IEEE Transactions on Network Science and Engineering, Available online10 June 2024
DOI: 10.1109/TNSE.2024.3410694

Rewiring Complex Networks to Achieve Cluster Synchronization using Graph Convolution Networks with Reinforcement Learning

Mengbang Zou, Weisi Guo*, *Senior Member, IEEE*, Kai-Fung Chu, *Member, IEEE*

***Abstract*—Synchronization on complex networks depends on a myriad of factors such as embedded dynamics, initial conditions, network structure, etc. Current literature simplifies analysis of cluster synchronization leveraging conditions on network topology such as input-equivalence, network symmetries, etc., of which external equitable partition (EEP) is one of the most relaxed conditions. One practical problem is that for a dynamic system, how to alter a network to reach arbitrary achievable cluster synchronization and remaining faithful to the original structure. To solve this problem, we represent graph dynamics in Graph Convolution Network (GCN) modules that sit within an Actor-Critic Reinforcement Learning (AC-RL) framework under the condition of EEP. This allows the framework to select a good policy to sequentially rewire the network, where the sequence of moves matters. We test our method on two types of high-dimensional networked systems, Rossler dynamic networks and Hindmarsh-Rose neuronal circuits, with different network sizes. Our research opens up a way for the discovery of achievable cluster synchronization configurations by altering the network structure in any given networked dynamics.**

***Index Terms*—complex network; synchronizability; reinforcement learning; graph neural network**

I. Introduction

SYNCHRONIZATION is a common phenomenon in cyber-physical and biological systems such as load balancing in wireless networks and power grids [1]–[3], and neural networks [4]–[6]. Synchronization can be exhibited either as a global state in which all components achieve the same trajectory or as cluster states where every cluster reaches synchronization and different clusters cannot reach synchronization [7]. These two synchronization patterns are separately called global synchronization and cluster synchronization, where the former is a special case of latter. Examples of cluster synchronization in wireless networks including re-configuring the base station association matrix such that they can achieve synchronized performance (e.g., load balancing or interference cancellation [1], [8]) at a desirable equilibrium point within self-organizing clusters.

Significant research has studied the criterion to achieve the stable global or cluster synchronized state [9]–[15]. For example, the master stability function proposed by [9] relates the stable global synchronized state to the spectral properties of the Laplacian matrix of the network, which provides the objective criterion for global synchronizability; [11] presented a criterion for stability of cluster synchronization that leverage the notion of external equitable partition (EEP). Based on the criterion for stability of synchronization, how the network topology affects the synchronization also has attracted a lot of attention [7], [16]–[19] recently.

One relevant and practical problem is how to alter complex networks to achieve different synchronization patterns. Since the complex network consists of numerous coupling components with nonlinear dynamics, even slight changes in network structure make complex network dynamics challenging to predict. Though the master stability function relates the global synchronization to the spectral properties of the Laplacian matrix, it is still complicated to rewire the network to make the spectral properties belong to the stable region. This is because the effects of structure perturbations on spectral properties depend on both the local structure of relevant nodes and the global structure of networks [20]. Some researchers have studied improving global synchronizability by rewiring the network structure. [21] proposed adding edges to the node whose nearest neighbors have small degrees to enhance the synchronizability. [22] improved the synchronizability by choosing edges based on the spectral properties of the network Laplacian to reduce eigenratio. Similarly, [23] designed an accept-reject mechanism based on the spectral properties of the network Laplacian to improve the synchronizability of Kuramoto model. [24]proposed a rewiring method based on nodes' similarity. These studies are relevant to improve the synchronizability, but a little different to achieve synchronization, i.e. if the extreme eigenvalues of original networks do not belong to the stable region, the decrease of the extreme eigenratio in a new network cannot guarantee that the extreme eigenvalues belong to the stable region. Besides, the above methods improve the global synchronizability mainly based on a specific network property, such as the degree distribution or eigenvectors of extreme eigenvalues. The design based on a specific network property is beneficial to computational efficiency but ignores other local or global information of the network which may cause the loss of better solutions.

Another challenge is to achieve cluster synchronization. Recent studies have linked the appearance of cluster synchronization to the existence of symmetries [10], [25] and EEP [15] in the network structure. [25] studied the stability condition

M. Zou is with Cranfield University, Cranfield, MK43 0AL, U.K.
W. Guo (corresponding author) is with Cranfield University, MK43 0AL, U.K. (e-mail: weisi.guo@cranfield.ac.uk).
K.-F. Chu is with Cranfield University, MK43 0AL, U.K. and Department of Engineering, University of Cambridge, CB2 1PZ, U.K.
The work is supported by EPSRC CHEDDAR: Communications Hub For Empowering Distributed ClouD Computing Applications And Research (EP/X040518/1) (EP/Y037421/1).

© 2024 IEEE. This is the Author Accepted Manuscript issued with: Creative Commons Attribution License (CC:BY 4.0).
The final published version (version of record) is available online at DOI:10.1109/TNSE.2024.3410694. Please refer to any applicable publisher terms of use.

for cluster synchronization based on the symmetric structure. [26] proposed a method to achieve cluster synchronization by controlling the symmetry of the network. By this method, nodes within the same cluster can synchronize to the same state. But only controlling the symmetry structure cannot guarantee that nodes in different clusters achieve different states. Besides, for given dynamic functions with determined coupling weight, controlling symmetry cannot always lead to synchronization. Controlling coupling strength is needed after controlling symmetry to ensure that the eigenvalues belong to the stable region. The problem we want to solve is for given dynamic functions with determined coupling weight, how to achieve cluster synchronization only by optimizing the network. EEP is a more relaxed condition compared with symmetry. Based on EEP, the criterion for cluster synchronization has been established [11], where the network structure must satisfy three conditions at the same time: 1) external equitable equation; 2) not all eigenvalues of the original network $\lambda^{(i)}$ belong to the stable region calculated by largest Lyapunov exponent; 3) all eigenvalues of the original network, which not belong to the eigenvalue spectral of the quotient graph of the original network, should belong to the stable region. Therefore, optimizing networks to achieve cluster synchronization is a big challenge.

Reinforcement learning (RL) has been generally applied in optimization problems like optimal control problems [27]–[29], combinatorial optimization problems [30], as well as the optimization problem of complex networks. [31] employed a deep reinforcement learning method to alter the network topology to improve the performance of communication network. [32] proposed a framework for improving the resilience of complex networks, by introducing the network construction Markov decision process. Inspired by the successes of RL in the study of complex networks, RL can be considered as a potential method to alter complex networks to achieve different synchronization patterns. We formulate the process of network optimization as a Markov decision process (MDP). The reward function is designed according to the criterion for different synchronization patterns which builds the relationship between synchronization and the network topology. For example, according to the criterion for global synchronization [9], for given dynamic functions, the networked system can achieve global synchronization if the extreme eigenvalues of the network belong to the stable region. The rewiring action which enables the extreme eigenvalues close to the stable region is high reward. Otherwise, the action is given a low reward. The objective to alter the network to achieve synchronization is converted to obtain a sequence of actions to rewire the network with higher accumulated rewards to make the extreme eigenvalues belong to the stable region by RL. This is an effective way to reduce computation complexity since it avoids computing dynamic functions to calculate the state errors of all nodes after each action taken to the network. Instead, it only requires the computation of extreme eigenvalues of the network after each rewiring. Furthermore, compared with the rewiring methods based on a specific network property, this method has a better exploration ability for solutions. Besides, this method is flexible to alter the network to achieve different synchronization patterns. This can be realized by designing the reward function in MDP according to the corresponding synchronization criterion without any other new design of the MDP.

The action space in network optimization contains edge addition and deletion. For a network with N nodes, the size of the action space is proportional to N^2. Considering the large action space on graph, policy-based framework like actor-critic network is more suitable than value-based framework like deep Q network. This is because value-based RL suffers from the problem as a specific ranking function is applied over all actions to pick the one with the highest score when making a decision. Actor-critic network uses an actor network to output the distribution of actions to reduce the complexity. To reduce the action space, the edge selection is decomposed into two steps: firstly selecting the starting node, and then selecting the ending node. Two neural networks are employed to generate the starting and ending nodes separately. Then a critic network is used to evaluate the state. Since edge selection is closely related to the network structure, it is natural for us to consider embedding the network structure into the actor and critic networks. Graph neural network as a deep learning-based method that operates on graph can be used as actor and critic network. Therefore, in this paper, we propose a framework which combines an actor-critic network with graph convolutional network (GCN) to alter network to achieve different synchronization patterns. According to our knowledge, this is the first time that a reinforcement learning method is applied to network synchronization problems.

The contribution of this paper is as follows. First, we propose a framework to alter the network structure to achieve different synchronization patterns (shown in Fig. 1). We formalize the network as a Markov decision process (MDP) and use an Actor-Critic framework with GCN to achieve the objective. Two GCNs are employed to generate starting and ending nodes separately, which effectively reduces the action space of adding or deleting edges in the network. Compared with heuristic algorithms based on some specific network properties, our method has a better exploration ability for optimal solutions. Secondly, prior knowledge of the criterion for synchronization is used to design the reward function in MDP. If there exists an achievable synchronized state, the initial network can be always altered by our method to satisfy the criterion to synchronize. The first advantage of this design is that it avoids complex computation on dynamic functions of nodes after every rewiring. The second advantage is that this design is flexible to alter the network to achieve different synchronization patterns by designing the reward function in MDP according to the corresponding synchronization criterion.

