Structural Features and Robustness of Coupled Software Networks

WANG Ershen[1,2], *TONG Zeqi*[1], *HONG Chen*[3,4*], *WANG Yanwen*[5], *MEI Sen*[6], *XU Song*[1], *NA La*[1]

1. School of Electronic and Information Engineering, Shenyang Aerospace University, Shenyang 110136, P. R. China; 2. Henan Collaborative Innovation Center of Geo-information Technology for Smart Central Plains, Zhengzhou 450052, P. R. China; 3. Multi-agent Systems Research Centre, Beijing Union University, Beijing 100101, P. R. China; 4. College of Robotics, Beijing Union University, Beijing 100101, P. R. China; 5. School of Computer Science and Engineering, Northeastern University, Shenyang 110169, P. R. China; 6. Shenyang Wuju Technology Co., Ltd., Shenyang 110170, P. R. China

(Received 2 April 2025; revised 5 September 2025; accepted 14 September 2025)

Abstract: Software systems play increasing important roles in modern society, and the ability against attacks is of great practical importance to crucial software systems, resulting in that the structure and robustness of software systems have attracted a tremendous amount of interest in recent years. In this paper, based on the source code of Tar and MySQL, we propose an approach to generate coupled software networks and construct three kinds of directed software networks: The function call network, the weakly coupled network and the strongly coupled network. The structural properties of these complex networks are extensively investigated. It is found that the average influence and the average dependence for all functions are the same. Moreover, eight attacking strategies and two robustness indicators (the weakly connected indicator and the strongly connected indicator) are introduced to analyze the robustness of software networks. This shows that the strongly coupled network is just a weakly connected network rather than a strongly connected one. For MySQL, high in-degree strategy outperforms other attacking strategies when the weakly connected indicator is used. On the other hand, high out-degree strategy is a good choice when the strongly connected indicator is adopted. This work will highlight a better understanding of the structure and robustness of software networks.

Key words: software network; software structure; software robustness; software system; complex network

CLC number: TN925 **Document code:** A **Article ID:** 1005-1120(2025)06-0801-12

0 Introduction

As the centerpiece of information technology, software systems play important roles in the modern world. How to develop a high-quality software system is always the major concern of software industry, especially for large-scale software systems. It is known that software systems are typical artificial complex systems which can be denoted as complex networks[1-5].

In the past decades, numerous achievements have been made in the application of complex network theory for the analysis of complex software systems[6-9]. Valverde et al.[10] constructed an undirected network to represent software systems. The result showed that the software network is of obvious small-world[11-12] and scale-free characteristics. Han et al.[13] analyzed the software network structure with different granularity levels. Results indicated that the network exhibited self-similar feature at different granularity levels. Shan et al.[3] constructed a directed weighted software network which can quantify the complexity of software systems. Peng et al.[14] constructed a software network from the multi-granularity perspective and analyzed the evolution of the software network. It is found that there are note-

*Corresponding author, E-mail address: xxthongchen@buu.edu.cn.

How to cite this article: WANG Ershen, TONG Zeqi, HONG Chen, et al. Structural features and robustness of coupled software networks[J]. Transactions of Nanjing University of Aeronautics and Astronautics, 2025,42(6):801-812.

http://dx.doi.org/10.16356/j.1005-1120.2025.06.007

worthy differences in the evolution process under different granularity levels. Ma et al.[15] proposed an evolution model based on local events to simulate the evolution process of software networks. Zhang et al.[16] proposed a model based on important nodes in software networks to analyze the modularity feature under different software versions. Results implied that the software evolution process followed a trend of high-cohesion and low-coupling.

With the increasing importance of large-scale software systems, research on the robustness of software systems has become a hotspot in recent years[17-18]. The robustness of a software system refers to its ability to maintain its function when it is disturbed. So far, the robustness of software systems has been studied from different aspects, including the models for software cascading faults[17,19], the cascade fault in weighted software network[18] and so on. Wang et al.[17] established a cascading fault model of software system by introducing the fault tolerance and the fault strength. The result showed that the weak fault strength can decelerate the fault propagation speed and reduce the range of faults. He et al.[18] proposed a cascading failure diffusion analyzing algorithm with respect to software networks, which can evaluate the impact of dynamic faults.

