyi (ER) Random Network Generation Model**

The pseudo code (shown in Figure 5) for the above procedure of generating a random graph under the ER model could be adapted from the pseudo code described in Figure 2. The overall time-complexity of the algorithm to generate a random graph of $N$ nodes and $L$ links under the ER model is $\Theta(N^2)$ as it is the time-complexity to run the *for* loop from lines 1-5 plus the time-complexity of the *while* loop from lines 6-13, both of which could be done in $\Theta(N^2)$ time.

### 3.2. Analytics Metrics

In this sub section, we briefly introduce the metrics used in the analysis.

***Clustering Coefficient***: The clustering coefficient of a node is the probability that any two neighbors of a node are connected [3]. Quantitatively, the clustering coefficient of a node is calculated as the ratio of the actual number of links connecting the neighbors of the node divided by the maximum number of possible links between the neighbors of the node. For a node with degree $K$ (i.e., $K$ neighbors), the maximum number of possible links between the neighbors of the node is $K(K$-1)/2. The clustering coefficient of a network (used in formula 1) is the average of the clustering coefficients of the nodes. The clustering coefficient of a node with an average degree of $K = 2L/N$ could be computed in $\Theta(K^2) = \Theta(L^2/N^2)$ time as it would require to explore the neighborhood of each of the *K* neighbors of the node. The time-complexity to compute the clustering coefficient of a network would be then $\Theta(NK^2) = \Theta(L^2/N)$.

***Path Length***: The length of a path between two nodes *u* and *v* is the minimum number of hops on the shortest path between *u* and *v*. The average path length (used in formula 1) is the average of the path length across all node pairs in the network. The path length of all node pairs could be represented in the form of a distance matrix and could be efficiently computed by running the Floyd's All Pairs Shortest Path algorithm [7] of time-complexity $\Theta(N^3)$ on a graph of *N* vertices.

***Degree***: The degree of a node (i.e., the number of neighbors for a node) is the number of links incident on the node. The average degree of a network (used in formula 1) is the average of the degree values of all the nodes in the network. The degree of a node can be computed in $\Theta(K) = \Theta(L/N)$ time and the average degree of a network of *N* nodes could be computed in $\Theta(NK) = \Theta(L)$ time.

**Iteration 1**

$$\begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 2 \\ 3 \\ 2 \end{bmatrix} \equiv \begin{bmatrix} 0.213 \\ 0.426 \\ 0.426 \\ 0.639 \\ 0.426 \end{bmatrix}$$

Normalized Value = 4.69

**Iteration 2**

$$\begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} 0.213 \\ 0.426 \\ 0.426 \\ 0.639 \\ 0.426 \end{bmatrix} = \begin{bmatrix} 0.426 \\ 0.852 \\ 1.065 \\ 1.278 \\ 1.065 \end{bmatrix} \equiv \begin{bmatrix} 0.195 \\ 0.389 \\ 0.486 \\ 0.584 \\ 0.486 \end{bmatrix}$$

Normalized Value = 2.19

**Iteration 3**

$$\begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} 0.195 \\ 0.389 \\ 0.486 \\ 0.584 \\ 0.486 \end{bmatrix} = \begin{bmatrix} 0.389 \\ 0.779 \\ 1.07 \\ 1.361 \\ 1.07 \end{bmatrix} \equiv \begin{bmatrix} 0.176 \\ 0.352 \\ 0.484 \\ 0.616 \\ 0.484 \end{bmatrix}$$

Normalized Value = 2.21

**Iteration 4**

$$\begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} 0.176 \\ 0.352 \\ 0.484 \\ 0.616 \\ 0.484 \end{bmatrix} = \begin{bmatrix} 0.352 \\ 0.792 \\ 1.100 \\ 1.320 \\ 1.100 \end{bmatrix}$$

Normalized Value = 2.21 converges

**Figure 6. Example to Illustrate the Execution of the Power Iteration Method to Compute the Spectral Radius of the Adjacency Matrix for a Network Graph**

