PLOS ONE

RESEARCH ARTICLE

A parallel adaptive quantum genetic algorithm for the controllability of arbitrary networks

Yuhong Li[1], Guanghong Gong[1☯], Ni Li[1,2☯*]

1 School of Automation Science and Electrical Engineering, Beihang University, Beijing, China, **2** State Key Laboratory of Virtual Reality Technology and Systems, Beihang University, Beijing, China

☯ These authors contributed equally to this work.
* lini@buaa.edu.cn

Check for updates

OPEN ACCESS

Citation: Li Y, Gong G, Li N (2018) A parallel adaptive quantum genetic algorithm for the controllability of arbitrary networks. PLoS ONE 13(3): e0193827. https://doi.org/10.1371/journal.pone.0193827

Editor: Irene Sendiña-Nadal, Universidad Rey Juan Carlos, SPAIN

Received: August 17, 2017

Accepted: February 20, 2018

Published: March 19, 2018

Copyright: © 2018 Li et al. This is an open access article distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

Data Availability Statement: The network models in our study were from existing references. All the underlying data set for our study could be available from the eleven sources listed below. 46. Erdős P, Rényi A. On the strength of connectedness of a random graph. Acta Math. Hungar. 1961;12 (1): 261–267. 47. Barabási AL, Albert R. Emergence of scaling in random networks. Science. 1999;286 (5439):509–512. 48. Newman MEJ, Watts DJ. Renormalization group analysis of the small-world network model. Phys. Lett. A. 1999;263:341-346. 49. Watts DJ, Strogatz SH. Collective dynamics of small-world networks. Nature.1998;393:440-442.

Abstract

In this paper, we propose a novel algorithm—parallel adaptive quantum genetic algorithm—which can rapidly determine the minimum control nodes of arbitrary networks with both control nodes and state nodes. The corresponding network can be fully controlled with the obtained control scheme. We transformed the network controllability issue into a combinational optimization problem based on the Popov-Belevitch-Hautus rank condition. A set of canonical networks and a list of real-world networks were experimented. Comparison results demonstrated that the algorithm was more ideal to optimize the controllability of networks, especially those larger-size networks. We demonstrated subsequently that there were links between the optimal control nodes and some network statistical characteristics. The proposed algorithm provides an effective approach to improve the controllability optimization of large networks or even extra-large networks with hundreds of thousands nodes.

Introduction

The real world consists of ubiquitous intricate and intertwined networks. Some are tangible, such as traffic networks [1, 2], power networks [3], and financial networks [4], whereas others are invisible networks that penetrate into every aspect of our lives, such as interpersonal relationship networks [5, 6], wireless networks [7, 8], and ecological networks [9]. The expected goal of research into a complex network is to be able to regulate and control it from the outside, achieve the desirable state or performance by injecting outward control signals to some network nodes (called driver nodes), and ultimately achieve the real controllability of a complex network.

Many studies have been conducted on the relationship between network topology and network controllability [10–13]. Researchers have proposed that all hub nodes with a high degree or betweenness centrality could be chosen as driver nodes [14]. Jalili et al. [15] found that an optimum driver node could not always be a hub node. To elucidate the configuration of driver nodes for the optimum network pinning control, a differential evolution method was used. The method worked well, but it was only suitable for undirected networks. Assuming that the

50. Batagelj V, Mrvar A. Pajek datasets. <URL: http://vlado.fmf.uni-lj.si/pub/networks/data/>. 2006. 51. Milo R, Itzkovitz S, Kashtan N, Levitt R, Shenorr S, Ayzenshtat I, et al. Superfamilies of designed and evolved networks. Science. 2004;303 (5663): 1538-1542. 52. Burt RS. Social Contagion and Innovation: Cohesion versus Structural Equivalence. American Journal of Sociology.1987;92(6):1287-1335. 53. Newman MEJ. Finding community structure in networks using the eigenvectors of matrices. Phys. Rev. E. 2006;74(3): 036104. 54. https://sparse.tamu.edu/. 55. Leskovec J, Lang KJ, Dasgupta A, Mahoney MW. Community structure in large networks: Natural cluster sizes and the absence of large well-defined clusters. Internet Mathematics. 2009;6 (1):29-123. 56. Leskovec J, Kleinberg J, Faloutsos C. Graph evolution: Densification and shrinking diameters. ACM Transactions on Knowledge Discovery from Data(TKDD). 2007;1(1):2.

Funding: This research is supported by the National Science Foundation of China with granting No.61773032.

Competing interests: The authors have declared that no competing interests exist.

objective network had finite-dimensional linear dynamics [16], the network could be structurally controlled by applying one time-varying input to the power dominating set. In the practical applications with economical and physical constraints, driver nodes can not always be freely selected to be injected to network nodes. In this context, Lo et al. [17] addressed a geometrical framework for the partial controllability issue of networks by solving an integer linear programme. The approach was also suitable to optimize the complete controllability of networks. The network permeability index provided a quantitative understanding of the challenge of controlling a network partially or completely.

The dynamics of the network could be expressed as a linear time-invariant system $\dot{x}(t) = Ax(t) + Bu(t)$, where x(t) is the state vector of the network. Assuming a network has N state nodes and P control nodes, $A \epsilon R^{N\times N}$ is the coupled matrix between state nodes, u(t) is the control or input vector forced on the network, and $B \epsilon R^{N\times P}$ is the input matrix. The general approach for the controllability problem $\dot{x}(t) = Ax(t) + Bu(t)$ is to determine a proper input matrix based on the Kalman rank condition such that the pair (A, B) is controllable [10]. However, this controllability problem has a large computational load with 2^N possibilities assuming each node can be either driven or not driven [11], and this exponential growth is especially rapid when the network size is large. To overcome this difficulty, Liu et al. [10] introduced the structural controllability concept [18], which ensured that the Kalman rank condition was verified. They first found that the number of driver nodes for the full controllability of a complex network mainly depended on the network degree distribution. The process controllability of network dynamics was explored by transforming node dynamics into edge switch dynamics [19] and resulted in similar controllability conclusions to those obtained by Liu et al. [10].

The structural controllability methods based on graphical analysis of pair (A, B) for the system $\dot{x}(t) = Ax(t) + Bu(t)$ [18] could identify n_D for arbitrary directed networks [10]. Several effective methods have been proposed to identify the minimum number of driver (control) nodes (n_D), for example, the maximum matching (MM) method [20], the cavity method [21], and an extremal optimization (EO) algorithm [22]. The computational load of determining n_D could be effectively reduced based on the MM method, which has the computational complexity of $O(\sqrt{N}L)$, where L is the number of linked edges between state nodes. EO [22] was proposed based on the Kalman rank condition to identify n_D for the full controllability of directed networks with the computational complexity of $O(N^4P^3)$. However, this structural controllability framework [10] is not applicable to undirected networks for the symmetric characteristic of the network matrix or networks with exact link weights [11, 23, 24]. These limitations prompt the development of exact network controllability theory, which is an exact controllability framework for the controllability of complex networks with arbitrary network structures and link weights. It optimizes the complete controllability of networks based on the Popov–Belevitch–Hautus (PBH) rank condition [25], which is an alternative criterion that is equivalent to the Kalman rank condition [26]. The PBH controllability method requires the sequential computation of the eigenvalues of the N × N matrix A and the rank of the N × (N + P) PBH matrix. The computational complexity of the eigenvalue computation of matrix A and the PBH matrix rank is $O(N^3)$, $O((N+P)^3)$, respectively [27]. Thus, the computational complexity of the PBH method is $O((N+P)^3)$. The Kalman controllability method does not require an eigenvalue computation. However, it requires the rank computation of the N × NP Kalman matrix with the computational complexity of $O(N^3P^3)$, which is larger than that of the PBH controllability method. Representative exact structural controllability methods consist of a maximum multiplicity theory (MMT) [11] and an effective self-adaptive genetic algorithm (GA) [28]. n_D was computed based on the MMT to be equal to the maximum geometric multiplicity of all eigenvalues of the network [11], and the computational complexity is O

($N^2(\log N)^2$). The GA [28] was studied to identify n_D of arbitrary networks, and the computational complexity is $O(2m \times (N + P)^3 \times l)$, where 2m is the population size and l is the number of different eigenvalues of the controlled network.

The authors demonstrated the evolution process of network topology of two networks (Erdős–Rényi (ER) and scale free (SF)) [28], i.e., the dynamic change of network topology with different number of control nodes being injected to network state nodes. The GA algorithm [28] exceeded the EO [22] much in terms of the convergence speed and iterations. However, networks with the more complexity than the scale 150 were not studied in [28] and the maximum network scale that EO processed was 200 [22]. And the convergence speed and iterations were still not satisfactory, for example, for ER, with both 150 state nodes and 150 control nodes, and the average degree 5.0, n_D converged at the 101st generation after 398.03 s [28]. The results showed that it remained a challenge for the two algorithms to optimize networks with hundreds of thousands or even larger networks. Additionally, almost all real networks have small-world (SW) properties with a large cluster coefficient and short average distance [29] (e.g., power grids, transportation networks, and social networks). The addition of the SW network controllability study is also significant for better mimicking reality.

Therefore, based on the PBH rank condition, we propose a parallel quantum genetic algorithm (PAQGA) to more rapidly determine the minimum number of control nodes. The proposed algorithm is suitable for arbitrary networks that comprise both control nodes and state nodes. The simulation results for a series of benchmark networks demonstrate the effectiveness of the algorithm. Furthermore, we demonstrate the relationship between the controllability of a network and its network properties such as network average degree, degree heterogeneity, power-law index, and clustering coefficient.

The remainder of this paper is organized as follows: In Section 2, we provide a description of the issue in which a network can be controlled through a small amount of control nodes. In Section 3, we introduce the PAQGA for the solution of the minimum number of control nodes to exactly control arbitrary networks. In Section 4, we analyze and discuss the performance and experimental results of the proposed framework by studying popular ER, SW, SF, and some real-life networks. We draw conclusions and suggest future work in Section 5.

Problem definition

In this paper, we provide a descriptive definition of the entire controllability problem of a directed weighted network.

Definition 2.1 [22]. A network that contains P control nodes and N state nodes can be expressed as a triple tuple G = (V,E,W), where $V = V_s \cup V_c$, $V_s = \{v_1,v_2,\ldots,v_N\} = \{x_1,x_2,\ldots,x_N\}$ is the set of state nodes, and $V_c = \{v_{N+1},v_{N+2},\ldots,v_{N+P}\} = \{u_1,u_2,\ldots,u_P\}$ is the set of control nodes; $E = E_s \cup E_c$, $E_s \in V_s \times V_s$ is the set of the linked edges between state nodes, and $E_c \in V_c \times V_s$ is the set of linked edges between control nodes and state nodes, where each state node can only be connected to one control node; and $W \in R^{(N+P)\times(N+P)}$ is the set of edge weights, $w_{ij} = 0$ if there is not a link between v_i and v_j; otherwise, w_{ij} (w_{ji}) represents the strength that v_i (v_j) could affect v_j (v_i), w_{ij} (w_{ji}) > 0 if the direction is i → j (j → i). Fig 1 shows an illustration of the definition.

