Network structure optimization for social networks by minimizing the average path length

Wei Du
Xi'an Jiaotong University
Gang Li
Xi'an Jiaotong University
Xiaochen He (xiaochenhe@xjtu.edu.cn)
Xi'an Jiaotong University

Research Article

Keywords: average path length (APL), network characteristics, small-world phenomena, scale-free property

Posted Date: August 11th, 2021

DOI: https://doi.org/10.21203/rs.3.rs-793079/v1

Version of Record: A version of this preprint was published at Computing on February 16th, 2022. See the published version at https://doi.org/10.1007/s00607-022-01061-w.

Network structure optimization for social networks by minimizing the average path length

Wei Du[1], **Gang Li**[1,2], **and Xiaochen He**[1,3,*]

[1]School of Public Policy and Administration, Xi'an Jiaotong University, Xi'an, 710049, P. R. China
[2]Center for Administration and Complexity Science, Xi'an Jiaotong University, Xi'an, 710049, P. R. China
[3]School of Economics and Finance, Xi'an Jiaotong University, Xi'an, 710061, P. R. China
[*]xiaochenhe@xjtu.edu.cn

ABSTRACT

Network structure plays an important role in the natural and social sciences. Optimization of network structure in achieving specified goals has been a major research focus. In this paper, we propose a definition of structural optimization in terms of minimizing the network's average path length (APL) by adding edges. We suggest a memetic algorithm to find the minimum-APL solution by adding edges. Experiments show that the proposed algorithm can solve this problem efficiently. Further, we find that APL will ultimately decrease linearly in the process of adding edges, which is affected by the network diameter.

Introduction

Social networks have been analyzed over the past several decades. Most social network research can be classified into the following three fields: (1) analysis of network characteristics, which focuses on finding indices to quantify features of network structure (such as modularity) and utilizing these indices to distinguish broad categories of network structure (such as small-world phenomena, scale-free property)[1–4]; (2) exploration of mechanisms of network structure formation using statistical or dynamic models such as the Watts-Strogatz or Barabási-Albert models[5–7]; (3) the dynamic evolution of networks, which focuses on the transformation of network structures based on propagation models[8–10]. Obviously, network structure plays a fundamental role in the analysis of social networks, and optimizing network structure can help to improve network performance.

Previous research has shown that a random network with density of 50% has high structural entropy[11]. Wang et al. showed that contagious duration can be related to the network spectral radius, so that optimizing the spectral radius by adjusting the network structure can constrain the contagious duration[12]. Watts and Strogatz found the small-world network structure by rewiring network edges through optimization of the clustering coefficient and APL[5]. Other scholars have concentrated on improving network characteristics by adjusting network structure[13–15]. In this paper, we call this process "network structure optimization". It is the process of adjusting the network structure (adding, deleting or rewiring edges) to improve performance with respect to a specified outcome. Although there are different methods of improving structural performance, a standard definition of structural optimization has yet to be proposed. Generally, a network can be defined as a graph $G = (V, E)$, where $\{v_i\} \in V$ represents the set of nodes; $\{e_{ij}\} \in E \rightarrow \{0, 1\}$ represents the set of edges, $e_{ij} = 1$ denotes a relationship between nodes i and j, and $e_{ij} = 0$ indicates no relationship between nodes i and j. The definition of "network structure optimization" is as follows.

Definition 1: For a specific performance P_G of a network G, structural optimization can be formulated as the process of optimizing P_G by changing the network structure $G = (V, E)$ to $G' = (V', E')$, which produces $P_{G'}$ such that $P_{G'} < P_G$.

Problems related to structural optimization have been widely discussed[13,16–18]. Barahona et al. treated balance as an optimization problem[19], and Wang et al. proposed that the transformation of an imbalanced network to a balanced one at the lowest cost is also an optimization problem[20]. In order to improve the connectivity of wireless ad hoc networks, Cavalcanti et al. designed an optimal method for adding shortcuts from a percolation perspective[21]. Yang and Wang proposed a decentralized small-world optimization method for improving search efficiency[22]. Schleich et al. found a high-quality solution to improve small-world properties on vehicular ad hoc networks[23]. Du et al. presented a more effective method for optimizing small-world network generating process[24] . In this paper, we explore a specific form of structural optimization, namely, optimization of the average path length (APL). This is useful as it correlates with a set of important indexes such as robustness or transmission efficiency. Several methods focusing on the optimization of APL have been proposed, but the effectiveness of the optimization can still be improved, especially for optimization of random networks. This paper proposes a more effective method to optimize APL. Moreover, some network features of the optimal networks need to be analyzed in order to determine the effect of optimizing APL on different networks, and this analysis can find out the added edges with which feature can decrease APL

more efficiently. In this paper, we compute several indices related to optimal networks, such as assortative index or network diameter that relate to the optimal network structure. In the analysis, we also find that APL ultimately decreases linearly in the process of adding edges, and this occurs because the network diameter affects the optimization of APL, which is ignored in the previous studies. In addition, this phenomenon can be used to design more effective methods to optimize APL.

To make structural optimization clearer, we discuss the problem in terms of minimizing a specific index. APL is a fundamental quantity that contains important information for understanding the behaviors of networks[25]. For example, APL can be used to analyze topological structures of real world systems such as the World Wide Web[7], to measure the transfer efficiency of a metabolic network[26], or to measure robustness with respect to random failures[27]. To minimize APL, we can either decrease communication cost[28], or increase synchronization[29–33]. Here, we take APL as the index to be optimized.