***Spectral Radius Ratio for Node Degree***: The spectral radius ratio for node degree is the ratio of the principal eigenvalue (a.k.a. spectral radius) of the adjacency matrix [8] of the network graph and the average degree of the nodes in the network. The spectral radius of a graph (denoted $\lambda_{sp}(G)$) is computed using the Power-Iteration method [8] of time-complexity $\Theta(N^3)$ involving at most *N* iterations; in each iteration, we compute the principal eigenvector of a graph based on its adjacency matrix (*A*). The principal eigenvector $X_{i+1}$ during the $(i+1)$th iteration is given by $AX_{i+1}/||AX_{i+1}||$, where $X_i$ is the principal eigenvector of the graph at the end of the *i*th iteration and $||AX_i||$ is the normalized value of the product of *A* and $X_i$. To start with, $X_i = [1, 1, ..., 1]$ - a column vector of all 1s corresponding to the number of vertices in the graph.

The Power-iteration method (example illustrated in Figure 6) stops when the normalized value of $||AX_i||$ converges, and the converged normalized value is the spectral radius. If $K_{min}$, $K_{avg}$ and $K_{max}$ are respectively the minimum, average and maximum values for the node degree, it has been established that $K_{min} \leq K_{avg} \leq \lambda_{sp}(G) \leq K_{max}$ [9]. Accordingly, the spectral radius ratio for node degree (SRK), calculated as $\lambda_{sp}(G)/K_{avg}$, will always be greater than or equal to 1.0.

## 3.3. Real-World Network Datasets

In this section, we introduce the six real-world network datasets [10-11] studied in the paper. We consider these six networks to be representatives of real-world networks with a broader range of values for the spectral radius ratio for node degree (i.e., to represent real-world networks with a broader range of values in the variation of node degree). The real-world networks are listed (in the increasing order of their spectral radius ratio for node degree) as follows (the abbreviations used to refer to the networks are indicated in parenthesis, immediately following the network name):

(i) US Football Network (FN): This is a network of 115 teams (representing Division I-A colleges) who played in the Fall 2000 Football season in the US. The nodes represent the teams and there is an edge between two nodes if the corresponding teams played against each other during the season [10].

(ii) US Politics Books Network (PN): This is a network of 105 books related to US Politics sold in Amazon.com. Each book represents a node and there is an edge between two nodes *u* and *v* if someone who bought a book corresponding to node *u* also bought the book corresponding to node *v* and vice-versa [10].

(iii) Karate Club Network (KN): This is a social network of 34 members of a karate club at a US university in the 1970s. The members represent the nodes and there is an edge between two nodes if the corresponding members are friends [10].

(iv) C. Elegans Neural Network (EN): This is a network of 297 neurons in the hermaphrodite Caenorhabditis Elegans [10]. Each neuron is a node and there is an edge between two nodes if the corresponding neurons interact with each other (in the form of chemical synapses, gap junctions and neuromuscular junctions).

(v) Les Miserables Network (LN): This is a co-occurrence network of 77 characters in the novel "Les Misèrables" by Victor Hugo. Each character represents a node and there is an edge between two characters if they co-occurred in at least one chapter of the book [10].

(vi) US Airports Network (AN): This is a network of 332 airports in the US during 1997. Each node is an airport and there is an edge between two nodes if there is a direct flight between the two corresponding airports [11].

In Figure 7, we depict the six real-world networks using the Gephi [12] visualization and analysis tool. The layout algorithm used is the Fruchterman Reingold layout [13] algorithm. The color of the vertices (varied from white to black: grayscale) is a measure of the clustering coefficient of the vertices - the darker/blacker the color of a vertex, the larger is its clustering coefficient and vice-versa. The size of the vertices is a measure of the degree of the vertices - the bigger the size of a vertex, the larger is its degree and vice-versa. For all the networks, except the US Football Network (FN), we observe that nodes of larger size (i.e., nodes having a larger degree) are pale (white or close to white) in color; whereas, the nodes of smaller size (i.e., nodes having a smaller degree) are relatively more darker (black) in color. This illustrates the inverse correlation between node degree and clustering coefficient typically observed in real-world networks. A key motivation for the work in this paper is to develop a random network model that can also display such an inverse correlation between node degree and clustering coefficient, unlike the well-known ER model (according to which the clustering coefficient is independent of node degree).

