Journal of Computer and Communications, 2022, 10, 26-40
https://www.scirp.org/journal/jcc
ISSN Online: 2327-5227
ISSN Print: 2327-5219

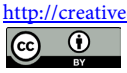
Research on the Controllability of Software Networks Based on the Source-Driven Model

He Tian[1,2], Xianwei Meng[1], Nan Hu[1], Yanchao Wang[1], Tianyi Yang[1]

[1]School of Sugon Big Data, Liaoning Institute of Science and Technology, Benxi, China
[2]College of Information, Liaoning University, Shenyang, China
Email: tianher@sina.cn

How to cite this paper: Tian, H., Meng, X.W., Hu, N., Wang, Y.C. and Yang, T.Y. (2022) Research on the Controllability of Software Networks Based on the Source-Driven Model. *Journal of Computer and Communications*, **10**, 26-40.
https://doi.org/10.4236/jcc.2022.108003

Received: July 21, 2022
Accepted: August 12, 2022
Published: August 15, 2022



Abstract

The complexity of software system has been increasing with software evolution, which affects the stability of software structure. Most of the existing measurement methods focus on the analysis of the macro-characteristics of the network topology, but lacked a certain depth and expansion to explore the nature of the complexity of the software structure, for this purpose, the complex network control theory was applied to the study of software network controllability. Firstly, the Source-Driver (SD) model was established based on the system control theory, the driver node sets were obtained by the minimum input theorem in the control process of software network topology; Then the relationship between the degree and center degree, the relationship between the in-degree and the out-degree of the software network topology were further analyzed owing to the non-uniqueness of the driver node sets; Finally, the values of the four indicators in the software system were compared. Experimental results show that the driver node sets in the software networks are mainly composed of nodes with low degree values, but it does not mean that the nodes whose in-degree values and out-degree values are also low; The action on control nodes and driver nodes are not random, the controllability of the driver nodes is closely related to the in-degree, when selecting the driver node sets, the network topology characteristics should be considered comprehensively, and the nodes with high degree and center degree are the first choice. The results have important guiding significance for the control, maintenance and redesign of software architecture.

Keywords

Complex Networks, Software Networks, Driver Node Sets, Network Control

1. Introduction

In the context of the Internet era, with the vigorous development of big data

technology, software systems based on network have rapidly increased in scale, number of users, and interaction of component elements have become a kind of complex system [1]. Through the interdisciplinary integration of software engineering and complex systems, abstract software systems into the artificial complex networks (*i.e.* software networks) from the perspective of complex systems and complex networks, which explore and discover the structural characteristics, evolution rules and behavior characteristics of complex software systems from the whole and global perspectives, it is helpful to understand the essential characteristics of software scientifically and comprehensively [2].

Software networks have been in development for more than a decade, and many research methods have achieved results in the measurement of software static structure characteristics [3] [4] and software dynamic behavior [5] [6]. However, the application environment is becoming more and more complex, which increases the risk of software development. The quality of software products is hard to be guaranteed. With the deepening of the research on complex networks theory, people explore the methods to improve the complex networks [7] [8]. Since Liu *et al.* [9] first studied the structural control of complex networks from the idea of linear system cybernetics in 2011, people began to explore how to effectively control the structure and behavior of complex dynamic systems. He established the controllability model of complex networks by introducing the matching theory of directed graph and the structural controllability theorem, inputting the control signals to the least driver node sets to completely control the entire network. On this basis, Yuan *et al.* [10] determined the number of control input nodes using PHB matrix criterion and eigenvalues of network dynamics coupling matrix. Zhang *et al.* [11] proposed a kind of random matching method to obtain the minimum driver node sets in networks and analyze the structure of nodes in them. Pósfai *et al.* [12] defined the correlation coefficient between in-degree and out-degree, in-degree and in-degree, out-degree and in-degree, out-degree and out-degree, and analyzed their relationship with network controllability. Liu *et al.* [13] researched the influence of the correlation between out-degree and in-degree (*i.e.* degree correlation) using controllable limit theory on the controllability of real network edges, through the analysis and calculation, it showed that the upper and lower controllable limits were applicable to all types of real networks, which deepened the understanding of the controllability of real network edges.

The research on complex networks provides strong support for exploring the structural characteristics of large-scale software systems [14] [15], Liu *et al.* [16] used an improved box covering algorithm to analyze the multi-fractal of software networks to enhance the quality of software systems. Ren *et al.* [17] proposed a new method for mining top-k key nodes in directional weighted complex software networks based on semi global information, and analyzed the stability, reliability and robustness of software networks by studying very few key functional nodes. Zhang *et al.* [18] proposed an effective method to evaluate software leaks by identifying vulnerable nodes in different software through the

interdependence of functions. At present, the application of software system complexity measurement, topology characteristics and information transmission has been relatively mature, but there is little research on the controllability of software network topology. Applying the network controllability theory [19] to software networks is a bold attempt, which expands the field of the research on complex network control. Software networks, as typical complex networks [20], have similar network topology characteristics, and also exist some structural control problems inevitably. Whether it shows similar characteristic rules with the control aspects of complex networks needs to be further considered.

Based on the research on the controllability of complex networks and the previous experience and analysis methods of software measurement, in this paper, according to the idea of control theory, the "Source-Driver" (SD) model is established, which intuitively and simply expresses the control process of the networks; then, apply the software networks to SD model and study the controllability of software networks combined with the characteristics and rules of the software network topology. Finally, the distribution relation on in-degree, out-degree, degree and center-degree are analyzed deeply to explore how to choose driver node sets in the networks effectively. The obtained results will have guiding significance for the maintenance of software system.