Minimizing APL has been the subject of much research. Donetti et al. proposed a new class of networks, named "entangled networks", which had small APL and large girths[34]. To speed up the convergence in Donetti's model, Xuan et al. suggested a simulated annealing method to achieve the optimal network by minimizing APL[35]. Keren explored the reduction of APL in binary decision diagrams using a spectral technique based on properties of the Walsh spectrum of a Boolean function and its autocorrelation function[36]. In this paper, we minimize APL by adding edges; this process can be applied to real optimization problems such as enterprise mergers or location planning. Specifically, for a connected network $G = (V,E)$ with $n = \|V\|$ nodes and $\|E\|$ the initial number of edges, we add k edges $(1 \le k \le \frac{n(n-1)}{2} - \|E\|)$ to produce a network $G' = (V,E')$ with the minimum APL. Our objective function can be expressed as Eq. (1):

$$\begin{aligned} &min\ APL \\ &s.t. \|E'\| - \|E\| = k \end{aligned} \tag{1}$$

It should be noted that optimization of APL suffers from combinatorial explosion problem. There can be $\binom{\frac{n(n-1)}{2} - \|E\|}{k}$ ways of adding k edges to a network with n nodes. Since the time to compute APL is $O(n^3)$ finding the best solution takes time $O(n^{2k+3})$, which is high even for a small size network. This problem can be seen as an amendment to link recommendations, namely to select from the candidates link recommendations that improve the APL, and many related algorithms can be employed to accomplish this[37,38]. Meyerson and Tagiku first defined this link recommendation as Average Shortest Path Distance Minimization (ASPDM) and proposed an approximation method (ASPDM Algorithm) to solve this problem[39]; this method finds a source vertex S, from which there is a lower value of the sum of path lengths to other nodes, and then adds edges from S to k other nodes to give a lower APL. The complexity of this method is $O(n^3)$, which is quite time-saving, and the solution was proved to be at least as good a solution, but there still exists a large distance from the global optimal solution. Upon adding a new edge, the path length of some pairs of nodes will be changed, while the path length of other pairs remains unchanged; Parotsidis et al. proposed the EdgeEffect Algorithm to maximize the reduction of those changed pairs' lengths[40]. The complexity of the EdgeEffect Algorithm is $O(n^3 \cdot \|E\|)$, which is higher than that of the ASPDM Algorithm, but the solution is more effective. Different variants of these algorithms have been considered, leading to qualitatively similar results[41,42]. A greedy algorithm adds edges one by one, and has proved to be the most efficient method in optimizing APL[40,43,44]. In this paper, we modify this greedy algorithm as follows: a new edge is added at each step; specifically, for the first step, we choose the edge giving the minimum APL from the $(\frac{n(n-1)}{2} - \|E\|)$ nonexistent edges, and for the second step we choose the best edge from the remaining $(\frac{n(n-1)}{2} - \|E\| - 1)$ nonexistent edges, and for the k^{th} step we choose the best edge from the remaining $(\frac{n(n-1)}{2} - \|E\| - k + 1)$ edges. The total time complexity of this greedy algorithm is $O(n^5 \cdot k)$. This algorithm has difficulty finding the globally optimal solution since it performs a local search. Its solution by adding k edges depends completely on the solution of adding $k-1$ edges. Therefore, a new optimizing algorithm that uses a global search technique is required to find more precise solutions.

The Algorithm

Since most structural optimizations suffer from the combinatorial explosion problem, evolutionary algorithms (such as a genetic algorithm or simulated annealing) that combine global and local search techniques have been widely used and can be effective in preventing combinatorial explosion[45,46]. In this section, we propose a memetic algorithm combined with a genetic algorithm and a heuristic local search to minimize the APL.

Framework

Algorithm 1 gives the framework of our proposed algorithm. We first input the necessary parameters and load the network matrix to be optimized. Then we generate an initial population by the function Initial_Population(). Before the iteration number reaches Imax, or the objective function (APL) remains unchanged for 40 iterations, we repeat a process for finding the best solution. In this repeated process, we first select the parent population for genetic operations by use of the function Tournament_Selection(); then we operate crossover and mutation in the Genetic_Operation() to generate the offspring population; next, we use Local_Search() to find a better solution after the Genetic_Operation(), which can speed up the convergence; Update_Population() sorts the objective function of all chromosomes and selects better performing ones to construct a new population.

Algorithm 1 Framework of our algorithm

Input: maximum number of iterations: I_{max}; size of population: S_{pop}; size of mating pool: S_{pool}; size of tournament: S_{tour}; probability of crossover: P_c; probability of mutation: P_m.

Output: the position of added edges; the updating network's matrix; APL of the updating network.

Load: the initial network adjacency matrix;
$P \leftarrow$ Initial_Population(S_{pop});
repeat
  $P_{parent} \leftarrow$ Tournament_Selection(P, S_{pool}, S_{tour});
  $P_{offspring} \leftarrow$ Genetic_Operation (P_{parent}, P_c, P_m);
  $P'_{offspring} \leftarrow$ Local_Search $(P_{offspring})$;
  $P \leftarrow$ Update_Population $(P, P'_{offspring})$;
until Termination.

Representation and Initialization

Our target is to determine in which positions we should add new edges to reduce the APL. To this end, we first find the positions of those pairs of nodes that are not connected (nonexistent edges). Then we give a set of sequence numbers to these positions; thus our solutions can be encoded as chromosomes consisting of these corresponding sequence numbers. An example is shown in Fig. 1, where the second entry of the chromosome changes from 2 to 3, so the added edge between Node 1 and Node 4 changes to the edge between Node 2 and Node 4. At initialization, we generate a population and randomly assign a set of numbers to every element of the chromosomes in the population. To speed up convergence and retain population diversity, we add two new sub-populations: (1) each element of the chromosomes (i.e., the added edge) connects the two highest degree nodes; (2) each element of the chromosomes connects the highest degree node and the lowest degree node. In other words, the initial population can be divided into three parts: solutions with random assignment, assortative connecting assignment and disassortative connecting assignment.

Genetic operation

The genetic operation consists of two parts: crossover and mutation.

In the crossover procedure, we randomly select two chromosomes from the parent population with probability P_c and generate two offspring chromosomes. The common elements of the two offspring chromosomes remain unchanged. For the remaining elements that differ between the chromosomes we give an ordering, and for every unshared element, crossover operates with probability a: if $a < 0.5$, the element remains unchanged; and if $a \geq 0.5$, the corresponding elements (with the same ordering number) of the two offspring are swapped. Fig. 2 gives an illustration of crossover. The sequence numbers 11 and 7 are elements common to both offspring chromosomes, so they do not need to exchange. For the other four elements, assuming that the probability a equals 0.4, 0.8, 0.7, and 0.2, respectively, the first and fourth unshared element remain unchanged since $a < 0.5$, and the second and third unshared elements are swapped between the two chromosomes since $a \geq 0.5$.

