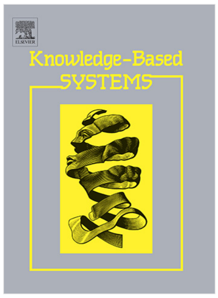
Accepted Manuscript



A novel edge rewiring strategy for tuning structural properties in networks

Junfang Mu, Wenping Zheng, Jie Wang, Jiye Liang

PII: S0950-7051(19)30169-8
DOI: https://doi.org/10.1016/j.knosys.2019.04.004
Reference: KNOSYS 4734

To appear in: *Knowledge-Based Systems*

Received date : 25 September 2018
Revised date : 18 January 2019
Accepted date : 9 April 2019

Please cite this article as: J. Mu, W. Zheng, J. Wang et al., A novel edge rewiring strategy for tuning structural properties in networks, *Knowledge-Based Systems* (2019), https://doi.org/10.1016/j.knosys.2019.04.004

A Novel Edge Rewiring Strategy for Tuning Structural Properties in Networks

Junfang Mu[a,b], Wenping Zheng[a,b], Jie Wang[a,b], Jiye Liang[a,b,*]

[a] *School of Computer and Information Technology, Shanxi University, Taiyuan 030006, Shanxi, China*
[b] *Key Laboratory of Computational Intelligence and Chinese Information Processing of Ministry of Education, Taiyuan 030006, Shanxi, China*

Abstract

Synthetic networks can be generated to mimic the dynamics and evolution of complex interconnected systems in real world. Many network models have been established based on various structural and topological characteristics, such as degree distribution, clustering coefficient, mixing parameter, etc. These generated network models can serve as null models in hypothesis testing to assess nontrivial results about real-world data in terms of statistical significance and generality. Therefore, researchers have actively pursued the development of network generation models with some given topological characteristics. So far, Standard Monte Carlo method and Simulated Annealing method are popular to adjust the clustering coefficient and average path length of the existing networks. However, these methods require a large number of calculations and are easy to fall into local extremes, which might limit the adjusting range of the algorithm. In order to reduce the amount of calculation and expand the range of adjustment, we propose a local structure based edge rewiring method to adjust the clustering coefficient and average path length of the network. By selecting of an appropriate local neighborhood of the node, we compute the 'local' clustering coefficient and 'local' average path length on the "local neighborhood", and then calculating cost in each adjusting iteration is greatly reduced. Focusing on the "local neighborhood" strategy helps the algorithm escape from local

*Corresponding author. E-mail: ljy@sxu.edu.cn; Tel.: +86-351-7010-566

extreme. Therefore, our edge rewiring strategy provides a border adjustment range of clustering coefficient and average path length in reasonable computing time. Experiment results show that our edge rewiring strategy can provide a boarder adjusting range for clustering coefficient and average path length than standard Monte Carlo method and the Simulated Annealing method under the same computation condition.

Keywords: network generation model, edge rewiring, clustering coefficient, average path length, community structure

1. Introduction

Complex networks[1, 2] are currently being studied across many fields of science and engineering. A complex network is a set of items, with connections between them. Examples of complex networks include the Internet[3], WWW, social networks[4], protein interaction network[5], gene-regulatory network and economic network. Real complex networks cannot be easily accessed or even duplicated and may grow too slow for decisions based on their structure to be taken. Therefore, researchers have actively pursued the development of network generation models to mimic the creation and evolution of complex networks emerging from a variety of real world interconnected systems. Network generation models have a number of benefits and applications[6], as they can serve as a null model in hypothesis testing, allowing nontrivial results regarding real world data to be easily assessed in terms of statistical significance and generality.

It is necessary to study and comprehend the structural characteristics of real-world complex networks, and then establish appropriate mathematical network models. Many cases studying on various real-world networks have been reported from different perspectives. The networks with small-world effect[7] always have higher clustering coefficient and shorter average path length; the networks with scale-free feature[8] obey power-law degree distribution; the networks with community structures[9, 10] could be divided into some groups such that many links connecting nodes of the same group and comparatively few links joining nodes of different groups. Among various structural characteristics to depict the topology and dynamics of a complex network, the clustering coefficient and average path length of a network are the two important attributes containing significant information concerning its topological structure. The clustering coefficient of a network indicates how well connected a node is to its neighbors and how compact the network is locally. The average path length expresses a global characteristic of the network regarding the average number of steps required to reach any two nodes. The coincidence of short average path length and high clustering coefficient is a general feature of a complex network. How to adjust the clustering coefficient and average path length of a network model has attracted more and more interest. Standard Monte Carlo method and Simulated Annealing method are popular to adjust the clustering coefficient and average path length of the existing networks. However, these methods require a large number of calculations and are easy to fall into local extremes, which might limit the adjusting range of the algorithm.

In this paper, we propose a local structure based edge rewiring strategy to adjust the clustering coefficient and average path length of the network. By selecting of an appropriate local neighborhood of the node, we compute the 'local' clustering coefficient and 'local' average path length on the "local neighborhood", instead of computing clustering coefficient and average path length on the whole network. By doing that, we save calculating costs in each adjusting iteration. What more, the adjustment of one pair of edges might not affect the clustering coefficient or average path length of the whole network, which might lead an algorithm fall into local extreme. The adjustment of one pair of edges has a larger probability to affect the local clustering coefficient or local average path length, which might help the algorithm escape from local extreme. Therefore, our edge rewiring strategy can provide border adjustment range of clustering coefficient and average path length in reasonable computing time. Experiment results show that our edge rewiring strategy can provide a boarder adjusting range for clustering coefficient and average path length than standard Monte Carlo method and the Simulated Annealing method under the same computation condition.

The rest of the paper is organized as follows. In Section 2, we provide some basic terminologies and notations used in this paper, and introduce some dominant edge rewiring methods in the literatures briefly. In Section 3, we present our edge rewiring method (ERS) in detail. In Section 4, we show experiment results of our edge rewiring method (ERS) compared with standard Monte Carlo method and the Simulated Annealing method. We conclude possible future directions of our research in Section 5.

2. Related Work

2.1. Notations

A *network* (or a *graph*) G with N nodes and M edges can be denoted as $G = (V, E)$, where $V = \{v_1, v_2, \cdots, v_N\}$ and E is the edge set of G. We only consider simple graph here. The *neighborhood of node* $v_i \in V$ is denoted as $N_G(v_i) = \{v_j \mid v_j \in V, v_jv_i \in E\}$. Let $d_G(v_i) = |N_G(v_i)|$ represents the *degree* of node v_i. The *degree sequence* $\mathbf{D}$ of G is the non-increasing sequence of its node degrees, say $\mathbf{D} = (d_G(v_1), d_G(v_2), \cdots, d_G(v_N))$. A sequence $\mathbf{d} = \{d_1, d_2, \cdots, d_n\}$ of nonnegative integers is called a *graphical sequence* if there is a simple graph $G = (V, E)$ with degree sequence $\mathbf{d}$. In this case we also say that G *realizes* $\mathbf{d}$. We use k_G, $\Delta(G)$ and $\delta(G)$ to denote *average degree*, *maximum degree* and *minimum degree* of G, respectively. An *induced subgraph* $G[S]$ is a graph whose node set is $S \subseteq V$ and whose edge set consists of all of the edges in E that have both endpoints in S. We write $[S]$ to denote the induced subgraph by node subset S when without causing confusion. Readers are referred to [11] for terminations not mentioned here in detail.

The *degree distribution* is defined by a probability function, $p(d)$, which can be understood as the probability that a randomly picked node has degree d, where each node has an equal probability to be picked. A network is scale-free if its degree distribution has a power-law form and is independent of the connectivity scale[11, 12]. In a scale-free network, the possibility for a node with

Table 1: Typical statistical indicators of the complex network instances[2]

Network	Type	N	$\overline{k}$	APL	C_G	α
physics coauthorship	Undirected	52909	9.27	[illegible]	0.56	–
Student relationship network	Directed	573	1.66	16.01	0.001	–
WWW nd.edu	Directed	269504	5.55	11.27	0.29	2.1/2.4
word co-occurrence	Undirected	460902	70.13	–	0.44	2.7
software classes	Directed	1377	[illegible]	[illegible]	0.012	–
electronic circuits	Undirected	24097	[illegible]	[illegible]	0.03	3
protein interactions	Undirected	2115	[illegible]	[illegible]	0.071	2.4
freshwater food web	Directed	92	[illegible]	1.90	0.087	–

* Basic statistics for a number of published networks. The properties measured are: type of graph, directed or undirected; total number of nodes N; average degree $\overline{k}$; average path length APL; clustering coefficient C_G; exponent α of degree distribution if the distribution follows a power law (or "–" if not; in/out-degree exponents are given for directed graphs).

degree d is $P(d) \sim d^{-\alpha}$, where α is a constant determined by the given network. Different complex networks have different power law exponent even if the same network in the evolution process.