Remark 2.1 [22]. The set W can be expressed by the representation of a block matrix that contains A and B as follows:

$$W = \begin{bmatrix} A & B \\ 0 & 0 \end{bmatrix}, \tag{1}$$

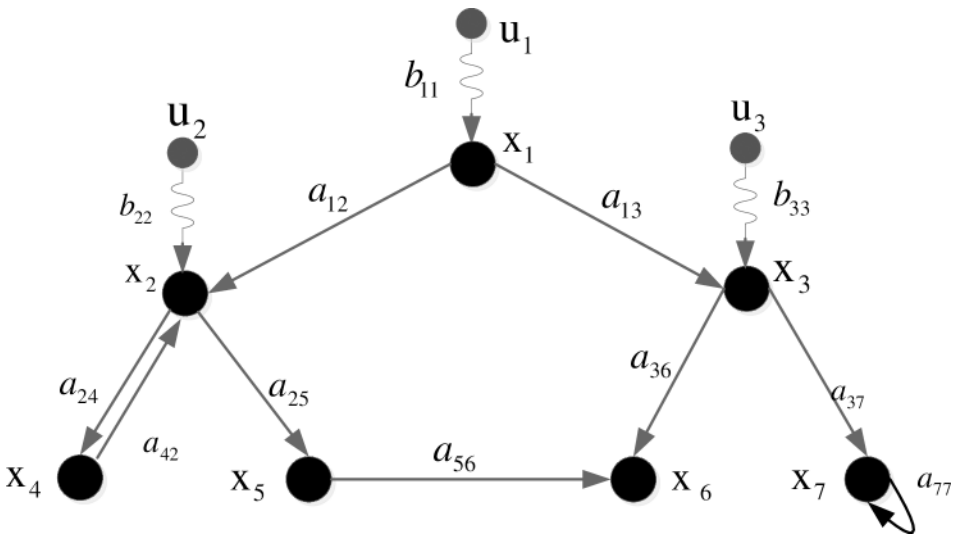


Fig 1. Example of a directed network, where $V_s = \{x_1,x_2,x_3,x_4,x_5,x_6,x_7\}$, $V_c = \{u_1,u_2,u_3\}$, $E_s = \{(x_1,x_2),(x_1,x_3),(x_2,x_4),(x_2,x_5),(x_3,x_6),(x_3,x_7),(x_4,x_2),(x_5,x_6),(x_7,x_7)\}$, $E_c = \{(u_1, x_1),(u_2, x_2),(u_3, x_3)\}$, a_{ij} $(i = 1,2,\ldots,7; j = 1,2,\ldots,7) \in V_s \times V_s$, $b_{ij}(i = 1,2,3; j = 1,2,\ldots,7) \in V_c \times V_s$., $w_{24} > 0$ and $w_{42} > 0$.

https://doi.org/10.1371/journal.pone.0193827.g001

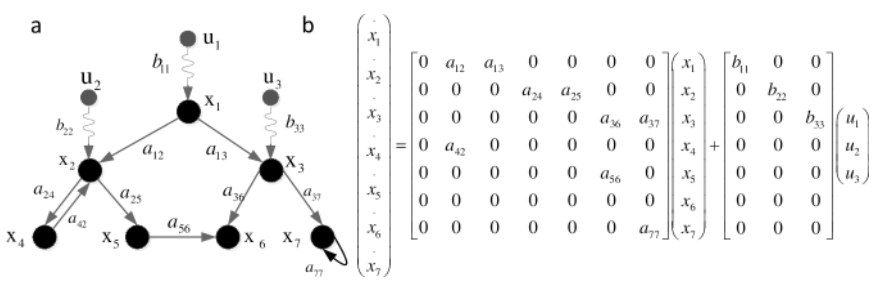
Definition 2.2 [10, 22]. The network G = (V,E,W) can be represented by

$$\dot{x}(t) = Ax(t) + Bu(t), \tag{2}$$

where $\mathbf{x}(t) = (x_1(t),x_2(t),\ldots,x_N(t))^T$ is the state vector of the network, $A \epsilon R^{N\times N}$ is the coupled matrix between state nodes, $u(t) = (u_1(t),u_2(t),\ldots,u_P(t))^T$ is the control or input vector forced on the network, $B \epsilon R^{N\times P}$ is the input matrix, and $B = \{b_{ij}\}$, b_{ij} is the weight of a directed link that the input signal u_j(j = 1,2,. . .,P) points to the network state node x_i (i = 1,2,. . .,N). For simplicity, hereafter the time symbol (t) will be omitted. Fig 2 shows the equation expression of the network in Fig 1.

Definition 2.3 [22]. A control scheme D of a network G = (V,E,W) is determined by the selected control nodes with definite number and their acting position. D could be represented by a binary diagonal matrix as $D = \mathrm{diag}\{d_1,d_2,\ldots,d_P\}$, where d_i(i = 1,2,. . .,P) is a variable of value zero or one, and $d_i = 1$ means that the control node u_i is chosen to be a component of the network control strategy; otherwise, u_i is removed together with its associated links.

Remark 2.2 [22]. Based on Definition 2.3, a novel control scheme D^* is determined for which a different set of control nodes is selected. Then a novel network topology is generated



$$\begin{pmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \\ \dot{x}_5 \\ \dot{x}_6 \\ \dot{x}_7 \end{pmatrix} = \begin{bmatrix} 0 & a_{12} & a_{13} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & a_{24} & a_{25} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & a_{36} & a_{37} \\ 0 & a_{42} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & a_{56} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & a_{77} \end{bmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{pmatrix} + \begin{bmatrix} b_{11} & 0 & 0 \\ 0 & b_{22} & 0 \\ 0 & 0 & b_{33} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix}$$

Fig 2. Corresponding dynamics equation of the network in Fig 1, where $x = \{x_1, x_2, x_3,x_4,x_5,x_6,x_7\}$, $u = \{u_1,u_2,u_3\}$, a_{ij} $(i = 1,2,\ldots,7; j = 1,2,\ldots,7)$ is the connection weight from state node i to state node j, and $b_{ij}(i = 1,2,3; j = 1,2,\ldots,7)$ is the connection weight from control node i to state node j.

https://doi.org/10.1371/journal.pone.0193827.g002

as $G^* = (V^*,E^*,W^*)$, where $V^* \in V$, $E^* \in E$, and $W^* \in R^{(N+r)\times(N+r)}$, where r is the number of selected control nodes. Accordingly, the network dynamics are also changed as

$$\dot{x} = Ax + B^* u^*, \tag{3}$$

where $B^* \in R^{N\times r}$ is the new input matrix that represents the connections between new chosen control nodes and network state nodes, and $u^* \in R^r$ is a time-variable input vector that contains r control nodes.

Definition 2.4 [22]. $M \epsilon R^{P\times r}$ is the index set of the selected control nodes, $M = \{m_{ij}\}$, and $m_{ij} = 1$ means that the j_{th} chosen control node is u_i, $i = 1,2,\ldots,P$, $j = 1,2,\ldots,r$, $r \leq P$.

Remark 2.3 [22]. M is constructed by the nonzero columns of the control scheme D^*. For example, if $u^* = \{u_1,u_2,u_3\}$ is chosen from a previous control node set $u = \{u_1,u_2,u_3, u_4\}$ to be a new control scheme $D^* = diag\{1,1,1,0\}$, then M is obtained from this D^* as

$$\mathrm{M} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}, \tag{4}$$

Remark 2.4 [22]. The evolving input matrix and input vector can be revised as $B^* = BD^*\mathrm{M}$ and $u^* = \mathrm{M}^{\mathrm{T}}u$, respectively. Then Eq (3) can be rewritten as

$$\dot{\mathrm{x}} = \mathrm{Ax} + \mathrm{BD}^*\mathrm{M}(\mathrm{M}^{\mathrm{T}}\mathrm{u}), \tag{5}$$

Fig 3 shows a simple case that illustrates how a control scheme influences the network topology.

Remark 2.5. Based on the PBH rank condition [25], the network G = (V,E,W) (Eq. (2)) can be steered to any desired state within a finite time, that is, G is fully controllable if and only if

$$rank(\lambda_i I_N - A, B) - N = 0, \tag{6}$$

and the new system (Eq. (5)) is fully controlled if and only if

$$rank(\lambda_i I_N - A, \mathrm{BD}^*\mathrm{M}) - N = 0, \tag{7}$$

is satisfied for each different eigenvalue λ_i of the state matrix A, where $I_N \in R^{N\times N}$ is an identity matrix.

For an arbitrary network G = (V,E,W), our purpose is to determine the minimum control nodes to guarantee its full control. Based on the above analysis, the controllability problem can be transformed into a single target restricted optimization problem as

$$\min_D \sum_{j=1}^{P} d_j, \tag{8}$$

subject to

$$rank(\lambda_i I_N - A, BDM) - N = 0, \forall \lambda_i \in eig(A), \tag{9}$$

$$d_j = \{0,1\}, j = 1,2,\ldots,P, \tag{10}$$

where A is the state matrix, B is the original input matrix, D is the original control scheme, M is the indicator matrix that is derived from the nonzero column of D, N and P are the dimensions of A and B, respectively, λ_i is the eigenvalue that belongs to A, $eig(A)$ is the set of different eigenvalues of A, d_j is the element of D, and $d_j = 1$ when u_j is selected and $d_j = 0$ otherwise.

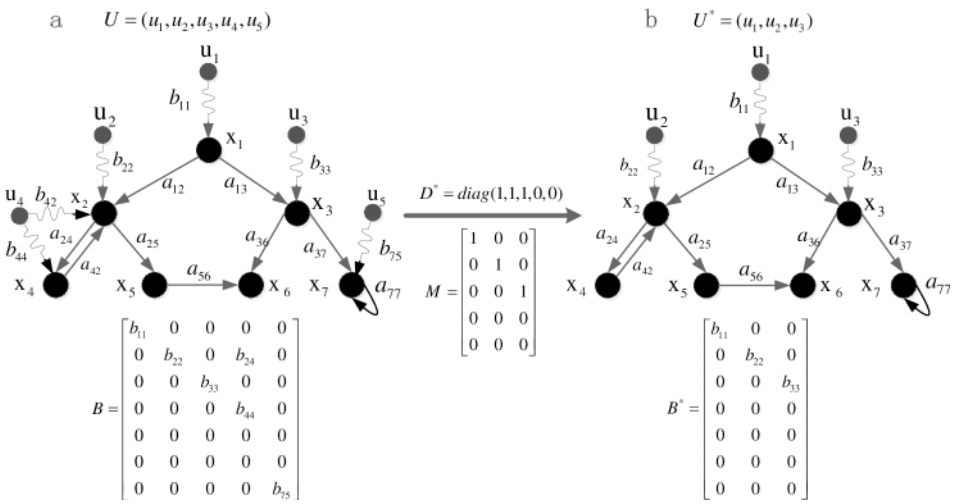


Fig 3. Illustration of how new control scheme D* functions in the network topology. (a) Original network with seven state nodes and five candidate control nodes. (b) Input matrix changes into $B^* = BD^*M$ after choosing $u^* = \{u_1, u_2, u_3\}$ as new control nodes. New network has seven state nodes and three control nodes.

https://doi.org/10.1371/journal.pone.0193827.g003

Solution framework

Overview of the QGA

GA is a global optimization method that can optimize problems with multiple parameters to reach near the global optima [30–33]. However, in some practical applications, it often requires multiple iterations because of the slow convergence speed and prematurity features, and easily falls into the local minima [34, 35]. Additionally, for many complex problems, a large population is required to obtain the optimal solution. The convergence of GA mainly depends on the selecting operation, which largely affects the convergence speed. Additionally, its searching capability mainly relies on crossover and mutation operations, which primarily affect the occurrence of the premature phenomenon. Therefore, regarding enhancing GA search performance, the approach used to choose suitable selecting, crossover, and mutation strategies has been always an urgent and pivotal issue in the study and application of GA [33, 36].