In the mutation procedure, we randomly select a chromosome with probability P_m, and for a selected element of the chromosome, we randomly choose a different sequence number to replace that element.

Local search

By incorporating some prior knowledge, a local search can efficiently reduce useless explorations and speed up the convergence of genetic algorithms[47]. In this section, we employ Path Length Learning and Neighbor Learning, respectively.

Path Length Learning is shown in Algorithm 2. We first update the network matrix by decoding the chromosome. Then, we compute the degrees of nodes connected by the added edges and delete the edge with the minimum sum degree of those pairs of nodes. Finally, we add a new edge to the pair of nodes that has the longest path length. This learning technique helps produce a better result.

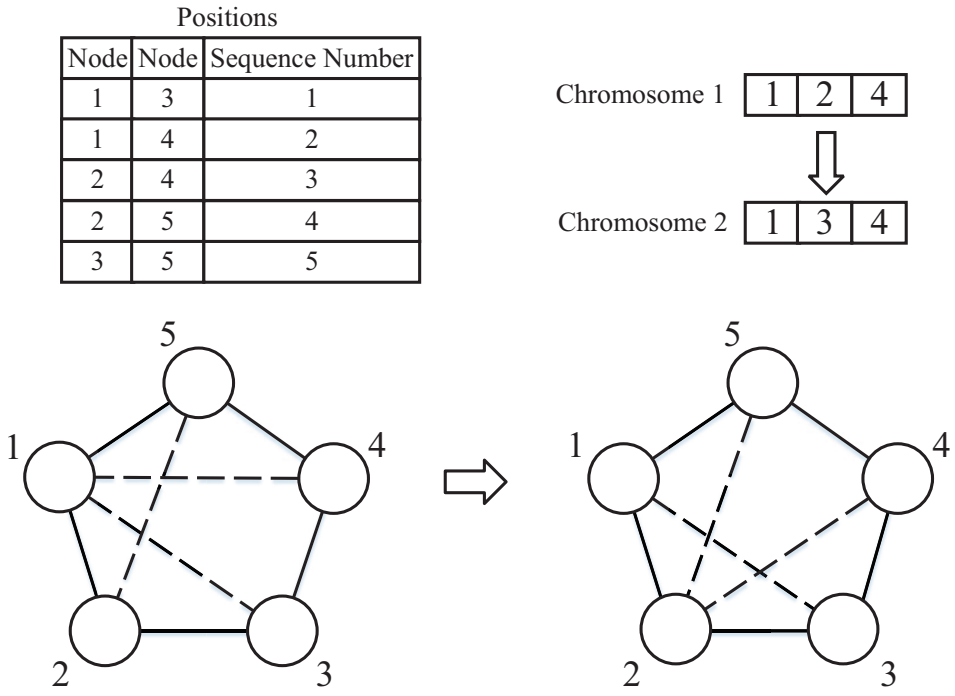


Figure 1. Illustration of the representation. The upper left table is the position of disconnected edges represented by a set of sequence numbers. Upper right is an illustration of chromosomes. The lower part is the topological graph corresponding to the chromosomes. The solid lines represent the existing edges, the dashed lines represent the added edges, and the number near the node is the sequence number of the node.

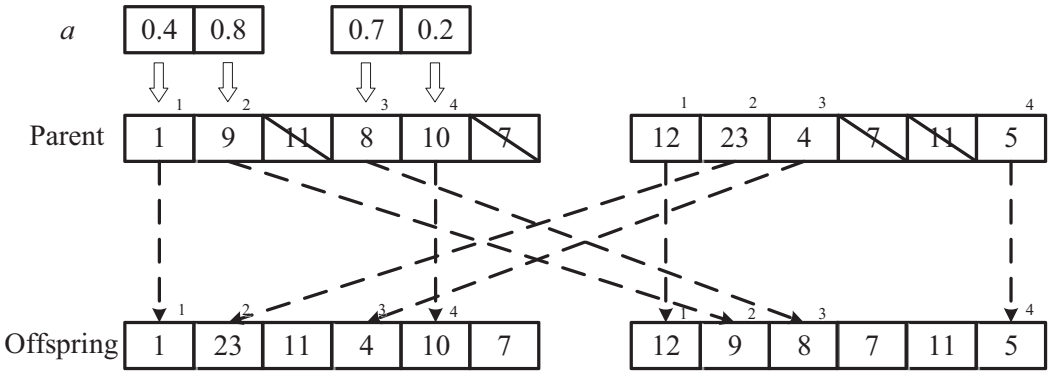


Figure 2. Illustration of crossover. The first line is the generalized probability a. The upper two chromosomes are parental chromosomes while the lower two are offspring chromosomes. The small numbers near the elements are the ordering numbers of the unshared elements, and the elements with the same ordering number are corresponding elements. We check every unshared element of the two parent chromosomes; if the generalized $a < 0.5$, the element remains unchanged in the offspring chromosomes, while if $a \geq 0.5$, the corresponding elements are swapped.

Algorithm 2 Path Length Learning

Input: the chromosome $X_{offspring}$; the adjacency matrix of the initial network M.
Output: $X_{offspring2}$.

$X_{offspring2} = X_{offspring}$;
Update the matrix M by decoding $X_{offspring}$;
Find the position of the pair of nodes with longest path length of the updated matrix (the position is represented by the sequence number g);
Check every element of $X_{offspring}$ and compute the degrees of nodes connected by the added edge decoded by the element;
Find the element i with the minimum sum value of nodes pair degrees;
$X_{offspring2}(i) = g$;

In Neighbor Learning, illustrated in Algorithm 3, for each element in the offspring chromosome, we choose the value of its best neighbor (a neighbor of an added edge is one of the nonexistent edges sharing one of the same nodes) to judge whether this choice produces a better result. If the new choice decreases APL, then we accept the new element, and the result improves.