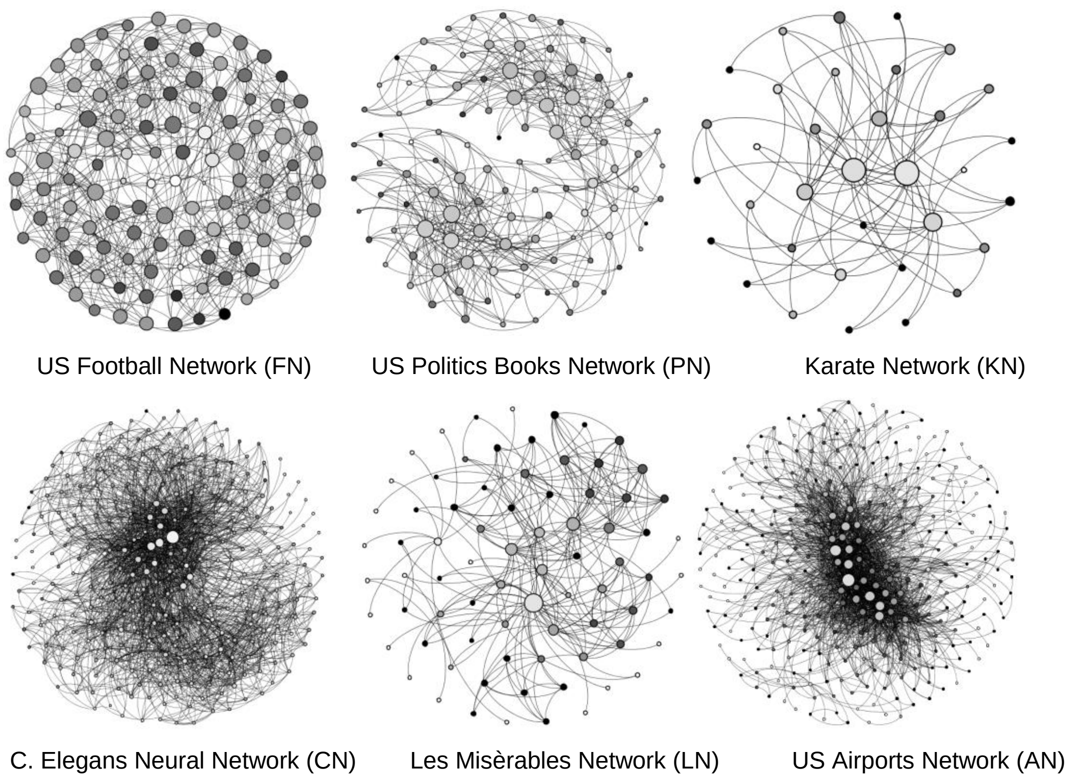
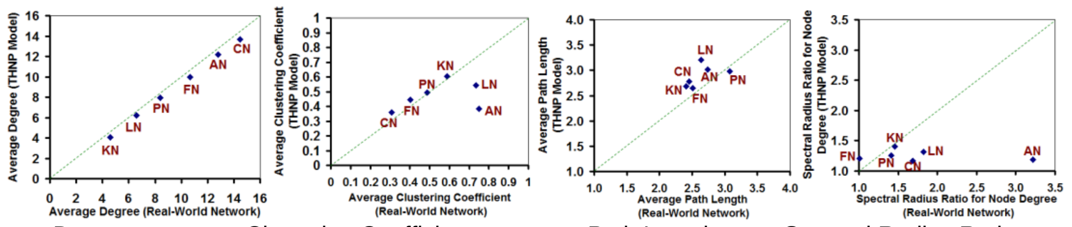

US Football Network (FN)
US Politics Books Network (PN)
Karate Network (KN)
C. Elegans Neural Network (CN)
Les Misèrables Network (LN)
US Airports Network (AN)

**Figure 7: Visualization of the Real-World Networks (Degree vs. Clustering Coefficient)**

In order to model the real-world network as a random graph under the ER and THNP models, we estimate the probability of link ($p_{link}$) between any two nodes in the theoretically modeled network to be $<K_{RwG}>$ / $N$-1, where $<K_{RwG}>$ and $N$ are respectively the average degree of the vertices and the total number of nodes in the real-world network. Figure 8 presents a comparative look at $p_{link}$ and the analytics metrics with respect to the average degree of the six real-world networks. We observe $p_{link}$ and average correlation coefficient to decrease with increase in the average degree; whereas, the average path length and spectral radius ratio for node degree appear to be almost independent of the average node degree.