II. Methods

A. Synchronization of complex networks with nonlinear dynamics

Consider an undirected network $G = (\mathbf{V}, \mathbf{E})$ consists of a node set $\mathbf{V} = \{v_1, v_2, \ldots, v_N\}$ and an edge set $\mathbf{E}$, where $\mathbf{E} \subseteq \mathbf{V} \times \mathbf{V}$ (notations are shown in Table (I)). The dynamics

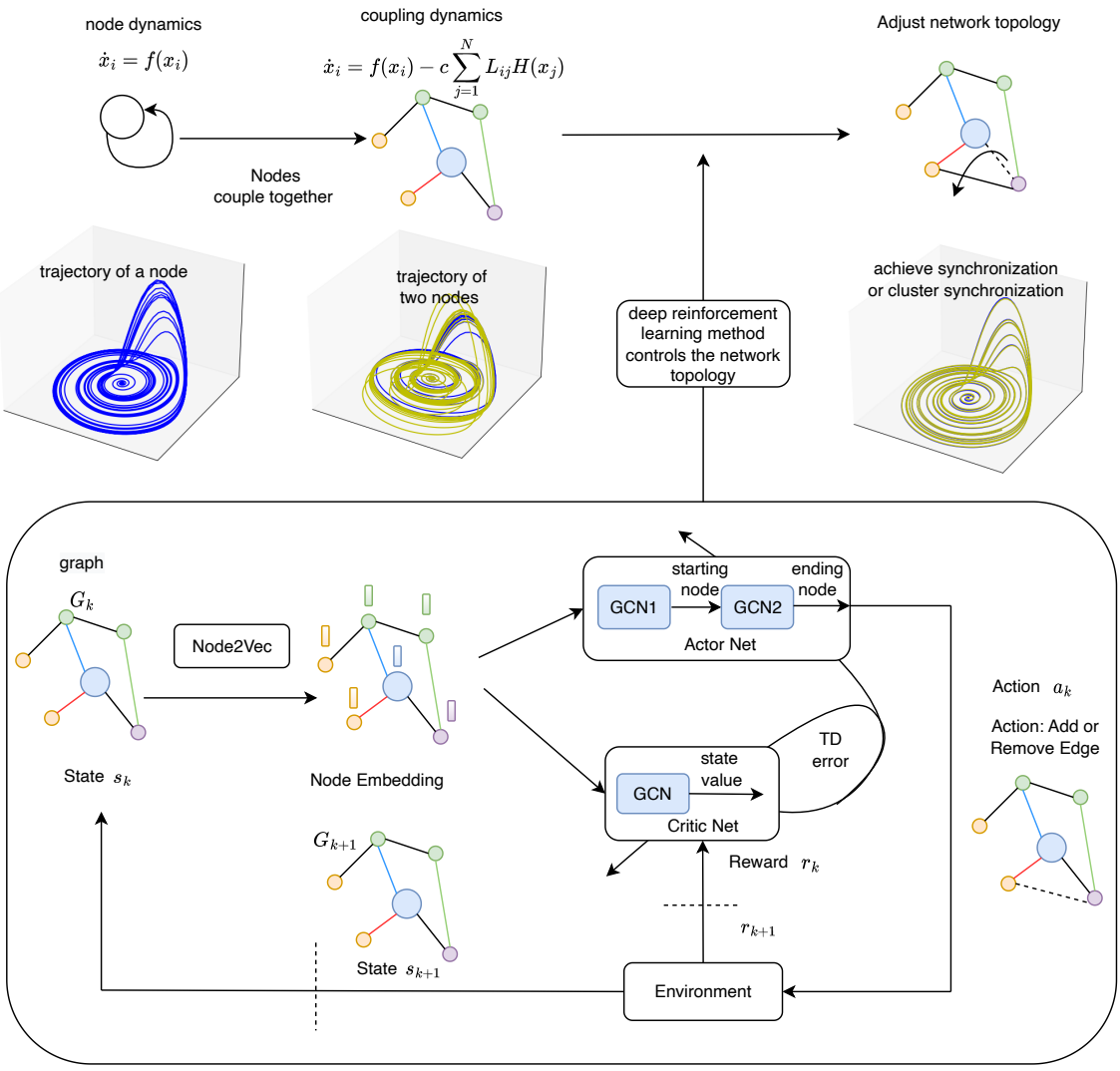


Fig. 1. Graph construction or optimization by the actor-critic graph convolutional network. In a complex network consisting of nodes with nonlinear dynamics, nodes can achieve different patterns of synchronization like cluster or global synchronization by adjusting the network topology. The observable state is the input network G_k. Feature vector of each node i is obtained by Node2Vec. Input adjacency matrix of network and feature matrix of all nodes to Actor Net to select an edge as an action. Apply the action to G_k and obtain a new network G_{k+1} as the next state s_{k+1}. In each step, the state s_k will be inputted to the critic net to estimate the state value. This process will repeat until the network achieve synchronization. Calculate the temporal difference (TD) error according to the reward and state values to update parameters of Actor Nets and Critic Net

of each node is described by

$$\dot{\mathbf{x}}_i = \mathbf{f}(\mathbf{x}_i) - c\sum_{j=1}^{N} L_{ij}\mathbf{H}(\mathbf{x}_j), i = 1, 2, 3, \ldots, N \qquad (1)$$

where $\mathbf{x}_i = (x_i^{(1)}, x_i^{(2)}, \ldots, x_i^{(n)})^\top \in \mathbb{R}^n$ is the state vector of node v_i, $\mathbf{f}(\cdot) : \mathbb{R}^n \to \mathbb{R}^n$ controls self dynamics of node v_i, $c > 0$ is the coupling strength, $\mathbf{H}(\cdot) : \mathbb{R}^n \to \mathbb{R}^n$ is the inner coupling function. $\mathbf{L}$ is the Laplacian matrix of the network. $\mathbf{L} = \mathbf{D} - \mathbf{A}$, where $\mathbf{A}$ is the adjacency matrix and $\mathbf{D}$ is the diagonal matrix of degrees. $D_{ij} = \sum_{j=1}^{N} A_{ij}$ if $i = j$. Otherwise, $D_{ij} = 0$. If an edge exists between node v_i and node v_j, $A_{ij} = A_{ji} = 1$. Otherwise, $A_{ij} = 0$. The matrix $\mathbf{L}$ satisfies $L_{ii} = -\sum_{j=1, j\neq i}^{N} L_{ij}, i = 1, 2, \ldots, N$. If the complex network is connected, then $\mathbf{L}$ is irreducible. Zero is an eigenvalue of $\mathbf{L}$ with multiplicity 1 and all other eigenvalues are strictly positive, denoted by

$$0 = \lambda^{(1)} < \lambda^{(2)} \leq \lambda^{(3)} \leq \ldots \leq \lambda^{(N)}. \qquad (2)$$

Global Synchronization. The system reaches the state of global synchronization if $\mathbf{x}_1(t) = \mathbf{x}_2(t) = \ldots = \mathbf{x}_N(t) = \mathbf{x}_s(t)$ when $t \to \infty$, where $\mathbf{x}_s(t)$ can be an equilibrium point, a periodic orbit, or a chaotic attractor [33]. The stability of the synchronized state $\mathbf{x}_1(t) = \mathbf{x}_2(t) = \cdots \mathbf{x}_N(t)$ can be determined by the master stability function [9]

$$\dot{\boldsymbol{\zeta}} = [D\mathbf{f}(\mathbf{x}_s(t)) - \alpha D\mathbf{H}(\mathbf{x}_s(t))]\boldsymbol{\zeta}, \qquad (3)$$

where $\boldsymbol{\zeta}$ is the collection of variations and $\boldsymbol{\zeta} = (\boldsymbol{\zeta}_1, \boldsymbol{\zeta}_2, ..., \boldsymbol{\zeta}_N)$. $\alpha = c\lambda^{(i)}$. $\boldsymbol{\zeta}_i$ is the variation on the i-th node. $D\mathbf{f}(\mathbf{x}_s(t))$ and $D\mathbf{H}(\mathbf{x}_s(t))$ are the Jacobian matrix of functions $\mathbf{f}$ and $\mathbf{H}$ at $\mathbf{x}_s(t)$. The evolution of $\boldsymbol{\zeta}$ is described on average as $\|\boldsymbol{\zeta}(t)\| \sim e^{\Omega_{\max}(\alpha)t}$, where $\Omega_{\max}$ is the largest Lyapunov exponent and the state is stable with $\|\boldsymbol{\zeta}(t)\| \to 0$, if $\Omega_{\max}(\alpha) < 0$. $\Omega_{\max}(\alpha)$

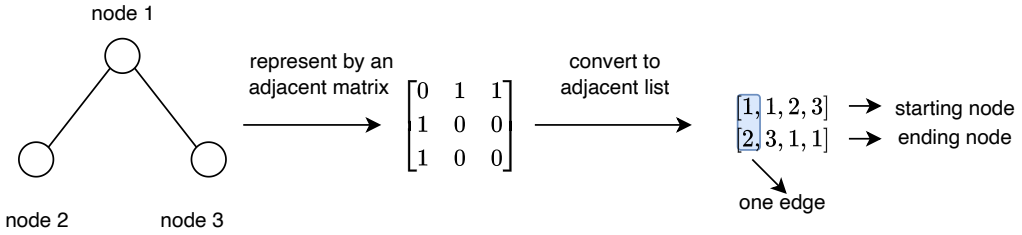


Fig. 2. Represent the adjacency matrix as an adjacency list. In adjacency list, one edge is represented by a starting node and an ending node.

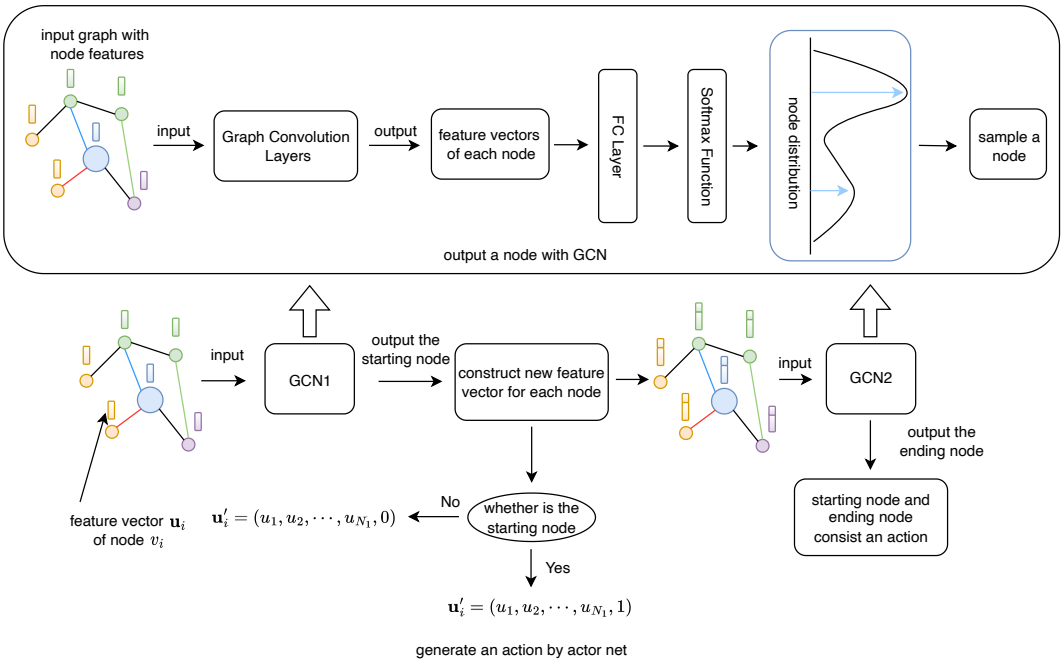


Fig. 3. This figure shows the structure of the actor net. The actor net consists of 2 GCN. The first GCN generates the starting node according to the input adjacency matrix of complex network and the feature vector $\mathbf{u}_i$ of each node. The Second GCN generates the ending node according to the adjacency matrix and the new feature vector of each node $\mathbf{u}'_i$. The new feature vector $\mathbf{u}'_i = (u_1, u_2, \ldots, u_{N_1}, u_{N_1+1})$. $u_{N_1+1} = 1$ if node i is selected as the starting node, otherwise $u_{N_1+1} = 0$. The graph with node features is input to the GCN layers to get feature vectors of each node. Then the feature vectors are input to the full connection layer and softmax function to get the node distribution, which used to output a node as starting node or ending node.