Algorithm 3 Neighbor Learning

Input: the chromosome $X_{offspring2}$; the number of added edges k; the adjacency matrix of the initial network M.
Output: $X_{offspring2}$.

$X_{offspring3} = X_{offspring2}$
Update the matrix M by decoding $X_{offspring3}$;
for $i = 1; i \leq k; i++$ **do**
    for $m = 1$; number of $X_{offspring3}(i)$'s neighbor; $m++$ **do**
        $X_{offspring3}(i)$ = $X_{offspring3}(i)$'s neighbor;
    end for
    find $X_{offspring3}(i)$'s neighbor q with minimum APL($X_{offspring}$);
    $X_{offspring3}(i) = q$;
    if APL($X_{offspring3}$) $<$ APL($X_{offspring2}$) **then**
        APL($X_{offspring2}$) = APL($X_{offspring3}$)
    end if
end for

Complexity analysis

If we add k edges to a network of size n, the time complexity of the proposed algorithm can be estimated as follows: At each iteration, we first need to carry out the crossover $\frac{S_{pool}}{2}$ times and mutation S_{pool} times, where S_{pool} is the size of the mating pool for the genetic operation. Since the time complexity of computing the APL is $O(n^3)$, the genetic operation will cost $O(S_{pool}(k+n^3))$. Second, on performing the path learning, updating the matrix costs time $O(k)$, finding the added edge with the minimum node degrees costs $O(kn)$, finding the nonexistent edge with longest path length costs $O(n^3)$, so the path learning will cost $O(n^3+kn+k)$. Third, to perform neighbor learning, we consider n neighbors of each added edge, and it will cost $O(kn^4)$ to compute the APL of all those neighbors. As a result, the complexity of our algorithm is $O(kn^4)$ at each iteration. If the network size n is big enough, the complexity of the Memetic Algorithm is less than that of Greedy Algorithm.

Results

In this section, we test our algorithm on different computer-generated networks. The experiments were carried out on a 2.40 GHz CPU, 4.00 GB Memory and Windows 10 operating system using MATLAB to execute the procedure. Since the algorithm we proposed is not parameter-free, we need to set some values for these parameters in advance, such as the size of population or mating pool. Some of these parameters are set to the fixed values that we found by trial and error in order to ensure that the proposed algorithm has excellent performance. Table. 1 shows the parameters used in the experiments. We first ran our algorithm on random networks with different network sizes, and compared it with five other methods: (1) add k edges at one time, where each of the added edges connects the two highest degree nodes (Denoted as Big-Big, which is an assortative connecting method); (2) add k edges at one time, where each of the added edges connects the highest degree node and the lowest degree node (denoted as Big-Small, which is a disassortative connecting method); (3) add k edges one by one, with

I_{max}	Maximum number of iterations	200
S_{pop}	Size of population	200
S_{pool}	Size of mating pool	100
S_{tour}	Size of tournament	2
P_c	Probability of Crossover	0.1
P_m	Probability of Mutation	0.9

Table 1. Parameters of the experiments.

each of the added edges giving the minimum APL as described in Section 2 (denoted as the Greedy Algorithm). We set k from 1 to 50; (4) ASPDM Algorithm proposed by Meyerson and Tagiku[39]; (5) EdgeEffect Algorithm proposed by Parotsidis et al[40]. Since evolutionary algorithms have inherent randomness, many of them, including our proposed algorithm, may fall into a local minimum in some cases. We ran the above-mentioned algorithms ten times. For the results of the mean APL for these ten runs, the Greedy Algorithm can find the best solutions, while the Memetic Algorithm can find better solutions than other methods. For finding the minimum APL for these ten runs, the Memetic Algorithm performs best. In order to make the experimental results easier to comprehend and for convenience, we list only the results of the minimum APL for the ten runs of these algorithms. We set the number of added edges to 10; Fig. 3 shows the optimizing result with different network sizes. Compared with the other methods, our algorithm and Greedy Algorithm can always find lower-APL solutions with different network sizes.