The remainder of the paper is organized as follows: **Section 2** introduces some basic theories about network control aspect and the network controllability analysis framework. In **Section 3**, the software network controllable model (SD model) is established. Experiments performed in **Section 4** discuss the control process of software network topology on SD model, and analyze the characteristic relations on different versions of software to probe the selection of driver node sets. Finally, the paper is concluded and the future direction is given in **Section 5**.

2. Control Method and Process

2.1. Basic Theory and Definition

Definition 1 Control nodes: It refers to the nodes where the control signals are input. Among them, control signals are used for receiving, and then forwarding immediately to any node in the networks.

Definition 2 Driver nodes [21]: Control nodes that do not share the same input nodes. Then, the minimum driver nodes required to fully control the states of all nodes in the networks are called the minimum driver node sets.

Definition 3 Center-degree [22]: It refers to the reciprocal of the sum of the shortest path lengths from a node to all other nodes multiply by $N-1$, expressed as:

$$C_i = (N-1) \Big/ \sum_{i=1}^{N} d_{ij} \tag{1}$$

where, d_{ij} is the shortest path lengths from node i to node j.

Theorem 1 Minimum input theorem [9]: The minimum input set (N_I) is

equivalent to the minimum output set (N_D). If the networks are completely matched, the minimum input set is any node in the networks; otherwise, it is the unmatched node set after the maximum matching, expressed as:

$$\left|N_I\right| = \left|N_D\right| = \max\left\{1, N - \left|M^*\right|\right\} \tag{2}$$

where $\left|M^*\right|$ is the maximum number of matched nodes in the directed networks.

Theorem 2 Kalman criterion [23]: For a $N \times NM$ controllable matrix

$$\boldsymbol{W} = \left[\boldsymbol{B}, \boldsymbol{AB}, \cdots, \boldsymbol{A}^{N\times 1}\boldsymbol{B}\right] \tag{3}$$

If and only if the matrix is full rank, *i.e.* $rank\left(\boldsymbol{W}\right) = N$, the system is controllable.

Theorem 3 Structural controllability theorem [24]: If there are no expansive structure and unreached nodes in the networks, the system structure is controllable.

2.2. Network Controllability

In the complex networks, the state of a node at time t can be represented by a set of N dimensional vectors $x\left(t\right) = \left(x_1\left(t\right), \cdots, x_N\left(t\right)\right)^{\mathrm{T}}$, expecting that the node reach the system state $x\left(t'\right) = \left(x_1\left(t'\right), \cdots, x_N\left(t'\right)\right)^{\mathrm{T}}$ at time t' . Nodes can be considered controllable if they can reach the desired state by inter-coupling. If some states are impossible to reach, the only way to force them is to input control signals from the outside, that situation is uncontrollable. As shown in **Figure 1**.

For the system equation

$$x'\left(t\right) = \boldsymbol{A}x\left(t\right) + \boldsymbol{B}u\left(t\right) \tag{4}$$

$u\left(t\right) = \left(u_1\left(t\right), \cdots, u_M\left(t\right)\right)^{\mathrm{T}}$ indicates the state of the input control signal at time *t*. $\boldsymbol{A}$ is the adjacency matrix of nodes in the networks, which describes the connectivity and interaction strength among nodes. $\boldsymbol{B}$ is the input matrix ($M \le N$), which describes the connective relation between the control signals and the nodes.

$$\boldsymbol{A} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ a_{21} & 0 & 0 & 0 \\ a_{31} & 0 & 0 & a_{34} \\ a_{41} & 0 & 0 & 0 \end{bmatrix}; \quad \boldsymbol{B} = \begin{bmatrix} b_1 & 0 \\ 0 & b_2 \\ 0 & 0 \\ 0 & 0 \end{bmatrix};$$

$$\boldsymbol{W} = \begin{bmatrix} b_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & b_2 & a_{21}b_1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & a_{31}b_1 & 0 & a_{34}a_{41}b_1 & 0 & 0 & 0 \\ 0 & 0 & a_{41}b_1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Dynamics equation of a single node is

$$x_i' = \sum_{k=1}^{N} a_{ik}x_k\left(t\right) + \sum_{j=1}^{M} a_{ij}u_j\left(t\right) \tag{5}$$

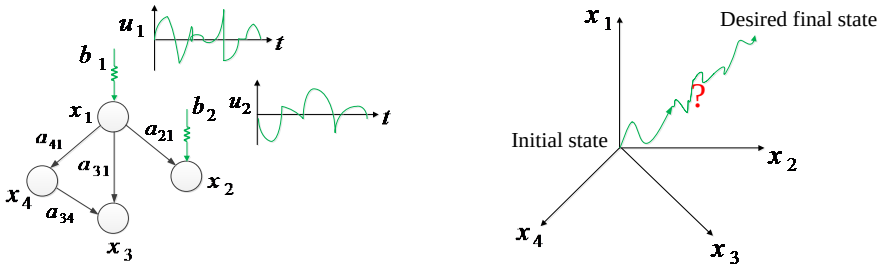


Figure 1. Schematic diagram of the state space of nodes.

If $rank(\boldsymbol{W}) = N$, *i.e.* satisfying the Theorem 2, the network is controllable.