For a network G, the *clustering coefficient of a node* $v_i \in V(G)$ is given by the proportion of edges between the nodes within its neighbourhood divided by the number of edges that could possibly exist between them, denoted as

$$C_G(v_i) = \frac{2|E([N_G(v_i)])|}{d_G(v_i)(d_G(v_i)-1)}. \tag{1}$$

The *clustering coefficient* of G is the average of the local clustering coefficients of all nodes of G, i.e.

$$C(G) = \frac{1}{N}\sum_{i=1}^{N} C_G(v_i). \tag{2}$$

Let $l_G(v_i, v_j)$ be the shortest distance between v_i and v_j in G, the *average path length* (APL) of G is defined as the average of the distance between all node pairs, defined as

$$APL(G) = \frac{\sum_{i \neq j} l_G(v_i, v_j)}{N(N-1)}. \tag{3}$$

The clustering coefficient of a network indicates how well connected a node is to its neighbors and how compact the network is locally. The average shortest path length expresses a global characteristic of the network regarding the average number of steps required to reach any two nodes. The coincidence of short average shortest path length and high clustering coefficient is a general feature of a complex network. The clustering coefficient of a small world network is much larger than that of the random network, $C \gg C_{ER}$, whereas the average path length of a small world network increases logarithmically with the number of nodes, $APL \sim \ln N$. Table 1 shows the basic statistical indicators of some complex network instances.

An important characteristic of these networks is the presence of community structures[13–15], i.e., with many links connecting nodes of the same group and comparatively few links joining nodes of different groups. Newman in 2004 proposed *modularity*[9, 10] to measure the community structure of a given network. Specifically, suppose V is partitioned into a set $\mathcal{C} = \{\mathcal{C}_1, \mathcal{C}_2, \cdots, \mathcal{C}_c\}$ of c non-overlapping communities, with union $\bigcup_{\mathcal{C}_i \in \mathcal{C}} \mathcal{C}_i = V$. Generally, $c \ll N$. Here, we define community size s_k represent the number of the nodes which belong to community k, i.e., $s_k = |\mathcal{C}_k|$. The function $\tau(v_i)$ represents label of community which node v_i belongs to, namely the range of values for $\tau(v_i)$ is $1 \leq \tau(v_i) \leq c$. The modularity of network calculation formula is (4), where the function $\omega(\tau(v_i), \tau(v_j))$ indicates whether the node v_i and the node v_j belong to the same community, as shown in formula (5).

$$m = \frac{1}{2M} \sum_{i=1}^{N} \sum_{j=1}^{N} (a_{ij} - \frac{d(v_i)d(v_j)}{2M}) \omega(\tau(v_i), \tau(v_j)) \tag{4}$$

$$\omega(\tau(v_i), \tau(v_j)) = \begin{cases} 1, \ \tau(v_i) = \tau(v_j) \\ 0, \ \tau(v_i) \neq \tau(v_j) \end{cases} \tag{5}$$

Many network models have been established based on various structure and topological characteristics, such as degree distribution, clustering coefficient, mixing parameter, etc. ER random network, created by Erdös and Rényi, is a completely random network[16], whose degree distribution follows a Poisson distribution. Watts and Strogatz proposed WS small world network[7] that have both features of high clustering coefficients along with short average path lengths. Barabási and Albert proposed BA scale-free network[8] to reflect "rich gets richer" phenomenon. There are many other models which are generalizations of these famous models, such as Leinberg navigable small world model[17], EBA[18] model, fitness model[19], local world model[20], HK model[21], etc. GN-Benchmark[10] proposed by Girvan and Newman in 2004 is one of the most popular model with communities structures. And there are many generalization model[22–28] of GN-Benchmark are widely used in practice, such as Weighted GN model[22], heterogeneous GN model[23] and LFR-Benchmark model[24, 25], etc.

2.2. *Works closely related to edge rewiring*

The topological characteristics of many networks change over time. In order to capture the empirically observed ones, there are some network generation models that control the clustering coefficient or adjust clustering coefficient or average path length in existing networks.

In 2002, Holme and Kim[21] extended standard BA scale-free network model to include a "triad formation step" when introducing new nodes. In HK model, the clustering coefficient could be tunable by changing control parameter m_t–the average number of triad formation trials per time step. In 2003, Newman proposed a model of a network[29] that has both a tunable degree distribution and a tunable clustering through bipartite project method, projecting bipartite graph into the individuals with probability p of knowing others with whom they share a group. In 2004, Volz[30] used a Markov chain Monte Carlo technique to generate both a given degree distribution and a clustering coefficient by constructing the appropriate queue to construct a triangle with a certain probability. In 2005, Badham and Stocker[31] proposed an algorithm based on configuration model with triangle formation for adjusting three properties of networks, containing the degree distribution, the clustering and the assortativity. In 2006, Guo and Zhou[32] proposed a simple rule that generates scale-free small-world networks with tunable assortative coefficient by controlling parameter p that is a probability of choosing neighbors. In 2010, Badham and Stocker[33] presented a spatially constructed algorithm that generates networks with constrained but arbitrary degree distribution, clustering and assortativity by controlling probability p in create edge process. The above methods can control clustering coefficient in generation of a network. However, they can not be used to adjust topological attributes(including clustering coefficient) of an existing network.

In 2002, Maslov and Sneppen[34] proposed an edge exchange method that randomly choose two edges (say, connecting nodes A and B, and nodes C and D), and then alter the original edges AB and CD to AC and BD, provided that none of these edges already exist in the network. The important property of the edge exchange method is that this process does not change the degree of each node. However, a blind repetition of the above edge exchanges have been shown to destroy all degree-degree correlations.

In order to study the performance of networks of artificial neurons with focus on the role of the clustering coefficient, Kim in 2004 introduced an algorithm[35] to control the clustering coefficient of a given network with the degree of each node kept fixed and guarantee the direction of each adjustment. Kim's algorithm randomly chooses two edges and then rewires to have different end nodes, and accepts the edge trial only when the new network configuration has higher (or lower) clustering coefficient. This is the standard Monte Carlo(denoted as KMC in following) simulation at zero temperature with the Hamiltonian H:

$$H = \sum_{v} c_v$$

where c_v is the clustering coefficient of the node v, Kim's algorithm could guarantee the direction of each adjusting of clustering coefficient with the degree of nodes kept unchanged. The detailed description of KMC algorithm is shown in Algorithm 1. In KMC method, we need to input a user-specified value, called desire clustering coefficient. If the cluster coefficient is to be increased (or decreased), then we would accept the edge trial that could lead to a higher (or lower) clustering coefficient of the network. The edge trial would be performed iteratively until the desired clustering coefficient or the maximum iteration is reached.

Algorithm 1 Monte Carlo simulation (KMC) for adjusting clustering coefficient

Input: Graph G, the desired value f' of clustering coefficient,
the maximum iteration $maxt = 100000$, threshold $\varepsilon = 0.0001$;

Output: Graph G' with the value of clustering coefficient approximately equal to f'.

```
Let t=0, G^(0) = G;
Calculate f(G^(t)), the value of clustering coefficient of G^(0);
Calculate E(C) = |f(G_t) − f'|
while (|f(G^(t)) − f'| ≥ ε) and (t < maxt) do
    t ← t + 1;
    Select randomly an edge pair ⟨x1x2, x3x4⟩ from G satisfying x_i x_j ∉ E for
    i ∈ {1, 2} and j ∈ {3, 4};
    Let G'_t = G ∪ {x1x3, x2x4} − {x1x2, x3x4};
    Calculate E(C') = |f(G'_t) − f'|
    if (E(C') < E(C)) then
        G_t = G'_t;
    else
        G_t = G'_t − {x1x3, x2x4} ∪ {x1x2, x3x4};
    end if
end while
```

To study the influence of average path length on the emergency dynamics of the majority-rule model, Andreas et al. in 2015 proposed an edge rewiring method[36] based on Simulated Annealing (denoted as ASA in following) to tuning the average path length of a network to a user-specified value. The objective of ASA method is to minimize the difference between the current average path length and the target average path length, i.e., $E(L) = ||L - L^{target}||$. The algorithm selects randomly two edges AB and CD such that each of them do not have any common neighbors. Then, rewiring the edges and evaluating the new average path length of the network $L^{'}$ and the corresponding objective function $E(L^{'})$. Then, algorithm ASA accepts or rejects the new configuration using the Metropolis procedure, i.e., if $E(L^{'}) < E(L)$, ASA would accept the edge exchanges; otherwise, it would accept the edge exchanges with a probability $e^{-\frac{E(L^{'})-E(L)}{Temp}}$, where $Temp$ is the system's pseudo-temperature and would decrease in the way of annealing scheme. In ASA, the initial systems pseudo-temperature was set to $Temp$=10, and the pseudo-temperature decreased 10% every 200 steps. The initial pseudo-temperature and the drop of temperature of ASA might limit the adjusting of average path length to a very small range. The detailed description of ASA algorithm is shown in Algorithm 2.

Among the above method, adjusting the clustering coefficient or average path length of the whole network is computationally costly, and easy to fall into local extremum. In order to make use of local information of a network to reduce the computational cost, as well as help our algorithm escaping from local extreme, we propose an edge rewiring strategy(ERS) to adjust the clustering coefficient and average path length in a local region of a network. The proposed ERS method could provide a boarder adjusting range for clustering coefficient and average path length of the network under consideration.