For small and medium-sized applications, the solution could be achieved within a tolerance range using GA. However, a gene (typically encoded with a 0–1 string) in a GA chromosome typically delivers certain information, which limits the population diversity. It performs worse in multivariate issues, for example, the controllability study of complex networks, which mostly has complex structures, and large-size nodes and links.

Combining quantum computing and GA, and adopting qubits as the representation of chromosome genes [37], QGA is a proper intelligent optimization algorithm for solving the network controllability problem [38]. These QGA qubits cover all possibilities for the linear superposition property of quantum information, which could reduce the algorithm's complexity and promote the achievement of the optimal solution under a smaller population [37].

In quantum computing, $|0\rangle$ and $|1\rangle$ signify two basic states of microscopic particles. According to the principle of the superposition property, the superposition state of quantum information could be the linear combination of the two basic states [39], which can be written as

$$|\varphi\rangle = \alpha|0\rangle + \beta|1\rangle,\ |\alpha|^2 + |\beta|^2 = 1, \tag{11}$$

where α and β are the state probability amplitudes of a qubit, and α^2 and β^2 are the probability that a qubit changes to be state $|0\rangle$ and state $|1\rangle$, respectively. One qubit also can be expressed as $\begin{bmatrix} \alpha \\ \beta \end{bmatrix}$.

Assume the number of optimization variables is n and the population size is 2m. The i_{th} chromosome is denoted by G_i(i = 1,2,. . .,m) as

$$G_i = \begin{bmatrix} \alpha_{i1} & \alpha_{i2} & \dots & \alpha_{in} \\ \beta_{i1} & \beta_{i2} & \beta_{in} & \end{bmatrix}, \tag{12}$$

where $\alpha_{i1}^2 + \alpha_{i2}^2 + \beta_{i1}^2 + \beta_{i2}^2 = 1, i = 1, 2, \dots, m$. G_i contains two parallel gene chains or individuals (α_{i1},α_{i2},. . .,α_{in} and β_{i1},β_{i2},. . .,β_{in}). Each individual is a candidate solution of an optimization problem:

$$G_i = \begin{bmatrix} G_{i1} \\ G_{i2} \end{bmatrix}, \quad \begin{cases} G_{i1} = [\alpha_{i1},\ \alpha_{i2}, .., \alpha_{ij}, .., \alpha_{in}] \\ G_{i2} = [\beta_{i1},\ \beta_{i2}, \dots, \beta_{ij}, .., \beta_{in}] \end{cases}, \tag{13}$$

QGA and enhanced QGA have already been studied to optimize many combinational problems [40–42]. For example, QGA overmatches classic GA with less complexity and higher performance in 0–1 combinational optimization problems [39]. Adaptive QGA models were proposed and tested on classical combinational problems, such as knapsack, maxcut and onemax [38], the multi-aircraft cooperative target allocation problem, and constrained engineering design problems [43]. However, the time efficiency was not seriously stressed. To increase the speed, a parallel QGA was developed and effectively applied to a knapsack problem [44]. It divided the entire population into subpopulations on different parallel processors and used the migration rate for the information exchange of these subpopulations. However, the Q-gate rotation was implemented according to a fixed lookup table, which did not take full advantage of the dynamic differences between individuals during the iterating process.

Inspired by current achievements, to quickly and efficiently solve the controllability problem of complex networks, we investigated a PAQGA scheme, in which: 1) partial programs of the algorithm are executed in parallel; 2) a set of adaptive Q-gate rotation rules are proposed and adaptive crossover operation are used; and 3) population catastrophe is implemented to accelerate convergence.

Workflow of the PAQGA for network controllability

Based on the above, each control scheme *D* is a diagonal matrix, whose elements on the primary diagonal are either zero or one. Therefore, we adopt the binary mechanism to encode the algorithm chromosome G_i(i = 1,2,. . .,m), and each binary gene chain can represent a control scheme, as shown in Fig 4.

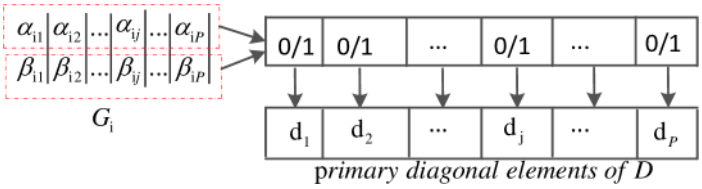


Fig 4. Chromosome encoding in quantum genetic algorithm.

https://doi.org/10.1371/journal.pone.0193827.g004

To apply PAQGA conveniently, a penalty term $Pen_i(D)$ is defined to convert the optimization problem described in Eqs (8), (9) and (10) into an unconstrained optimization problem. $Pen_i(D)$ is used to evaluate the i_{th} perturbation for a specific control scheme D:

$$Pen_i(D) = \sigma_i \times (rank(\lambda_i I_N - A, BDM) - N)^2, i = 1, 2, \ldots, l, \tag{14}$$

where σ_i is the penalty coefficient defined as $\sigma_i = \{c\sigma_{i-1}\}$ ($\sigma_1 = 10P$, $c > 0$) is a strictly increasing positive sequence to reduce the calculation burden of minimizing the penalty function, and l is the total number of distinct eigenvalues of A. The overall penalty of D is the sum of $Pen_i(D)$ expressed as

$$Pen(D) = \sum_{i=1}^{l} Pen_i(D), \tag{15}$$

According to the PBH rank condition, when the network G = (V,E,W) is fully controllable, $Pen(D)$ should be zero and vice versa. Therefore, the fitness function can be achieved by merging the penalty term into the optimization Eqs (8), (9) and (10):

$$\begin{aligned} \mathrm{f(D)} &= \sum_{j=1}^{P} d_j + Pen(D) \\ &= \sum_{j=1}^{P} d_j + \sum_{i=1}^{l} \sigma_i \times (rank(\lambda_i I_N - A, BDM) - N)^2, \end{aligned} \tag{16}$$

For the optimization problem, our objective is to minimize f(D) based on the 0–1 integer values of d_j (j = 1,2,. . .,P). Fig 5 shows the fitness evaluation of different control schemes on a simple network.

After defining the chromosome representation and fitness function, the network controllability problem can be optimized using the following steps. Fig 6 shows the flow chart of the proposed PAQGA for the optimization problem.

Step 1: The population at the t_{th} generation is denoted as $\mathrm{Q(t)} = \{G_1^t, G_2^t, \ldots, G_m^t\}$,

$$G_i^t = \begin{bmatrix} \alpha_{i1}^t & \alpha_{i2}^t & \ldots & \alpha_{iP}^t \\ \beta_{i1}^t & \beta_{i2}^t & \beta_{iP}^t & \end{bmatrix}, i = 1, 2, \ldots, m, t = 0, 2, \ldots, maxgen - 1, \tag{17}$$

where N is the number of qubits, that is, the number of network state nodes, and *maxgen* is the maximum iterating generation.

Initialize the initial population as

$$\mathrm{Q}(t_0) = \{G_1^0, G_2^0, \ldots, G_m^0\}, \tag{18}$$

where $G_i^0 = \begin{bmatrix} \alpha_{i1}^0 & \alpha_{i2}^0 & \ldots & \alpha_{iP}^0 \\ \beta_{i1}^0 & \beta_{i2}^0 & \beta_{iP}^0 & \end{bmatrix}, i = 1, 2, \ldots, m.$

All the quantum states (α_{ik} and β_{ik}) in the PAQGA are initialized in parallel with the value $\frac{1}{\sqrt{2}}$, i = 1,2,. . .,m, k = 1,2,. . .,N. Additionally, set $D_{best} = D_0$, the iterative generation t = 1, and $\sigma_1 = 10P$.

When we set the initial control scheme $D_0 = diag\{d_j = 1\}$,j = 1,2,. . .,P, the initial best fitness value is $f(D_{best}) = P + 0 = P$. It is easily proved that with $\sigma_i = \{c\sigma_{i-1}\}$ ($\sigma_1 = 10P$, $c > 0$), the fitness $f(D) \leq P$ if and only if the control scheme D always satisfies the constraint Eq (9). With the initialization $f(D_{best}) = P$, whenever D_{best} is updated by D_i, we have $f(D_i) < f(D_{best}) = P$, which means D_i meets the constraint Eq (9). Thus, D_{best} always evolves in the feasible region that makes the network entirely controllable.

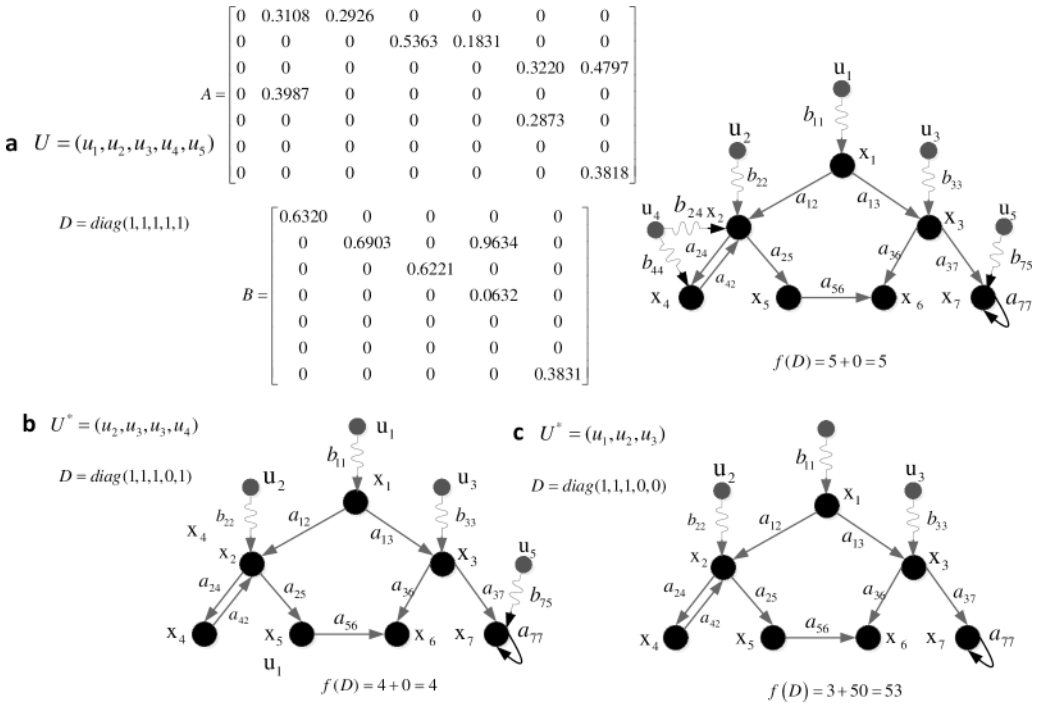


Fig 5. Illustration of the fitness evaluation for different control schemes on a directed weighted network with self-loop. (a) Initial network with seven state nodes and five control nodes. Connecting weights are randomly assigned between zero and one. Fitness of initial D = diag{1,1,1,1,1} is f(D) = 5 and the penalty term is zero; thus, the network is entirely controllable. (b) When D changes to new D = diag{1,1,1,0,1}, f(D) is four, the penalty is zero, and the network is still fully controllable. (c) u_4 is removed from (b). For simplicity, c is set to 1, and the penalty term $\mathrm{pen(D)} = 10P * \sum_{i=1}^{l} Pen_i(D) = 50 * 1 \neq 0$ indicates that the network with this topology cannot be fully controlled.

https://doi.org/10.1371/journal.pone.0193827.g005

Step 2: Observe the qubits of each individual of $Q(t_0)$ in parallel following the rules in Section 3.3 and obtain the binary strings.

