Analysis of Watts-Strogatz Networks

Ruowen Liu, Porter Beus, Steven Madler, Bradley Bush

April 15, 2015

Abstract

This report implements an algorithm to generate random Watts-Strogatz networks based on a modified (unbiased) rewiring procedure. The small-world properties of the generated networks are verified with various rewiring probability β. A Matlab and a R package are also included to visualize Watts-Strogatz networks.

1 Introduction

A Watts-Strogatz network is a random graph that obtained by rewiring links in a circle in which only neighbors are connected initially. Watts-Strogatz networks possess small-world properties as the rewiring probability β is big enough. The Watts-Strogatz model was first introduced by Duncan J. Watts and Steven Strogatz in their joint work published in Nature (1998). In our work, we numerically verify the small-world properties of Watts-Strogatz networks of 100 nodes.

2 Method

2.1 The Original Algorithm[1]

A Watts-Strogatz network can be generated by:

1. Construct a regular ring lattice with N nodes each connected to $2m$ neighbors (m on each side). In this work, $m = 2$ by default.
2. For every node $n_i = n_1, \ldots, n_N$ take every edge (n_i, n_j) with $i < j$, and rewire it with probability β. Rewiring is done by replacing (n_i, n_j) with (n_i, n_k) where k is chosen with uniform probability from all possible values that avoid self-loops ($k \neq i$) and link duplication (there is no edge (n_i, n'_k) with $k' = k$ at this point in the algorithm).

However, the original Watts-Strogatz algorithm is order-dependent, i.e., an edge (n_i, n_j) with $i < j$ can only be rewired to (n_i, n_k) but (n_j, n_k). For example, suppose $\beta \approx \frac{1}{2N}$ is a very small value that only one link is rewired, then we can never obtain a network which rewires $(1, N)$ to $(2, N)$.

2.2 Modified (Unbiased) Algorithm

1. Construct a regular ring lattice with N nodes each connected to $2m$ neighbors (m on each side). In this work, $m = 2$ by default.
2. For every node $n_i = n_1, \ldots, n_N$ take every edge (n_i, n_j) with $i < j$, and rewire it with probability β. Rewiring is done by replacing (n_i, n_j) with (n_p, n_q) where p, q are chosen with uniform probability from all possible values that avoid self-loops and link duplication ($p = i$ or $q = j$ is possible but not simultaneously).

The method actually implemented removes the bias that is implicit in the original Watts-Strogatz method. Two results of this modified method are: one, rewired edges cannot be rewired to their original nodes and two, there is no bias due to the linear way in which the former algorithm operated (i.e. being the first node does not give you any special privileges).

3 Implementation

We focus on the modified (unbiased) algorithm, which places no restriction on the nodes when rewiring edges. Only the upper (or lower) triangular part of the adjacent matrix is needed for rewiring. Note that, in order to avoid self-loops, we first reset the diagonal of the adjacent matrix to -1 (or any value other than 0, 1). The procedure is listed below.

1. For a given rewiring probability β, generate an array of uniform random values simultaneously, $\{r_i\}$ for all links, i.e., for the positions of ones in the triangular adjacent matrix.
2. Assign the random numbers to the $2N$ links in the original network.
3. Mark all $r_i < \beta$ and randomly dart ones to the positions of zeros in the triangular adjacent matrix.
4. Change the marked ones to zeros.

Main part of Matlab codes:

```
A = diag(ones(N-1,1),1)+diag(ones(N-2,1),2)+diag(ones(2,1),N-2)+diag(ones(1),N-1);
% initial adjacent matrix
r = rand(N*2,1); % generate random numbers for the links
idx = find(A==1); % positions of ones in A
rewired = idx(r<p); % mark to-be-rewired links
A = tril(-1*ones(N))+A; % assign the lower triangular and diagonal to -1 (we only need upper)
available_pos = find(A==0);
rand_idx = randsample(available_pos,length(rewired));
A(rewired) = 0; A(rand_idx) = 1;
A=triu(A,1); % restore A to upper matrix
A=A+A'; % if you need full matrix
```

4 Visualizing the Network

Though in the end visualization of networks with more than 50 nodes is of somewhat limited utility, there are methods available for this type of visual analysis on Watts-Strogatz Networks.