3. Edge Rewiring Method

The main idea of our edge rewiring strategy (ERS) is to adjust the clustering coefficient and the average path length in a local region of a given network. The scheme of the proposed ERS method can be divided into two steps. In the first step, we randomly choose edge pairs and then rewire each pair to have different end nodes, provided that none of new edges already exist in the network. In the second step, we accept the edge pair with highest local efficiency function to execute edge exchange operation based on standard Monte Carlo simulation at zero temperatures to save calculating costs and escape from local extreme. One can adjust the clustering coefficient or average path length of a network to a user-specified value by running the edge rewiring strategy iteratively.

Algorithm 2 Simulated Annealing (ASA) for adjusting average path length

Input: Graph G, the desired value f' of average path length,
the maximum iteration $maxt = 100000$, threshold $\varepsilon = 0.0001$,
the initial systems pseudo-temperature, say Temp(Temp=10),
the annealing scheme, the pseudo-temperature decreased 10%
every 200 steps;

Output: Graph G' with the value of average path length approximately equal to f'.

Let t=0, $G^{(0)} = G$;
Calculate $f(G^{(t)})$, the value of average path length approximately of $G^{(0)}$;
Calculate $E(APL) = |f(G_t) - f^{'}|$
while $(|f(G^{(t)}) - f^{'}| \geq \varepsilon)$ and $(t < maxt)$ **do**
    $t \leftarrow t + 1$;
    Select randomly an edge pair $\langle x_1x_2, x_3x_4 \rangle$ from G satisfying:
        (i)$x_ix_j \notin E$ for $i \in \{1, 2\}$ and $j \in \{3, 4\}$;
        (ii) $N_G(x_i) \bigcap N_G(x_j) = \emptyset$ for $i, j \in \{1, 2, 3, 4\}$
    Let $G_t' = G \cup \{x_1x_3, x_2x_4\} - \{x_1x_2, x_3x_4\}$;
    Calculate $E(APL') = |f(G_t') - f'|$
    if $(E(APL') < E(APL))$ **then**
        $G_t = G_t'$;
    else
        Calculate $probability = e^{\frac{-(E(L')-E(L))}{Temp}}$
        if (Random number < probability) **then**
            $G_t = G_t'$;
        else
            $G_t = G_t' - \{x_1x_3, x_2x_4\} \cup \{x_1x_2, x_3x_4\}$;
        end if
    end if
    Reduce the system pseudo-temperature according to the annealing schedule
end while

An edge exchange operation on edge pair $\langle x_1x_2, x_3x_4\rangle$ is to rewire the edges between x_i's, that is to say, delete edges $\{x_1x_2, x_3x_4\}$ from G and add edges $\{x_1x_3, x_2x_4\}$ to G. Obviously, edge exchange operations keep the degree of every node unchanged. Hence, we can execute edge exchange operations to adjust the value of the concerning topological properties without changing the degree distribution of the network. If an edge exchange operation on edge pair $\langle x_1x_2, x_3x_4\rangle$ could change the value of the concerning topological property (here, we only take clustering coefficient or average path length in consideration) of the network to the desired direction, we call it an effective exchange; Otherwise, it is an ineffective exchange. To determine whether an edge exchange operation be effective, we should calculate the clustering coefficient (or average path length) of the network before and after the edge exchange operation. This would require a large number of calculations and it is easy to fall into local extremes, which might limit the adjusting range of the algorithm.

Calculating the local effectiveness of an edge change operation is a good choice for saving calculating costs and escapes from local extreme. Therefore, we construct efficiency function to show the local effectiveness of an edge exchange operation on clustering coefficient and average path length. We randomly select two parallel edges $x_1x_2 \in E$ and $x_3x_4 \in E$ from G, satisfying that $x_ix_j \notin E$ for $i = 1, 2$ and $j = 3, 4$. Then we accept the edge exchange operations based on standard Monte Carlo simulation at zero temperatures to reach a user-specified value by running edge rewiring strategy iteratively.

3.1. Local effectiveness of edge pairs on adjusting clustering coefficient

For a network G, let x_1x_2 and x_3x_4 be two edges of G. Let G' be the new graph obtain from G by executing the edge exchange operation on edge pair $\langle x_1x_2, x_3x_4\rangle$. We denote $C_G(x_i)$ be the clustering coefficient of node v_i in G and $C_{G'}(x_i)$ be the clustering coefficient of node v_i in G'. We use an efficiency function $LC(x_1x_2, x_3x_4)$ to estimate the local effectiveness of an edge pair $\langle x_1x_2, x_3x_4\rangle$ on adjusting the clustering coefficient of the network, defined as

$$LC(x_1x_2, x_3x_4) = \sum_{i=1}^{4} \big(C_G(x_i) - C_{G'}(x_i)\big). \tag{6}$$

3.2. Local effectiveness of edge pairs on adjusting average path length

For a network G, let x_1x_2 and x_3x_4 be two edges of G. Let G' be the new graph obtain from G by executing the edge exchange operation on edge pair $\langle x_1x_2, x_3x_4\rangle$. Let $N_1 = \bigcup_{i=1}^{4} N_G(x_i)$, then $N_1 = \bigcup_{i=1}^{4} N_{G'}(x_i)$. We consider the variation of shortest path length between nodes in N_1 after executing edge exchange operation on $\langle x_1x_2, x_3x_4\rangle$. We use a efficiency function $LP(x_1x_2, x_3x_4)$ to estimate the local effectiveness of an edge pair $\langle x_1x_2, x_3x_4\rangle$ on adjusting the average path length of the network, defined as

$$LP(x_1x_2, x_3x_4) = \sum_{i\neq j, v_i, v_j \in N_1} \big(l_G(v_i, v_j) - l_{G'}(v_i, v_j)\big). \tag{7}$$

3.3. Adjusting strategy for retaining community structure

For a network G with community structure, it is always expected that adjusting the clustering efficient (or average path length) of G as much as possible without changing the original community structures of G. It turns out the clustering coefficient or the average path length increases with the community structure strength [37].

Let G' be another network having identical degree sequence and community structure with G by executing a sequence of edge exchange operations from G. If we alter some edges between communities to edges within communities, then the clustering coefficient (or the average path length) of G' is more likely greater than that of G. If we alter some edges within communities to edges between communities, then the clustering coefficient (or the average path length) of G' is more likely smaller than that of G.

Based on the above analysis, to keep the community structure of a network unchanged as much as possible after edge exchange operations, the probability that an edge selected between communities should be greater than an edge selected within communities when increasing the clustering coefficient (or the average path length), whileas the probability that an edge selected between communities should be smaller than an edge selected within communities when decreasing the clustering coefficient (or the average path length).

However, an increased (a decreased) local effectiveness of an edge pair on adjusting clustering coefficient or the average path length might not indicate the edge pair lies in between (within) communities, owing to that an increase (a decrease) of local effectiveness might not lead to an increase (a decrease) of the corresponding topological property value of the whole network. If we execute edge exchange operation once the change of local effectiveness of an edge pair coinciding to the desired direction, then the probability of an edge between communities selected might be approximately equal to the probability of an edge within communities selected after frequent edge exchange operations. Thus, the community structure of the network might be weaken after frequent edge exchange operations. Moreover, the convergence speed of adjusting might be limited. Thereby, the adjusting range of the clustering coefficient and the average path length might be further affected.

To retain the community structure as much as possible, we randomly select multiple parallel edge pairs from G at one time, such that each edge pair $\langle x_1x_2, x_3x_4\rangle$ satisfies $x_ix_j \notin E$ for $i \in \{1,2\}$ and $j \in \{3,4\}$ and nodes involved in the edge pair are mutually distinct. Then we choose the edge pair with maximum (or minimum) value of local effectiveness to execute edge exchange operation on the edge pair. We select ten parallel edge pairs from G at one time in following experiments.

Algorithm 3 gives the edge rewiring strategy ERS for adjusting clustering coefficient or average path length described above. We use "TP" to represent the corresponding topological parameter, which depends on actual situation.

We provide an illustration example to explain why the proposed edge rewiring strategy can retain the community structure. As is shown in Figure 1, the clustering coefficient is increased from 0.1849 to 0.3407 by our edge rewiring strategy, community structure has not changed too much. In our edge rewiring strategy, we select ten parallel edge pairs from G at one time, then choose the edge pair

Algorithm 3 Edge rewiring strategy for adjusting TP

Input: Graph G, the desired value f' of TP,
the maximum iteration $maxt = 100000$, threshold $\varepsilon = 0.0001$;

Output: Graph G' with the value of TP approximately equal to f'.

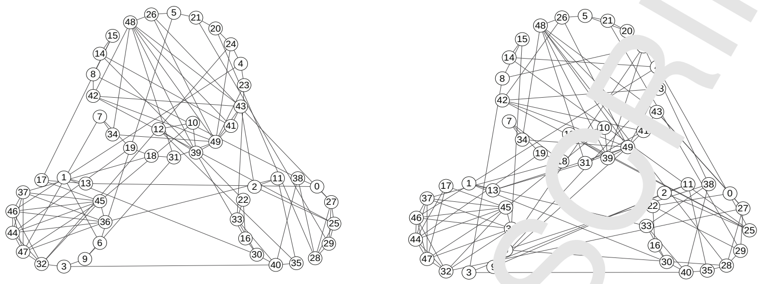
```
Let t=0, G^(0) = G;
Calculate f(G^(t)), the value of TP of G^(0);
while (|f(G^(t)) - f'| >= ε) and (t < maxt) do
    t <- t + 1;
    for iter=1 to 10 do
        Select randomly an edge pair ⟨x1x2, x3x4⟩ from G satisfying xi xj ∉ E
        for i ∈ {1, 2} and j ∈ {3, 4};
        Let G'_iter = G ∪ {x1x3, x2x4} - {x1x2, x3x4};
    end for
    Let G^t = G'_m, where G'_m = arg max{f(G'_y) | y = 1, 2, ..., 10};
end while
```

with maximum value of local effectiveness to execute edge exchange operation on the edge pair. Therefore, the probability that an edge selected between communities should be greater than an edge selected within communities when improving the clustering coefficient. In the process, we can adjust clustering coefficient faster and keep the community structure of a network unchanged as much as possible after edge exchange operations. When the original network has the ground-truth community structure information, our edge rewiring strategy would keep the original community structure as much as possible.