However, most previous studies on software networks have focused on the scene of a single or an isolated network. Actually, there are a number of coupled modules in many software systems[20-21]. Especially, for a large-scale complex software system, there exists many public functions defined in one module but called in other modules, reflecting a coupled relationship between different modules. Hence, complex software systems can be well represented as coupled network[22]. In recent years, the study of coupled networks has attracted a great of interest[23-30]. Buldyrev et al.[31] proposed a coupled network model to characterize the dynamic behavior between two real interdependent networks. Tan et al.[32] proposed a cascading failure model on interconnected network and studied the cascading failure dynamics in the network. Cui et al.[33] proposed a method to study the network robustness by reasonably allocating limited costs to some links. The result indicated that adding both connectivity links and dependence links can bring stronger robustness to the coupled networks. Gao et al.[34] proposed a deliberate attacking model, and studied the robustness of interdependent networks with different coupling modes.

Note that coupled networks often exhibit different features and robustness compared with isolated networks[35]. Meanwhile, due to the intrinsic interconnected relations between software modules, it is very meaningful to study software networks from the perspective of coupled networks. In this paper, based on the source code of Tar and MySQL, we propose a method to generate coupled software networks and establish three kinds of directed software networks: The function call network, the weakly coupled network and the strongly coupled network. By introducing eight attacking strategies and two robustness indicators, the structural properties and robustness of these software networks are examined extensively. The results show that nodes with high in-degree or high out-degree are of crucial influence on the robustness of directed software networks.

While most software network studies primarily focus on either single function call networks or inter-module dependency networks, this work addresses a critical gap by constructing coupled software networks based on cross-module call relationships manifested in source code. Our hierarchical construction method transitions from function call networks to weakly coupled networks, and finally to strongly coupled networks, effectively capturing complex dependencies formed by shared functions between software modules.

Through extensive analysis of Tar and MySQL systems, we reveal that all constructed networks exhibit small-world properties and demonstrate that average influence and average dependence remain consistent across directed software networks. To quantitatively assess network resilience, we introduce eight distinct attack strategies alongside weak and strong connectivity indicators, providing a comprehensive framework for evaluating software system robustness under various failure scenarios.

The paper is organized as follows. In Section 1,

we demonstrate three kinds of directed software networks, eight attacking strategies and two network robustness indicators in detail. In Section 2, the simulation results and discussion are provided. Finally, the work is summarized in Section 3.

1 Models

1.1 Function call networks

As a file packaging tool on Unix and Unix-like systems, Tar is an important open source software system. MySQL is a famous large-scale open source software system as a well-known database system. Therefore, in this paper, we select Tar and MySQL to generate software networks.

The specific generation scheme of the function call networks is described as follows. Firstly, we download the source code of Tar-1.23 (http://ftp.gnu.org/gnu/tar/) and MySQL-5.7.13 (https://downloads.mysql.com/archives/installer/) from website, then use Cflow (http://www.gnu.org/software/cflow/) to decompose the call relationships among different functions. Afterwards, Tree2dotx (https://github.com/hawkinchina/tiny-club-linux-0.11-lab/blob/master/tools/tree2dotx) is used to convert function call relationships to dot format. Here, functions in source code are abstracted as nodes, and call relationships between functions are represented as directed links. Finally, we use Gephi (https://gephi.github.io/users/download/) to parse the dependence of nodes in the dot format, and export the information of edges to form the function call network.

Tar consists of five major modules: src, gnu, lib, tests and rmt. There are 1 204 nodes and 3 285 directed edges in the function call network of Tar (FCNT), and the topological structure of FCNT is shown in Fig.1(a). In Fig.1, different modules in software networks are shown in different colors. MySQL comprises nineteen major modules (client, scripts, cmd-line-utils, libevent, storage, libmysql, sql-common, sql, dbug, zlib, libmysqld, extra, strings, regex, unittest, testclients, mysys, plugin, vio). There exist 4 598 nodes and 16 018 directed edges in the function call network of MySQL (FCNM), and Fig.1(b) displays the topological structure of FCNM.