4.1 Matlab Visualization

Simple but effective Matlab code can place nodes in a ring and, based on a given adjacency matrix, connect the nodes according to the network structure. This is done using the ***gplot*** function.

```
for i = 1:N % Build Circle
    x = r*cos(2*pi/N*i);
    y = r*sin(2*pi/N*i);
    R(i,:) = [x y];
end
% plot
```

```
gplot(A,R,'-*') % A is an N by N adjacency matrix
```

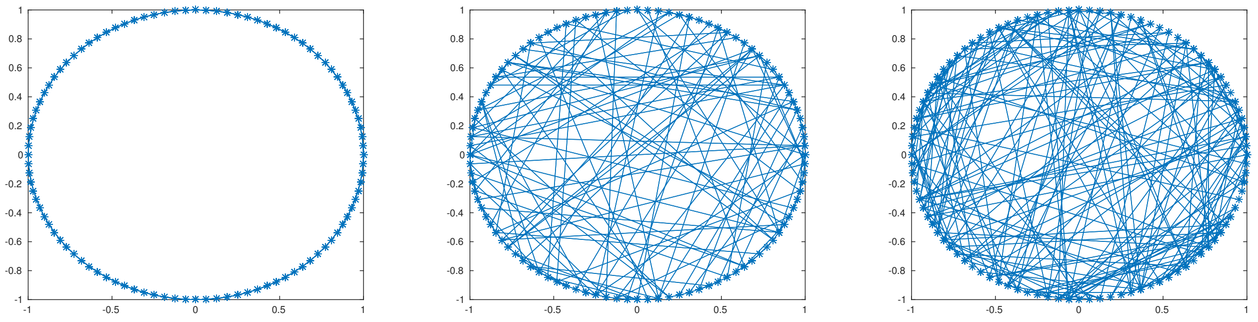

Figure 1: Some typical Watts-Strogatz ring networks using varying rewiring probabilities ($\beta = 0, 0.5, 1.0$ from left to right).

4.2 R Visualization

There is a well-built library in R, called ***igraph***. This tool visualizes ring networks more clearly because it can plot links as curves instead of straight lines to avoid overlap. The main part of R codes are shown below.

```
# Create a Watts-Strogatz network with rewiring
# probabilities of 0, 0.5, and 1 respectively
# Note: must load igraph before using the following commands

ws_graph_0 <- watts.strogatz.game(1, 50, 2, 0)
ws_graph_0.5 <- watts.strogatz.game(1, 50, 2, 0.5)
ws_graph_1 <- watts.strogatz.game(1, 50, 2, 1)

# Plot the Watts-Strogatz networks using igraph
plot.igraph(ws_graph_0, layout=layout.circle, vertex.label=NA, vertex.size=5,
edge.width = 2, edge.curved=1)
plot.igraph(ws_graph_0.5, layout=layout.circle, vertex.label=NA, vertex.size=5,
edge.width = 2, edge.curved=1)
plot.igraph(ws_graph_1, layout=layout.circle, vertex.label=NA, vertex.size=5,
edge.width = 2, edge.curved=1)
```