3.4. Complexity analysis

It takes $O(N*(\overline{k})^2)$ to compute the clustering coefficient and $O(N*(N+M))$ to compute the average path length of the whole network after each edge rewiring operation in average for a network with N nodes and M edges. Let $\overline{k}$ be the average degree of the network.

Let t_{KMC} be the iteration number for adjusting clustering coefficient using



(a) (b)

Figure 1: The dynamics of an example network using ERS strategy

KMC algorithm, then the time complexity of KMC algorithm takes is $O(t_{KMC} * N * \overline{k}^2)$. Let t'_{ASA} be the iteration number for adjusting average path length using ASA algorithm, then the average time complexity of ASA algorithm takes is $O(t'_{ASA} * N * (N + M)) = O(t'_{ASA} * \overline{k} * N^2)$.

In ERS algorithm, we calculate the local effectiveness of an edge change operation to determine whether an edge exchange operation should be accepted instead of global characteristic values. The average time complexity of calculating the clustering coefficient of local structure is $O(4 * \overline{k}^2)$, and the time complexity of calculating the average path length of local structure is $O(4\overline{k} * (4\overline{k} + 2\overline{k} * \overline{k})) = O(8\overline{k}^3)$. Let t_{ERS} and t'_{ERS} be the iteration number for adjusting clustering coefficient and average path length by ERS algorithm, respectively. Then ERS algorithm with take $O(4 * t_{ERS} * \overline{k}^2)$ and $O(8 * t'_{ERS} * \overline{k}^3)$ for adjusting clustering coefficient and average path length by ERS algorithm, respectively.

We have $t_{ERS} < t_{KMC} *$ and $t'_{ERS} < t'_{ASA}$, since our local strategy could help algorithm escaping from local extreme efficiently to save calculating costs. For a large network, we always have $\overline{k} << N$, hence we have

$$4 * t_{ERS} * \overline{k}^2 << t_{KMC} * N * \overline{k}^2;$$

$$8 * t'_{ERS} * \overline{k}^3 << t'_{ASA} * \overline{k} * N^2.$$

Hence, our ERS could provide border adjustment range of clustering coefficient and average path length in reasonable computing time. Table 2 shows the algorithm complexity of KMC, ASA and ERS methods.

Table 2: Summary of complexity analysis

Method	Adjust C	Adjust APL
ERS	$O(4 * t_{ERS} * \overline{k}^2)$	$O(3 * t_{ERS} * \overline{k}^3)$
KMC	$O(t_{KMC} * N * \overline{k}^2)$	–
ASA	–	$O(t'_{ASA} * \overline{k} * N^2)$

* The number of nodes of network N, the average degree of network $\overline{k}$. t_{ERS} and t'_{ERS} are the iteration number for adjusting clustering coefficient and average path length by ERS algorithm. t_{KMC} is the iteration number for adjusting clustering coefficient using KMC algorithm. t'_{ASA} is the iteration number for adjusting average path length using ASA algorithm.

4. Experimentation

4.1. Artificial network models

We construct two artificial network models with community structure to validate the feasibility and reliability of the proposed edge rewiring strategy ERS. Note that real networks are characterized by heterogeneous distributions of node degree and community sizes, i.e., the tails of the distributions of the networks' node degree and community sizes can be fairly well approximated by a power law. When constructing artificial network models, we should take into account the heterogeneous distributions in networks to mimic the real-world networks. We generate artificial networks whose degree sequence and community size sequence which obey scale free distribution.

We firstly generate a reference degree sequence $\mathbf{D} = \{d_1, \ldots, d_N\}$ with probability $p(d_i)$ according to the power law distribution given by (8),

$$p(d_i) = \frac{\alpha - 1}{\delta(G)} \left(\frac{d_i}{\delta(G)} \right)^{-\alpha}. \tag{8}$$

where $d_1 \geq d_2 \ldots \geq d_N$, $\Delta(G) \geq d_i \geq \delta(G)$, α is the power exponent and $\alpha = 3$ in the following generation process. We might trim slightly some items of the generated sequence to make $\mathbf{D}$ graphical.

Similarly, we generate community size sequence $\{s_1, \ldots, s_c\}$ obeys power law distribution, where c is the predefined number of communities, $s_1 \leq s_2 \ldots \leq s_c$ and $\sum_{k=1}^{c} s_i = N$.

We use mixing parameter μ, the ratio between the external degree of a node with respect to its community and the total degree of the node, to control the level of community structures in our network model. A smaller μ results in networks with higher level of community structures.

For node v_i, we define its internal degree and external degree according to the mixing parameter μ as (9):

$$\begin{aligned} d^{in}(v_i) &= \lfloor d(v_i) \times (1-\mu) + 0.5 \rfloor, \\ d^{ex}(v_i) &= d(v_i) - d^{in}(v_i), \end{aligned} \tag{9}$$

where $d^{in}(v_i)$ denotes the number of adjacent nodes that lie in the same community as v_i, and $d^{ex}(v_i)$ denotes the number of adjacent nodes that lie in the different community from v_i.

Each node would be prior to be distributed the communities with smaller size, provided that the internal degree of the node is no larger than the size of the community. Edges connecting different communities are linked randomly according to the external degree sequence $\mathbf{D^{ex}} = \{d^{ex}(v_1), d^{ex}(v_2), \ldots, d^{ex}(v_N)\}$.

We add edges within communities by two different generation models: Havel-Hikimi model[38] to generate communities with high assortative mixing, which is defined as "a preference for high-degree nodes to attach to other high-degree nodes"; and configuration model[39] to generate communities with random links. See details in Section 4.2 and 4.3.

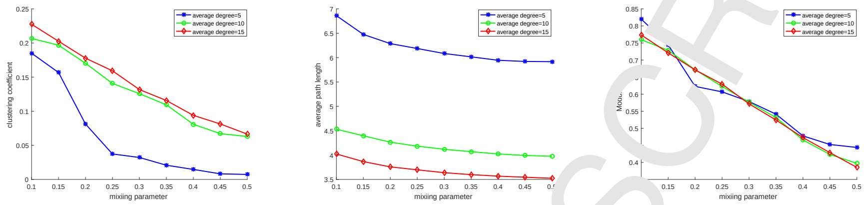
Our experimentation has been conducted on two computer-generated networks to prove the validity and effectiveness of the proposed method. We compare the performance of the proposed edge rewiring strategy to Monte Carlo methods given by Kim and Simulated Annealing method proposed by Andreas et al., respectively. All the experiments have been carried on Windows 10 platform, running on a PC with Intel Core CPU i7-2600@3.40 GHz and 8 GB RAM. The programming language is Java.

4.2. Effects of edge rewiring strategy on HH network model

Havel-Hakimi algorithm[38] can be used to generate a network from a graphical degree sequence. The networks generated by Havel-Hikimi algorithm are always assortative, that is to say, there is a preference for high-degree nodes to attach to other high-degree nodes. In generation our Havel-Hikimi network model, we construct edges within communities by Havel-Hikimi algorithm according to the inter-degree sequence of G, $\mathbf{D_k^{in}} = \{d^{in}(v_i)|\tau(v_i) = k\}$, $1 \leq k \leq c$. We also call it a HH network model for abbreviation.

The topological properties of HH network model are dependent on the average degree and the mixing parameter of the network. The average degree reflects the density of a network. The average degree increment of the network will cause general increasing clustering coefficient and general decreasing average path length, and vice versa. However, the community structure of a network has no relation with the average degree, but has a significant relation with the mixing parameter μ. Along with the increasing mixing parameter, the modularity of the network will decrease, which means the community structure of the network is disappearing gradually. Figure 2 shows the changes of topological properties (including clustering coefficient, average path length and modularity) which are associated with the change of the average degree and the mixing parameter μ in the generated HH networks. In all experiments, we set node number $N = 5000$, the number of communities $c = 10$, power-law exponent of degree distribution $\alpha = 3$, power-law exponent of community size distribution $\gamma = 2$.

From Figure 2, we can also observe that the mixing parameter increment of the network will cause general decreasing clustering coefficient and average path length. This is as expected because a higher value of mixing parameter will lead to formation of edges to long distance neighbours which reduces the global path length and decreases the chances of triads in networks.