(a) FCNT (b) FCNM

Fig.1 Topological structures of function call networks

In what follows, we will elucidate in detail the scheme of generating two kinds of coupled software networks: The weakly coupled network and the strongly coupled network.

1.2 Coupled software networks

For complex software systems, it is known that many public functions are called by different modules. For example, in the case of Tar, function xmalloc is defined in module gnu and called by some functions in lib, src and tests modules, indicating that these modules are of coupled relations. To describe this kind of relation, we define the nodes which are similar to function xalloc as coupling node. In the case of MySQL, analogously, function die is defined in module client and called in scripts module, and thus function die is a coupling node too. Here, we obtain 205 coupling nodes in the function call network of Tar and 634 coupling nodes in the function call network of MySQL. Fig.2 shows the schematic of different kinds of software networks, where coupling nodes are marked with red color.

To analyze the coupled relation among different modules, we first construct a simple kind of coupled software networks: The weakly coupled network, which is composed by all coupling nodes and some special non-coupling nodes. The weakly coupled network is refined from the function call network by removing some non-coupling nodes and their edges. Concretely, we remove the edges that do not contain any coupling node. For instance, ex-

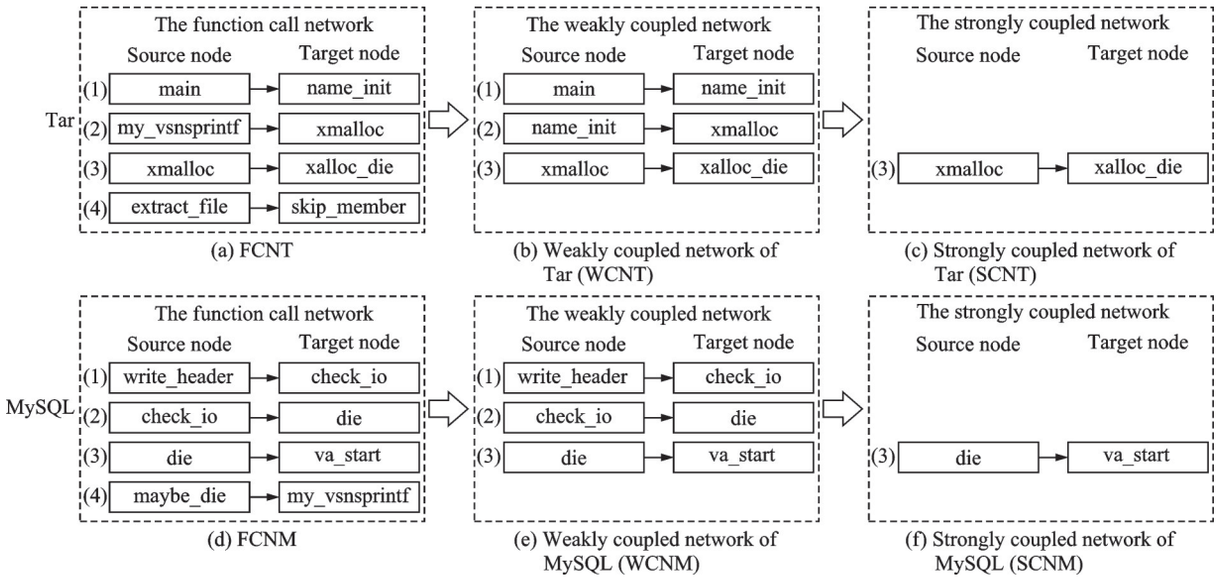

Fig.2 Schematic of different kinds of software networks

tract_file and skip_member are all non-coupling nodes (Fig.2(a)), so edge 4 is removed, namely, node extract_file, node skip_member and the directed link from extract_file to skip_member are all removed simultaneously (Fig.2(b)).