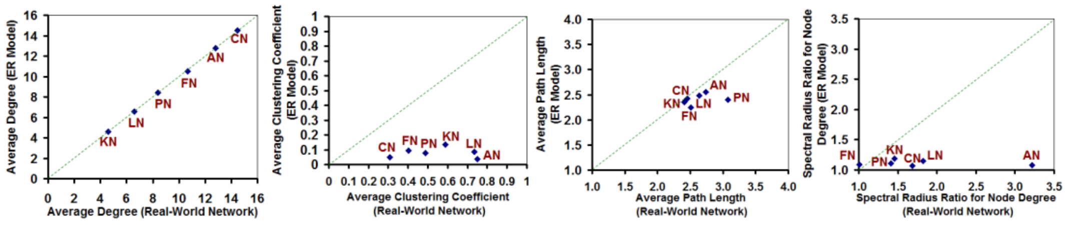
## 3.4. Proximity Evaluation

In this section, we evaluate the proximity (similarity) of the theoretically generated random graphs under the THNP and ER models with that of the corresponding real-world networks. We use the estimated probability for a link between any two nodes (estimated as discussed in Section 3.3) and the number of nodes for the real-world networks to generate the corresponding random networks under both the THNP and ER models. For each real-world network (number of nodes and estimated $p_{link}$ values as parameters), we generated 100 instances of the random networks (under each of the two models: THNP and ER) and averaged the values for the analytics metrics. For each real-world network and the corresponding two random networks, we evaluated the analytics metrics (degree, clustering coefficient, path length and spectral radius ratio for node degree) with respect to their average values and distribution, wherever applicable.



Degree Clustering Coefficient Path Length Spectral Radius Ratio Node Degree

**Figure 8: Visualization of the Proximity of the Real-World Networks and the Corresponding Random Networks Generated under the THNP Model (based on the Analytics Metrics)**



Degree Clustering Coefficient Path Length Spectral Radius Ratio Node Degree

**Figure 9. Visualization of the Proximity of the Real-World Networks and the Corresponding Random Networks Generated under the ER Model (based on the Analytics Metrics)**

Figures 8 and 9 show respectively the proximity of the average values of the analytics metrics for the random networks under the THNP and ER models in comparison with that obtained under the real-world networks. We observe the THNP model to generate random networks whose average clustering coefficient is very much aligned with that of the corresponding real-world networks (in four of the six cases); whereas, the average clustering coefficient of random networks generated under the ER model are nowhere near the values observed for the real-world networks. We also observe the THNP-model random networks to exhibit relatively larger values for the spectral radius ratio for node degree (compared to the ER model) and incur SPR values that are relatively closer to that obtained for real-world networks. The tradeoff is that the average path length of the THNP-model random networks is slightly larger than that obtained for the corresponding real-world networks and the ER-model random networks. The average node degree values for the THNP-model random networks are slightly smaller than the average node degree values for the corresponding real-world networks and the ER-model random networks. This is attributed to the preference for triangle closure using two-hop neighbors (as is observed in the example shown in Figure 4). Though triangle closure involving three nodes (say nodes *u*, *v* and *w*) could facilitate a path length of one hop involving any two of these three nodes, we observe that preference for triangle closure (rather than adding the link to arbitrarily connect any two nodes) only increases the chances of observing a larger path length between any two nodes in the network.