can be calculated by $\Omega_{\max}(\alpha) = \lim_{t\to\infty} \frac{1}{t} \ln \frac{\|\boldsymbol{\zeta}(t)\|}{\|\boldsymbol{\zeta}(0)\|}$ [16]. The region Ξ of α which makes $\Omega_{\max}$ negative is called the synchronized region. The criterion to achieve a stable global synchronized state is

$$-c\lambda^{(i)} \in \Xi, i = 2, 3, \ldots, N \tag{4}$$

For given self and coupling dynamics, the complex network can achieve the global synchronized state by rewiring the network to make the eigenvalues satisfy the equation (4).

Cluster Synchronization. In this paper, we assume the existence of external equitable partitions (EEP) [34] in the network structure. Given the network G, a partition $Q = \{\mathbf{V}_1, \mathbf{V}_2, \ldots, \mathbf{V}_M\}$ is a map of nodes that groups them into M distinct clusters, $\mathbf{V}_1, \mathbf{V}_2, \ldots, \mathbf{V}_M$, with $\bigcup_{l=1}^{M} \mathbf{V}_l = \mathbf{V}$ and $\mathbf{V}_i \cap \mathbf{V}_j = \varnothing$, for $i \neq j$ [12]. If for any pairs of clusters $\mathbf{V}_l$ and $\mathbf{V}_k$, with $l \neq k$, each node $i \in \mathbf{V}_l$ has exactly b_{lk} neighbours in $\mathbf{V}_k$, Q is an external equitable partition [11]. The characteristic matrix $\mathbf{P}$ of an EEP $Q = \{\mathbf{V}_1, \mathbf{V}_2, \ldots, \mathbf{V}_M\}$, is defined as the $N \times M$ matrix with elements

$$P_{ij} = \begin{cases} 1, & \text{if } v_i \in \mathbf{V}_j, \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

For example, $\mathbf{V} = \{v_1, v_2, v_3\}$ and $Q = \{\{v_1, v_2\}, \{v_3\}\}$ is a partition of $\mathbf{V}$ with clusters $\mathbf{V}_1 = \{v_1, v_2\}, \mathbf{V}_2 = \{v_3\}$. The characteristic matrix of Q is

$$\mathbf{P}^\top = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

$\mathbf{P}^\top\mathbf{P}$ is diagonal with the jth element equal to the number of nodes in the cluster $\mathbf{V}_j$. Since the diagonal terms are nonzero as the clusters are not empty, $\mathbf{P}^\top\mathbf{P}$ is invertible. The quotient graph of G is defined as G^Q. Given an EEP, the associated quotient graph has a number of nodes equal to the size of the partition and directed edge weights given by the number of nodes in the neighbor clusters. The elements in adjacency matrix of the quotient graph $\mathbf{A}^Q$ are defined as $A^Q_{ij} = |\mathbf{V}_j|$, where $|\mathbf{V}_j|$ is the number of nodes in cluster $\mathbf{V}_j$, if there exist

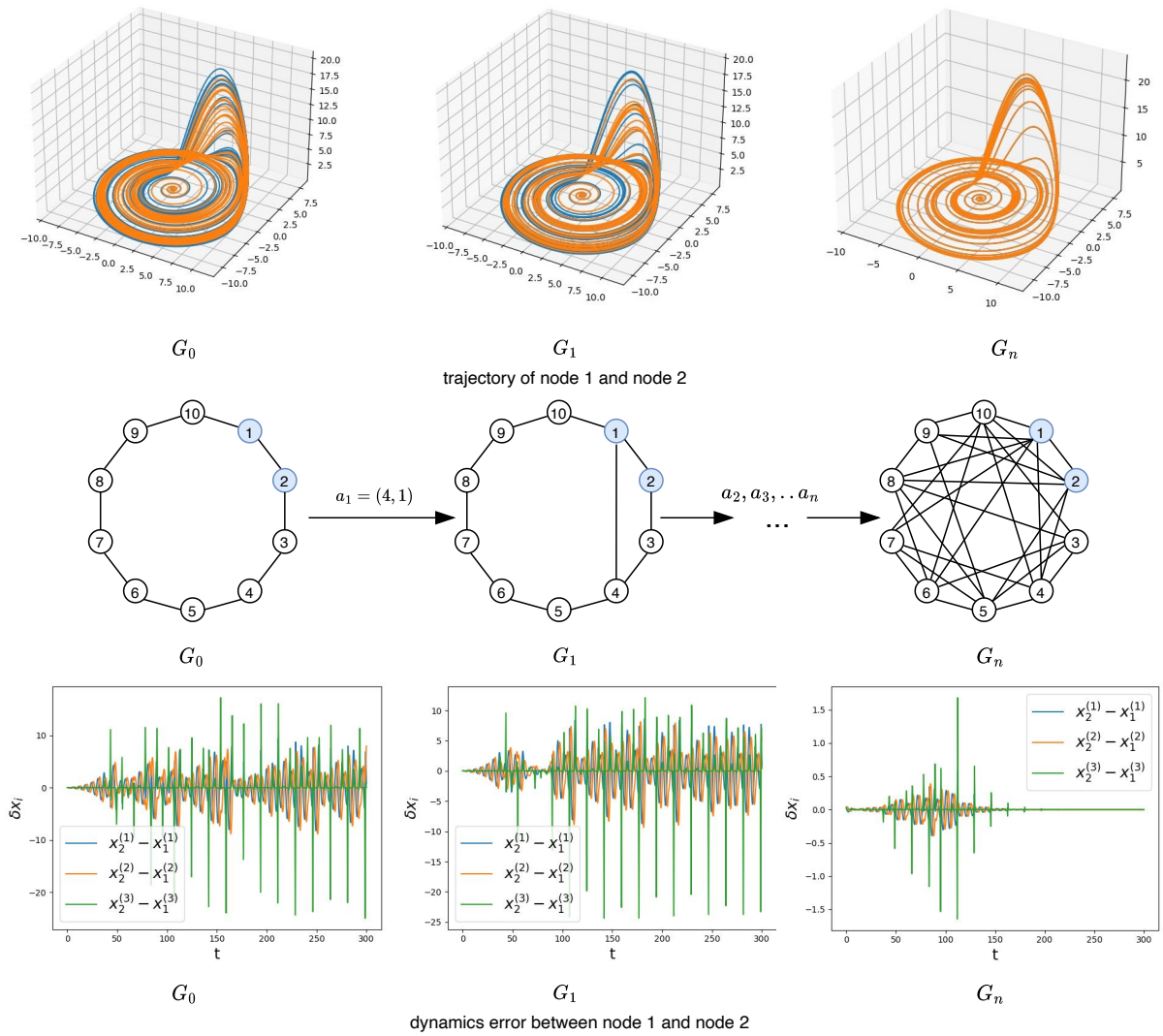


Fig. 4. This figure shows the synchronization process of nodes after several actions to the network. Nodes in the initial network G_0 do not satisfy $x_i - x_j = 0$, which means they cannot synchronize to the state. Then the two Actor Nets output starting node and ending node to form the action $a_1 = (4, 1)$. The edge (4, 1) is added to the network G_0 according to the action and forms a new network G_1. In this state, nodes in G_1 still do not synchronize, and the Actor Nets continue to output actions $a_2, a_3, ..., a_n$ to the network. The Actor Nets stop outputting actions until all the nodes in network G_n satisfy $x_i - x_j = 0$, which means all nodes synchronize.

edges between $\mathbf{V}_i$ and $\mathbf{V}_j$. Otherwise, $A_{ij}^Q = 0$. The Laplacian matrix of the quotient graph G^Q is indicated as $\mathbf{L}^Q$. If Q is an EEP of G, $\mathbf{LP} = \mathbf{PL}^Q$, and $\mathbf{L}^Q = (\mathbf{P}^\top\mathbf{P})^{-1}\mathbf{P}^\top\mathbf{LP}$ [35]. The eigenvalues of $\mathbf{L}^Q$ are a subset of the eigenvalues of $\mathbf{L}$.

Let $\mathbf{s}_l(t)$ be the synchronous manifold of the l-th cluster. The small perturbations on node i which belongs to cluster l is $\delta\mathbf{x}_i = \mathbf{x}_i - \mathbf{s}_l$. Whether the nodes within the cluster l are synchronizable is determined by

$$\begin{aligned}\delta\dot{\mathbf{x}}_i = D\mathbf{f}(\mathbf{s}_l)\delta\mathbf{x}_i + c\sum_{j\in V_l} a_{ji}[D\mathbf{H}(\mathbf{s}_l)\delta\mathbf{x}_j - D\mathbf{H}(\mathbf{s}_l)\delta\mathbf{x}_i] \\ + c\sum_{m\neq l}\sum_{j'\in V_m} a_{ij'}[D\mathbf{H}(\mathbf{s}_m)\delta\mathbf{x}'_j - D\mathbf{H}(\mathbf{s}_l)\delta\mathbf{x}_i],\end{aligned} \tag{6}$$

where $\mathbf{s}_m$ is the synchronized manifold of the m-th cluster and $\delta\mathbf{x}_{j'} = \mathbf{x}_{j'} - \mathbf{s}_m$ (with $j' \in \mathbf{V}_m$). $D\mathbf{f}(\cdot)$ and $D\mathbf{H}(\cdot)$ are the Jacobi matrix of $\mathbf{f}(\cdot)$ and $\mathbf{H}(\cdot)$. In equation (6), $c\sum_{j\in\mathbf{V}_l} a_{ji}[D\mathbf{H}(\mathbf{s}_l)\delta\mathbf{x}_j - D\mathbf{H}(\mathbf{s}_l)\delta\mathbf{x}_i]$ represents the effect of nodes in cluster l on node i. $c\sum_{m\neq l}\sum_{j'\in\mathbf{V}_m} a_{ij'}[D\mathbf{H}(\mathbf{s}_m)\delta\mathbf{x}j' - D\mathbf{H}(\mathbf{s}_l)\delta\mathbf{x}_i]$ represents the effects of nodes from other clusters on node i. The condition to reach stability state for nodes in different clusters is that $\delta\mathbf{x}_i$ approaches to 0.