In general, an input signal $u_i(t)$ may have many driver nodes. The main goal of network controllability technology is to make the networks globally controllable with the least input control signals. Liu *et al.* [9] transformed this question into driver nodes required by the control network with the least amount, thus, solving the minimum driver node sets resolve the key problem of network control. According to the **Theorem 1** and the **Theorem 3**, solve the maximum matching nodes and unmatched nodes in the directed graph. If all the unmatched nodes in the network are controlled by a direct input signal, the whole network is completely controllable [25], the unmatched nodes are driver nodes, as shown in **Figure 2**.

Take a small network as an example in **Figure 2(a)**. Firstly, the network is transformed into a bipartite graph structure, as shown in **Figure 2(b)**, observing the maximum matching edge of this bipartite graph [26], the maximum matching edge set of one combination is the red edge in **Figure 2(c)**, it can be seen that v_2, v_3, v_4 and v_5 are matching nodes, v_1 is an unmatched node, *i.e.* a driver node. The controllable state is as shown in **Figure 2(d)**. v_1 makes all nodes reachable, so as to control the state of all nodes in the entire network.

3. Controllability of Software Networks

3.1. Extraction of Class-Level Software Networks

Analyzing the software source codes is the first step to obtain structural information. In object-oriented software, for instance, in **Figure 3(a)**, classes are the most basic elements. Abstract the classes in the software source codes as nodes and the interaction relations between classes as edges in the networks. The whole extraction process is to get the class diagram in the software system first, seeing **Figure 3(b)**, and then abstract it into a network model, seeing **Figure 3(c)**.

3.2. Controllable Mode of Software Networks

In real life, most real systems can be driven by nonlinear processes, but in fact, the controllability of nonlinear systems is very similar to linear systems in many aspects, such as structure [27]. The controllable condition of Kalman criterion is applied to any complex network and can also be used to deal with most nonlinear systems [9].

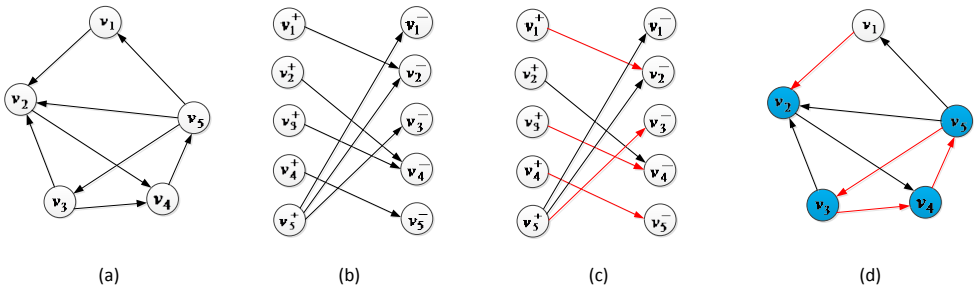


Figure 2. The generation of driven node sets: (a) Network topology (b) Corresponding bipartite diagram (c) A maximum matching edge set (d) Controllable state of the networks.

```
class RView{
        int x, y;
}
class RMax{
        void setphoto (char b, Rview n);
        void getphoto (RView n);
}
abstract class RPhotoobject{
        RView view;
}
class PhotoGrid extends RMax{
        RPhotoobject displayList[];
        void addPhotoobject (Rphotoobject obj);
        void deletePhotoobject (int dex);
}
class BorderedPhotoRGrid extends PhotoGrid{
         void setBorderCharacter (char b);
         void displayPhoto();
}
```

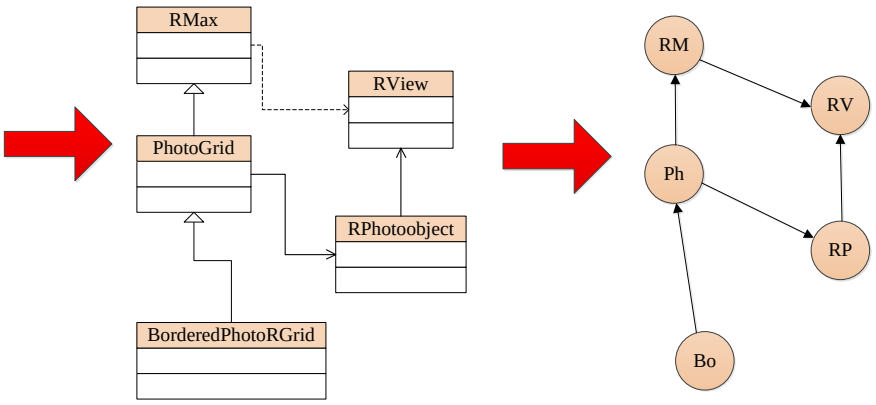


Figure 3. Software network of class: (a) A part of source codes; (b) Class diagram; (c) Network topology.

According to the idea of software engineering, class modules in the object-oriented software structure are regarded as nodes in the network, and the calling behaviors among classes are regarded as edges, so as to form software networks [28]. However, the interaction and information transmission among nodes is a control and being controlled process. Measure and control the software network topology from a macro perspective, so that the software system can run stably in various application fields. Therefore, how to control network topology with low cost and high efficiency is an urgent problem to be solved in software network structure control.

The Source-Driver (SD) model based on the analysis of the control process of complex networks is established, and the process of information transmission in the software network topology can be extracted SD model, as shown in **Figure 4**. It is divided into two layers, namely source layer α and driver layer β. Nodes, as controllable targets, are planned in two layers. Nodes in layer α are source nodes (control nodes), they selectively send control signals to other nodes. Nodes in layer β are matching nodes and driver nodes. The driver nodes directly receive the control signals sent by the source nodes, and then transmit them to other nodes in the subnets at layer β.