(a) Impact of mixing parameter (μ) and average degree ($\overline{k}$) on clustering coefficient

(b) Impact of mixing parameter (μ) and average degree ($\overline{k}$) on average path length

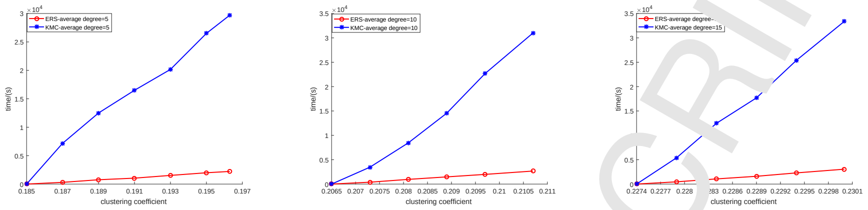
(c) Impact of mixing parameter (μ) and average degree ($\overline{k}$) on modularity

Figure 2: Structural properties of networks generated by HH model

4.2.1. Adjusting clustering coefficient in HH network model

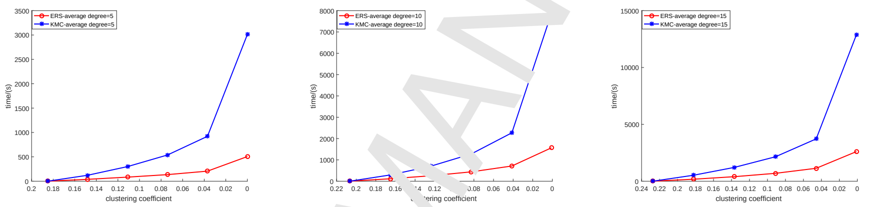
In this section, we adjust clustering coefficient by edge rewiring strategy ERS. Through several iterations, we can adjust the clustering coefficient of the network effectively without changing the degree distribution. We compare the performance of our edge rewiring strategy on adjusting clustering coefficient with that of Kim's Monte Carlo method(KMC). In each iteration, KMC method selects a pair of parallel edges to execute edge rewiring as long as the edge exchange would change the global clustering coefficient to the desired direction. In our method, we select a pair of parallel edges in each iteration to calculate its local efficiency on local clustering coefficient according to (6). We choose the edge pair with the highest local efficiency to execute edge rewiring every ten iterations.

Figures 3-4 show the adjusting performance on clustering coefficient in networks with average degree varying from 5 to 15 with $N = 5000$, $\mu = 0.1$ and $c = 10$, where N is node number, μ is mixing parameter, and c is community number. The red lines correspond to the results of our method and the blue lines to those of KMC method. For average degree $\bar{k} = 5, 10$ and 15, the clustering coefficient of the initial HH network equals to 0.1850, 0.2065 and 0.2274, respectively.



(a) $\overline{k} = 5$ (b) $\overline{k} = 10$ (c) $\overline{k} = 15$

Figure 3: Comparison results of ERS and KMC in time on HH model when increasing clustering coefficient under different average degree



(a) $\overline{k} = 5$ (b) $\overline{k} = 10$ (c) $\overline{k} = 15$

Figure 4: Comparison results of ERS and KMC in time on HH model when decreasing clustering coefficient under different average degree

For average degree $\bar{k} = 5$, the KMC method increased the clustering coefficient of the HH network from 0.1850 to 0.1963 after 100,000 iterations in 29,995.50 seconds, the proposed ERS method increased the clustering coefficient from 0.1850 to 0.1963 after 100,000 iterations in 2,542.41 seconds. For average degree $\bar{k} = 10$, the KMC method increased the clustering coefficient of the HH network from 0.2065 to 0.2107 after 100,000 iterations in 31,455.78 seconds. The proposed ERS method increased the clustering coefficient from 0.2065 to 0.2107 after 100,000 iterations in 2,706.00 seconds. For average degree $\bar{k} = 15$, the KMC method increased the clustering coefficient of the HH network from 0.2274 to 0.2300 after 100,000 iterations in 33,922.76 seconds, the proposed ERS method increased the clustering coefficient from 0.2274 to 0.2297 after 100,000 iterations in 2,741.07 seconds.

For average degree $\bar{k} = 5$, the KMC method decreased the clustering coefficient of the HH network from 0.1850 to almost zero after 11,000 iterations in 3,012.42 seconds; the proposed ERS method can decrease the clustering coefficient from 0.1850 to almost zero after 20,000 iterations in 516.36 seconds. For average degree $\bar{k} = 10$, the KMC method decreased the clustering coefficient of the HH network from 0.2065 to almost zero after 25,000 iterations in 7,873.55 seconds and the proposed ERS method decrease the clustering coefficient from 0.2065 to almost zero after 59,000 in 1,567.17 seconds. For average degree $\bar{k} = 15$, the KMC method decreased the clustering coefficient of the HH network from 0.2274 to almost zero after 37,000 iterations in 12,892.20 seconds and the proposed ERS method decrease the clustering coefficient from 0.2274 to almost zero in 2605.56 seconds.

In Table 3, we show the comparisons on time consuming and effect on community structures from the initial clustering coefficient up to an given value, and in Table 4 we show comparison results from initial clustering coefficient down to an given value. It can be concluded that the proposed ERS method can increase or decrease the clustering coefficient of the HH network at a faster rate. What more, our method retains community structures well.

Table 3: Comparison results in increasing clustering coefficient for HH Networks

$N = 5000, \bar{k} = 5$					$N = 5000, \bar{k} = 10$					$N = 5000, \bar{k} = 15$				
C_G	ERS		KMC		C_G	ERS		KMC		C_G	ERS		KMC	
	time(s)	m	time(s)	m		time(s)	m	time(s)	m		time(s)	m	time(s)	m
0.1850	0	0.8202	0	0.8202	0.2065	0	0.7598	0	0.7598	0.2274	0	0.7735	0	0.7735
0.1940	1814	0.8115	[illegible]	0.8115	0.2095	1877	0.7580	15816	0.7578	0.2294	2293	0.7729	25394	0.7731
0.2030	3884	0.8003	[illegible]	0.7995	0.2125	4051	0.7546	46445	0.7549	0.2314	4570	0.7719	54425	0.7729
0.2120	6459	[illegible]	70821	0.7912	0.2155	6090	0.7518	69706	0.7519	0.2334	7226	0.7715	81266	0.7728
0.2210	925[illegible]	0.7742	—	—	0.2185	8437	0.7494	—	—	0.2354	10127	0.7714	—	—
0.2341	[illegible]	[illegible]	—	—	0.2243	13422	0.7451	—	—	0.2379	13473	0.7711	—	—

* Total number of nodes N; average degree $\overline{k}$; clustering coefficient C_G; Run time in seconds and modularity m; "-" means that the corresponding clustering coefficient can not be achieved within a reasonable period of time.

4.2.2. Adjusting average path length in HH network model

In this section, we adjust average path length by edge rewiring strategy ERS. Through several iterations, we can adjust the average path length of the network effectively without changing the degree distribution. We compare the

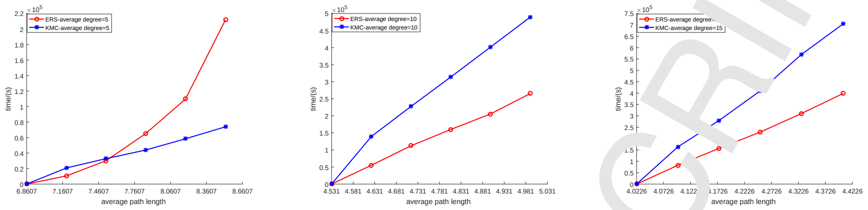
Table 4: Comparison results in decreasing clustering coefficient for HH Networks

$N = 5000, \bar{k} = 5$					$N = 5000, \bar{k} = 10$					$N = 5000, \bar{k} = 15$				
C_G	ERS		KMC		C_G	ERS		KMC		C_G	ERS		KMC	
	time(s)	m	time(s)	m		time(s)	m	time(s)	m		time(s)	m	time(s)	m
0.1850	0	0.8202	0	0.8202	0.2065	0	0.7598	0	0.7598	0.2274	0	[illegible]	0	0.7735
0.1480	38	0.7994	122	0.7766	0.1652	114	0.7325	293	0.7108	[illegible]	176	0.7448	528	0.7182
0.1110	84	0.7740	300	0.7276	0.1239	255	0.7002	706	0.6498	[illegible]	398	0.7089	1218	0.6530
0.0740	135	0.7453	538	0.6739	0.0826	439	0.6573	1281	0.5774	[illegible]	692	0.6615	2146	0.5758
0.0370	205	0.7069	897	0.6139	0.0413	712	0.5958	2270	[illegible]	0.0458	[illegible]	0.5923	3721	0.4723
0	506	0.5864	3012	0.5395	0	1565	0.4470	7873	[illegible]	0	2605	0.4141	12892	0.3003

* Total number of nodes N; average degree $\overline{k}$; clustering coefficient C_G; Run time in seconds and modularity m.

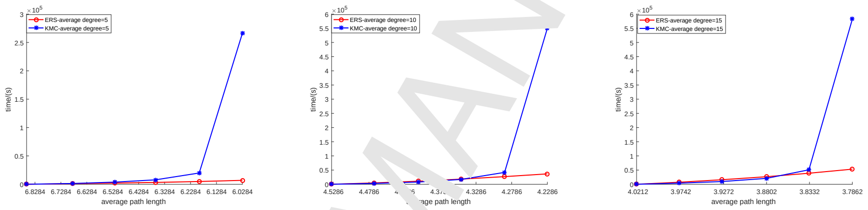
performance of our edge rewiring strategy on adjusting average path length with that of Andreas's Simulated Annealing method (ASA). In each iteration, ASA method selects a pair of parallel edges to execute edge rewiring as long as the edge exchange would change the global average path length to the desired direction. In our method, we select a pair of parallel edges in each iteration to calculate its local efficiency on local average path length according to (7). We choose the edge pair with the highest local efficiency to execute edge rewiring every ten iterations.