$\Delta\mathbf{X} = [\Delta\mathbf{X}_1, \Delta\mathbf{X}_2, \ldots, \Delta\mathbf{X}_M]^\top$ is set as the network perturbation vector, where $\Delta\mathbf{X}_l = [\delta\mathbf{x}_1, \delta\mathbf{x}_2, \ldots, \delta\mathbf{x}_{n_l}]^\top$ is the perturbation vector associated to cluster l, and n_l is the number of nodes in cluster $\mathbf{V}_l$. The variational equation (6) can be rewritten as

$$\Delta\dot{\mathbf{X}} = [\sum_{l=1}^{M} \boldsymbol{\mathcal{E}}^l \otimes D\mathbf{f}(\mathbf{s}_l) + c\mathbf{L}\sum_{l=1}^{M} \mathbf{E}^l \otimes D\mathbf{H}(\mathbf{s}_l)]\Delta\mathbf{X}. \tag{7}$$

$\boldsymbol{\mathcal{E}}^l$ is an n_l-dimensional diagonal matrix, with $\mathcal{E}_{ii}^l = 1$ if $i \in \mathbf{V}_l$, and $\mathcal{E}_{ii}^l = 0$ otherwise. The key step to analyze the stability of the cluster synchronization is to construct a matrix $\mathbf{T}$ to

Algorithm 1 Actor-Critic method with Graph Convolutional Network

Initialize parameters $\theta^{(1)}, \theta^{(2)}$ in actor net 1 and 2, parameters ω in critic net, learning rate $\alpha_{\theta 1}, \alpha_{\theta 2}, \alpha_{\omega}$;
for episode $i = 1, I$ **do**
  Input the initial graph G_0 to get the initial state s_0;
  Input s_0 to Node2Vec to obtain the feature vector $\mathbf{u}_i$ for each node;
  while step $k = 0, K$ **do**
    Input feature vectors and state s_k to critic net to estimate the value of current state r_k;
    Input feature vectors and state s_k to actor net 1 to generate starting node v_i by policy π_θ;
    Generate a new feature vector $\mathbf{u}_i'$ for each node according to feature vector $\mathbf{u}_i$ and starting node.
    Input node feature vector $\mathbf{u}_i'$ and state s_k into actor net 2 to generate ending node v_j according to policy π_θ;
    State transits from s_k to the next state s_{k+1} after taking action $a_k = (v_i, v_j)$;
    Input feature vectors $\mathbf{u}_i'$ and state s_{k+1} to critic net to estimate the value of current state r_{k+1};
  end while
  Sample trajectory $\{s_0, a_0, r_0, s_1, a_1, r_1, \ldots, s_K, r_K\}$
  Compute TD error $\delta_k = r_k + \gamma V^{\pi_\theta}(s_{k+1}) - V^{\pi_\theta}(s_k)$
  Update parameters of critic net by $\omega = \omega + \eta_\omega \sum_k \phi_k \nabla_\omega \mathcal{V}(s_k)$
  Update parameters of actor net 1 and 2 by $\theta^{(1)} = \theta^{(1)} + \alpha_{\theta 1} \nabla J(\theta^{(1)})$ and $\theta^{(2)} = \theta^{(2)} + \alpha_{\theta 2} \nabla J(\theta^{(2)})$
end for

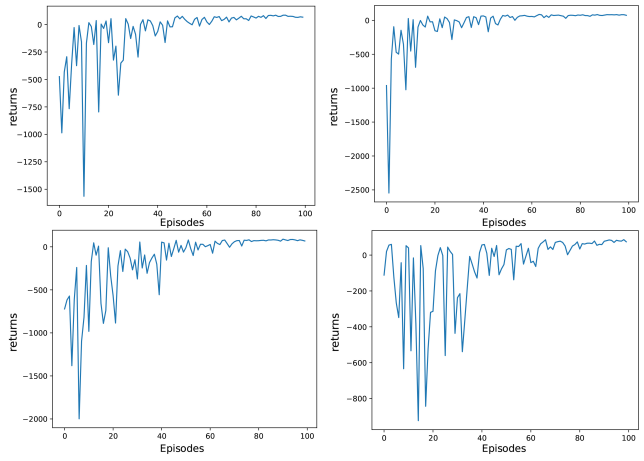


Fig. 5. This figure shows the training process of the Rossler system with different initial network structures.

decouple the clusters from each other in a block form as

$$\mathbf{L}' = \mathbf{T}^{-1}\mathbf{L}\mathbf{T} = \begin{bmatrix} \mathbf{C} & 0 & \ldots & 0 \\ 0 & \mathbf{B}_1 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & \mathbf{B}_M \end{bmatrix}, \tag{8}$$

so that the stability of the clusters can be treated individually

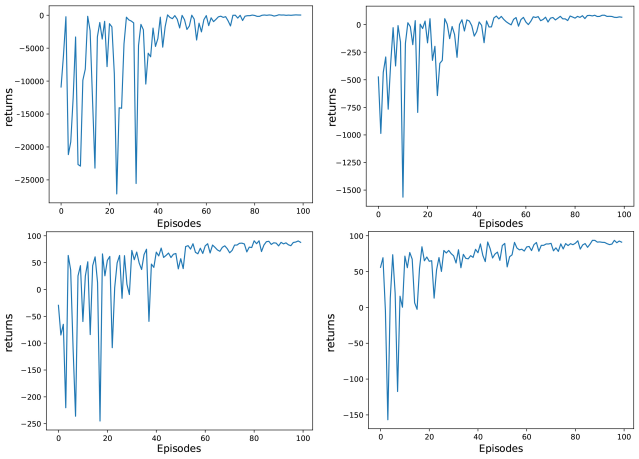


Fig. 6. This figure shows the training process of the Rossler system with different parameters, $c = 0.04, 0.05, 0.06, 0.07$.

TABLE I
TABLE OF SYMBOLS

Symbol	Description	Symbol	Description
G	undirected network	$\mathbf{V}$	node set of network
$\mathbf{E}$	edge set of network	N	number of nodes
$\top$	transpose of matrix	$\mathbf{x}_i$	state of node v_i
$\mathbf{f}(\cdot)$	self dynamics function	$\mathbf{H}(\cdot)$	inner coupling function
$\mathbf{L}$	Laplacian matrix	Ξ	synchronized region
$\mathbf{D}$	diagonal matrix of degrees	$\mathbf{A}$	adjacency matrix
$\lambda^{(i)}$	eigenvalue of matrix $\mathbf{L}$	$D\mathbf{f}(\cdot)$	Jacobian matrix of $\mathbf{f}$
$D\mathbf{H}(\cdot)$	Jacobian matrix of $\mathbf{H}$	$\boldsymbol{\zeta}$	collection of variations
$\Omega_{\max}$	largest Lyapunov exponent	c	coupling strength between nodes
ξ_i	bound of Ξ	Q	partition of network G
$\mathbf{V}_l$	cluster of nodes	M	number of clusters
$\mathbf{P}$	characteristic matrix	G^Q	quotient graph of G
$\mathbf{A}^Q$	adjacency matrix of quotient graph	$\mathbf{L}^Q$	Laplacian matrix of quotient graph
$\mathbf{s}_l$	synchronous manifold of the l-th cluster	n_l	number of nodes in cluster $\mathbf{V}_l$
$\otimes$	Kronecker product	$\Delta\mathbf{X}$	perturbation vector
$\boldsymbol{\mathcal{E}}^l$	an N-dimensional diagonal matrix of cluster l	$\mathbf{I}$	identity matrix
$\mathbf{C}$	a block of a matrix	$\mathbf{B}_l$	the l-th block of a matrix
$\mathcal{S}$	state space	$\mathcal{A}$	action space
r_k	reward of the state	$\mathcal{P}$	state transition function
$\mathcal{R}$	reward function	R_k	sum of discounted rewards
γ	discount factor	π_θ	policy of agent
$\mathcal{V}^{\pi_\theta}(s)$	state value function	$\boldsymbol{u}_i$	feature vector of node v_i
$\boldsymbol{\mathcal{H}}^l$	embedding feature matrix of nodes in neural network layer	ϕ	a nonlinear activation function
J	objective function	∇	gradient
$\mathcal{V}_\omega$	state value	η	learning rate

(The details of decoupling the clusters in a block form are shown in [11], [25]). With $\Delta\mathbf{Y} = \mathbf{T}^{-1}\Delta\mathbf{X}$, equation (7) can be converted to

$$\Delta\dot{\mathbf{Y}} = [\sum_{l=1}^{M} \widetilde{\boldsymbol{\mathcal{E}}}^l \otimes D\mathbf{f}(\mathbf{s}_l) + c\mathbf{L}' \sum_{l=1}^{M} \widetilde{\boldsymbol{\mathcal{E}}}^l \otimes D\mathbf{H}(\mathbf{s}_l)]\Delta\mathbf{Y}, \tag{9}$$

where $\widetilde{\boldsymbol{\mathcal{E}}}^l = \mathbf{T}^{-1}\mathbf{E}^l\mathbf{T}$. The equation for each cluster is

$$\Delta\dot{\mathbf{Y}}_l = (\mathbf{I}_{n-1} \otimes D\mathbf{f}(\mathbf{s}_l) + c\mathbf{B}_l \otimes D\mathbf{H}(s_l))\Delta\mathbf{Y}_l \tag{10}$$