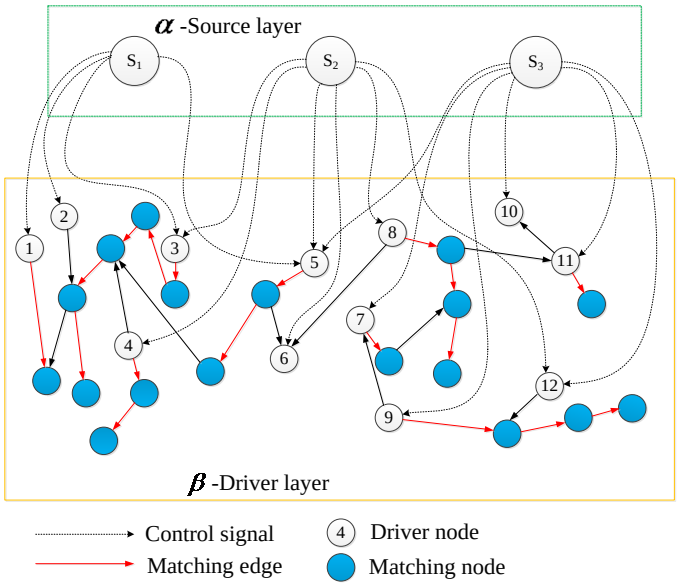


Figure 4. SD model.

In Figure 4, it describes a control situation of a directed controllable software network on the SD model. Assume there are three independent source nodes S_1, S_2 and S_3 in layer α, they first send control signals to the driver nodes in the networks at layer β to control the whole network. The generation of driver node sets is not unique in complex networks, moreover, the signals sent by the source nodes to others in layer β are not completely random. Certainly, the matching between driver nodes and other nodes in layer β is not completely random, which is different from the previous researches on the controllability of complex networks. In the software networks, the relations, connections and information transmission among nodes are based on the dependencies and calls among various modules in the software systems. Therefore, source nodes selectively send control signals to the nodes in layer β on SD model in software networks to serve them. Naturally, a driver node is not controlled by only one source node, on the other hand, the more driver nodes a source nodes controls, the stronger their controlling force over the whole network.

Source nodes directly call driver nodes in SD model, relative to the huge networks, the number of driver nodes is less, it can be controlled the driver nodes to control the whole network. Thus, for software, the maintenance cost will be reduced, and the measurement efficiency will be improved.

4. Empirical Analyses

In the object-oriented software networks, center-degree is a characteristic parameter for discovering core nodes, which are locate in the central position, with the greatest influence in the entire network. Degree reflects the connection with adjacent nodes and depicts the mutual call behaviors among local modules in the structure of software system [29]. Out-degree and in-degree represent the direct control and being controlled states among various modules. Seeing the SD mod-

el framework, the size of driver node set in software networks is closely related to the feature of degree distribution.

Select 10 different scales of software to carry on the experiments, involving developed by object-oriented programming languages, such as C++ and Java. Extract them into software networks for different scales, and their basic statistics of network topology are shown in **Table 1**.

Make statistics on the characteristics of software networks, including the average degree of networks, average out-degree, average degree, average center-degree and driver node ratio. Their results are shown in **Figure 5**.

It can be seen intuitively in **Figure 5** that the average degree values of various software are all small, which indicates that the degree values of most nodes are low, and the average in-degree is greater than the average out-degree in all software networks. Except for Mysql, the average out-degree is basically similar to the average degree in other software networks, but for all of them, the average center-degree is the lowest. It explains that the design and interaction of modules in the software reduce the overall complexity. However, the driver node ratio in all software networks is about 0.5, which is slightly greater than that in other complex networks. One of the main goals of designing software at the beginning is to reduce the complexity of software structure, which is also convenient for subsequent software updates, just as the computed results presented in **Figure 4**, most nodes' degree values are relatively low in software networks, driving nodes and matching nodes will be generated here. On the contrary, if degree values are high for most nodes, the calling relationships (connection edges) between modules within the software structure will be more. It can be concluded that the driver node sets tend to be composed of nodes with low degree value. That is, from the perspective of control cost, the better driver node is not with high value, which is basically consistent with the conclusion in literature [9].

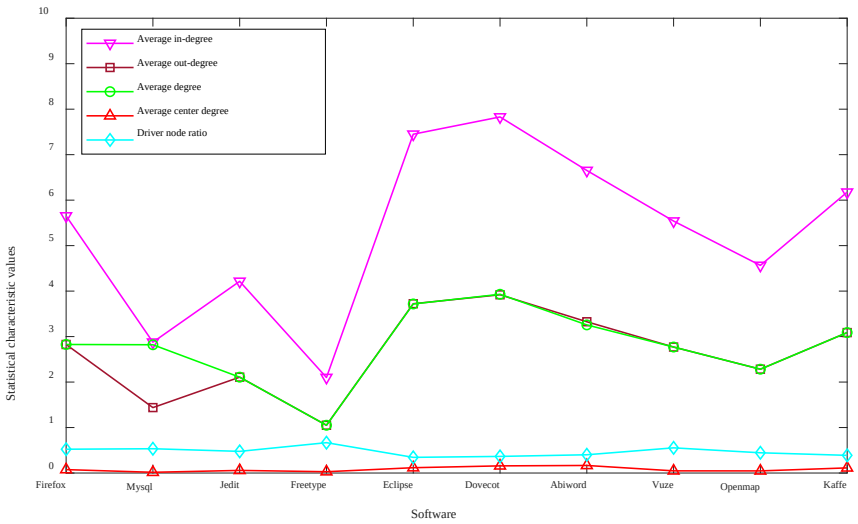


Figure 5. Topological characteristic statistics of software network.