Step 3: Evaluate each individual of $Q(t_0)$ in parallel and save the optimal individual as the evolving goal in the next generation.

Step 4:

While (*maxgen* is not reached), do the following:

(a) Observe the qubit value of each individual of Q(t) in parallel following the rules in Section 3.3.

(b) To increase the diversity of the population and inherit the excellent genes from the previous population, the adaptive crossover operation is performed in parallel in accordance with Section 3.4.

(c) Evaluate each individual of Q(t) in parallel and store the optimal individual as the evolving goal in the next generation.

(d) Perform the Q-gate rotation operation in parallel and obtain the offspring population Q (t+1).

For each individual, two parallel gene chains update simultaneously. Rotation angle θ is first computed based on Section 3.5 and then the qubits are updated by Q-gate rotation. The

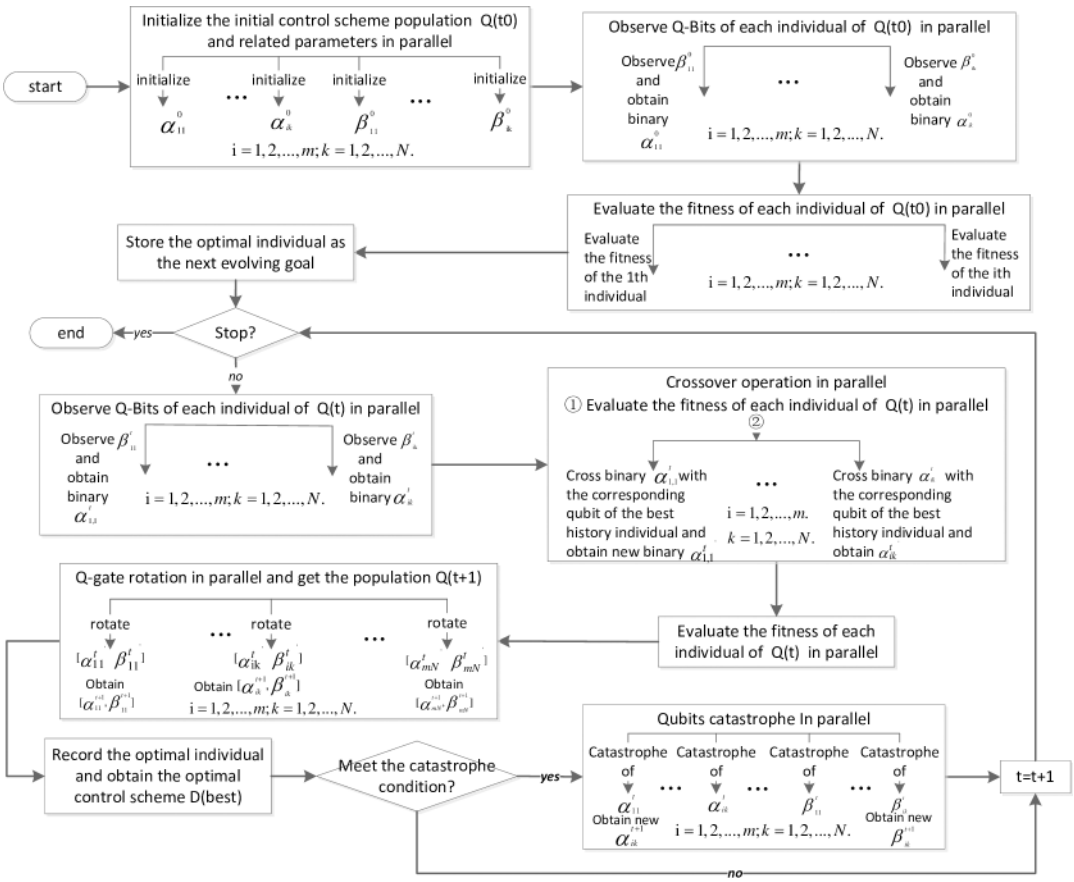


Fig 6. Flow chart of the PAQGA.

https://doi.org/10.1371/journal.pone.0193827.g006

Q-gate is expressed as [45]

$$\text{Q}-\text{gate} = \begin{bmatrix} cos\theta & -sin\theta \\ sin\theta & cos\theta \end{bmatrix}, \tag{19}$$

The Q-gate rotation operation is

$$\begin{cases} \alpha_{ik}^{t+1} = cos\theta * \alpha_{ik}^{t\,\prime} - sin\theta * \beta_{ik}^{t\,\prime} \\ \beta_{ik}^{t+1} = sin\theta * \alpha_{ik}^{t\,\prime} + cos\theta * \beta_{ik}^{t\,\prime} \end{cases} \tag{20}$$

where $i = 1,2,\ldots,m$, $k = 1,2,\ldots,N$, $t = 1,2,\ldots,maxgen - 1$, and $\theta = \delta * s$, where δ is the rotation angle value and s is its sign.

(e) Record D_{best} and $f(D_{best})$.

(f) If the optimal values of the past several successive generations are the same, then perform the parallel population catastrophe operation.

The fitness function evaluation and the crossover operation are the two most time-consuming steps in the process of the flow execution. Assume that η parallel processors are used, the cost is $O\left(\frac{m\times(N+p)^3\times l}{\eta}\right)$ and $O\left(\frac{m\times(N+p)^3\times l}{\eta^2}\right)$, respectively, where l is the number of different

eigenvalues of the controlled network. Therefore, the computational complexity of the PAQGA is $O\left(\frac{m\times(N+p)^3\times I}{\eta}\right)$.

Observing operation

Each qubit of the chromosome can be adjusted to be at a stationary state using an observation operation. We adopt a random observing method by running the following pseudocode in parallel:

```
start
      if(rank(k) ≥ (β_ik^t)^2), i = 1,2,...,m; k = 1,2,...,N
         return binary α_ik^t = 1;
      else
         return binary α_ik^t = 0;
end
```

where rand(k) is a random digit. If rand(k) is not less than $(\beta_{ik}^t)^2$ (the probability to be state $|1\rangle$), then the observed value of the qubit α_{ik}^t is 1 and 0 otherwise.

Crossover operation

The crossover operator is an important operation of GA. Information about individuals can be exchanged using the operation. Subsequently, excellent genes could be reserved for population evolution to move in a better direction. To increase the diversity of the population and improve the optimization performance of PAQGA, the crossover operator is introduced. We obtain novel binary values α'_{ik} by crossing each binary qubit value α_{ik},i = 1,2,. . .,m, k = 1,2,. . .N with corresponding information on the historically best control scheme, that is, $D_{best}(k,k)$, based on a certain crossover probability. The specific crossover mode is

$$\alpha'_{ik} = \begin{cases} D_{best}(k,k), \ if \ rand(k) < p_c \\ \alpha_{ik}, \ otherwise, \ i = 1,2,\ldots,m; k = 1,2,\ldots,N, \end{cases} \quad (21)$$

where i is the i_{th} individual, $rand(k)$ is a random number between [0, 1], and p_c is the crossover probability. Fig 7 shows a simple crossover example with 10 qubits to explain the rule (21).

In the early days of population evolution, there existed relatively big differences between individuals. Therefore, the crossover possibility to produce better offspring should have been bigger. Moreover, if we increased the crossover probability at this time, the evolution process would have been accelerated. By contrast, in the late stages of evolution, differences between individuals became smaller as the best solution was approaching. The crossover probability should have been correspondingly diminished to reserve the good genes. We design an adaptive crossover operator as

$$p_c(i) = \begin{cases} \frac{m}{Q_i} p_{c0} \exp\left(-\frac{|f_{max} - f(G_i)|}{f_{max} - f_{min}}\right), \ f_{max} \neq f_{min} \\ \frac{m}{Q_i} p_{c0}, \ f_{max} = f_{min} \end{cases}, \ i = 1,2,\ldots,m, \quad (22)$$

where $p_c(i)$ is the crossover probability of the i_{th} current individual, Q_i is the number of those individuals whose fitness is better than that of the historically best individual, p_{c0} is the initial crossover probability, f_{max} and f_{min} are the previous worst fitness and best fitness, respectively, and $f(G_i)$ is the fitness of the i_{th} current individual.

We can observe that $p_c(i)$ becomes bigger when the control scheme G_i becomes worse and vice versa. Moreover, $p_c(i)$ is inversely proportional to Q_i, which means that if there are not so

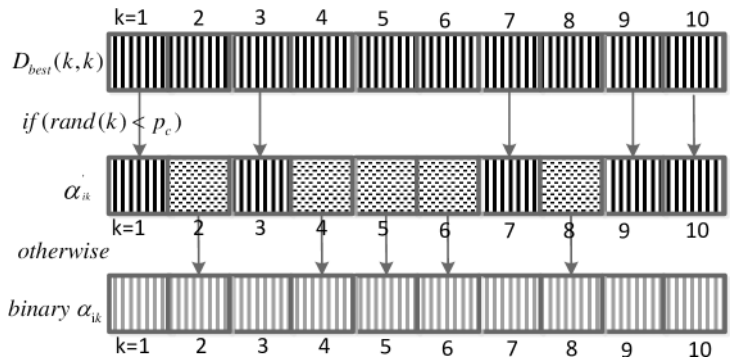


Fig 7. Simple crossover example.

https://doi.org/10.1371/journal.pone.0193827.g007

many good individuals, $p_c(i)$ should be bigger to produce a greater number of better individuals; otherwise, it should be smaller because the evolving individuals are becoming better. The improved adaptive crossover operation from Eq (20) is

$$\alpha'_{ik} = \begin{cases} D_{best}(k,k), & if\ rand(k) < p_c(i) \\ \alpha_{ik}, & otherwise \end{cases}, i = 1, 2, \ldots, m; k = 1, 2, \ldots N, \tag{23}$$

A better population is determined after the crossover operation following the pseudocode.

```
start
     obtain fitness(i) in parallel;
     find f(max) and f(min);
     obtain pc(i) according to formula (22) in parallel;
     obtain new binary population;
end
```

where fitness(i) is the fitness value of the ith individual.