With the existence of EEP, the eigenvalues that the original

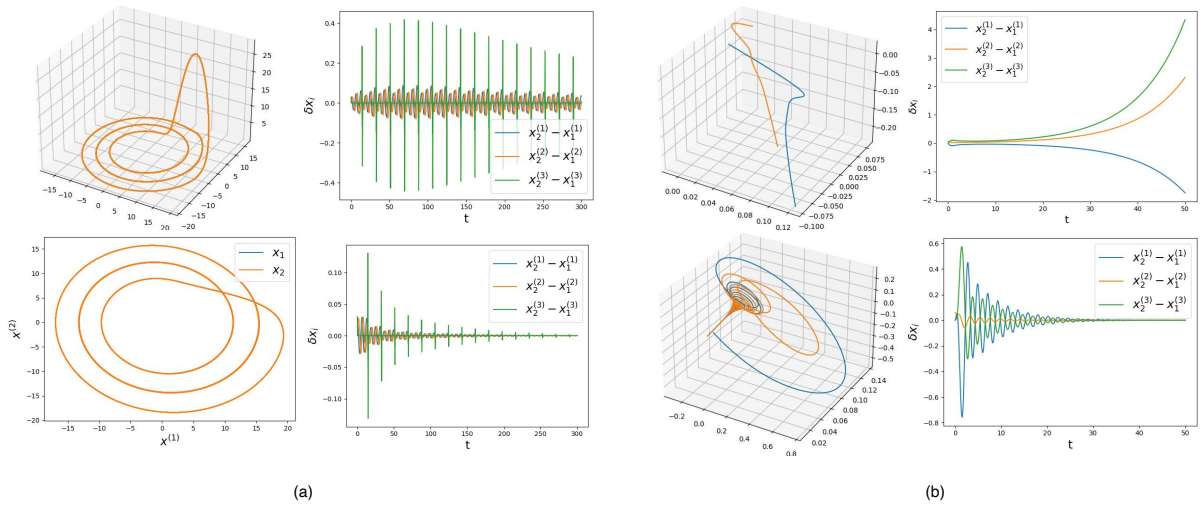


Fig. 7. (a) Alter network of Rossler system to synchronize to the limit cycle; (b) Alter network of Chua system to synchronize to the fixed point.

graph G shares with the quotient graph, $\lambda(\mathbf{L}^Q)$, are contained in the block $\mathbf{C}$, while the remaining in $\mathbf{B}_1, \mathbf{B}_2, \cdots, \mathbf{B}_M$. One criterion for stability of cluster synchronization with EEP [11] requires that Lyapunov exponents in equation (11) are negative. The variation of i-th node in the l-th cluster of equation (10) is

$$\delta\dot{\boldsymbol{\zeta}}_i = (D\mathbf{f}(\mathbf{s}_l) + c\lambda_{i,l} D\mathbf{H}(\mathbf{s}_l))\delta\boldsymbol{\zeta}_i, \tag{11}$$

where $\lambda_{i,l}$ with $i = 1, \ldots, n_l - 1$ indicate the eigenvalues of $\mathbf{B}_l$, which clearly are a subset of $\lambda(\mathbf{L})\backslash\lambda(\mathbf{L}^Q)$. Therefore, to achieve cluster synchronization, the complex network needs to satisfy (1) the existence of EEP; (2) not all $-c\lambda(\mathbf{L}^Q) \subseteq \Xi$ but $-c(\lambda(\mathbf{L})\backslash\lambda(\mathbf{L}^Q)) \subseteq \Xi$. For given self and coupling dynamics, the complex network can achieve the cluster synchronized state by adjusting the network topology according to this criterion. Global synchronization can be regarded as a particular kind of cluster synchronization, where all nodes of the network are grouped into a single cluster. In this situation, the partition $Q = \{\{v_1, v_2, \ldots, v_N\}\}$. The characteristic matrix $\mathbf{P} = [1, 1, \ldots, 1]^\top$. $\mathbf{L}^Q = (\mathbf{P}^\top\mathbf{P})^{-1}\mathbf{P}^\top\mathbf{L}\mathbf{P}$ and $\lambda(\mathbf{L}^Q) = 0$. Since $\lambda^{(0)} = 0$ in $\mathbf{L}$, $-c(\lambda(\mathbf{L})\backslash\lambda(\mathbf{L}^Q)) \subseteq \Xi$ equals to $\lambda^{(2)}, \lambda^{(3)}, \ldots, \lambda^{(N)}$ in $\mathbf{L}$ are belong to Ξ. This conclusion is the same with equation (4).

B. Network construction as a Markov decision process

As we mentioned above, for given self and coupling dynamics the complex network can achieve different synchronized patterns by adjusting the network topology. Given an initial network $G_0 = (\mathbf{V}, \mathbf{E}_0)$, the aim of optimizing the network is to perform a series of edge additions or deletions to G_0 such that the resulting network structure satisfies the corresponding synchronization criterion with minimally rewiring the network. The construction or optimization problem of networks can be cast as a sequential decision-making process. Therefore, we can construct or formalize networks as a Markov decision process (MDP), which is one possible formalization of a decision-making process, to get an ideal network structure to realize the objective. A decision maker is called the agent and the surrounding with which it interacts is called environment.

The MDP is defined by the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}(\cdot), \mathcal{R}(\cdot))$ as follows:

$\mathcal{S}$ is the set of states called the state space. The fully observable state is formulated as $s_k = G_k$, where s_k is a state of $\mathcal{S}$ and G_k is the current input network at step k. Normally, the network is represented by an adjacency matrix. Here we use an elegant and memory-efficient way of representing the sparse matrix as adjacency lists. These adjacency lists describe the connectivity of edge $e_{i,j}$ between nodes v_i and v_j as a tuple (i, j). (shown in Fig. 2)

$\mathcal{A}$ is the set of actions called the action space. The action imposed on the network is to add or delete an edge of the network. An action sequence consists of several actions $(a_1, a_2, ...a_k)$. If the selected edge $e_{i,j}$ formed by nodes v_i, v_j does not exist in the current network, $e_{i,j} \notin \mathbf{E}_k$, where $\mathbf{E}_k \subseteq \mathbf{V} \times \mathbf{V}$ is the edge set of graph G_k at step k, the action is to add $e_{i,j}$ to the network. Otherwise, $e_{i,j} \in \mathbf{E}_k$, the action is to delete $e_{i,j}$ from the network. The network must be connected after each action. For example, in Fig. (2), removal of the edge between node 1 and node 3 is not allowed because node 3 will be disconnected to the remaining nodes. The action sequence $(e_{2,3}, e_{1,3})$ is valid but $(e_{1,3}, e_{2,3})$ is not allowed. This means that the sequence of actions matters.

$\mathcal{P}(s_{k+1}, a_k, s_k)$ is the state transition function, which governs the probability of transiting from state s_k to the next state s_{k+1} after taking action a_k.

$\mathcal{R}(s_k, a_k, s_{k+1})$ is the reward function. $r_k = \mathcal{R}(s_k, a_k, s_{k+1})$ is the reward. The reward function is designed according to the pattern of synchronization. To achieve global synchronization, the extreme eigenvalues of the complex network must satisfy the criterion of equation

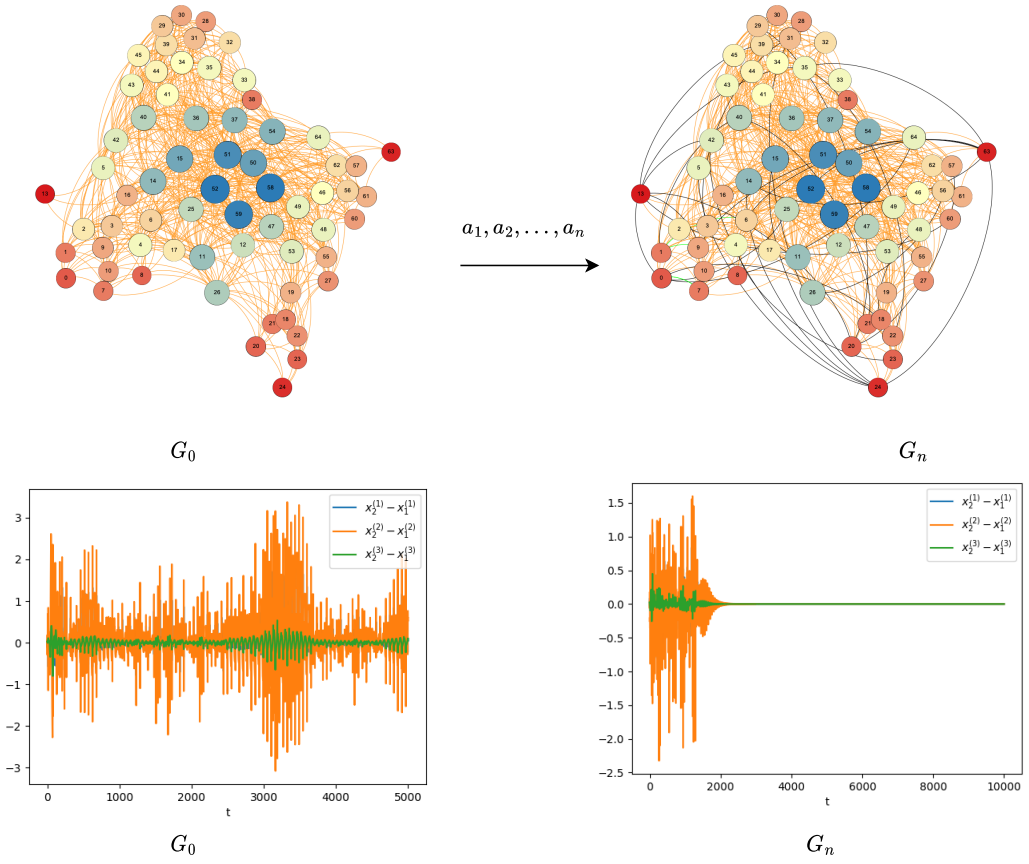


Fig. 8. This figure shows the synchronization process of cat's brain network. In the initial network G_0 and G_n edges of orange color are original edges. Black edges represent new edges added to the network and green edges represent edges which have been removed from the network. In the initial network, nodes cannot achieve the same state, $x_i - x_j \neq 0$. After rewiring the network according to the action sequence obtained from our method, all nodes in the network synchronize to the state, $x_i - x_j = 0$.