Table 1. The scales of software networks.

	Firefox	Mysql	Jedit	Freetype	Eclipse	Dovecot	Abiword	Vuze	Openmap	Kaffe	Firefox	Mysql
N	10,116	3457	930	358	17,604	371	1300	3678	1932	7849	10,116	3457
L	29,672	5480	3359	387	32,718	729	2117	10,525	4518	12,120	29,672	5480

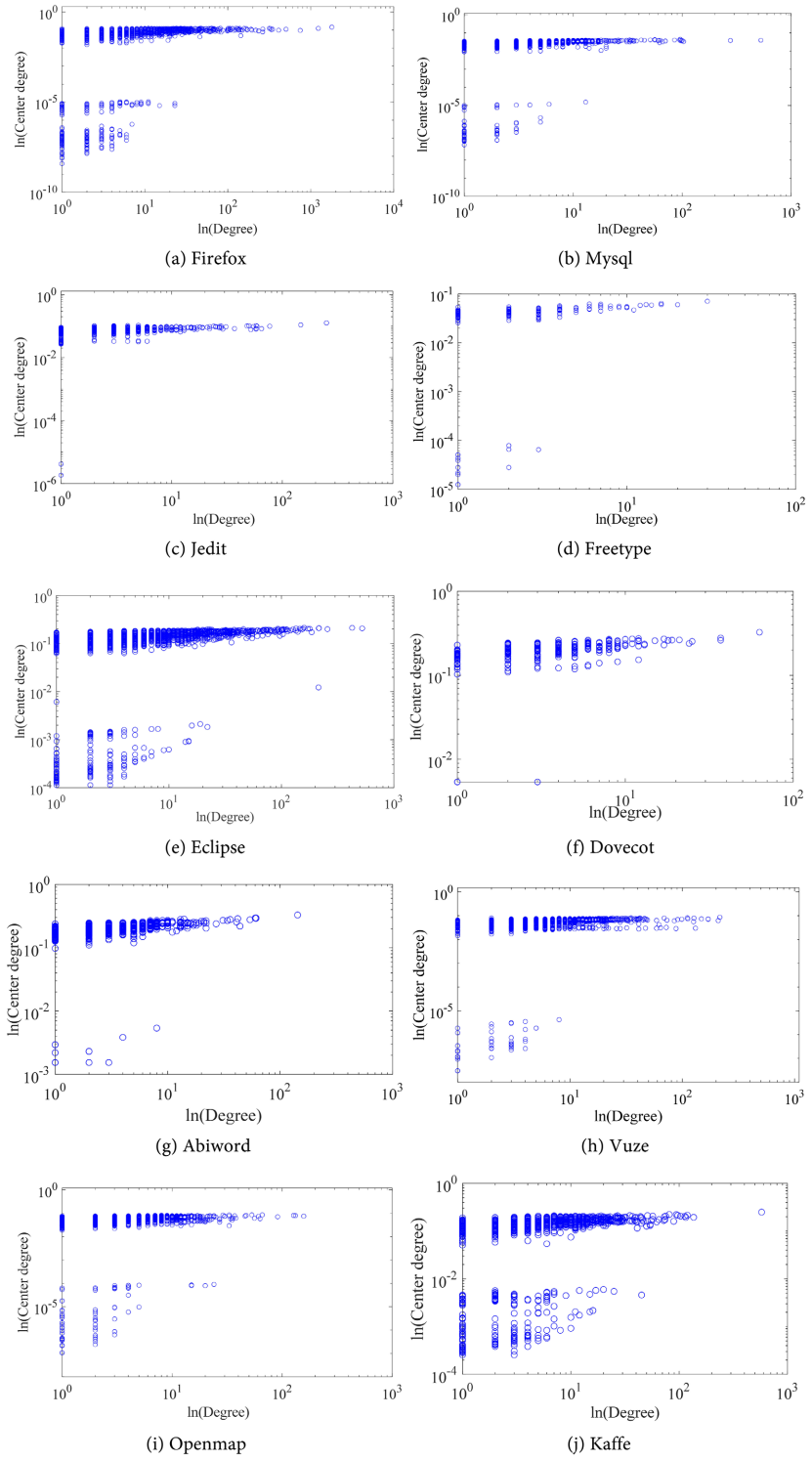


Figure 6. Distribution relationship between degree and centrality.

However, for a node, degree indicator cannot fully explain the importance and control force. Although there may be a large number of low degree nodes in the driver node sets, nodes with middle degree or high degree can still be found. Software networks, as scale-free networks, the degree distribution of nodes obeys the power-law distribution [30]. Reducing the degree value is to avoid overly complex construction in the software system structure. The selection of driver node sets should also consider the attribute characteristics of a node itself. In software networks, center-degree (**Definition 3**) is an indicator evaluated the importance and influence of nodes. The larger its value is, the closer it is to the central position of the networks, also is the core position. As shown in **Figure 6**.