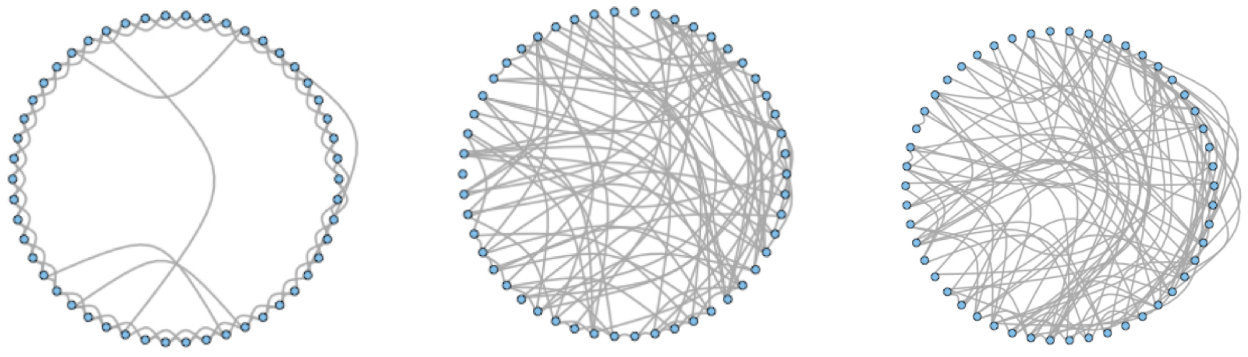

Figure 2: Some typical Watts-Strogatz ring networks plotted by R ($\beta = 0.05, 0.5, 1.0$ from left to right).

5 Network Properties

The Matlab Tools for Network Analysis (2006-2011) [a] are used to compute the properties of networks generated above. Notice that we modified the Matlab codes for a better performance for undirected simple networks but the copyright still goes to Massachusetts Institute of Technology.

Different rewiring probability β are tested in our work.

5.1 Mean Diameter

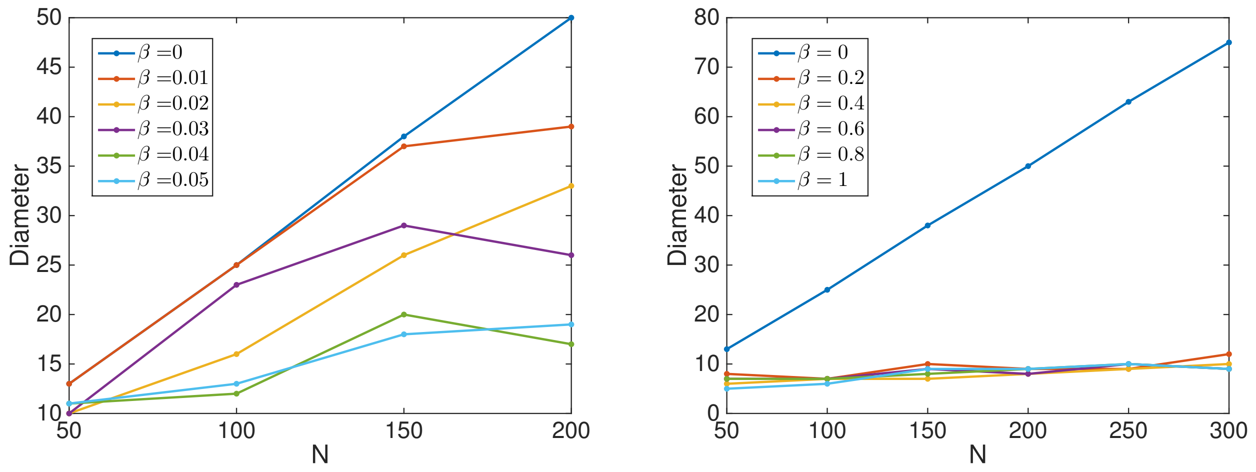

Figure 3: The mean diameter of the rewired network drops quickly when β arises. When $\beta \geq 0.2$, the diameter has shown a small network property.