Rotation angle updating rules

Learning from the solid lookup rules [39], we present a set of adaptive rotation angle updating rules in Table 1. The rotation angle θ_i ($\theta_i = \delta_i{}^* s(\alpha_i,\beta_i)$), $i = 1,2,\ldots,m$ dynamically varies according to the evolution process.

If $D_c(i) \neq D_{best}(i)$, the rotation angle δ_i is adaptively proportional to $\frac{f(D_c)}{f(D_{best})}$. If $f(D_c) < f(D_{best})$, the angle will be smaller; otherwise, it will be bigger. Initially, a big initial angle is set.

Table 1. Rotation angle updating rules.

$D_c(i)$	$D_{best}(i)$	$f(D_c) < f(D_{best})$	$s(\alpha_i,\beta_i)$				
			δ_i	$\alpha_i\beta_i>0$	$\alpha_i\beta_i<0$	$\alpha_i = 0$	$\beta_i = 0$
0	0	false	0	0	0	0	0
0	0	true	0	0	0	0	0
0	1	false	$\frac{f(D_c)}{f(D_{best})} \cdot 0.03\pi$	+1	−1	0	±1
0	1	true	$\frac{f(D_c)}{f(D_{best})} \cdot 0.01\pi$	−1	+1	±1	0
1	0	false	$\frac{f(D_c)}{f(D_{best})} \cdot 0.03\pi$	−1	+1	±1	0
1	0	true	$\frac{f(D_c)}{f(D_{best})} \cdot 0.01\pi$	+1	−1	0	±1
1	1	false	0	0	0	0	0
1	1	true	0	0	0	0	0

https://doi.org/10.1371/journal.pone.0193827.t001

As the iteration proceeds, the differences between individuals decrease and δ_i becomes smaller. In this way, the probability amplitude evolves in the direction of the optimal solution.

Population catastrophe

When the best individuals in several successive generations are identical, it shows that the algorithm falls into a local minimum. At this moment, catastrophe operations for the current population should be performed to take it out of the constraint and start a new search. Specifically, the successive best individual is retained in the new population Q(t + 1) and the remaining individuals in Q(t + 1) are regenerated as a large disturbance. The pseudocode of the catastrophe operation is as follows:

```
start
     obtain the best individual corresponding to the optimal fitness;
     keep this best individual;
     rebuild the rest in parallel;
end
```

The strategy would prefer that the population eliminate its dull state rather than make it degenerate, which is an effective means to commence a new search.

Simulations and analyses

We used the orthodox ER random [46], SF [47], SW networks of NW type [48] and some real-world networks as benchmarks to illustrate the feasibility of the PAQGA for optimizing the controllability of arbitrary networks that encompass control nodes and state nodes. Additionally, we also conducted an analysis of the relationship between the network topology and number of control nodes. ER and SF networks were obtained from the static model [49] with N state nodes and P candidate control nodes ($N = P$). Each control node pointed to state nodes with uniform probability and the weights of all edges were randomized between zero and one. SW networks were generated from randomized adding edges [48, 49]. The characteristics of random regular networks, ER, SF, and SW networks are illustrated in Fig 8.

We define the number of selected control nodes that correspond to the current best control scheme as n_c and the density of these selected control nodes as N_c, where $N_c = n_c/N$. The minimum number of selected control nodes after the optimization process is denoted as n_{cm}, and the minimum control node density is $N_{cm} = n_{cm}/N$. To implement the parallel strategy, we performed the following simulations on eight MATLAB® workers. The parameters of the PAQGA were set to 2m = 30, maxgen = 100, and $p_{c0} = 0.06$. All the following experimental results are the average of 10 independent simulations and the standard deviation is 0.01.

Performance of the PAQGA

To show that the optimal solution (D_{best}) at each generation always evolves in the feasible region, that is, $Pen_i(D_{best}) = 0$, we conducted experiments on different networks. All these networks were directed with 100 state nodes and 100 candidate control nodes. The experimental results are shown in Fig 9. The figure shows that the best fitness quickly converged to the minimum value after approximately the first few generations. The mean current fitness fluctuated dramatically because of the operations of qubit cross, qubit catastrophe, and Q-gate rotation. The penalty was always equal to zero, which means that D_{best} always met the PBH rank condition throughout the entire optimization process.

We compare the performance of PAQGA of optimizing network controllability with that of EO [22] and adaptive GA [28] on a list of popular networks and real-life networks. The comparison results are tabulated in Table 2.

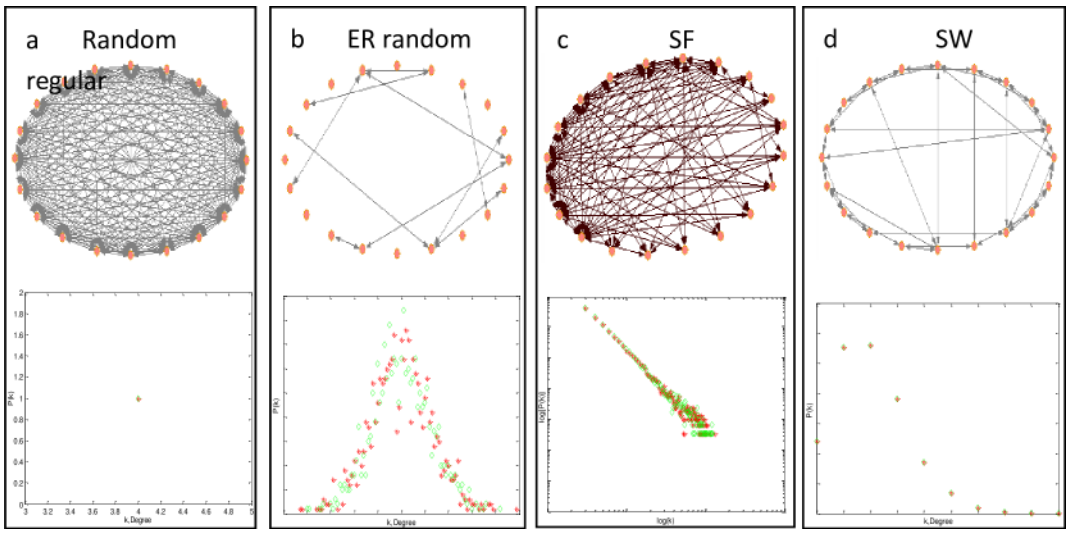


Fig 8. Characteristics of the addressed networks. Red stars represent the node in-degree denoted by $\langle k_{in} \rangle$ and the green diamonds represent the node out-degree denoted by $\langle k_{out} \rangle$. (a) Random regular networks with homogeneous degree distribution of $\langle k_{in} \rangle = \langle k_{out} \rangle = 4$. (b) ER random networks with Poisson degree distribution; the degree heterogeneities rely on the average degree denoted by $\langle k \rangle$. (c) SF networks with power-law degree distribution, which results in large degree heterogeneities. (d) SW networks with long-tail degree distribution, which decreases much slower than the SF distribution.

https://doi.org/10.1371/journal.pone.0193827.g008

From the columns of n_{cm}, it can be observed that the three algorithms almost converged to the same value, which demonstrates that PAQGA, GA, and EO all had a good ability to determine the optimal control nodes. When the size of the network was small (e.g., $N = P \leq 50$),

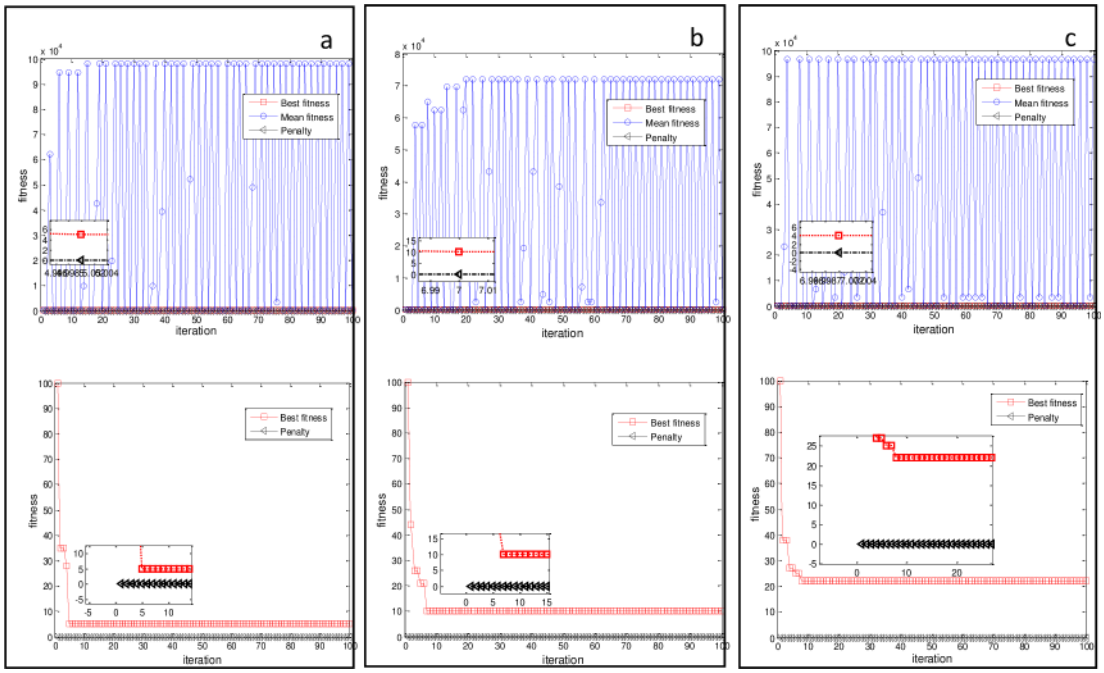


Fig 9. Fitness and penalty curves as a function of iterating generation for (a) ER with $\langle k \rangle = 4.0$, (b) SF with $\langle k \rangle = 4.0$ and $\gamma = 2.1$, and (c) SW with $\langle k \rangle = 4.0$. The red dotted line with a square is the best fitness corresponding to D_{best} at the current generation, the blue dashed line with a circle is the mean fitness of all control schemes at each generation, and the black line with a triangle is the penalty term corresponding to D_{best} at each generation.

https://doi.org/10.1371/journal.pone.0193827.g009

Table 2. Performance comparison of PAQGA, GA, and EO on different networks in terms of n_{cm}, the minimum iterating generations, and computational time. Power-law index of SF networks in these experiments was γ = 2.1. '/' indicates that the corresponding results were not available for the computational time limit. For data sources, see Supplementary information S1 Dataset.