Figures 5-6 show the adjusting performance on average path length in networks with average degree varying from 5 to 15 with $N = 5000$, $\mu = 0.1$ and $c = 10$, where N is node number, μ is mixing parameter, and c is community number. The red lines corresponds to the results of our method and the blue lines to those of ASA method. For average degree $\bar{k} = 5, 10$ and 15, the average path length of the initial HH network equals to 6.8607, 4.5310 and 4.0226, respectively.



(a) $\overline{k} = 5$ (b) $\overline{k} = 10$ (c) $\overline{k} = 15$

Figure 5: Comparison results of ERS and ASA in time on HH model when increasing average path length under different average degree



(a) $\overline{k} = 5$ (b) $\overline{k} = 10$ (c) $\overline{k} = 15$

Figure 6: Comparison results of ERS and ASA in time on HH model when decreasing average path length under different average degree

For average degree $\bar{k} = 5$, the ASA method increased the average path length of the HH network from 6.8507 to 8.5197 after 100,000 iterations in 211,961 seconds, the proposed ASA method increased the average path length from 6.8507 to 8.5197 in 73,893 seconds. For average degree $\bar{k} = 10$, the ASA method increased the average path length of the HH network from 4.5310 to 4.9915 after 100,000 iterations in 490,077 seconds, the proposed ERS method increased the average path length from 4.5310 to 5.2331 after 100,000 iterations in 4[illegible]8,220 seconds. For average degree $\bar{k} = 15$, the ASA method increased the average path length of the HH network from 4.0226 to 4.4050 after 100,000 iterations in 704,894 seconds, the proposed ERS method increased the average path length from 4.0226 to 4.5775 after 100,000 iterations in 625,442 seconds.

For average degree $\bar{k} = 5$, the ASA method decreased the average path length of the HH network from 6.8602 to 6.0278 after 100,000 iterations in

272,321 seconds; the proposed ERS method can decrease the average path length from 6.8602 to 5.2800 after 100,000 iterations in 174,739 seconds. For average degree $\bar{k} = 10$, the ASA method decreased the average path length of the HH network from 4.5310 to 4.2282 after 100,000 iterations in 550,742 seconds and the proposed ERS method decrease the average path length from 4.5310 to 3.8065 after 100,000 in 378,879 seconds. For average degree $\bar{k} = 15$, the ASA method decreased the average path length of the HH network from 4.0226 to 3.7865 after 100,000 iterations in 588,255 seconds and the proposed ERS method decrease the average path length from 4.0226 to 3.4691 in 505,043 seconds.

In Table 5, we show the comparisons on time consuming and effect on community structures from the initial average path length up to an given value, and in Table 6 we show comparison results from initial average path length down to an given value. It can be concluded that the proposed ERS method can increase or decrease the average path length of the HH network at a faster rate. What more, our method retains community structures well.

Table 5: Comparison results in increasing average path length for HH Networks

$N = 5000, \bar{k} = 5$					$N = 5000, \bar{k} = 10$					$N = 5000, \bar{k} = 15$				
APL	ERS		ASA		APL	ERS		ASA		APL	ERS		ASA	
	time(s)	m	time(s)	m		time(s)	m	time(s)	m		time(s)	m	time(s)	m
6.8607	0	0.8203	0	0.8203	4.5310	0	0.7598	0	0.7598	4.0226	0	0.7735	0	0.7735
7.1946	10614	0.8280	[illegible]	[illegible]	[illegible]	88237	0.7906	189703	0.7486	4.1338	116092	0.7927	215998	0.7789
7.5233	30126	0.8355	[illegible]	0.5533	4.8204	167338	0.8105	324392	0.7592	4.2444	221877	0.8132	398953	0.7871
7.8545	65154	[illegible]	[illegible]	[illegible]	4.9645	247774	0.8232	462915	0.7659	4.3556	338066	0.8248	624121	0.7932
8.1862	109701	0.8464	58417	[illegible]	5.1093	349472	0.8334	—	—	4.4663	458903	0.8332	—	—
8.5197	211961	[illegible]	[illegible]	0.4970	5.2331	468220	0.8396	—	—	4.5775	625442	0.8395	—	—

* Total number of nodes N; average degree $\overline{k}$; average path length APL; Run time in seconds and modularity m; "-" means that the corresponding average path length can not be achieved within a reasonable period of time.

4.3. Effects of edge rewiring strategy on RL network model

The configuration model[39] describes a way to construct an undirected graph on N nodes. For each node generates a degree independently from a random variable with distribution F and creates "stubs". Pick two "stub" randomly among all "stubs" in the graph and join them. Obviously, there may be self-loops and multiple edges in the construction process. Here, we avoid the

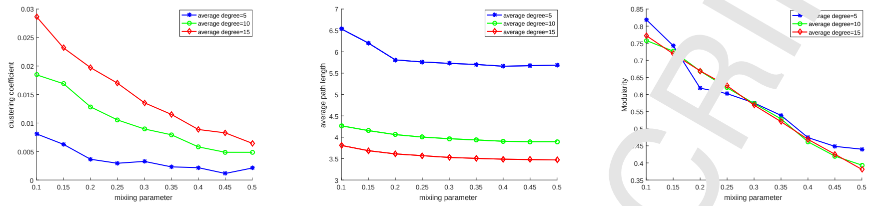
Table 6: Comparison results in decreasing average path length for H[illegible] Networks

$N = 5000, \bar{k} = 5$					$N = 5000, \bar{k} = 10$					$N = 5000, \bar{k} = 15$				
APL	ERS		ASA		APL	ERS		ASA		APL	ERS		ASA	
	time(s)	m	time(s)	m		time(s)	m	time(s)	m		time(s)	m	time(s)	m
6.8607	0	0.8203	0	0.8203	4.5310	0	0.7598	0	0.7598	4.0226	0	[illegible]	0	0.7735
6.5457	1970	0.7996	3403	0.6884	4.3869	13248	0.7338	10518	0.7049	[illegible]	22290	0.7469	15441	0.7344
6.2266	4513	0.7713	16185	0.5562	4.2416	34424	0.6941	162436	0.6358	[illegible]	56144	0.7085	—	—
5.9136	8549	0.7292	—	—	4.0969	64727	0.6348	—	—	[illegible]	105910	0.6519	—	—
5.5968	16134	0.6559	—	—	3.9513	115552	0.5376	—	—	[illegible]	[illegible]	0.5509	—	—
5.2800	174739	0.1952	—	—	3.8065	378879	0.2534	—	—	[illegible]	[illegible]	0.3233	—	—

* Total number of nodes N; average degree $\overline{k}$; average path length APL; Run time in seconds and modularity m; "-" means that the corresponding average path length can not be achieved within a reasonable period of time.

multiple edges and self-loop by modifying the degree of nodes in the construction process. In generation our random network model, we construct edges within communities by configuration model according to the internal degree sequence of G $\mathbf{D}_{\mathbf{k}}^{\mathbf{in}} = \{d^{in}(v_i)|\tau(v_i) = k\}$, $1 \leq k \leq c$. We also call it a RL network model for abbreviation.

Analogously, the topological properties of RL network model are dependent on the average degree and the mixing parameter of the network. Figure 7 shows the changes of topological properties (including clustering coefficient, average path length and modularity) which are associated with the change of the average degree and the mixing parameter μ in the generated RL networks. In all experiments, we set node number $N = 5000$, the number of communities $c = 10$, power-law exponent of degree distribution $\alpha = 3$, power-law exponent of community size distribution $\beta = 2$. We can also observe that the mixing parameter increment of the network will cause general decreasing clustering coefficient and average path length.



(a) Impact of mixing parameter (μ) and average degree ($\overline{k}$) on clustering coefficient

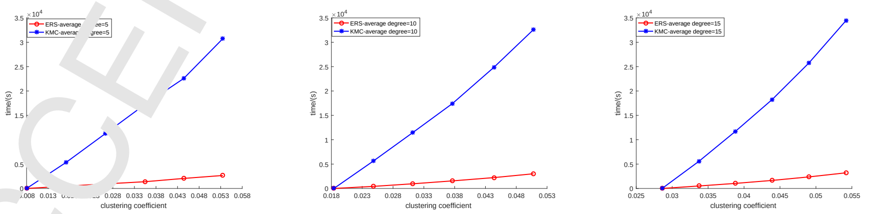
(b) Impact of mixing parameter (μ) and average degree ($\overline{k}$) on average path length

(c) Impact of mixing parameter (μ) and average degree ($\overline{k}$) on modularity

Figure 7: Structural properties of networks generated by RL model

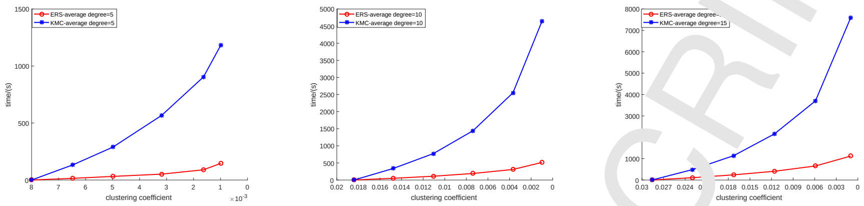
4.3.1. Adjusting clustering coefficient in RL network model

In this section, we adjust clustering coefficient by edge rewiring strategy ERS. Through several iterations, we can adjust the clustering coefficient of the network effectively without changing the degree distribution. We compare the performance of our edge rewiring strategy on adjusting clustering coefficient with that of Kim's Monte Carlo method (KMC). Figures 8-9 shows the adjusting performance on clustering coefficient in networks with average degree varying from 5 to 15 with $N = 5000$, $\mu = 0.1$ and $c = 10$, where N is node number, μ is mixing parameter, and c is community number. The red lines correspond to the results of our method and the blue lines to those of KMC method. For average degree $\overline{k} = 5, 10$ and 15, the clustering coefficient of the initial RL network equals to 0.0080, 0.0184 and 0.0286, respectively.