In our revised approach, we introduce the concept of semi-coupled nodes. These are non-coupling nodes that, although not directly representing cross-module interactions, play a vital role in linking two or more coupling nodes and thus preserving the overall network semantics. Formally, the function call network is represented as a directed graph $G(V,E)$ with $C\subset V$ denoting the set of coupling nodes. If a non-coupling node $v\in V\backslash C$ (node v belongs to V, but not to C) satisfies that there exist $u,w\in C$ such that $(u,v)\in E$ or $(v,w)\in E$, v is defined as a semi-coupled node. These semi-coupled nodes are retained in the weakly coupled network (WCN) to ensure that every edge in the network involves at least one coupling node, thereby maintaining the intrinsic connectedness and rich semantic information of the original network.

When the overall removing process is completed, the weakly coupling network is just composed by coupling nodes and some special non-coupling nodes which must be linked with a coupling node. This shows that at least one coupling node is included in each edge of the weakly coupled network. Furthermore, some coupling nodes are still connected via non-coupling nodes, for instance, the call path between function main and function xmalloc is still connected via non-coupling node name_init (Fig.2(b)). There are 695 nodes and 1 677 edges in WCNT, and the topological structure of WCNT is shown in Fig.3(a).

For MySQL, in the same manner, edge 4 is removed since maybe_die and my_vsnsprintf are all non-coupling nodes (Fig.2(c)), and the path between coupling node write_header and coupling node die is still connected via non-coupling node check_io (Fig.2(e)). There are 2 682 nodes and 8 790 edges in WCNM, and the topological structure of WCNM is shown in Fig.3(b).

To further study the coupled relation among different modules, we generate another kind of coupled software networks: The strongly coupled network. Suppose that the strongly coupled network is only comprised by coupling nodes, which is refined from the weakly coupled network by removing all non-coupling nodes and corresponding links. In the case of Tar, edge 1 and edge 2 are all removed since there exist non-coupling nodes in above two edges (Fig.3(c)). There are 109 nodes and 163 edges in SCNT, and the topological structure of SCNT is shown in Fig.3(c). Noteworthily, in this case, some coupling nodes such as main and xmalloc are also removed since they are only connected with a non-coupling node. Similarly, to construct the strongly cou-

pled network of MySQL, edge 1 and edge 2 are all discarded (Fig.3(d)). SCNM consists of 393 nodes and 1 288 edges, and Fig.3(d) displays the topological structure of SCNM.

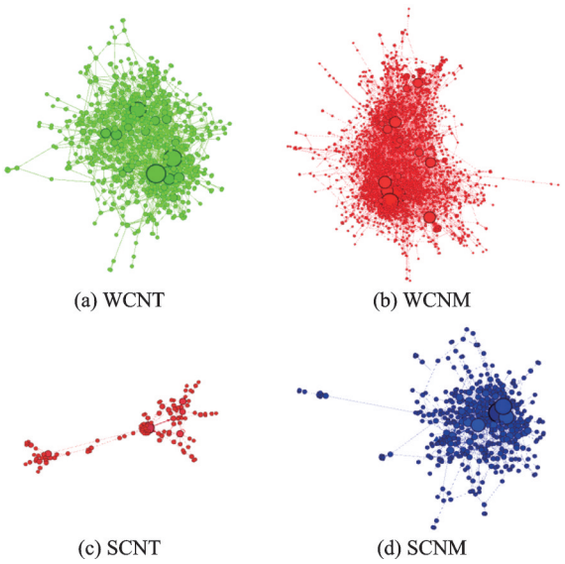

Fig.3 Topological structures of coupled software networks

In constructing the strongly coupled network, we only retain those coupling nodes that bear explicit identifiers in cross-module calls, thereby excluding all non-coupling nodes that do not directly participate in cross-module calls. Although this simplification, to some extent, disregards certain "connector" or "scheduling center" nodes that may play a bridging role in actual systems and will result in the loss of certain semantic information. However, to deeply investigate the "strong" coupling features of realistic networks, this simplification approach is a carefully considered methodological choice. Specifically, the crucial goal is to identify and analyze direct strong coupling relationships between modules, and the simplification will provide a more focused and actionable analysis framework, making the identification of strong coupling relationships more accurate and practical.