Figure 10 illustrates a quantitative comparison of the relative proximity of the random networks generated under the THNP and ER models to the corresponding real-world network. The proximity values are computed using equation (1) with regards to the average values for the four analytics metrics: degree, clustering coefficient, path length and the spectral radius ratio for node degree. The ideal proximity value desired is 0; we observe the THNP model to incur a proximity value that is less than 0.5 for five of the six real-world networks analyzed and less than 0.25 for three of the six networks. On the other hand, the ER model incurs a proximity value that is greater than 0.5 for all the six real-world networks analyzed.

The proximity values for the random networks under the ER model are larger than the proximity values for the random networks under the THNP model by factors of 2-7 for five of the six real-world networks. We also observe the THNP model to be of very close proximity to the real-world networks that exhibit a moderate variation in node degree (i.e., for spectral radius ratio for node degree values of the real-world networks in the range of 1.1 to 1.5). These are the categories of real-world networks (like the Politics Books Network and the Karate Network) that are neither completely random nor completely scale-free, and the ER model or the Barabasi-Albert (BA) model [4] for scale-free networks cannot effectively model them. This is where the proposed THNP model fits the bill. It could effectively model random networks that have a longer tail and/or a longer head (i.e., those networks that look like a combination of random networks and scale-free networks).

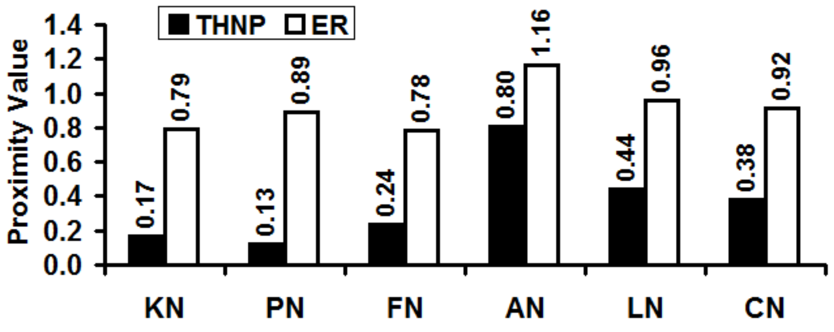


**Figure 10. Quantitative Evaluation of the Proximity of the Random Networks under the THNP and ER Models with that of the Corresponding Real-World Networks (based on the Analytics Metrics)**

## 4. Related Work

In [14], the authors proposed a model to generate random graphs with degree distribution that follows power-law [4]. In [15], the authors propose a model for scale-free random graphs (the network size increases with the addition of one node at a time) in which each new vertex is added to $m$ of the existing nodes with a probability proportional to the degree of the existing nodes plus a strictly positive constant (the latter is associated with introducing randomness in node selection). It was observed that the expected clustering coefficient of the scale-free random network generated with the above approach is asymptotically proportional to $\log n/n$. The THNP model differs from the above model because the candidate list of nodes available for setting up a link for a node $u$ are the two-hop neighbors of $u$ and a randomly chosen node $v$ that is originally considered for the link.

In [16], the authors propose a model for creating random intersection graph as follows: Let $W$ be the set of all $n$ vertices; let $D_1$, $D_2$, ..., $D_n$ be randomly created subsets of the $n$ vertices; two vertices $u$ and $v$ are considered to be linked in the graph if they are in at least $s$ of the $n$ subsets of vertices, where $s \geq 1$ is a model parameter. It is observed in [17] that the random intersection graphs exhibit a linear inverse relationship between clustering coefficient and node degree (i.e., the clustering coefficient of a node is proportional to $k^{-1}$ where $k$ is the node degree). In this paper, we observe that the clustering coefficient-degree relationship need not be simply linear. The random intersection graph model is not sufficient to model the complex inverse relationship between clustering coefficient and node degree, as seen in several real-world networks.