(4). The reward r_k at step k is defined as follows:

$$r_k = \begin{cases} 100, & \xi_1 < \lambda^{(2)} < \lambda^{(N)} < \xi_2 \\ \dfrac{1}{\lambda^{(N)} - \xi_2 + 1} - 1, & \xi_1 < \lambda^{(2)} < \xi_2 < \lambda^{(N)}, \\ \dfrac{1}{\xi_1 - \lambda^{(2)} + 1} - 1, & \lambda^{(2)} < \xi_1 < \lambda^{(N)} < \xi_2, \\ -2, & \text{otherwise}, \end{cases} \tag{12}$$

where $\Xi = (\xi_1, \xi_2)$ is the stable region.

To achieve the cluster synchronized state, the reward r_k is defined as :

$$r_k = \begin{cases} 100, & \Psi_1 = 1, \\ -(M - M_1), & \Psi_1 = 0 \,\&\, \Psi_2 = 0, \\ -1, & \Psi_1 = 0 \,\&\, \Psi_2 = 1. \end{cases} \tag{13}$$

We set $\Psi_2 = 1$ if EEP exists, otherwise $\Psi_2 = 0$. If the complex network satisfies (1) existence of EEP; (2)$\exists - c\lambda^{(i)}(\mathbf{L}^Q) \notin \Xi$; (3)$-c(\lambda(\mathbf{L}) \backslash \lambda(\mathbf{L}^Q)) \subseteq \Xi$ at the same time, the network achieves cluster synchronize and $\Psi_1 = 1$. Otherwise $\Psi_1 = 0$. M is the number of clusters under the partition $Q = \{\mathbf{V}_1, \mathbf{V}_2, ..., \mathbf{V}_M\}$. M_1 is the number of clusters $\mathbf{V}_l$ where each node $v_i \in \mathbf{V}_l$ has exactly b_{lk} neighbours in $\mathbf{V}_k$ with $l \neq k$. In MDP, return $\mathcal{R}_k$ is defined as the sum of discounted rewards from step k to the end of the process with discount factor $\gamma \in [0, 1]$. The objective to alter the network to achieve synchronization is realized by obtaining higher returns after training. Return R_k is calculated by

$$R_k = r_k + \gamma r_{k+1} + \gamma^2 r_{k+2} + ... = \sum_{j=0}^{\infty} \gamma^j r_{k+j}. \tag{14}$$

The policy π_θ of the agent is a function which maps states to actions. $\pi_\theta(a|s)$ is the distribution of actions over the states s. $\mathcal{V}^{\pi_\theta}(s)$ is the state value function of state s based on policy π_θ

C. Learning to alter complex networks

According to the MDP, we know that each time the agent takes an action, the structure of the complex network will change. The action taken by the agent is closely related to the current network structure. In a traditional actor-critic method, the neural networks are used as actor net and critic net to generate policy and estimate the state value. Traditional neural network methods cope with graph structured data by preprocessing the graph-structured information to a simpler

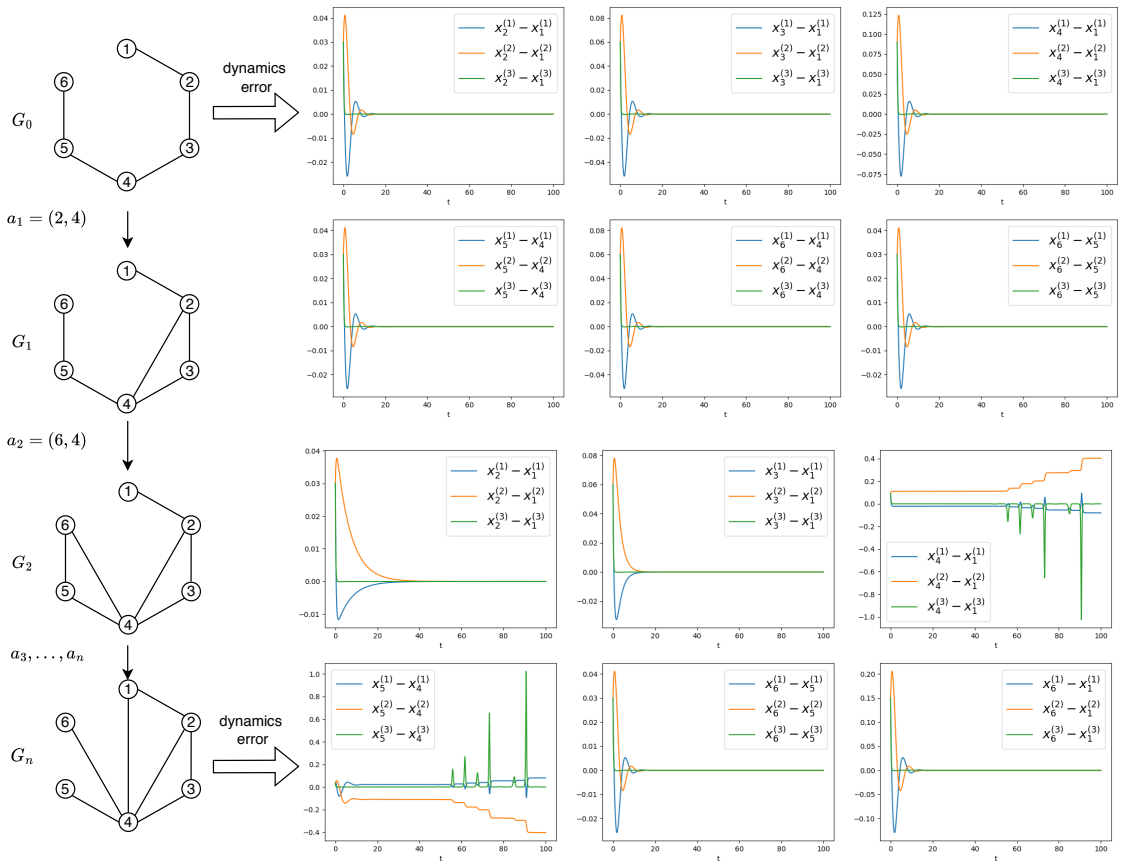


Fig. 9. This figure shows the process of cluster synchronization. Our goal is to enable nodes 1 ,2 ,3 ,5 ,6 group into a cluster and synchronize to the same state. Node 4 forms a cluster alone and achieves a different state from the first cluster. In the initial network G_0, all nodes synchronize to the same state. After several actions by our method, a new network G_n is constructed. Nodes 1, 2, 3, 5, 6 in this complex network synchronize to the same state but node 4 achieves to a different state.

representation, e.g. vectors of reals. The preprocessing step first 'squashes' the graph-structured data into a vector of reals and then deals with the preprocessed data using a list-based data processing technique. However, important information, e.g., the topological dependency of information on each node may be lost during the preprocessing stage. Graph neural networks can preserve the graph-structured nature of the data by encoding the underlying graph-structured data using topological relationships among the nodes of the graph [36]. Compared with the traditional neural network, graph neural network is more suitable for dealing with graph. Graph convolutional network as a kind of graph neural network method which is highly efficient and accurate in classification and prediction on graph is employed here as actor net and critic net to generate actions and estimate state value respectively. The algorithm procedure is shown in algorithm (I).

Node Embedding Typical machine learning algorithms require a set of informative, discriminating, and independent features. This means that one has to construct a feature vector representation for nodes and edges in prediction problems on networks. Traditional methods often use handcrafted domain-specific features based on expert knowledge to represent nodes and graphs, such as global or local degree distribution. Here, we leverage Node2Vec [37] to construct the feature vector representation. Node2Vec as a node embedding method refers to the process of mapping nodes to a low-dimensional space of features which capture the graph structure information by random walks through a graph. Nodes with similar attributes or connectivity e.g., highly interconnected, have similar structure roles in the graph, are mapped closer together in the feature space [37]. The similarity of nodes affects the synchronization of the network. In [24], they pointed out that adding links between nodes with lower similarity of structure can improve synchronizability of networks more, compared to adding links between nodes with higher similarity of structure. A feature vector for node v_i is set as $\mathbf{u}_i$.

Actor Net The Actor Net is used to generate actions to delete or add edges to the complex network. The number of potential actions imposed on the complex network with N nodes is proportional to N^2. To reduce action space, the edge selection is decomposed into two steps: first selecting the starting node and then the ending node (shown in Fig. (3)). Selecting the starting and ending nodes are predicted by two independent GCN, respectively. Every neural network layer in GCN can be written as a nonlinear function

$$\boldsymbol{\mathcal{H}}^{(l+1)} = \phi(\boldsymbol{\mathcal{H}}^{(l)}, \mathbf{A}), \tag{15}$$

with $\boldsymbol{\mathcal{H}}^{(0)} = \mathbf{U}$, where $\boldsymbol{\mathcal{H}}^{(l)}$ is the embedding feature matrix of nodes in l-th layer. $\mathbf{A}$ is the adjacency matrix of the input complex network. $\mathbf{U} = (\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_N)$ is a $N \times N_1$ feature

matrix obtained by Node2Vec. l represents the l-th layer of neural network. The specific models then differ only in how $\phi(\cdot)$ is chosen and parameterized.

Critic Net The value of the state $\mathcal{V}_\omega(s_k)$ is predicted by an GCN. Policy gradient expression is defined as:

$$\nabla_\theta J(\theta) = \mathbb{E}[\sum_{k=0}^{K} \psi_k \nabla_\theta \log \pi_\theta(a_k|s_k)]. \tag{16}$$

ψ_k has a lot of forms to choose, and temporal difference (TD) error is used in this paper as follows:

$$\psi_k = r_k + \gamma \mathcal{V}_\omega(s_{k+1}) - \mathcal{V}_\omega(s_k). \tag{17}$$

The TD error is used to update parameters of actor net and critic net. Parameters θ in actor network are updated by $\theta = \theta + \eta_\theta \nabla_\theta J(\theta)$ and parameters ω in critic network are updated by $\omega = \omega + \eta_\omega \sum_k \phi_k \nabla_\omega \mathcal{V}(s_k)$.