(a) $\overline{k} = 5$ (b) $\overline{k} = 10$ (c) $\overline{k} = 15$

Figure 8: Comparison results of ERS and KMC in time on RL model when increasing clustering coefficient under different average degree



(a) $\bar{k} = 5$ (b) $\bar{k} = 10$ (c) $\bar{k} = 15$

Figure 9: Comparison results of ERS and KMC in time in RL model when decreasing clustering coefficient under different average degree

For average degree $\bar{k} = 5$, the KMC method increased the clustering coefficient of the RL network from 0.0080 to 0.0535 after 100,000 iterations in 30,768 seconds, the proposed ERS method increased the clustering coefficient from 0.0080 to 0.0538 after 100,000 iterations in 2,699 seconds. For average degree $\bar{k} = 10$, the KMC method increased the clustering coefficient of the RL network from 0.0184 to 0.05[illegible] after 100,000 iterations in 32,736 seconds, the proposed ERS method increased the clustering coefficient from 0.0184 to maximum 0.0487 after 100,000 iterations in 2,739 seconds. For average degree $\bar{k} = 15$, the KMC method increased the clustering coefficient of the RL network from 0.0286 to 0.054[illegible] after 100,000 iterations in 34,504 seconds, the proposed ERS method increased the clustering coefficient from 00.0286 to 0.0513 after 100,000 iterations in 2,7[illegible]4 seconds.

For average degree $\bar{k} = 5$, the KMC method decreased the clustering coefficient of the RL network from 0.0080 to almost zero after 3,889 iterations in 1,182 seconds; the proposed ERS method can decrease the clustering coefficient from 0.0080 to almost zero after 10,350 iterations in 272 seconds. For average degree $\bar{k} = 10$, the KMC method decreased the clustering coefficient of the RL network from 0.0184 to almost zero after 13,787 iterations in 4,648 seconds and the proposed ERS method decrease the clustering coefficient from 0.0184 to almost zero after 19,300 iterations in 514 seconds. For average degree $\bar{k} = 15$, the KMC method decreased the clustering coefficient of the RL network from 0.0286 to almost zero after 21,683 iterations in 7,599 seconds and the proposed

ERS method decrease the clustering coefficient from 0.0284 to almost zero after 40,250 iterations in 1,121 seconds.

In Table 7, we show the comparisons on time consuming and effect on community structures from the initial clustering coefficient up to an given value, and in Table 8 we show comparison results from initial clustering coefficient down to an given value. It can be concluded that the proposed ERS method can increase or decrease the cluster coefficient of the RL network at a faster rate. What more, our method retains community structures well.

Table 7: Comparison results in increasing clustering coefficient for RL Networks

$N = 5000, \bar{k} = 5$					$N = 5000, \bar{k} = 10$					$N = 5000, \bar{k} = 15$				
C_G	ERS		KMC		C_G	ERS		KMC		C_G	ERS		KMC	
	time(s)	m	time(s)	m		time(s)	m	time(s)	m		time(s)	m	time(s)	m
0.0081	0	0.8192	0	0.8192	0.0184	0	0.7581	0	0.7581	0.0286	0	0.7724	0	0.7724
0.0348	1353	0.8008	17070	0.7935	0.0347	1276	[illegible]	14664	0.7313	0.0410	1269	0.7513	14150	0.7502
0.0615	3485	0.7702	38284	0.7707	0.0510	[illegible]	0.7082	32927	0.7013	0.0534	3089	0.7256	33082	0.7204
0.0882	5993	0.7359	—	—	0.0673	5435	[illegible]	56611	0.6678	0.0658	5488	0.6967	58517	0.6973
0.1149	9290	0.6992	—	—	0.0836	8789	0.6413	—	—	0.0782	8954	0.6619	—	—
0.1416	13569	0.6643	—	—	[illegible]	[illegible]	[illegible]	—	—	0.0905	13749	0.6257	—	—

* Total number of nodes N; average degree $\overline{k}$; clustering coefficient C_G; Run time in seconds and modularity m. '-' means that the corresponding clustering coefficient can not be achieved within a reasonable period of time.

Table 8: Comparison results in decreasing clustering coefficient for RL Networks

$N = 5000, \bar{k} = 5$					$N = 5000, \bar{k} = 10$					$N = 5000, \bar{k} = 15$				
C_G	ERS		KMC		C_G	ERS		KMC		C_G	ERS		KMC	
	time(s)	m	time(s)	m		time(s)	m	time(s)	m		time(s)	m	time(s)	m
0.0081	0	[illegible]	0	0.8192	0.0184	0	0.7581	0	0.7581	0.0286	0	0.7724	0	0.7724
0.0065	13	[illegible]	[illegible]	0.8162	0.0148	49	0.7469	326	0.7417	0.0229	108	0.7549	486	0.7451
0.0049	31	[illegible]	288	0.8129	0.0112	107	0.7353	753	0.7249	0.0172	241	0.7332	1138	0.7147
0.0033	50	0.8117	[illegible]	0.8117	0.0076	182	0.7204	1362	0.7072	0.0115	408	0.7059	2156	0.6793
0.0017	[illegible]	[illegible]	855	0.8088	0.0040	259	0.7068	1362	0.7072	0.0058	656	0.6698	3700	0.6424
0	[illegible]	[illegible]	1183	0.8071	0	514	0.6812	4648	0.6744	0	1119	0.6278	7599	0.6092

* Total number of nodes N; average degree $\overline{k}$; clustering coefficient C_G; Run time in seconds and modularity m.

4.3.2. Adjusting average path length in RL network model

In this section, we adjust average path length by edge rewiring strategy ERS. Through several iterations, we can adjust the average path length of the network effectively without changing the degree distribution. We compare the performance of our edge rewiring strategy on adjusting average path length with that of Andreas's Simulated Annealing method(ASA). In each iteration, ASA method selects a pair of parallel edges to execute edge rewiring as long as the edge exchange would change the global average path length to the desired direction. In our method, we selects a pair of parallel edges in each iteration to calculate its local efficiency on local average path length according to (7). We choose the edge pair with the highest local efficiency to execute edge rewiring every ten iterations.

In Table 9, we show the comparisons on time consuming and effect on community structures from the initial average path length up to an given value, and in Table 10 we show comparison results from initial average path length down to an given value. However, there is one thing when improving average path length by ASA method. Average path length will appear to decline and then rise. The reason is that the ASA method needs to set the initial temperature and the drop rate of temperature, and calculate the accept probability. When the parameter setting is not reasonable, it will have a greater probability to accept the opposite situation. As the number of iterations increases, the temperature decreases gradually, producing a smaller probability of accepting the opposite. Therefore, adjust process of improving average path length appear first decline and then rise. In Table 9, we find that average path length does not increase after 100,000 iterations in ASA method. However, ASA method decreasing average path length more faster than our ERS method in Table 10. The reasons may be related to the different structures of the network.

5. Conclusion

In this paper, we propose a local structure based edge rewiring strategy to adjust the clustering coefficient and average path length of a network. The adjustment of one pair of edges has a larger probability to affect the local clustering coefficient or local average path length, which might help the algorithm escape from local extreme and reduce the computational cost. Therefore, our edge

Table 9: Comparison results in increasing average path length for RL Networks

$N = 5000, \bar{k} = 5$					$N = 5000, \bar{k} = 10$					$N = 5000, \bar{k} = 15$				
APL	ERS		ASA		APL	ERS		ASA		APL	ERS		ASA	
	time(s)	m	time(s)	m		time(s)	m	time(s)	m		time(s)	m	time(s)	m
6.5352	0	0.8192	0	0.8192	4.2682	0	0.7581	0	0.7581	3.8080	0	0.7724	0	0.7724
6.8642	6969	0.8259	122270	0.4048	4.4407	93836	0.7931	—	—	[illegible]	[illegible]	0.7956	—	—
7.1934	19299	0.8332	162491	0.4048	4.4631	185925	0.8157	—	—	[illegible]	229563	[illegible]	—	—
7.5226	35938	0.8400	203814	0.3778	4.7863	289069	0.8306	—	—	[illegible]	338066	[illegible]	—	—
7.8518	71349	0.8441	242440	0.3516	4.9582	395133	0.8406	—	—	[illegible]	[illegible]	0.8375	—	—
8.3849	154465	0.8503	—	—	5.1310	526487	0.8473	—	—	[illegible]	[illegible]	0.8403	—	—

* Total number of nodes N; average degree $\overline{k}$; average path length APL; Run time in seconds and modularity m; - means that the corresponding clustering coefficient can not be achieved within a reasonable period of time.