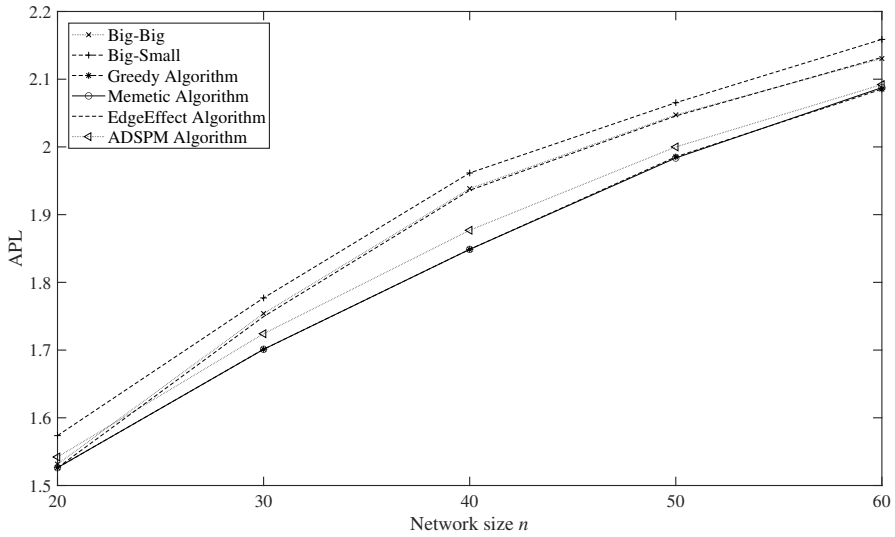
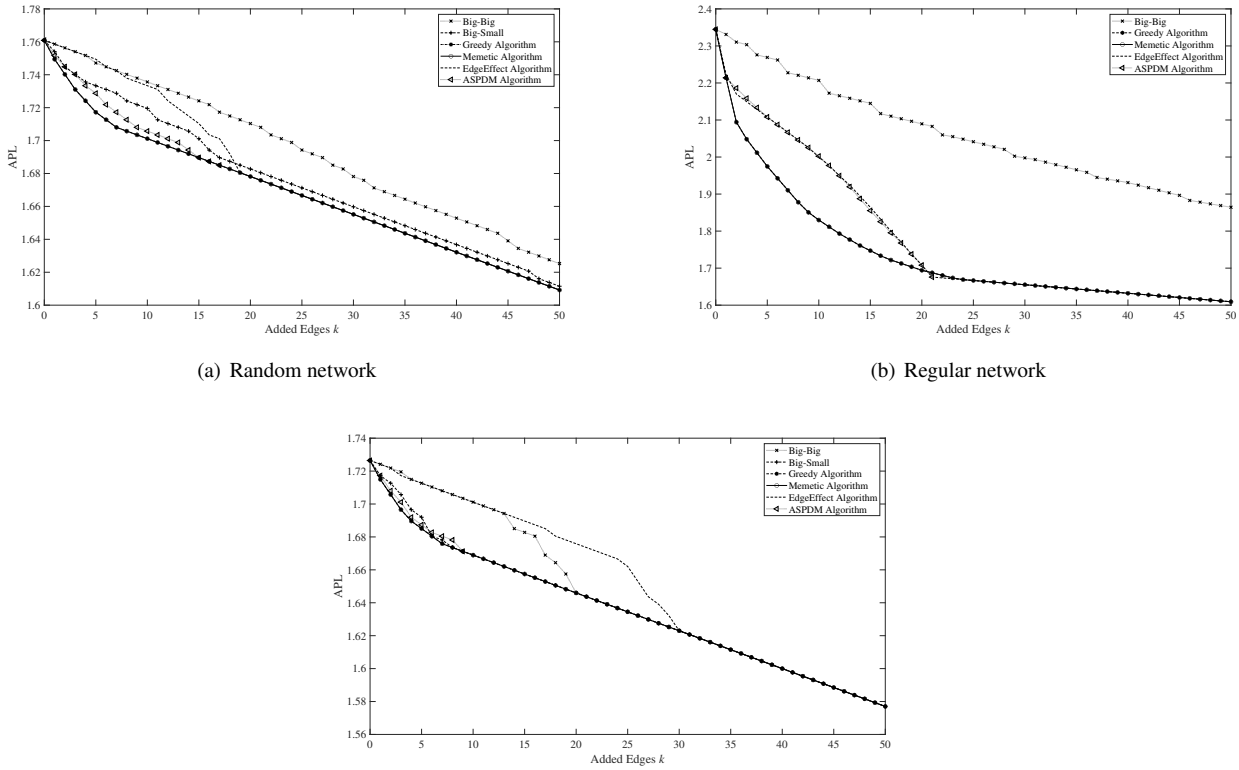


Figure 3. Optimizing result of APL with different network size.

Then we ran our algorithm on three different networks (random network, regular network and scale-free network of size 30) with different numbers of added edges to explore the effect of each added edges. Fig. 4 shows the optimizing results for different networks. Since the degrees of regular network nodes are all equal, the results of Big-Big and Big-Small are the same; here we record just the result of Big-Big on the regular network. Since ASPDM Algorithm adds edges from a source vertex S to other nodes, which do not connect to S, when all nodes are connected to S in the process of adding edges no more edges will be added as shown in Fig. 4.



(a) Random network

(b) Regular network

(c) Scale-free network

Figure 4. Optimizing results of APL for three different networks.

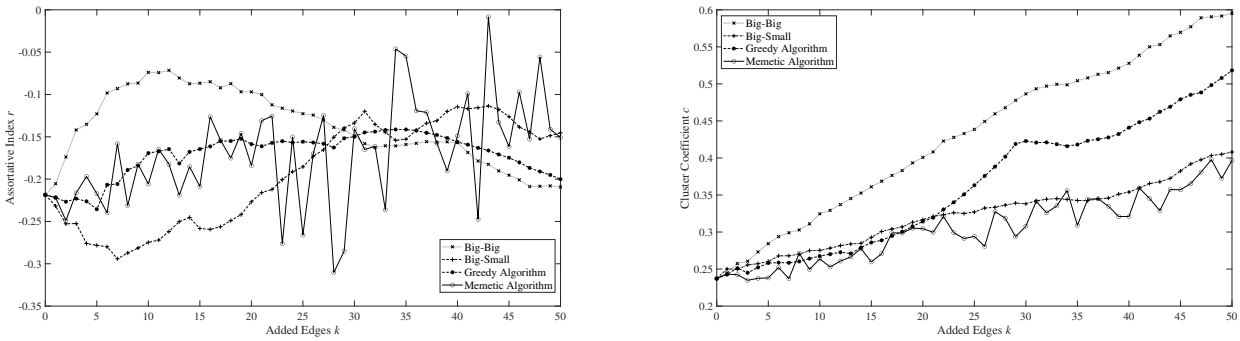
Compared with the other five methods, our algorithm can find lower-APL solutions except for a few solutions of the Greedy Algorithm and EdgeEffect Algorithm which do better than the Memetic Algorithm on the regular network (this is because our priori knowledge is less useful on regular networks). We conclude that in most cases, our proposed algorithm can effectively minimize APL. Further, we find that the solutions of Big-Small are close to the optimizing solutions in Figs. 4(a) and 4(c). In a sense, disassortative connecting can find good solutions for minimizing APL. Moreover, we find when the number of added edges increases to a threshold value, APL begins to decline linearly no matter which method is used. This phenomenon can be explained by the feature of Critical Diameters[48,49]. Specifically, when we add enough edges to the network, the longest path length of the network becomes 2 (i.e. the network diameter is 2), in which case, if we randomly add a new edge between a pair of nodes, we can only change the path lengths of these two nodes to 1, and thus the APL can be reduced by just $\frac{2}{n(n-1)}$ for each step, which constitutes a linear decline.