III. Results

Case 1: Consider a Rossler system [38] for all nodes. The dynamic function of the i-th oscillator is

$$\begin{aligned} \dot{x_i}^{(1)} &= -(x_i{}^{(2)} + x_i{}^{(3)}) \\ \dot{x_i}^{(2)} &= x_i{}^{(1)} + 0.2 x_i{}^{(2)} \\ \dot{x_i}^{(3)} &= 0.2 + x_i{}^{(3)}(x_i{}^{(1)} - 5.7). \end{aligned} \tag{18}$$

The coupling strength $c = 0.05$. Nodes are coupled by $x^{(1)}$ and The inner function H is

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}. \tag{19}$$

According to the largest Lyapunov exponent, the stability region is $\Xi = (-5.4, -0.2)$. The initial complex network G_0 is set as

$$\begin{aligned} &[1,1,2,2,3,3,4,4,5,5,6,6,7,7,8,8,9,9,10,10] \\ &[2,10,1,3,2,4,3,5,4,6,5,7,6,8,7,9,8,10,9,1]. \end{aligned} \tag{20}$$

The extreme eigenvalues of the initial network G_0 are $\lambda^{(2)} = 0.38$ and $\lambda^{(N)} = 4.0$. $-c\lambda^{(i)} \notin \Xi$, so nodes in the initial network G_0 cannot achieve the same stable state (shown in Fig. (4)). To achieve global synchronized state, an action sequence is obtained by our proposed method after training 100 episodes (shown in Fig. (5) (a)). After actions taken to G_0, G_n is obtained and the extreme eigenvalues of G_n are $\lambda^{(2)} = 4.03, \lambda^{(N)} = 9.35$. $-c\lambda^{(i)} \in \Xi$ and all nodes achieve the same state shown in Fig. (4)

TABLE II
Effects of initial connectivity on resulting network

c	Initial Network		Resulting Network		$-\Xi/c$	Synch-ronize
	$\lambda^{(2)}$	$\lambda^{(N)}$	$\lambda^{(2)}$	$\lambda^{(N)}$		
0.05	0.38	4.0	4.03	9.35	$(4.0, 108)$	Yes
0.05	0.35	4.38	4.40	9.69	$(4.0, 108)$	Yes
0.05	0.38	4.50	4.33	9.25	$(4.0, 108)$	Yes
0.05	0.44	4.56	4.26	9.07	$(4.0, 108)$	Yes

To check whether our method is valid in different initial networks, the resulting networks G_n with different initial networks G_0 are shown in Table (II). Their training process is shown in Fig. (5). It clearly shows that for different connectivity of G_0, the training process and resulting network are different, but all resulting networks satisfy $-c\lambda^{(i)} \in \Xi$ and they all achieve the synchronized state at last. This means that our method is valid in different initial networks.

TABLE III
Effects of different system parameters on resulting network

c	Initial Network		Resulting Network		$-\Xi/c$	Synch-ronize
	$\lambda^{(2)}$	$\lambda^{(N)}$	$\lambda^{(2)}$	$\lambda^{(N)}$		
0.04	0.38	4.0	5.24	9.71	$(5.0, 135)$	Yes
0.05	0.38	4.0	4.40	9.69	$(4.0, 108)$	Yes
0.06	0.38	4.0	3.5	9.02	$(3.33, 90)$	Yes
0.07	0.38	4.0	2.88	8.89	$(2.86, 77.1)$	Yes

TABLE IV
Comparison of methods in different networked systems

		Number of nodes					
		10	20	50	100	200	500
Small-world Network	Method 1	13	41	61	136	257	387
	Method 2	10	32	57	102	227	342
	Method 3	9	25	33	53	124	191
	RL	**8**	**23**	**32**	**52**	**122**	**187**
Scale-free Network	Method 1	9	22	75	102	246	329
	Method 2	8	18	52	89	207	296
	Method 3	**6**	11	41	57	127	141
	RL	**6**	**9**	**38**	**54**	**125**	**139**

Second, we explore how the variation of system parameters affects the resulting network, for example, different values of parameter c. $c = 0.04, 0.05, 0.06, 0.07$ and the initial network G_0 for these 4 cases are set the same as equation (20). The resulting networks are shown in Table (III) and the training processes are shown in Fig. (6). Though the training processes and resulting networks are different, all resulting networks achieve the synchronized state after optimizing by our method. Eigenvalues should satisfy $-c\lambda^{(i)} \in \Xi$ and $\Xi = (-5.4, -0.2)$ in equation (18). To achieve a synchronized state, eigenvalues of different networks satisfy $\lambda^{(i)} \in (0.2/c, 5.4/c)$. It requires more rewiring to the initial network to make $\lambda^{(i)} \in \Xi$ with smaller c, which causes lower returns. That is why the lowest returns of the training process increase with c. If c is very small, $0.2/c$ becomes very large, which may cause no networks satisfying $\lambda^{(2)} > 0.2/c$. No achievable synchronized state exists in this situation and our method is invalid. Therefore, our method is valid with different values of parameter c, if the achievable synchronized states exist. Our method is also tested to alter the network to synchronize to a fixed point and a limited cycle in Fig. (7).

Case 2: In this case, we consider a large-scale complex network in the real world - the cat's brain network (the real data is available in networkrepository) [39]. This network consists of 65 nodes and about 1000 edges. The nodes in the network represent neurons and edges represent fiber tracts that connect one node to another. The dynamics of neurons are described by the Hindmarsh-Rose (HR) neuronal circuit [6],

[40] as

$$\begin{aligned}\dot{x_i}^{(1)} &= x_i^{(2)} - a{x_i^{(1)}}^3 + b{x_i^{(1)}}^2 - x_i^{(3)} + I_e \\ \dot{x_i}^{(2)} &= d - e{x_i^{(1)}}^2 - x_i^{(2)} \\ \dot{x_i}^{(3)} &= r[s(x_i^{(1)} + 1.6) - x_i^{(3)}].\end{aligned} \tag{21}$$

a, b, d, e, r, s are the system parameters which are set as $a = 1, b = 3, d = 1, e = 5, r = 6 * 10^{-2}, s = 4$. $I_e = 320\mu A$ is the external forcing current. The inner coupling function H is the same as (19) and the coupling strength c is set as 1/30. According to the largest Lyapunov exponent, the stable region Ξ is $(-\infty, -0.23)$. The extreme eigenvalues of the initial complex network G_0 are 2.88 and 46.53. $-c\lambda^{(2)} \notin \Xi$, so nodes in the initial complex network cannot synchronize to the same state (shown in Fig. (8)). To achieve global synchronized state, an action sequence is obtained by our proposed method after training. G_n is obtained according to the action sequence. The extreme eigenvalues of G_n are 7.03 and 47.24. $-c\lambda^{(i)} \in \Xi$ and all nodes synchronize to the same state (shown in Fig. (8)).

Our method is compared with other existing methods in different network models with different sizes, such as accept-reject rewiring mechanism based on random selection [41], rewiring method based on nodes degree [21], accept-reject rewiring mechanism based on eigenvector [23] corresponding to the method 1, 2, 3 in Table (IV). The table shows that our method rewires the network with minimum number of rewiring edges compared with other methods. Another advantage of our method is that the exploration ability is better than greedy algorithms based on network properties and can find a rich set of optimal solutions. The advantage of methods 2 and 3 is that they are time efficient since they are both a kind of greedy algorithm.

Case 3: In this case, we employ a simple example to show how the network achieves cluster synchronization. Considering the Rossler function (18) for all nodes The coupling strength $c = 1$. The initial network G_0 is shown in Fig. (9) and extreme eigenvalues of G_0 are $\lambda^{(2)} = 0.2679, \lambda^{(N)} = 3.7321$. According to the largest Lyapunov exponent, the stale region of this system is $\Xi = (-5.4, -0.2)$. $-c\lambda^{(i)} \in \Xi$ and all nodes synchronize to the same state. Now, we want nodes to synchronize to different states. For example, nodes $1, 2, 3, 5, 6$ as a cluster synchronize to one state, and node 4 synchronizes to a different state. To achieve the cluster synchronized state, the complex network needs to satisfy (1) the existence of EEP; (2) not all $-c\lambda(\mathbf{L}^Q) \subseteq \Xi$ but $-c(\lambda(\mathbf{L})\backslash\lambda(\mathbf{L}^Q)) \subseteq \Xi$. The graph convolution network with reinforcement learning method takes a sequence of actions to the network and the final network G_n is obtained after training as

$$[1, 1, 2, 2, 2, 3, 3, 4, 4, 4, 4, 4, 5, 6]$$
$$[2, 4, 1, 3, 4, 2, 4, 1, 2, 3, 5, 6, 4, 6].$$

The characteristic matrix $\mathbf{P}$ is

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}^\top.$$

The Laplacian matrix of the quotient graph is $\mathbf{L}^Q = (\mathbf{P}^\top\mathbf{P})^{-1}\mathbf{P}^\top =$

$$\begin{bmatrix} 1 & -1 \\ -5 & 5 \end{bmatrix}.$$

Eigenvalues of the Laplacian matrix $\mathbf{L}(G_n)$ of G_n are $0, 1, 1, 2, 4, 6$. Eigenvalues of the Laplacian matrix of quotient graph $\mathbf{L}^Q(G_n)$ are $0, 6$. In $\mathbf{L}(G_n)$, $-c\lambda^{(6)} \notin \Xi$ but $-c(\lambda(\mathbf{L}(G_n)\backslash\lambda(\mathbf{L}^Q(G_n)) \subseteq \Xi$. The dynamics process of G_n is shown in Fig. (9). It is clearly shown that nodes $1, 2, 3, 5, 6$ synchronize to the same state but are different from node 4. Therefore, our method is valid to alter the network to achieve a stable cluster synchronization state.

IV. Conclusions

In this paper, we have proposed a framework to alter network structure to achieve cluster synchronization. We construct the network as a Markov decision process and then use Actor-Critic reinforcement learning framework with graph convolutional network to learn how to alter complex network to achieve synchronization. The reward function has been designed based on the criterion for synchronization, which makes our method flexible to alter network to achieve different synchronization patterns by redesigning the reward function. This means that our method can be widely applied in different scenarios. Also, it is beneficial to reduce computation complexity by avoiding calculating dynamic equations of each node after every rewiring of the network. Our method has been verified in classical Rossler dynamic networks and Hindmarsh-Rose neuronal circuits. We set different initial network structures and dynamical system parameters to verify the generality of our method. The method is always valid to alter the network to achieve synchronization if the achievable synchronized states exist. Also, we have analyzed how the initial networks and system parameters affect the training process and resulting networks. In the future, we will study how to use reinforcement learning methods with reward function designed by the state error of nodes to discover different network structures besides EEP or symmetry to achieve cluster synchronization.Based on the new structures, it probably helps researchers to establish an analytic criterion for cluster synchronization with a more relaxed structure condition.

References

[1] G. Moutsinas and W. Guo, "Probabilistic stability of traffic load balancing on wireless complex networks," *IEEE Systems Journal*, vol. 14, no. 2, pp. 2551–2556, 2020.