Select 10 different scales of software to analyze the relationship between degree and center-degree of nodes in network topology. In order to observe clearly, take the double logarithmic coordinates of **Figure 6**. It can be seen that there are three types of nodes in the six software networks: 1) Nodes with high degree and center-degree, which may be no more than 5 such nodes in the software networks. They are generally core nodes, and are responsible for completing the main functions of the software systems. 2) Nodes with low degree and high center-degree. Although such nodes are part of the main functional modules of the software system, their constructions are not complex, for example, the login interface in a common software system is this kind of module. 3) Nodes with low degree and center-degree. Most of these nodes correspond to the basic units within the software systems to complete some basic functions, and the logical compositions of these modules are simple. It can be concluded that nodes with high degree must have high center-degree, but the opposite may not be true, that is to say, nodes with high center-degree do not certainly have high degree. Therefore, when selecting the driver node sets, the first type of nodes is the primary consideration.

Combining with SD model, the control extent of nodes is closely related to their in-degree and out-degree. The driver nodes use the in-degree indicator to characterize the received control signals, and out-degree for output signals. So in-degree and out-degree affect the behaviors of driver nodes together. In that way, it is necessary to analyze the relationship between the in-degree and out-degree of nodes. Orphan nodes and the leaf nodes of these 10 software networks are dislodged, and then take the double logarithmic coordinates as the distribution diagram, as shown in **Figure 7**.

It can be seen that the in-degree and out-degree of most nodes present linear relation in the directed software networks and a few nodes have deviation, which is also the reason why the nodes in software networks are generally with low degree. Moreover, the nodes in a small area below the straight line reduce the matching process among nodes. Digging deep into the architecture of a software system, some interesting phenomena will be found. Take Firefox as an example, extract its network topology, as shown in **Figure 8**.

In the visualized diagram, the size of a node represents its degree value, and different colorings represent different degree values, the larger the size of a node, the

higher the degree value. The figure presents a local aggregation phenomenon. They are connected to a few nodes. It also shows that the degree values of most nodes are not large, illustrating that the interaction preference of nodes, that is, the nodes with small degree values are attached to for those with large degree values.

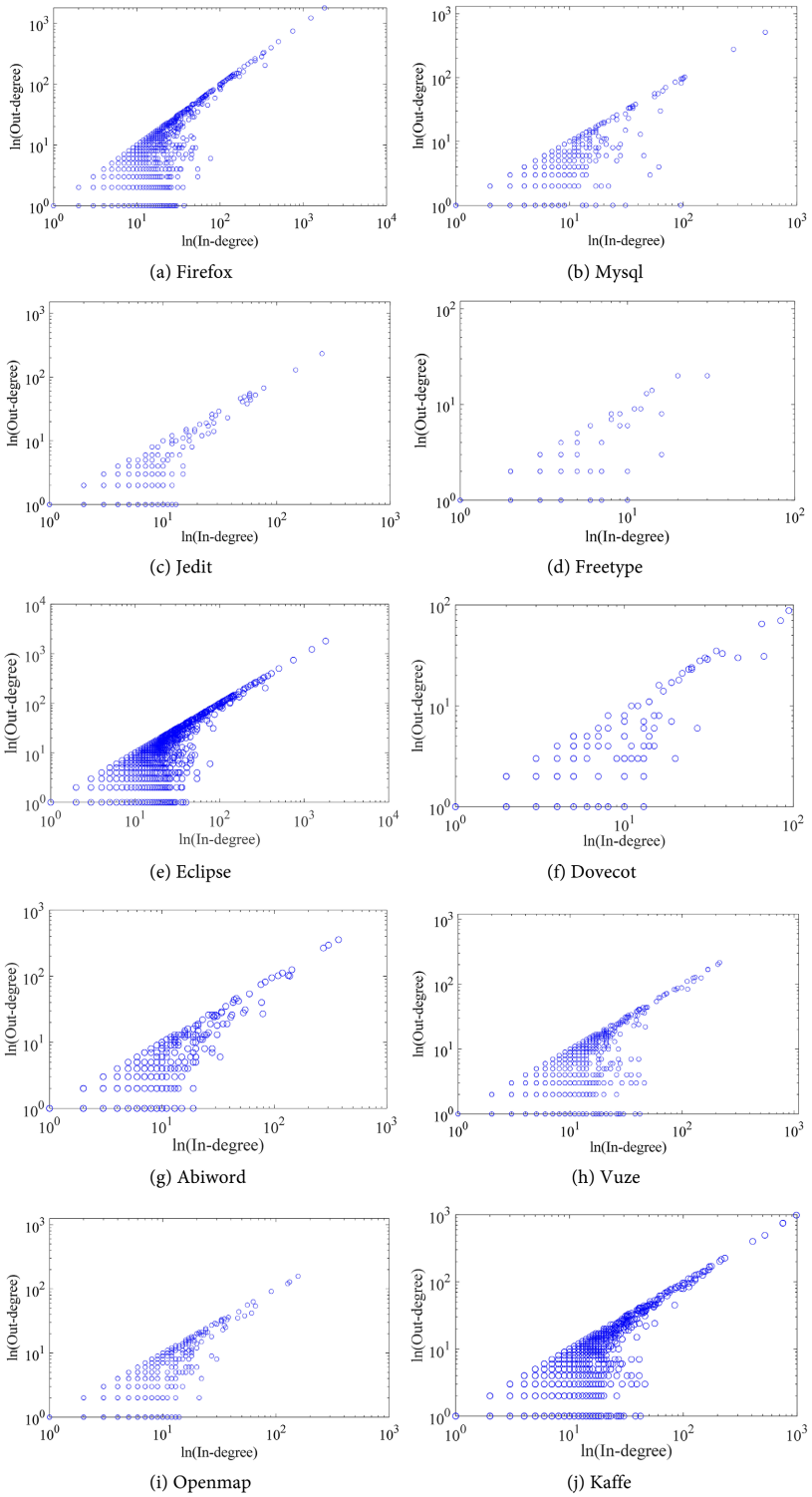


Figure 7. Distribution relationship between in-degree and out-degree.