In this section, we analyze the structure of the optimal networks to determine the effect of optimizing APL on different networks. The assortative index r and cluster coefficient c are important indices for measuring network characteristics[50]. The assortative index r measures the characteristic of assortative (or disassortative) mixing, and can be formulated as Eq.(2):

$$r = \frac{M^{-1}\sum_i j_i k_i - \left[M^{-1}\sum_i \frac{1}{2}(j_i + k_i)\right]^2}{M^{-1}\sum_i \frac{1}{2}(j_i{}^2 + k_i{}^2) - \left[M^{-1}\sum_i \frac{1}{2}(j_i + k_i)\right]^2} \tag{2}$$

where j_i, k_i are the degrees of the vertices at the ends of the i^{th} edge.[51] The cluster coefficient c measures average local density ($c = \frac{\sum_v CC_v}{n}$), where $CC_v = \frac{|E(\Gamma_v)|}{k_v(k_v-1)}$, $|E(\Gamma_v)|$ is the number of edges in the relational neighborhood of node v, and $(k_v(k_v-1)$ is the number of possible edges in the relational neighborhood of node v. As APL decreases monotonically, the Memetic Algorithm results for r and c fluctuate because there can be more than one best solution with different connection rules that minimizes APL when more edges are added as shown in Figs 5-7. Further, the variation in r and c appears to be quite different in the three different network models.

For the random network experiments, as shown in Fig. 5, r does not show a regular trend, but the optimal networks turn out to be disassortative since most r values are negative, while c shows an increasing trend as more edges are added to the initial network, which means that a random network is gradually evolving into a small-world network. This figure shows two interesting phenomena. First, disassortative connection for adding edges may contribute more to the decrease of APL. Second, decreasing APL in random networks can simultaneously increase the cluster coefficient, which is helpful in constructing small-world networks.

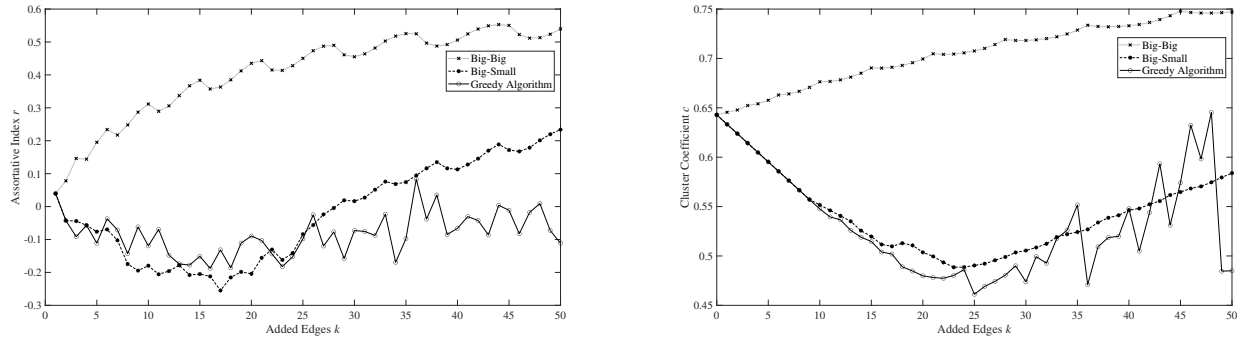


(a) Assortative index r as a function of added edges.

(b) Cluster coefficient c as a function of added edges.

Figure 5. Assortative index r and cluster coefficient c results for the random network.

For the regular network experiments, r and c both decline with the first few edges added as shown in Fig. 6. Thus, we can conclude that when adding a small number of edges, a disassortative connection is a good choice to decrease APL. Since a regular network initially has a high cluster coefficient, c will decrease when we add edges in a disassortative way. However, when we add a larger number of edges to a regular network, the disassortative connection becomes less effective. This conclusion is different from that for the random network.

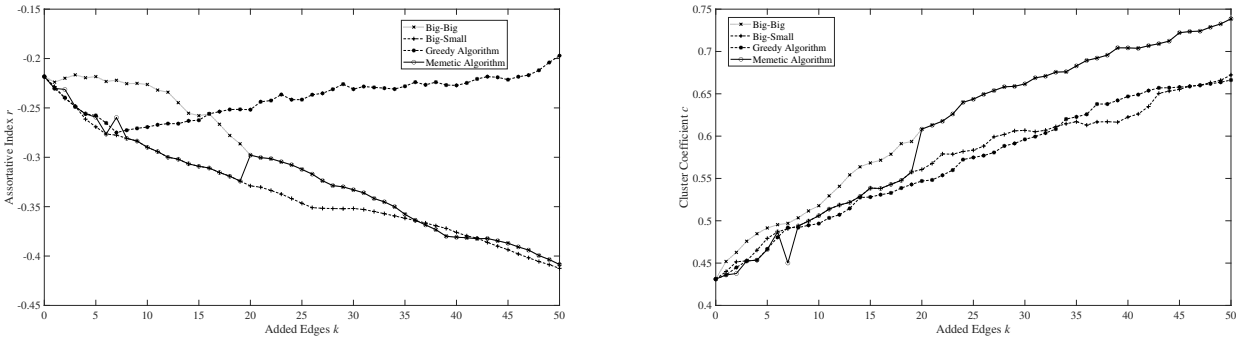


(a) Assortative index r as a function of added edges.

(b) Cluster coefficient c as a function of added edges

Figure 6. Assortative index r and cluster coefficient c results for the regular network.

For optimization of the scale-free network as shown in Fig. 7, all signs of r are negative, which indicates that optimal networks are all disassortative. However, c increases as new edges are added, because adding more edges to a scale-free network, especially linking high-degree nodes with other nodes, has a high probability of constructing more triangles, which will result in a higher c. The property of connections in scale-free networks is similar to that in random networks, and the disassortative feature of the connections is more obvious in scale-free networks.



(a) Assortative index r as a function of added edges.

(b) Cluster coefficient c as a function of added edges

Figure 7. Assortative index r and cluster coefficient c results for the scale-free network.