network	N/P	<k>	PAQGA			Adaptive GA [28]			EO [22]		
			ncm	iterations	time (s)	ncm	iterations	time (s)	ncm	iterations	time (s)
ER	25	1.5	2	3	1.86	2	30	0.23	3	22	0.45
ER	50	3	3	4	4.03	3	34	4.19	3	22	7.79
ER	100	4	5	5	7.38	5	75	266.26	6	24	121.62
ER	200	6	31	5	12.49	31	128	873.05	31	29	1545.84
ER	300	8	22	11	18.31	25	44	1708.81	/	/	/
ER	500	10	43	17	31.82	/	/	/	/	/	/
ER	1000	16	64	26	65.05	/	/	/	/	/	/
SF	25	1.5	3	5	2.36	3	20	0.43	4	22	0.75
SF	50	3	5	6	5.38	5	32	4.83	5	21	8.37
SF	100	4	10	7	8.22	10	69	278.55	10	21	134.54
SF	200	6	35	5	13.06	35	116	892.16	36	34	1623.13
SF	300	8	83	11	19.57	83	132	1823.12	/	/	/
SF	500	10	192	17	34.92	/	/	/	/	/	/
SF	1000	16	416	26	68.04	/	/	/	/	/	/
Rhode [50]	20	2.65	2	5	2.76	2	20	0.52	2	22	0.86
Maspalomas [50]	24	3.417	3	6	5.56	3	32	5.24	3	21	8.93
Michigan [50]	39	5.667	13	7	6.45	13	69	5.52	14	21	12.62
Circuit-s208 [51]	122	3.126	29	9	18.22	29	116	913.14	30	34	1745.83
Friend-rev [52]	228	4.01	52	10	20.45	52	121	1201.54	54	45	2733.61
Circuit-s420 [51]	252	3.21	59	13	23.65	59	132	1962.92	/	/	/
Circuit-s838 [51]	512	3.44	119	18	39.03	/	/	/	/	/	/
Roget [50]	1022	4.966	396	27	75.66	/	/	/	/	/	/

https://doi.org/10.1371/journal.pone.0193827.t002

PAQGA took slightly more time than GA and EO to determine the best solution. This paradoxical phenomenon is attributed to the launching of the MATLAB® distributed server, and the launching time was approximately 2s. However, once the server started, PAQGA showed a greater advantage in processing large-size networks over GA and EO. For example, for the ER network with $\langle k \rangle = 6.0$ and N = P = 200, PAQGA obtained n_{cm} at the fifth generation and took 12.49s; for the same network, GA took 873.05s at the 128th generation and, and EO required 29 generations and 1545.84 s. By comparing the computational time, PAQGA saved 98.57% more than GA and 99.19% more than EO.

A parallel version of GA was transformed from the adaptive GA [28] using η MATLAB® workers with the computational complexity of $O\left(\frac{2m\times(N+P)^3\times l}{\eta}\right)$, where 2m is the population size, l is the number of different eigenvalues of the controlled network. We compared it with the proposed PAQGA, and the results are shown in Table 3.

It can be inferred from Table 3 that the parallel computation (allowing for multiple processors) contributes to the performance of algorithms. However, it is not the only important factor. The computational efficiency of EO, GA, parallel GA and PAQGA could be reflected by their computation complexity. First, the computation of the PBH rank matrix in GA, parallel GA and PAGQA and the Kalman rank matrix in EO is the most time-consuming. This is the main factor affecting their speedability. The rank computation of Kalman matrix takes much

Table 3. Performance comparison of PAQGA and parallel GA on different networks using eight MATLAB® workers in terms of n_{cm}, the minimum iterating generations, and computational time. For data sources, see Supplementary information S1 Dataset.

network	N/P	<k>	PAQGA			parallel GA		
			ncm	iterations	time (s)	ncm	iterations	time (s)
ER	25	1.5	2	3	1.86	2	19	2.71
ER	50	3	3	4	4.03	3	28	5.19
ER	100	4	5	5	7.38	5	35	53.31
ER	300	8	22	11	18.31	25	37	89.54
SF	25	1.5	3	5	2.36	3	18	3.52
SF	50	3	5	6	5.38	5	27	6.65
SF	100	4	10	7	8.22	10	34	42.73
SF	300	8	83	11	19.57	83	126	243.12
Rhode [50]	20	2.65	2	5	2.76	2	18	3.51
Maspalomas [50]	24	3.417	3	6	5.56	3	28	8.32
Michigan [50]	39	5.667	13	7	6.45	13	66	11.47
Circuit-s208 [51]	122	3.126	29	9	18.22	29	98	220.78
Friend-rev [52]	228	4.01	52	10	20.45	52	107	275.67
Circuit-s420 [51]	252	3.21	59	13	23.65	59	116	295.63
Circuit-s838 [51]	512	3.44	119	18	39.03	119	122	413.41
Roget [50]	1022	4.966	396	27	75.66	396	135	511.76

https://doi.org/10.1371/journal.pone.0193827.t003

more time than that of PBH matrix. Second, PAQGA adopts qubits representation, where each chromosome contains two individuals. This expands the space of feasible solution. And the adaptive Q-gate rotation operation and crossover operation help to improve the algorithm efficiency.

Moreover, comparison tests among the PAQGA, MMT and MM are conducted to observe the performance of the PAQGA on much larger real networks. The experimental results are shown in Table 4.

It can be seen that N_{cm} of PAQGA agrees with that of MMT on these real-world networks, and is slightly greater than or equal to that of MM. The experimental results show the

Table 4. Performance comparison of PAQGA, MMT, and MM on several large real-directed, -weighted and–unweighted networks in terms of N_{cm} and computational time. For data sources, see Supplementary information S1 Dataset.

network	class	N/P	PAQGA		MMT [11]		MM [10]	
			Ncm	time (s)	Ncm	time (s)	Ncm	time (s)
Coauthorships [53]	Directed weighted	1461	0.3436	85.12	0.3436	67.03	0.3425	34.16
SciMet [54]	Directed unweighted	2729	0.4251	126.35	0.4251	83.29	0.4236	52.91
Kohonen [54]	Directed unweighted	3772	0.562	173.46	0.562	106.62	0.5604	73.85
Wiki-Vote [55]	Directed unweighted	7115	0.6656	392.44	0.6656	228.73	0.6656	167.59
P2P-3 [56]	Directed unweighted	8717	0.5778	473.19	0.5778	279.15	0.5774	206.52
P2P-1 [56]	Directed unweighted	10876	0.5531	685.78	0.5531	359.83	0.552	268.96

https://doi.org/10.1371/journal.pone.0193827.t004

efficiency of PAQGA in identifying the minimum control nodes. Nevertheless, PAQGA is at a disadvantage in computational time compared with MMT and MM, although such defect could be improved by adding the number of processors or using computer groups. For example, the cost of calculating N_{cm} of Wiki-vote network is reduced to 301.34s with 12 processors.

PAQGA is an intelligent probabilistic optimization algorithm that provides approximate solutions. The optimal solution cannot be guaranteed to be found. Moreover, the optimal solution of a problem is typically unknown in advance. We used the structural controllability theory [10, 18] as the benchmark to compute n_{cm} to test the validation of PAQGA. The results are shown in Fig 10.

We can observe that for ER, SF, and SW, the obtained n_{cm} is the same as the benchmark result, which indicates that the proposed PAQGA was effective in determining the minimum control nodes of complex networks.

Discussion and analysis of results

Applying PAQGA, the optimization results and evolution process of network topology can be achieved. The results are intuitively displayed in Fig 11.

From Fig 11(A), N_c of different networks quickly converged to a steady minimum value, which indicates the effectiveness of the PAQGA. For example, N_c of ER with <k> = 4.0 converged to 0.05 at the fifth generation, which demonstrates that five control nodes were sufficient to maintain network controllability. For SF with <k> = 6.0 and γ = 2.1, N_c rapidly decreased to a minimum value of 0.07 at the seventh generation, which means that at least seven control nodes were required to fully control the network. Fig 11(B)–11(D) together capture the evolution of the SW network with <k> = 6.0 at the zeroth, fifth, and seventh iteration, and the convergence trend of the control nodes can be acquired.

We also found that two networks with different <k> required different N_{cm}. For example, for ER with <k> = 4.0 and <k> = 6.0, N_{cm} of the network with <k> = 4.0 was 0.05 and with <k> = 6.0, N_{cm} = 0.02. Additionally, N_{cm} of SW networks with <k> = 4.0 and <k> = 6.0 was 0.25 and 0.22, respectively. Second, for networks with the same <k> and different γ, N_{cm} also differed. For example, consider SF with same <k> = 6.0, and different γ = 2.1 and γ = 3.0. The two networks had N_{cm} = 0.06 and N_{cm} = 0.04, respectively. Third, for networks with the same γ and different <k>, N_{cm} was also different, which can be determined from Table 2. These results led us to conjecture that N_{cm} had a relationship with <k> and γ.

To confirm our hypothesis, we performed simulations on a set of different networks and plotted N_{cm} as the function of <k> and γ. The results are shown in Fig 12.

From Fig 12(A), it is obvious that N_{cm} of networks with fixed γ decreased monotonically with <k> until N_{cm} became slowly flat. Additionally, the downward trend was of asymptotic exponential dependence, which suggests that the sparse network required more control nodes to maintain full controllability. From Fig 12(B), we can observe that N_{cm} with fixed <k> decreased as γ increased. The results indicate that N_{cm} may be influenced by the degree heterogeneity, denoted by H, which is the standard deviation of the network node degree distribution [57]. In this paper, H is defined as

$$\mathrm{H} = \left(\sum_i (k_i - \langle k \rangle)^2)/\mathrm{N}\right)^{1/2},\ i = 1, 2, \ldots, N, \tag{24}$$

where k_i is the degree of state node i.

To determine the relationship between N_{cm} and H, we examined N_{cm} as a function of H and obtained the results shown in Fig 13.

From Fig 13(A), it can be observed that a larger N_{cm} always corresponded to a larger H and smaller γ. Fig 13(B) shows that the network with a smaller ⟨k⟩ and larger H typically required a

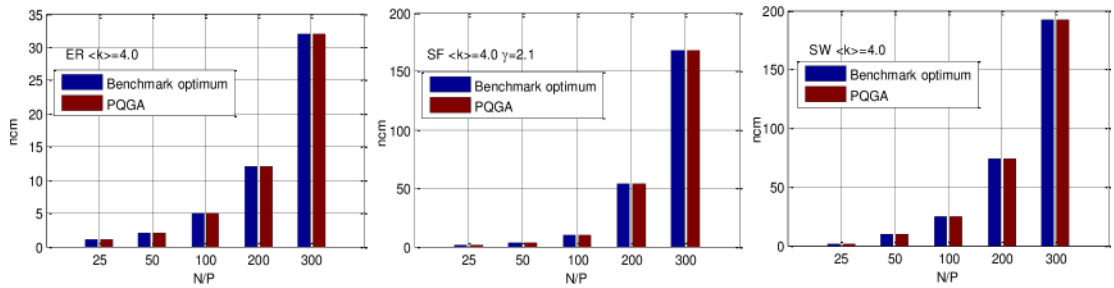


Fig 10. n_{cm} comparison between the structural controllability theory and PAQGA on (a) ER with ⟨k⟩ = 4.0, (b) SF with ⟨k⟩ = 4.0 and γ = 2.1, and (c) SW with ⟨k⟩ = 4.0.

https://doi.org/10.1371/journal.pone.0193827.g010

larger N_{cm}. The results suggest that the larger the differences between node degrees, the more control nodes were required to entirely control the network.