[a] `http://strategic.mit.edu/downloads.php?page=matlab_networks`

5.2 Clustering Coefficient

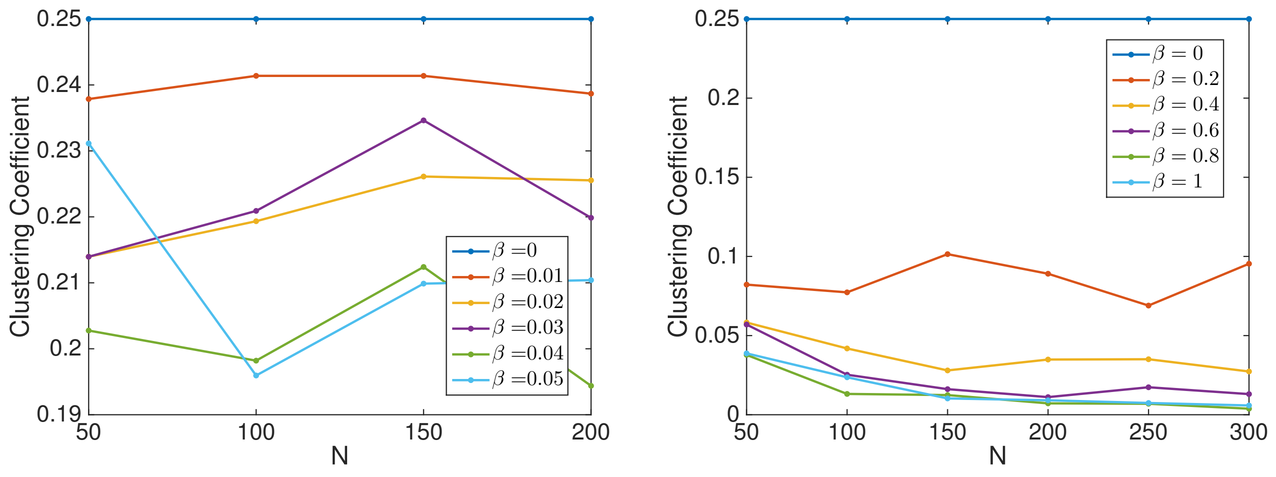

Figure 4: As can be proven, the clustering coefficient for $\beta = 0$ is 0.25 for all values of N. As β increases, the clustering coefficient drops because rewiring edges leads to breakdown of cliques with neighbors.

5.3 Average Shortest Path

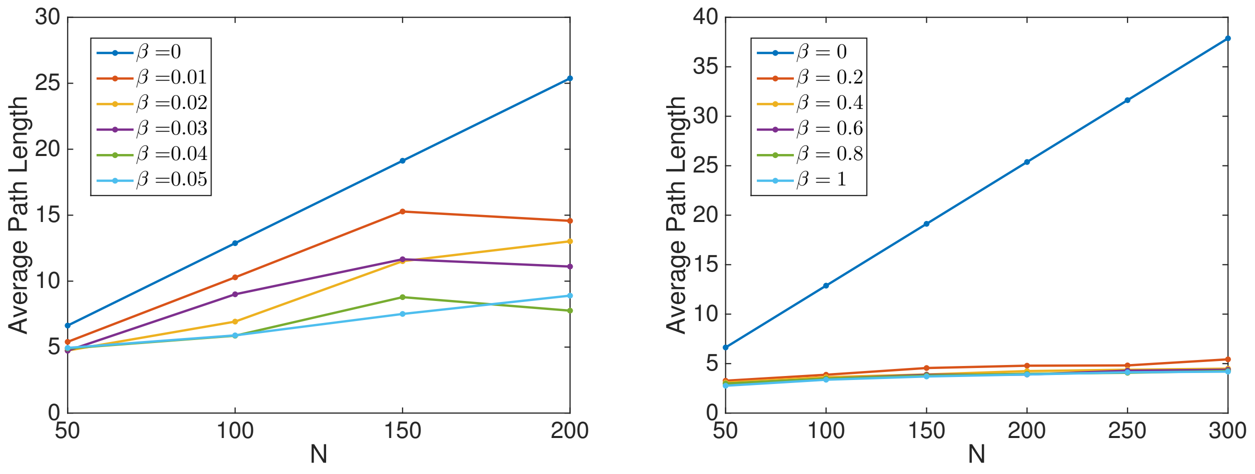

Figure 5: As with the mean diameter, the average shortest path of the rewired network drops quickly when β rises. When $\beta \geq 0.2$, the average shortest path shows typical small world network behavior.

6 Conclusion

Due to the simplicity of its generation and the small world properties it exhibits, Watts-Strogatz and Watts-Strogatz like graphs are common place in simulation across a variety of fields. As has been shown, the generation of these networks can be vectorized for greatly decreased computational intensity. Furthermore it has been shown that they exhibit small world properties. The work here shows that for a small rewiring probability $\beta \approx 0.2$ the Watts-Strogatz random networks already exhibit small-world properties. For practitioners that use such networks to simulate Osotua or other real-world applications, they can use a guideline as $\beta \approx 0.2$ instead of a large β to sufficiently guarantee small-world properties.

References

[1] D. J. Watts and S. H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393(6684):440–442, 1998.