Conclusion

In this paper, we have focused on the optimization of a network by minimizing APL by adding edges, and we propose a memetic algorithm to find optimal solutions. The experimental results show that our algorithm can minimize APL efficiently. In the analysis, we also find that APL will ultimately decrease linearly in the process of adding edges, which is directly affected by the network diameter. Further, we compute the assortative index and cluster coefficient for optimal networks with different initial network structures and find that these two properties of optimal solutions may be quite different. We optimize only one indicator, APL, in this paper, but in many cases we need to optimize more than one feature of networks, for example minimizing APL and maximizing the cluster coefficient at the same time. This multi-objective structural optimization should be explored in future work.

References

1. Girvan, M. & Newman, M. E. J. Community structure in social and biological networks. *Proc. Natl. Acad. Sci.* **99**, 7821–7826, DOI: 10.1073/pnas.122653799 (2002).
2. Gómez, S., Jensen, P. & Arenas, A. Analysis of community structure in networks of correlated data. *Phys Rev E Stat Nonlin Soft Matter Phys* **80**, 016114, DOI: 10.1103/PhysRevE.80.016114 (2009).
3. Leicht, E. A. & Newman, M. E. Community structure in directed networks. *Phys Rev Lett* **100**, 118703, DOI: 10.1103/PhysRevLett.100.118703 (2008).
4. Zhang, X., Nadakuditi, R. R. & Newman, M. E. Spectra of random graphs with community structure and arbitrary degrees. *Phys Rev E Stat Nonlin Soft Matter Phys* **89**, 042816, DOI: 10.1103/PhysRevE.89.042816 (2014).
5. Watts, D. J. & Strogatz, S. H. Collective dynamics of 'small-world' networks. *Nature* **393**, 440–442, DOI: 10.1038/30918 (1998).
6. Barabási, A.-L. & Albert, R. Emergence of scaling in random networks. *Science* **286**, 509–512, DOI: 10.1126/science.286.5439.509 (1999).
7. Albert, R. & Barabási, A.-L. Statistical mechanics of complex networks. *Rev. Mod. Phys.* **74**, 47–97, DOI: 10.1103/revmodphys.74.47 (2002).
8. Schelling, T. C. Dynamic models of segregation†. *The J. Math. Sociol.* **1**, 143–186, DOI: 10.1080/0022250x.1971.9989794 (1971).
9. Bailey, N. T. J. *The Mathematical Theory of Infectious Diseases and Its Applications* (Hafner Press, New York, 1975), 2nd edn.
10. Anderson, R. & May, R. *Infectious Diseases of Humans* (Oxford University Press, Oxford, 1992).
11. Meng., C., Haifeng., D. & Feldman, M. W. A new network structure entropy based on maximum flow. *Acta Phys. Sinica* **63**, 060504, DOI: 10.7498/aps.63.060504 (2014).
12. Yang, W., Chakrabarti, D., Chenxi, W. & Faloutsos, C. Epidemic spreading in real networks: an eigenvalue viewpoint. 25–34, DOI: 10.1109/reldis.2003.1238052 (IEEE Comput. Soc, 2003).

13. Wang, Y., Zeng, A., Di, Z. & Fan, Y. Enhancing synchronization in growing networks. *EPL (Europhysics Lett.* **96**, DOI: 10.1209/0295-5075/96/58007 (2011).

14. Shi, D., Chen, G., Thong, W. W. K. & Yan, X. Searching for optimal network topology with best possible synchronizability. *IEEE Circuits Syst. Mag.* **13**, 66–75, DOI: 10.1109/mcas.2012.2237145 (2013).

15. Sun, Y., Du, H., Gong, M., Ma, L. & Wang, S. Fast computing global structural balance in signed networks based on memetic algorithm. *Phys. A: Stat. Mech. its Appl.* **415**, 261–272, DOI: 10.1016/j.physa.2014.07.071 (2014).

16. Kułakowski, K., Gawroński, P. & Gronek, P. The heider balance: A continuous approach. *Int. J. Mod. Phys. C* **16**, 707–716, DOI: 10.1142/s012918310500742x (2005).

17. Antal, T., Krapivsky, P. L. & Redner, S. Dynamics of social balance on networks. *Phys Rev E Stat Nonlin Soft Matter Phys* **72**, 036121, DOI: 10.1103/PhysRevE.72.036121 (2005).

18. Facchetti, G., Iacono, G. & Altafini, C. Computing global structural balance in large-scale signed social networks. *Proc Natl Acad Sci U S A* **108**, 20953–8, DOI: 10.1073/pnas.1109521108 (2011).

19. Barahona, F. On the computational complexity of ising spin glass models. *J. Phys. A: Math. Gen.* **15**, 3241–3253, DOI: 10.1088/0305-4470/15/10/028 (1982).

20. Wang, S. *et al.* Optimizing dynamical changes of structural balance in signed network based on memetic algorithm. *Soc. Networks* **44**, 64–73, DOI: 10.1016/j.socnet.2015.06.004 (2016).

21. Cavalcanti, D., Agrawal, D., Kelner, J. & Sadok, D. *Exploiting the Small-World Effect to Increase Connectivity in Wireless Ad Hoc Networks*, book section Chapter 53, 388–393. Lecture Notes in Computer Science (2004).

22. Yang, X. & Xiaohua, W. Decentralized small-world optimization strategy. *J. Suzhou Univ. (Engineering Sci. Ed.* **27**, 41–46, DOI: 10.3969/j.issn.1673-047X.2007.03.010 (2007).

23. Schleich, J., Danoy, G., Dorronsoro, B. & Bouvry, P. Optimising small-world properties in vanets: Centralised and distributed overlay approaches. *Appl. Soft Comput.* **21**, 637–646, DOI: 10.1016/j.asoc.2014.03.045 (2014).

24. Du, H., Fan, J., He, X. & Feldman, M. W. A genetic simulated annealing algorithm to optimize the small-world network generating process. *Complexity* **2018**, 1–12, DOI: 10.1155/2018/1453898 (2018).

25. Yu, F., Li, Y. & Wu, T.-J. A temporal ant colony optimization approach to the shortest path problem in dynamic scale-free networks. *Phys. A: Stat. Mech. its Appl.* **389**, 629–636, DOI: 10.1016/j.physa.2009.10.005 (2010).