## 5. Conclusions

The high-level contributions of this paper is the proposal of a random network graph model that captures the inverse relationship between node degree and clustering coefficient as well as exhibits a relatively larger variation in node degree matching close to real-world networks whose degree distribution exhibits a random network-like Poisson curve with a long tail and/or long head. We accomplish the above by preferring triangle closure for link formation during the evolution of the random network; a node gives preference to attach to its two-hop neighbors (to facilitate triangle closure) rather than to an arbitrary node. Accordingly, we refer to the proposed model as Two-Hop Neighborhood Preference (THNP)-based random network model. The THNP model is very much suitable for modeling real-world social networks wherein associations (like Friendships in Facebook) are predominantly initiated based on the two-hop neighborhood information (like Friends of Friends as in Facebook). The THNP model can be run just with the knowledge of the number of vertices and number of edges in the real-world network graph. To the best of our knowledge, the proposed THNP model is the first such model for random networks wherein the clustering coefficient of the nodes decreases with increase in node degree (as in the case of real-world networks) and still exhibits a Poisson-style degree distribution.

## Acknowledgements

This paper is a revised and expanded version of a paper entitled " A Random Network Model with High Clustering Coefficient and Variation in Node Degree," presented at the 4th International Conference on Digital Contents and Applications, (DCA 2015), Jeju Island, Korea, December 16-19, 2015.

## References

[1] N. Alon and J. H. Spencer, "The Probabilistic Method", 3rd edition, Wiley-Interscience, **(2008)**.
[2] B. Bollobas, "Random Graphs", 2nd edition, Cambridge University Press, **(2001)**.
[3] M. Newman, "Networks: An Introduction", 1st edition, Oxford University Press, **(2010)**.
[4] A. L. Barabasi and R. Albert, "Emergence of Scaling in Random Networks", Science, vol. 286, no. 5439, **(1999)**.
[5] P. Erdos and A. Renyi, "On Random Graphs I", Publicationes Mathematicae, vol. 6, **(1959)**.
[6] N. Meghanathan, "Spectral Radius as a Measure of Variation in Node Degree for Complex Network Graphs", Proceedings of the 3rd International Conference on Digital Contents and Applications, **(2014)**; Hainan, China.
[7] T. H. Cormen, C. E. Leiserson, R. L. Rivest and C. Stein, "Introduction to Algorithms", 3rd edition, MIT Press, **(2009)**.
[8] D. C. Lay, S. R. Lay and J. J. McDonald, "Linear Algebra and its Applications", 5th edition, Pearson Publishers, **(2015)**.
[9] B. K. Butler and P. H. Siegel, "Sharp Bounds on the Spectral Radius of Nonnegative Matrices and Digraphs", Linear Algebra and its Applications, vol. 439, no. 5, **(2013)**.
[10] M. Newman, http://www-personal.umich.edu/~mejn/netdata/, **(2015)**.
[11] Pajek Datasets, http://vlado.fmf.uni-lj.si/pub/networks/pajek/data/gphs.htm, **(2015)**.
[12] K. Cherven, "Mastering Gephi Network Visualization", 1st edition, Packt Publishing, **(2015)**.
[13] T. M. J. Fruchterman and E. M. Reingold, "Graph Drawing by Force-Directed Placement", Software Practice and Experience, vol. 21, no. 11, **(1991)**.
[14] T. Britton, M. Deijfen and A. Martin-Lof, "Generating Simple Random Graphs with Prescribed Degree Distribution", Journal of Statistical Physics, vol. 124, no. 6, **(2006)**.
[15] N. Eggemann and S. D. Noble, "The Clustering Coefficient of a Scale-Free Random Graph", Discrete Applied Mathematics, vol. 159, no. 10, **(2011)**.

[16] E. Godehardt, J. Jaworski and K. Rybarczyk, “Random Intersection Graphs and Classification”, Proceedings of the 30th Annual Conference of Data Analysis and Knowledge Organization, **(2006)**; Berlin, Germany.
[17] M. Bloznelis, “Degree and Clustering Coefficient in Sparse Random Intersection Graphs”, The Annals of Applied Probability, vol. 23, no. 3, **(2013)**.

## Authors

**Natarajan Meghanathan**, He is currently a Full Professor of Computer Science at Jackson State University, MS, USA. He graduated with a PhD in Computer Science from The University of Texas at Dallas in Spring 2005. He specializes in the areas of Network science and Graph theory, Wireless ad hoc and sensor networks and Cyber security.