SW networks have the remarkable characteristics of a large clustering coefficient, denoted by C, which represents the overlapping degree of friend circles of two adjacent state nodes and is defined as

$$C = \frac{1}{N}\sum_i \frac{E_i}{\frac{1}{2}k_i(k_i-1)},\ i = 1, 2, \ldots, N, \tag{25}$$

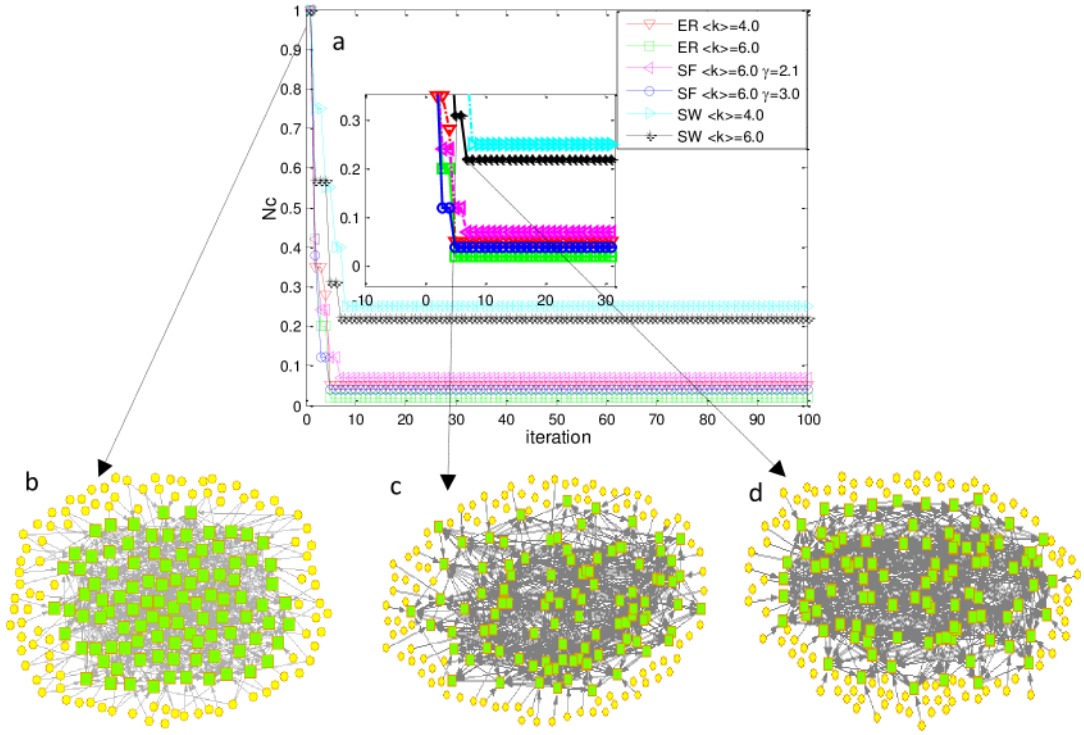


Fig 11. PAQGA optimization results and network topology evolution. (a) Convergence trend of N_c of directed ER, SF, and SW with N = P = 100. (b) Initial network topology (at the zeroth generation) of SW with ⟨k⟩ = 6.0. Yellow circles represent the candidate control nodes and green squares represent the state nodes. Selected control nodes are connected to state nodes with a row from circles to squares. Links between state nodes are arrowed lines between squares. (c) Network topology guided by 31 control nodes at the fifth generation. (d) Network topology with 22 control nodes at the first convergence generation (seventh generation).

https://doi.org/10.1371/journal.pone.0193827.g011

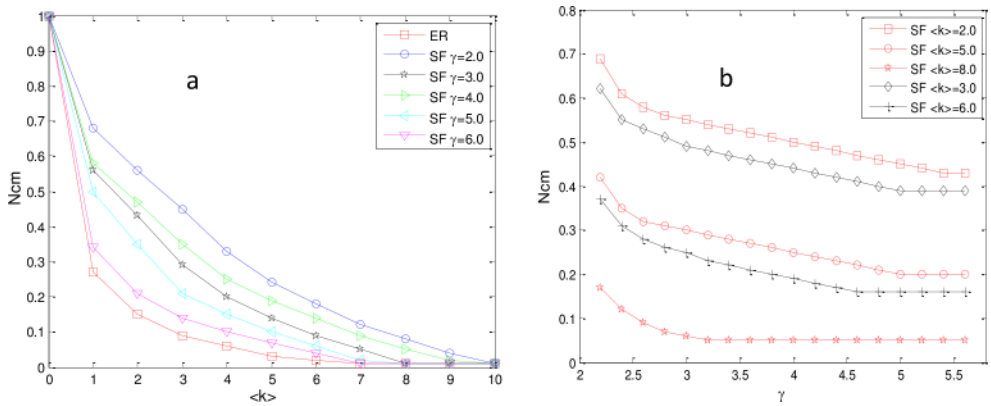


Fig 12. Impact of <k> and γ on N_{cm}. (a) N_{cm} as a function of <k> with fixed γ. (b) N_{cm} as a function of γ with fixed <k>. Networks are directed with N = P = 500.

https://doi.org/10.1371/journal.pone.0193827.g012

where i is node i, k_i is the number of edges between node i and other nodes, and E_i is the number of edges among the k_i nodes. For SW networks, N_{cm} may be affected by C. To explore the relationship between N_{cm} and C, we plot N_{cm} as a function of ⟨k⟩ and C,shown in Fig 14.

From Fig 14(A) and 14(B), we can determine that a larger C corresponded to a smaller N_{cm}, which indicates that the more interconnected the network, the fewer control nodes were required to control the network. For other networks, such as ER, SF, the conclusion also holds.

Considering the aforementioned analysis results together, we can determine that for a given network with both control nodes and state nodes: 1) the sparser the network, the more control nodes were required to control it; and 2) the more heterogeneous the network, the more control nodes were required to guarantee its full control. We reflect that the sparse and heterogeneous network is the most difficult for guiding its dynamic evolution (see Tables 2 and 4 and Figs 10(A) and 12(B)). The consistency between the results from our approach and from these existing methods [10, 11, 22, 28] confirms the similarity between them for directed networks, which not only further validate these existing methods, but also reflect the effectiveness of our method.

To evaluate the controllability of directed networks, the structural controllability framework based on the MM method is still the best for its error-free feature [11]. Like the MMT, the PAQGA also relies on the eigenvalues and the rank of the network matrix, the computation

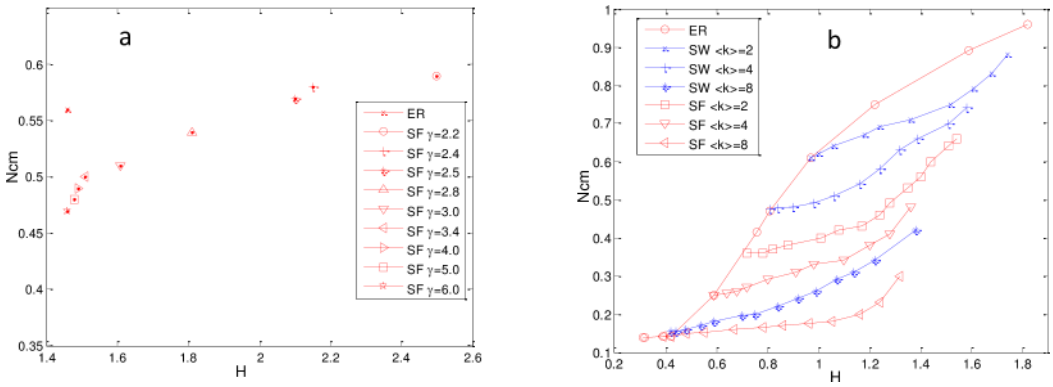


Fig 13. N_{cm} as a function of H. (a) N_{cm} as a function of H for ER and SF networks with fixed γ and variable ⟨k⟩. (b) N_{cm} as a function of H for ER, SF, and SW networks with variable γ and fixed ⟨k⟩. The networks are directed with N = P = 500.

https://doi.org/10.1371/journal.pone.0193827.g013

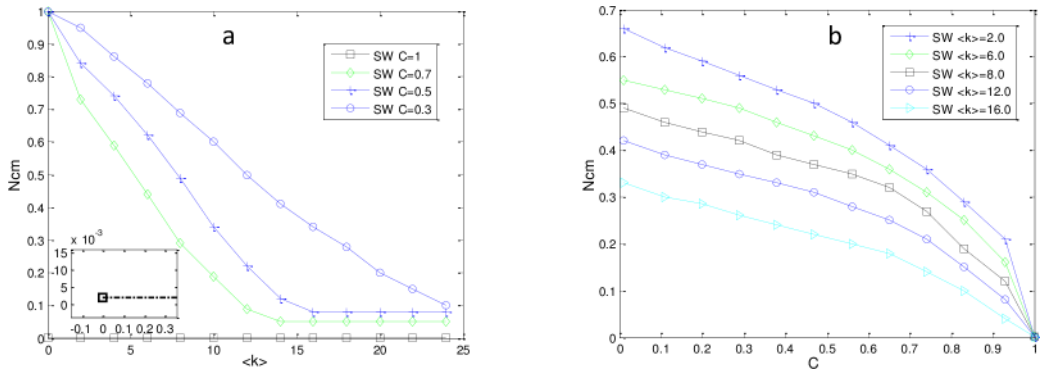


Fig 14. Impact of <k> and C on N_{cm} of SW networks. (a) N_{cm} as a function of <k> with fixed C. When C = 1, the network is fully connected and can be steered to any state with only one controller. (b) N_{cm} as a function of C with fixed ⟨k⟩. Networks are directed with N = P = 500.

https://doi.org/10.1371/journal.pone.0193827.g014

of which inevitably introduces numerical errors. Further, MM and MMT both surpass PAQGA in computational efficiency in identifying both the minimum set of driver nodes and the number of these driver nodes. However, the PAQGA can have a wider range of applications. For example, the PAQGA is valid for networks containing a number of self-loops with identical or different weights, and networks with bidirectional connections between two nodes. The PAQGA is also applicable to undirected networks, where the structural matrix assumption is slightly violated because of the network symmetry. Further, combined with advantages of computer hardware and the adaptive strategies itself, PAQGA has great room for improvement. Taken together, the PAQGA as an alternative exact structural controllability framework provides us deeper understanding of the controllability of complex networked systems.

Conclusions

In this paper, we introduced a PAQGA to optimize the controllability of arbitrary networks with control nodes and state nodes under the PBH rank condition. In addition to MATLAB® workers, more parallel mechanisms can be flexibly embedded in the PAQGA, for which more threads concurrently processing could further promote the time efficiency of generating a solution. Analyses and simulation comparisons demonstrated the effectiveness and applicability of the proposed PAQGA. Furthermore, we found that the minimum control nodes were affected by the network degree distribution, degree heterogeneity, and clustering coefficient. The sparse and heterogeneous network is the most difficult to be fully controlled.

In our study, the topology that comprises state nodes remained static during the entire evolution process. However, networks normally evolve over time, which manifests as the increasing or decreasing of different nodes and their links. In the future, we will focus on the controllability of dynamic networks. Furthermore, we hope to explore how to use the obtained minimum control nodes to steer an intermediate network to evolve into our desired network considering realistic energy constraints.