[2] F. Dörfler, M. Chertkov, and F. Bullo, "Synchronization in complex oscillator networks and smart grids," *Proceedings of the National Academy of Sciences*, vol. 110, no. 6, pp. 2005–2010, 2013.

[3] Y. Zong, X. Dai, Z. Wei, M. Zou, W. Guo, and Z. Gao, "Robust time synchronisation for industrial internet of things by h ∞ output feedback control," *IEEE Internet of Things Journal*, pp. 1–1, 2022.

[4] C. Chen, L. Li, H. Peng, Y. Yang, L. Mi, and H. Zhao, "A new fixed-time stability theorem and its application to the fixed-time synchronization of neural networks," *Neural networks*, vol. 123, pp. 412–419, 2020.

[5] X. Zhang, C. Li, and Z. He, "Cluster synchronization of delayed coupled neural networks: Delay-dependent distributed impulsive control," *Neural Networks*, vol. 142, pp. 34–43, 2021.

[6] S. Malik and A. H. Mir, "Synchronization of hindmarsh rose neurons," *Neural Networks*, vol. 123, pp. 372–380, 2020.

[7] T. Chen, "Synchronization of multi-cluster complex networks," *Neural Networks*, vol. 156, pp. 239–243, 2022.

[8] G. Moutsinas, W. Guo, and T. Charalambous, "On the stability of the foschini-miljanic algorithm with uncertainty over channel gains," in *2019 18th European Control Conference (ECC)*, 2019, pp. 1524–1530.

[9] L. M. Pecora and T. L. Carroll, "Master stability functions for synchronized coupled systems," *Phys. Rev. Lett.*, vol. 80, no. 10, p. 2109, 1998.

[10] L. M. Pecora, F. Sorrentino, A. M. Hagerstrom, T. E. Murphy, and R. Roy, "Cluster synchronization and isolated desynchronization in complex networks with symmetries," *Nature communications*, vol. 5, no. 1, pp. 1–8, 2014.

[11] L. V. Gambuzza and M. Frasca, "A criterion for stability of cluster synchronization in networks with external equitable partitions," *Automatica*, vol. 100, pp. 212–218, 2019.

[12] W. Lu, B. Liu, and T. Chen, "Cluster synchronization in networks of coupled nonidentical dynamical systems," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 20, no. 1, p. 013120, 2010.

[13] Z. Aminzare, B. Dey, E. N. Davison, and N. E. Leonard, "Cluster synchronization of diffusively coupled nonlinear systems: A contraction-based approach," *Journal of Nonlinear Science*, vol. 30, no. 5, 2020.

[14] G. Russo and J.-J. E. Slotine, "Symmetries, stability, and control in nonlinear systems and networks," *Physical Review E*, vol. 84, no. 4, p. 041929, 2011.

[15] M. T. Schaub, N. O'Clery, Y. N. Billeh, J.-C. Delvenne, R. Lambiotte, and M. Barahona, "Graph partitions and cluster synchronization in networks of oscillators," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 26, no. 9, p. 094821, 2016.

[16] A. Arenas, A. Díaz-Guilera, J. Kurths, Y. Moreno, and C. Zhou, "Synchronization in complex networks," *Physics reports*, vol. 469, no. 3, pp. 93–153, 2008.

[17] P. D. Rungta, A. Choudhary, C. Meena, and S. Sinha, "Are network properties consistent indicators of synchronization?" *EPL (Europhysics Letters)*, vol. 117, no. 2, p. 20003, 2017.

[18] M. Zou and W. Guo, "Local assortativity affects the synchronizability of scale-free network," *IEEE Systems Journal*, 2022.

[19] J. Aguirre, R. Sevilla-Escoboza, R. Gutierrez, D. Papo, and J. Buldú, "Synchronization of interconnected networks: the role of connector nodes," *Physical review letters*, vol. 112, no. 24, p. 248701, 2014.

[20] Z. He, C. Yao, J. Yu, and M. Zhan, "Perturbation analysis and comparison of network synchronization methods," *Physical Review E*, vol. 99, no. 5, p. 052207, 2019.

[21] S.-J. Wang, Z.-X. Wu, H.-R. Dong, and G. Chen, "Enhancing the synchronizability of scale-free networks by adding edges," *International Journal of Modern Physics C*, vol. 21, no. 01, pp. 67–77, 2010.

[22] A. Hagberg and D. A. Schult, "Rewiring networks for synchronization," *Chaos: An interdisciplinary journal of nonlinear science*, vol. 18, no. 3, 2008.

[23] S. H. Ghorban, B. Hesaam, and H. Sarbazi-Azad, "A rewiring mechanism to improve synchronization in kuramoto networks," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2022, no. 11, 2022.

[24] Y. Luan, X. Wu, and B. Liu, "Maximizing synchronizability of networks with community structure based on node similarity," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 32, no. 8, 2022.

[25] W. Lin, H. Fan, Y. Wang, H. Ying, and X. Wang, "Controlling synchronous patterns in complex networks," *Physical Review E*, vol. 93, no. 4, p. 042209, 2016.

[26] L. V. Gambuzza, M. Frasca, F. Sorrentino, L. M. Pecora, and S. Boccaletti, "Controlling symmetries and clustered dynamics of complex networks," *IEEE Transactions on Network Science and Engineering*, vol. 8, no. 1, pp. 282–293, 2020.

[27] Y. Yang, B. Kiumarsi, H. Modares, and C. Xu, "Model-free λ-policy iteration for discrete-time linear quadratic regulation," *IEEE Transactions on Neural Networks and Learning Systems*, 2021.

[28] Y. Yang, Y. Pan, C.-Z. Xu, and D. C. Wunsch, "Hamiltonian-driven adaptive dynamic programming with efficient experience replay," *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[29] Y. Yang, H. Modares, K. G. Vamvoudakis, and F. L. Lewis, "Cooperative finitely excited learning for dynamical games," *IEEE Transactions on Cybernetics*, 2023.

[30] J. Kallestad, R. Hasibi, A. Hemmati, and K. Sörensen, "A general deep reinforcement learning hyperheuristic framework for solving combinatorial optimization problems," *European Journal of Operational Research*, vol. 309, no. 1, pp. 446–468, 2023.

[31] Z. Li, X. Wang, L. Pan, L. Zhu, Z. Wang, J. Feng, C. Deng, and L. Huang, "Network topology optimization via deep reinforcement learning," *IEEE Transactions on Communications*, 2023.

[32] V.-A. Darvariu, S. Hailes, and M. Musolesi, "Goal-directed graph construction using reinforcement learning," *Proceedings of the Royal Society A*, vol. 477, no. 2254, p. 20210168, 2021.

[33] F. Comellas and S. Gago, "Spectral bounds for the betweenness of a graph," *Linear Algebra and its Applications*, vol. 423, no. 1, 2007.

[34] C. Godsil and G. F. Royle, *Algebraic graph theory*. Springer Science & Business Media, 2001, vol. 207.

[35] N. O'Clery, Y. Yuan, G.-B. Stan, and M. Barahona, "Observability and coarse graining of consensus dynamics through the external equitable partition," *Physical Review E*, vol. 88, no. 4, p. 042805, 2013.

[36] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE transactions on neural networks*, vol. 20, no. 1, pp. 61–80, 2008.

[37] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *ACM SIGKDD international conference on Knowledge discovery and data mining*, 2016, pp. 855–864.

[38] O. E. Rössler, "An equation for continuous chaos," *Physics Letters A*, vol. 57, no. 5, pp. 397–398, 1976.

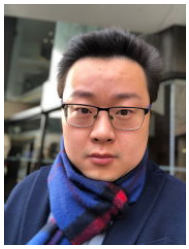
[39] R. A. Rossi and N. K. Ahmed, "The network data repository with interactive graph analytics and visualization," in *AAAI*, 2015. [Online]. Available: https://networkrepository.com

[40] R. Rose and J. Hindmarsh, "The assembly of ionic currents in a thalamic neuron. ii. the stability and state diagrams," *Proceedings of the Royal Society of London. B. Biological Sciences*, vol. 237, no. 1288, pp. 289–312, 1989.

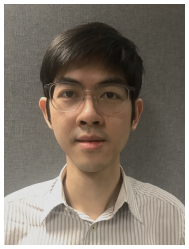
[41] P. S. Skardal, D. Taylor, and J. Sun, "Optimal synchronization of complex networks," *Physical review letters*, vol. 113, no. 14, 2014.



Mengbang Zou received his M.Sc degree from Huazhong University of Science and Technology in Mechanical Science and Engineering in 2019. He is now a Ph.D. student in Aerospace School of Cranfield University. He is interested in complex system with nonlinear dynamics including complex system's resilience, synchronization and region of attraction. He also studies the optimization problems on graphs by reinforcement learning methods and graph neural network.



Weisi Guo (S07, M11, SM17) received his MEng, MA, and Ph.D. degrees from the University of Cambridge. He is Chair Professor of Human Machine Intelligence at Cranfield University. He has published over 250 papers and is PI on over 12 research projects from EPSRC, Royal Society, EC H2020, and InnovateUK. His research has won him several international awards (IET Innovation 15, Bell Labs Prize Finalist 14 and Semi-Finalist 16 and 19). He was a Turing Fellow at the Alan Turing Institute and Fellow of Royal Statistical Society.



Kai-Fung Chu (S'17–M'20) received his Ph.D. degree in at the University of Hong Kong in 2020, and his M.Sc. and B.Eng. (First Class Honors) degrees both in Electronic and Information Engineering from The Hong Kong Polytechnic University in 2016 and 2013. He is a Marie Skłodowska-Curie Fellow at the University of Cambridge. He was a Research Assistant Professor at the Hong Kong Polytechnic University and a research fellow at Cranfield University. His research interests include machine intelligence, autonomous vehicles, and intelligent transportation systems.



https://dspace.org/

Rewiring complex networks to achieve cluster synchronization using graph convolution networks with reinforcement learning

Zou, Mengbang; Guo, Weisi; Chu, Kai-Fung

2024-09-01
Institute of Electrical and Electronics Engineers (IEEE)
https://doi.org/10.1109/TNSE.2024.3410694
Zou M, Guo W, Chu K-F. (2024) Rewiring complex networks to achieve cluster synchronization using graph convolution networks with reinforcement learning. IEEE Transactions on Network Science and Engineering. Volume 11, Issue 5, September-October 2024, pp. 4293-4304

Downloaded from a DSpace repository. https://dspace.org/