26. Ma, H. & Zeng, A. P. Reconstruction of metabolic networks from genome data and analysis of their global structure for various organisms. *Bioinformatics* **19**, 270–277, DOI: 10.1093/bioinformatics/19.2.270 (2003).

27. Wang, B., Tang, H., Guo, C., Xiu, Z. & Zhou, T. Optimization of network structure to random failures. *Phys. A: Stat. Mech. its Appl.* **368**, 607–614, DOI: 10.1016/j.physa.2005.12.050 (2006).

28. Ashton, D. J., Jarrett, T. C. & Johnson, N. F. Effect of congestion costs on shortest paths through complex networks. *Phys Rev Lett* **94**, 058701, DOI: 10.1103/PhysRevLett.94.058701 (2005).

29. Lago-Fernández, L. F., Huerta, R., Corbacho, F. & Sigüenza, J. A. Fast response and temporal coherent oscillations in small-world networks. *Phys. Rev. Lett.* **84**, 2758–2761, DOI: 10.1103/physrevlett.84.2758 (2000).

30. Gade, P. M. & Hu, C.-K. Synchronous chaos in coupled map lattices with small-world interactions. *Phys. Rev. E* **62**, 6409–6413, DOI: 10.1103/physreve.62.6409 (2000).

31. Jost, J. & Joy, M. P. Spectral properties and synchronization in coupled map lattices. *Phys. Rev. E* **65**, DOI: 10.1103/physreve.65.016201 (2001).

32. Hong, H., Choi, M. Y. & Kim, B. J. Synchronization on small-world networks. *Phys Rev E Stat Nonlin Soft Matter Phys* **65**, 026139, DOI: 10.1103/PhysRevE.65.026139 (2002).

33. Barahona, M. & Pecora, L. M. Synchronization in small-world systems. *Phys. Rev. Lett.* **89**, DOI: 10.1103/PhysRevLett.89.054101 (2002).

34. Donetti, L., Hurtado, P. I. & Munoz, M. A. Entangled networks, synchronization, and optimal network topology. *Phys Rev Lett* **95**, 188701, DOI: 10.1103/PhysRevLett.95.188701 (2005).

35. Xuan, Q., Li, Y. & Wu, T.-J. Optimal symmetric networks in terms of minimizing average shortest path length and their sub-optimal growth model. *Phys. A: Stat. Mech. its Appl.* **388**, 1257–1267, DOI: 10.1016/j.physa.2008.12.020 (2009).

36. Keren, O. Reduction of average path length in binary decision diagrams by spectral methods. *IEEE Transactions on Comput.* **57**, 520–531, DOI: 10.1109/tc.2007.70811 (2008).

37. Adamic, L. A. & Adar, E. Friends and neighbors on the web. *Soc. Networks* **25**, 211–230, DOI: 10.1016/s0378-8733(03)00009-1 (2003).

38. Liben-Nowell, D. & Kleinberg, J. The link-prediction problem for social networks. *J. Am. Soc. for Inf. Sci. Technol.* **58**, 1019–1031, DOI: 10.1002/asi.20591 (2007).

39. Meyerson, A. & Tagiku, B. *Minimizing Average Shortest Path Distances via Shortcut Edge Addition*, 272–285 (Springer Berlin Heidelberg, 2009).

40. Parotsidis, N., Pitoura, E. & Tsaparas, P. *Selecting Shortcuts for a Smaller World*, 28–36. Proceedings (Society for Industrial and Applied Mathematics, 2015). Doi:10.1137/1.9781611974010.4.

41. Papagelis, M., Bonchi, F. & Gionis, A. Suggesting ghost edges for a smaller world, DOI: 10.1145/2063576.2063952 (2011).

42. Lazaridou, K., Semertzidis, K., Pitoura, E. & Tsaparas, P. *Identifying Converging Pairs of Nodes on a Budget* (2015).

43. Lee, S. H. & Holme, P. A greedy-navigator approach to navigable city plans. *The Eur. Phys. J. Special Top.* **215**, 135–144, DOI: 10.1140/epjst/e2013-01720-8 (2013).

44. Papagelis, M. Refining social graph connectivity via shortcut edge addition. *ACM Transactions on Knowl. Discov. from Data* **10**, 1–35, DOI: 10.1145/2757281 (2015).

45. Brusco, M. & Steinley, D. A tabu-search heuristic for deterministic two-mode blockmodeling of binary network matrices. *Psychometrika* **76**, 612–33, DOI: 10.1007/s11336-011-9221-9 (2011).

46. Ross, B. J. & Zuviria, E. Evolving dynamic bayesian networks with multi-objective genetic algorithms. *Appl. Intell.* **26**, 13–23, DOI: 10.1007/s10489-006-0002-6 (2006).

47. Moscato, P. On evolution, search, optimization, genetic algorithms and martial arts - towards memetic algorithms. *Caltech Concurr. Comput. Program* (1989).

48. Du, H., He, X., Du, W. & Feldman, M. W. Optimization of the critical diameter and average path length of social networks. *Complexity* **2017**, 1–11, DOI: 10.1155/2017/3203615 (2017).

49. Du, H., Wang, J., He, X. & Du, W. A memetic algorithm to optimize critical diameter. *Swarm Evol. Comput.* **47**, 56–65, DOI: 10.1016/j.swevo.2017.10.001 (2019).

50. Niu, J. & Wang, L. Structural properties and generative model of non-giant connected components in social networks. *Sci. China Inf. Sci.* **59**, DOI: 10.1007/s11432-015-0790-x (2016).

51. Newman, M. E. J. Assortative mixing in networks. *Phys. Rev. Lett.* **89**, DOI: 10.1103/physrevlett.89.208701 (2002).

Author contributions

X.C.H. designed the project and developed the initial idea. W.D. and G.L. contributed to analyses and interpretations of the results. All authors wrote the manuscript together.

Additional information

Competing interests: The authors declare no competing interests