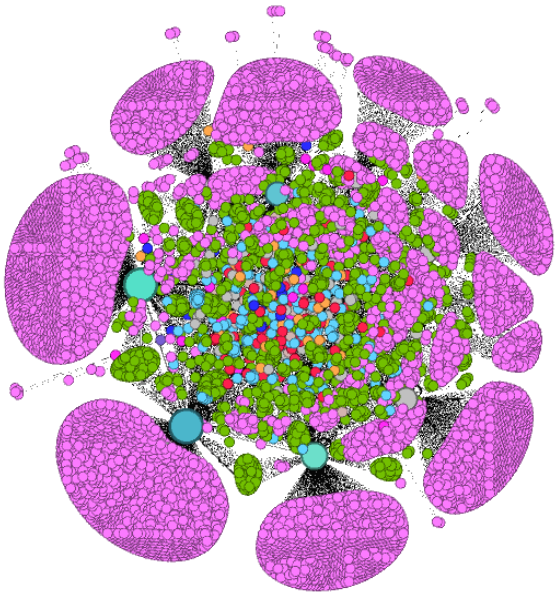


Figure 8. The software network topology of Firefox.

Table 2. Comparison of characteristic parameters of Firefox software network.

Node number	Node name	d^{in}	d^{out}	d	C
5784	"nsISecurityPref" interface	1797	1797	0	0.1439
3507	"nsCOMPtr" class	1233	1223	10	0.1232
4611	"nsICSSPseudoElement" class	3	2	1	0.1231
4364	"nsHypotheticalBox" struct	410	398	12	0.0978
568	"already_AddRefed" struct	143	143	0	0.0487
6951	"nsRefPtr" class	347	203	144	0.1095
9999	"XPCWrappedNativeScope" class	15	3	12	6E-06

Select 6 special nodes ulteriorly, listed in Table 2, and then contrast their in-degree d^{in}, out-degree d^{out}, degree d and center-degree C.

As a result, it can be found that the No. 5784 node has the largest value of center-degree, but its in-degree value is equal to out-degree value, they also are the largest, moreover, its degree value is 0, which is an extreme case. On the contrary, the No. 4611 node has the smallest in-degree value and out-degree value, but its center-degree value is very large. From those analyses, it can be concluded that the degree values cannot veritably reflect the situation of sending and receiving signals among nodes in the directed networks, control behaviors and states should be measured in in-degree and out-degree of nodes. That is to say, in-degree and out-degree are also important factors to select the driver node sets.

5. Conclusions

The control problem of complex networks is always a direction that people are

keen to research. While the theoretical research results are constantly abundant, broadening the application field is the practical significance of scientific research. Integrating software networks with control theory opens up a new perspective for measuring the structural complexity of software networks. In this paper, the controllability of software networks has been discussed in this direction.

The effective control of networks largely depends on the selection of the driver node sets. In this way, the SD model with two lays is established to analyze the control process of software network topology. Based on it, the relationship between degree and center-degree combined with the topological characteristics of software networks is analyzed, and the results illustrate that most of the driver node sets tend to be composed of nodes with a low degree, but the nodes with high center-degree should be given priority to consider. Subsequently, the correlation between in-degree and out-degree is analyzed, and the 4 metrics are contracted by going deep into the software system structure, it can be found that although the degree values of some nodes are very small, the in-degree and out-degree values are very different, corresponding the center-degree values are changeable, for this reason, such nodes should be paid close attention when controlling the software network structure. Especially, in-degree and center-degree are the key factors of node control.

There is still a lot of space for research on the controllability of software networks. In the future, the topological characteristics of the driver node sets in the software network and their information transmission will be studied deeply, so as to maintain the smooth and safe operation of software.

Acknowledgements

Project supported by the Basic Scientific Research Project of Education Department of Liaoning Province (Surface Project) (Grant No. LJKZ1065).

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Wang, J. (2019) Research on Complexity and Evolution Analysis of Open Source Operating System Software Based on Software Network Model. National University of Defense Technology, Changsha. (In Chinese)

[2] He, K.Q., Ma, Y.T., Liu, J., *et al.* (2008) Software Network. Science Press, Beijing. (In Chinese)

[3] Li, P., Zhao, H., Qiao, Y., *et al.* (2010) A Platform of Software Network Measurement Design and Implement. *The 2nd IEEE International Conference on Advanced Computer Control*, Vol. 2, 336-339. https://doi.org/10.1109/ICACC.2010.5486661

[4] Pan, H., Zhen, W., Zhang, Z.F., *et al.* (2019) Study on Fractal Features of Software Networks. *Computer Science*, **46**, 166-170.