Table 10: Comparison results in decreasing average path length for RL Networks

$N = 5000, \bar{k} = 5$					$N = 5000, \bar{k} = 10$					$N = 5000, \bar{k} = 15$				
APL	ERS		ASA		APL	ERS		ASA		APL	ERS		ASA	
	time(s)	m	time(s)	m		time(s)	m	time(s)	m		time(s)	m	time(s)	m
6.5352	0	0.8192	0	0.8192	4.2682	0	0.7581	0	0.7581	3.8080	0	0.7724	0	0.7724
6.2900	2783	0.7973	796	0.7789	4.1786	[illegible]	[illegible]	2691	0.7273	3.7252	23454	0.7516	4950	0.7425
6.0524	6609	0.7669	1883	0.7273	4.0875	2793	[illegible]	6629	0.6848	3.6419	62720	0.7153	12679	0.6988
5.8069	12737	0.7203	4284	0.6355	3.9978	54023	0.6625	12400	0.6282	3.5579	126116	0.6574	27950	0.6274
5.5675	23567	0.6413	9020	0.5113	[illegible]	[illegible]	[illegible]	23177	0.5402	3.4742	237625	0.5572	66196	0.5014
5.3175	148180	0.2339	34450	0.2825	3.8170	[illegible]	0.2580	58223	0.3738	3.3905	565787	0.3233	651458	0.2891

* Total number of nodes N; average degree $\overline{k}$; average path length APL; Run time in seconds and modularity m.

rewiring strategy can provide border adjustment range of clustering coefficient and average path length in reasonable computing time. Experiment results show that our edge rewiring strategy can provide a boarder adjusting range for clustering coefficient and average path length than standard Monte Carlo method and the Simulated Annealing method under the same computation condition.

As part of the future work, we can consider the numerous microscopic rules such as the preferential attachment and triadic closure when adjusting topological features of network. Besides, we can further consider the internal structure of the network, such as motif distribution.

Acknowledgments

This paper is supported by National Natural Science Foundation of China (Nos.U1435212, 61432011, 61876103 and 61572005), Shanxi Scholarship Council of China (2017-014), the Projects of Key Research and Development Plan of Shanxi Province (201603D111014), the 1331 Engineering Project of Shanxi Province, China.

References

[1] S. Boccaletti, V. Latora, Y. Moreno, M. Chavez, D.-U. Hwang, Complex networks: Structure and dynamics, Physics Reports 424 (4-5) (2006) 175–308.

[2] M. E. Newman, The structure and function of complex networks, SIAM Review 45 (2) (2003) 167–256.

[3] C. Perera, A. V. Vasilakos, A knowledge-based resource discovery for internet of things, Knowledge-Based Systems 109 (2016) 122–136.

[4] J. Ashraf, E. Chang, O. K. Hussain, F. K. Hussain, Ontology usage analysis in the ontology lifecycle: A state-of-the-art review, Knowledge-Based Systems 80 (2015) 34–47.

[5] X. Lei, J. Zhao, H. Fujita, A. Zhang, Predicting essential proteins based on rna-seq, subcellular localization and go annotation datasets, Knowledge-Based Systems 151 (2018) 136–148.

[6] J. Leskovec, D. Chakrabarti, J. Kleinberg, C. Faloutsos, Z. Ghahramani, Kronecker graphs: An approach to modeling networks, Journal of Machine Learning Research 11 (Feb) (2010) 985–1042.

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

[8] A.-L. Barabási, R. Albert, Emergence of scaling in random networks, Science 286 (5439) (1999) 509–512.

[9] M. E. Newman, M. Girvan, Finding and evaluating community structure in networks, Physical Review E 69 (2) (2004) 026113.

[10] A. Clauset, M. E. Newman, C. Moore, Finding community structure in very large networks, Physical Review E 70 (6) (2004) 066111.

[11] G. Chen, X. Wang, X. Li, Introduction to complex networks: models, structures and dynamics, Higher Education Press, 2012.

[12] A. Clauset, C. R. Shalizi, M. E. Newman, Power-law distributions in empirical data, SIAM Review 51 (4) (2009) 661–703.

[13] H. Li, B. Zhan, W. Zhen, J. Cao, Y. Shi, Enhance the performance of network computation by a tunable weighting strategy, IEEE Transactions on Emerging Topics in Computational Intelligence 2 (3) (2018) 214–223.

[14] J. Cao, B. Zhan, G. Gao, H. Tao, Weighted modularity optimization for crisp and fuzzy community detection in large-scale networks, Physica A Statistical Mechanics & Its Applications 462 (2016) 386–395.

[15] Z. Bu, J. Cao, H. J. Li, G. Gao, H. Tao, Gleam: a graph clustering framework based on potential game optimization for large-scale social networks, Knowledge & Information Systems 55 (3) (2017) 741–770.

[16] P. Erdös, A. Rényi, On random graphs i, Publicationes Mathematicae (Debrecen) 6 (1959) 290–297.

[17] J. M. Kleinberg, Navigation in a small world, Nature 406 (6798) (2000) 845.

[18] R. Albert, A. L. Barabási, Topology of evolving networks: local events and universality, Physical Review Letters 85 (24) (2000) 5234.

[19] G. Bianconi, A. L. Barabási, Bose-einstein condensation in complex networks, Physical Review Letters 86 (24) (2001) 5632–5635.

[20] X. Li, G. Chen, A local-world evolving network model, Physica A: Statistical Mechanics and its Applications 328 (1-2) (2003) 274–286.

[21] P. Holme, B. J. Kim, Growing scale-free networks with tunable clustering, Physical Review E 65 (2) (2002) 026107.

[22] Y. Fan, M. Li, P. Zhang, J. Wu, Z. Di, Accuracy and precision of methods for community identification in weighted networks, Physica A: Statistical Mechanics and its Applications 377 (1) (2007) 363–372.

[23] L. Danon, A. Díaz-Guilera, A. Arenas, The effect of size heterogeneity on community identification in complex networks, Journal of Statistical Mechanics: Theory and Experiment 2006 (11) (2006) P11010.

[24] A. Lancichinetti, S. Fortunato, F. Radicchi, Benchmark graphs for testing community detection algorithms, Physical Review E 78 (4) (2008) 046110.

[25] A. Lancichinetti, S. Fortunato, Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities, Physical Review E 80 (1) (2009) 016118.

[26] F. Zaidi, Small world networks and clustered small world networks with random connectivity, Social Network Analysis and Mining 3 (1) (2013) 51–63.

[27] M. Q. Pasta, Z. Jan, A. Sallaberry, F. Zaidi, Tunable and growing network generation model with community structures, in: Cloud and Green Computing (CGC), 2013 Third International Conference on, 2013, pp. 233–240.

[28] F. Zaidi, M. Q. Pasta, A. Sallaberry, G. Melançon, Social ties, homophily and extraversion–introversion to generate complex networks, Social Network Analysis and Mining 5 (1) (2015) 29.

[29] M. E. J. Newman, Properties of highly clustered networks, Physical Review E 68 (2) (2003) 026121.

[30] E. Volz, Random networks with tunable degree distribution and clustering, Physical Review E 70 (5) (2004) 056115.

[31] M. A. Serrano, M. Boguná, Tuning clustering in random networks with arbitrary degree distributions, Physical Review E 72 (3) (2005) 036133.

[32] Q. Guo, T. Zhou, J. G. Liu, W. J. Bai, B. H. Wang, M. Zhao, Growing scale-free small-world networks with tunable assortative coefficient, Physica A: Statistical Mechanics and its Applications 371 (2) (2006) 814–822.

[33] J. Badham, R. Stocker, A spatial approach to network generation for three properties: Degree distribution, clustering coefficient and degree assortativity, Journal of Artificial Societies and Social Simulation 13 (1) (2010) 11.

[34] A. T. Skjeltorp, A. V. Belushkin, Forces, Growth and Form in Soft Condensed Matter: At the Interface between Physics and Biology, Vol. 160, Springer Science & Business Media, Germany, 2005.

[35] B. J. Kim, Performance of networks of artificial neurons: The role of clustering, Physical Review E 69 (4) (2004) 045101.

[36] A. I. Reppas, K. Spiliotis, C. I. Siettos, Tuning the average path length of complex networks and its influence to the emergent dynamics of the majority-rule model, Mathematics and Computers in Simulation 109 (2015) 186–196.

[37] K. Orman, V. Labatut, H. Cherifi, An empirical study of the relation between community structure and transitivity, in: Complex Networks, Springer, 2013, pp. 99–110.

[38] H. Kim, Z. Toroczkai, P. L. Erdös, I. Miklós, L. A. Székely, Degree-based graph construction, Physics A : Mathematical and Theoretical 42 (39) (2009) 392001.

[39] M. Molloy, B. Reed, A critical point for random graphs with a given degree sequence, Random Structures & Algorithms 6 (2-3) (1995) 161–180.