Supporting information

S1 Dataset. Canonical and real-world network datasets for comparison experiments.
(RAR)

Author Contributions

Conceptualization: Yuhong Li, Guanghong Gong, Ni Li.

Data curation: Yuhong Li.

Formal analysis: Yuhong Li.

Funding acquisition: Guanghong Gong, Ni Li.

Investigation: Yuhong Li.

Methodology: Yuhong Li.

Software: Yuhong Li.

Supervision: Guanghong Gong, Ni Li.

Validation: Guanghong Gong, Ni Li.

Writing – original draft: Yuhong Li.

Writing – review & editing: Yuhong Li, Guanghong Gong, Ni Li.

References

1. An XL, Zhang L, Li YZ, Zhang JG. Synchronization analysis of complex networks with multi-weights and its application in public traffic network. Physica A Statistical Mechanics & Its Applications. 2014; 412(10):149–156.
2. Tang J, Wang Y, Liu F. Characterizing traffic time series based on complex network theory. Physica A Statistical Mechanics & Its Applications. 2013; 392(18):4192–4201.
3. Guo Y, Cao J, Duan R, Li S. Power Grid Vulnerability Identifying Based on Complex Network Theory. Second International Conference on Instrumentation, Measurement, Computer, Communication and Control IEEE. 2012;474–477.
4. Sinha S, Thess M, Markose S. How Unstable Are Complex Financial Systems? Analyzing an Interbank Network of Credit Relations. Econophysics of Systemic Risk & Network Dynamics. 2013;59–76.
5. Van NR. Online collaboration: Scientists and the social network. Nature. 2014; 512(7513):126–9. https://doi.org/10.1038/512126a PMID: 25119221
6. Hale SA. Global connectivity and multilinguals in the Twitter network. Conference on Human Factors in Computing Systems Proceedings. 2014;833–842.
7. Xiao WH, Cai XD. A Novel Wireless Sensor Network Model Based on Complex Network Theory. Advanced Materials Research. 2012; 546-547(1):1276–1282.
8. Jiang ZY. Modeling Wireless Network Topology Based on the Theory of Complex Network. Applied Mechanics & Materials. 2013; 321–324:2892–2896.
9. Xiao ZD, Hu H, Zhou GH. Study on the formation mechanism of ecological relationship in an Eco-production network based on the node correlation. IEEE International Symposium on Assembly and Manufacturing IEEE. 2013;32–36.
10. Liu Y Y, Slotine J J, Barabási A L. Controllability of complex networks. Nature. 2011; 473(7346): 167–173. https://doi.org/10.1038/nature10011 PMID: 21562557
11. Yuan Z, Zhao C, Di Z, Wang WX, Lai YC. Exact controllability of complex networks. Nature Communications. 2013; 4:2447. https://doi.org/10.1038/ncomms3447 PMID: 24025746
12. Zhao C, Wang WX, Liu YY, Slotine JJ. Intrinsic dynamics induce global symmetry in network controllability. Scientific Reports. 2015; 5:8422. https://doi.org/10.1038/srep08422 PMID: 25672476
13. Pósfai M, Liu YY, Slotine JJ, Barabási AL. Effect of correlations on network controllability. Scientific Reports. 2012; 3(3):1067.
14. Turci LF, Macau EE. Performance of pinning-controlled synchronization. Physical Review E Statistical Nonlinear & Soft Matter Physics. 2011; 84(1):1068–1072.
15. Jalili M, Askari SO, Yu X. Optimal pinning controllability of complex networks: Dependence on network structure. Physical Review E. 2015; 91(1).

16. Cowan NJ, Chastain EJ, Vilhena DA, Freudenberg JS, Bergstrom CT. Nodal dynamics, not degree distributions, determine the structural controllability of complex networks. Plos One. 2012; 7(6):e38398. https://doi.org/10.1371/journal.pone.0038398 PMID: 22761682

17. Lo IF, Garofalo F, Sorrentino F. Structural permeability of complex networks to control signals. Nature Communications. 2015; 6:8349 https://doi.org/10.1038/ncomms9349 PMID: 26391186

18. Lin CT. Structural Controllability. IEEE Transactions on Automatic Control. 1974; 19(3):201–208.

19. Nepusz T, Vicsek T. Controlling edge dynamics in complex networks. Nature Physics. 2011; 8(7):568–573.

20. Commault C, Dion JM, Woude JWVD, Van JW, Woude D. Characterization of generic properties of linear structured systems for efficient computations. Kybernetika -Praha-, 2002; 38503(5):503–520.

21. Zdeborová L, Mézard M. The number of matchings in random graphs. Journal of Statistical Mechanics Theory & Experiment. 2006; 2006(5):2006. https://doi.org/10.1088/1742-5468/2013/03/P03003

22. Ding J, Lu YZ, Chu J. Studies on controllability of directed networks with extremal optimization. Physica A Statistical Mechanics & Its Applications. 2013; 392(24):6603–6615.

23. Shields RW, Pearson JB. Structural controllability of multiinput linear systems. IEEE Transactions on Automatic Control. 1975; 21(2):203–212.

24. Dion JM, Commault C, Woude VD. Generic properties and control of linear structured systems: a survey. Automatica.2003; 39(7):1125–1144.

25. Hautus MLJ. Controllability and observability conditions of linear autonomous systems. Ned. Akad. Wetenschappen, Proc. Ser. A. 1969; 72(5): 443–448.

26. Kalman RE. Mathematical description of linear dynamical systems. J. Soc. Ind. Appl. Math. Ser. A: Control. 1963; 1(2):152–192.

27. Moler CB. Numerical Computing with MATLAB, Revised Reprint [2 ed.]. Society for Industrial and Applied Mathematics. 2008; 270–297.

28. Li XF, Lu ZM. Optimizing the controllability of arbitrary networks with genetic algorithm. Physica A. 2016; 447: 422–433.

29. Liljeros F, Edling C, Amaral L, Stanley H, Aberg Y. Human web of sexual contacts. Nature. 2001; 411: 907–908. https://doi.org/10.1038/35082140 PMID: 11418846

30. Holland JH, Nisbett RE, Holyoak KJ. Induction: Processes of Inference, Learning, and Discovery. MIT Press. 1989; 2:92–93.

31. Goldberg DE. Genetic Algorithm in Search, Optimization and Machine Learning. MA Publisher: Addison-Wesley. 1989.

32. Abo-Hammour ZS, Alsmadi OMK, Al-Smadi AM. ARMA model order and parameter estimation using genetic algorithms. Comp. Model. Dyn. 2012; 18(2):1–21.

33. Li XF, Lu ZM. Optimizing the controllability of arbitrary networks with genetic algorithm. Physica A Statistical Mechanics & Its Applications. 2016; 447:422–433.

34. Reynolds RG. Evolutionary computation: toward a new philosophy of machine intelligence. 3rd Edition, IEEE Press Series on Computational Intelligence, John Wiley and Sons Ltd. 2006.

35. Faghihia V, Reinschmidta KF, Kang JH. Construction scheduling using Genetic Algorithm based on Building Information Model. Expert Syst. Appl. 2014; 41(16): 7565–7578.

36. Srinivas M, Patnaik LM. Adaptive probabilities of crossover and mutation in genetic algorithms. IEEE Trans. Syst. Man Cybern. 1994; 24 (4):656–667.

37. Han KH, Kim JH. Genetic quantum algorithm and its application to combinatorial optimization problem. Evolutionary Computation, 2000. Proceedings of the 2000 Congress on IEEE. 2003; 2:1354–1360.

38. Han B, Jiang J, Gao Y, Ma J. A Quantum Genetic Algorithm to Solve the Problem of Multivariate. Information Computing and Applications. Springer Berlin Heidelberg. 2011;308–314.

39. Laboudi Z, Chikhi S. Comparison of Genetic Algorithm and Quantum Genetic Algorithm. International Arab Journal of Information Technology. 2012; 9(3):243249.

40. Han KH, Kim JH. Quantum-inspired evolutionary algorithm for a class of combinatorial optimization. IEEE Transactions on Evolutionary Computation. 2002; 6(6):580–593.

41. Wei FM, Zhang JP, Li B, Yang J. A Survey of Quantum Genetic Algorithm for Combinatorial Optimization Problems. Applied Mechanics & Materials. 2014; 568–570:822–826

42. Kong H, Ni L, Shen Y. Adaptive double chain quantum genetic algorithm for constrained optimization problems. Chinese Journal of Aeronautics. 2015; 28(1):214–228.

43. Chaturvedi J, Chaturvedi J. Adaptive Quantum Inspired Genetic Algorithm for Combinatorial Optimization Problems. International Journal of Computer Applications. 2014; 107(4):34–42.

44. Han KH, Park KH, Lee CH, Kim JH. Parallel quantum-inspired genetic algorithm for combinatorial optimization problem. Evolutionary Computation, 2001. Proceedings of the 2001 Congress on IEEE. 2001; 2:1422–1429.

45. Li SY, Li PC. Quantum genetic algorithm based on real encoding and gradient information of object function. J Harbin Inst Technol. 2008; 38(8):1216–1218+1223.

46. Erdős P, Rényi A. On the strength of connectedness of a random graph. Acta Math. Hungar. 1961; 12 (1): 261–267.

47. Barabási AL, Albert R. Emergence of scaling in random networks. Science. 1999; 286 (5439):509–512. PMID: 10521342

48. Newman MEJ, Watts DJ. Renormalization group analysis of the small-world network model. Phys. Lett. A. 1999; 263:341–346.

49. Watts DJ, Strogatz SH. Collective dynamics of small-world networks. Nature.1998; 393:440–442. https://doi.org/10.1038/30918 PMID: 9623998

50. Batagelj V, Mrvar A. Pajek datasets. <URL: http://vlado.fmf.uni-lj.si/pub/networks/data/>. 2006.

51. Milo R, Itzkovitz S, Kashtan N, Levitt R, Shenorr S, Ayzenshtat I, et al. Superfamilies of designed and evolved networks. Science. 2004; 303(5663): 1538–1542. https://doi.org/10.1126/science.1089167 PMID: 15001784

52. Burt RS. Social Contagion and Innovation: Cohesion versus Structural Equivalence. American Journal of Sociology.1987; 92(6):1287–1335.

53. Newman MEJ. Finding community structure in networks using the eigenvectors of matrices. Phys. Rev. E. 2006; 74(3): 036104.

54. https://sparse.tamu.edu/.

55. Leskovec J, Lang KJ, Dasgupta A, Mahoney MW. Community structure in large networks: Natural cluster sizes and the absence of large well-defined clusters. Internet Mathematics. 2009; 6(1):29–123.

56. Leskovec J, Kleinberg J, Faloutsos C. Graph evolution: Densification and shrinking diameters. ACM Transactions on Knowledge Discovery from Data(TKDD). 2007; 1(1):2.

57. Hu HB, Wang XF. Unified index to quantifying heterogeneity of complex networks. Physica A: Statistical Mechanics & Its Applications. 2008; 387(14):3769–3780.