[5] Pan, W.F., Jiang, B. and Li, B. (2013) Refactoring Software Packages via Community Detection in Complex Software Networks. *International Journal of Automation and Computing*, **10**, 157-166. https://doi.org/10.1007/s11633-013-0708-y

[6] Liu, H.Q. (2019) Research on Application of Network System Software Based on Large Data Analysis. *The 2nd International Conference on Computer Science and Advanced Materials*, Tianjin, 16 February 2019, 407-409. https://webofproceedings.org/proceedings_series/ECS/CSAM%202019/CSAM1982.pdf

[7] Wang, W.X., Ni, X., Lai, Y.C., *et al.* (2012) Optimizing Controllability of Complex Networks by Minimum Structural Perturbations. *Physical Review E*, **85**, Article ID: 026115. https://doi.org/10.1007/s11633-013-0708-y

[8] Chen, G.R. (2013) Problems and Challenges in Control Theory under Complex Dynamical Network Environments. *Acta Automatica Sinica*, **39**, 312-321. https://doi.org/10.1016/S1874-1029(13)60032-4

[9] Liu, Y., Slotine, J. and Barabási, A.L. (2011) Controllability of Complex Networks. *Nature*, **473**, 167-173. https://doi.org/10.1038/nature10011

[10] Yuan, Z.Z., Zhao, C., Di, Z.R., *et al.* (2013) Exact Controllability of Complex Networks. *Nature Communications*, **4**, 167-173. https://doi.org/10.1038/ncomms3447

[11] Zhang, X.Z., Lv, T.Y., Luan, H., *et al.* (2015) Analysis of Minimum Driver Node Set of Complex Network based on Random Matching. *Control and Decision*, **30**, 751-754.

[12] Posfai, M., Liu, Y.Y., Slotine, J.J., *et al.* (2013) Effect of Correlations on Network Controllability. *Scientific Reports*, **3**, Article No. 1067. https://doi.org/10.1038/srep01067

[13] Liu, L.S. and Pang, P.S. (2020) Effect of Degree Correlation on Edge Controllability of Real Networks. *Chinese Physics B*, **29**, Article ID: 100202. https://doi.org/10.1088/1674-1056/ab99ab

[14] Wang, W., Li, T., He, Y., *et al.* (2016) A Hybrid Approach for Ripple Effect Analysis of Software Evolution Activities. *Journal of Computer Research and Development*, **53**, 503-516.

[15] Ma, Y.T., He, K.Q., Li, B., *et al.* (2011) Empirical Study on the Characteristics of Complex Networks in Networked Software. *Journal of Software*, **22**, 381-407. https://doi.org/10.3724/SP.J.1001.2011.03934

[16] Liu, M.L., Qi, X.G. and Pan, H. (2022) Multifractal Analysis of the Software Evolution in Software Networks. *Chinese Physics B*, **31**, Article ID: 030501. https://doi.org/10.1088/1674-1056/ac1b8a

[17] Ren, J., Wu, H., Yin, T. and Bai, L. (2015) A Novel Approach for Mining Important Nodes in Directed-Weighted Complex Software Network. *Journal of Computational Information Systems*, **11**, 3059-3071.

[18] Zang, B., Sun, S.T. and Hao, X.B. (2020) Mining Important Functions in Software Network by Node Vulnerability. *Journal of Physics Conference Series*, **1453**, Article ID: 012015. https://doi.org/10.1088/1742-6596/1453/1/012015

[19] Chen, G.R. (2014) Pinning Control and Synchronization on Complex Dynamical Networks. *International Journal of Control, Automation and Systems*, **12**, 221-230. https://doi.org/10.1007/s12555-014-9001-2

[20] Li, H., Zhao, H., Cai, W., *et al.* (2013) A Modular Attachment Mechanism for Software Network Evolution. *Physica A: Statistical Mechanics & Its Applications*, **392**, 2025-2037. https://doi.org/10.1016/j.physa.2013.01.035

[21] Lv, T.Y., Piao, X.F., Xie, W.Y., *et al.* (2012) Controllability of Complex Networks

Based on Propagation Immunization. *Acta Physica Sinica*, **61**, Article ID: 170512. https://doi.org/10.7498/aps.61.170512

[22] Xu, J.Q., Gui, J.W., Zhao, H., *et al.* (2013) Evolution Analysis of Closeness of Software Macrotopology. *Journal of Northeastern University* (*Natural Science*), **34**, 646-649. http://xuebao.neu.edu.cn/natural/EN/Y2013/V34/I5/646

[23] Kalman, R.E. (1963) Mathematical Description of Linear Dynamical Systems. *Journal of the Society for Industrial & Applied Mathematics*, *Series A: Control*, **1**, 152-192. https://doi.org/10.1137/0301010

[24] Liu, C.T. (1974) Structural Controllability. *IEEE Transactions on Automatic Control*, **19**, 201-208. https://doi.org/10.1109/TAC.1974.1100557

[25] Ai, J. (2013) Research on Techniques of Target Node Analysis in Complex Networks. Northeastern University, Shenyang. (In Chinese)

[26] Tamás, N. and Tamás, V. (2012) Controlling Edge Dynamics in Complex Networks. *Nature Physics*, **8**, 568-573. https://doi.org/10.1038/nphys2327

[27] Li, Z., Ring, P., Kym, M., *et al.* (1991) Applied Nonlinear Control. Prentice-Hall, Hoboken. https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.70.7100&rep=rep1&type=pdf

[28] Gu, Q. and Chen, D.X. (2014) Validation and Simulation of Software System Evolution Rules Using Software Networks. *Science China: Information Science*, **44**, 20-26.

[29] Gu, Q., Xiong, S.J. and Chen, D.X. (2014) Correlations between Characteristics of Maximum Influence and Degree Distributions in Software Networks. *Science China* (*Information Sciences*), **57**, 21-32. https://doi.org/10.1007/s11432-013-5047-7

[30] Wang, J.R., Wang, J.P., He, Z., *et al.* (2015) Degree Distribution and Robustness of Cooperative Communication Network with Scale-Free Model. *Chinese Physics B*, **24**, Article ID: 060101. https://doi.org/10.1088/1674-1056/24/6/060101 https://iopscience.iop.org/article/10.1088/1674-1056/24/6/060101