To deeply investigate the "strong" coupling features of realistic networks, this simplification approach is a carefully considered methodological choice. Specifically, the crucial goal is to identify and analyze direct strong coupling relationships between modules, rather than building a complete system semantic representation.

1.3 Attacking strategies

To investigate the robustness of the above three kinds of networks, we introduce the following eight attacking strategies.

(1) High out-degree strategy (HODS)

The out-degree $k_i^{\text{out}} = \sum_{j=1}^{N} a_{ij}$ refers to the number of directed edges pointing outward from node i to its direct neighbor nodes, where N is the size of the network. If there is an edge from node i to node j, $a_{ij} = 1$, otherwise $a_{ij} = 0$. For HODS, we sort nodes in descending order of node's out-degree in the initial network, and select nodes to attack from the sorting list.

(2) High in-degree strategy (HIDS)

The in-degree of node i corresponds to the number of directed edges pointing inward from its direct neighbors to node i, which can be represented as $k_i^{\text{in}} = \sum_{j=1}^{N} a_{ji}$. In the case of HIDS, we sort nodes in descending order of node's in-degree in the initial network.

(3) Random strategy (RS)

In the case of RS, nodes in the network are removed randomly.

(4) High authority strategy (HAS)

Hyperlink-induced topic search (HITS)[36] is a classic algorithm for identifying key nodes in the directed complex network. In the algorithm, two novel indices (the authoritative value and the hub value) are used to metric the importance of nodes in the directed network. Under HAS, we will sort nodes in descending order of node's authoritative value in the initial network.

(5) High hub strategy (HHS)

For HHS, nodes are sorted in descending order of node's hub value in the initial network. Here the hub value of nodes is computed according to the HITS algorithm[36].

(6) High pagerank strategy (HPS)

Pagerank (PR)[37] is a famous algorithm used by Google search to rank web pages in their search engine results. The PR value has been widely adopted to identify crucial nodes in the directed network.

Under HPS, we sort nodes in descending order of node's PR value in the initial network.

(7) High dependence strategy (HDS)

For a directed network, if there exists a directed path with arbitrary length from node i to node j, we designate that node i can reach node j. The dependence value of node i is defined as the number of nodes in set $D(i)$, where $D(i)$ is the collection of nodes (with the exception of node i) that node i can reach. Under HDS, we sort nodes in descending order of node's dependence value in the initial network. Obviously, for software systems, the larger the dependence of a function, the more other functions it depends on, and the more vulnerable the function will be.

(8) High influence strategy (HIS)

The influence value of node i is denoted as the number of nodes in set $I(i)$, where $I(i)$ is the set of nodes (excluding node i) that can reach node i. For HIS, we sort nodes in descending order of node's influence value in the initial network.

It is noteworthy that, for the above eight attacking strategies, the edges linking the removed node are deleted simultaneously.

1.4 Robustness indicators

In order to evaluate the robustness of directed software networks, we use two robustness indicators: the weakly connected indicator and the strongly connected indicator.

Given a directed network, if each node can be reached when all edges in the network are regarded as undirected edges, the network is a weakly connected network. The weakly connected indicator is defined as $W=N_W/N$, where N_W is the number of nodes in the largest weakly connected network after attack and N the number of nodes in the initial network. In this paper, the attack rate is defined as $f=N'/N$, where N' is the total number of removed nodes. One can see that the larger the value of the weakly connected indicator is, the stronger the robustness of the directed network is.

For any pair of nodes u and v in a directed network, if there is a directed path from u to v and also exists a directed path from v to u, the network is a strongly connected network. The strongly connected indicator is denoted as $S=N_S/N$, where N_S is the number of nodes in the largest strongly connected network after attack. We can see that the larger the value of the strongly connected indicator is, the stronger the robustness of the directed network is.

For a directed software network, obviously, the strongly connected indicator S is stricter than the weakly connected indicator W, thus S value is always smaller than the value of W.

2 Simulation Results and Discussion

First, we discuss the structural features of Tar. Table 1 shows the network properties with respect to three kinds of networks of Tar: FCNT, WCNT, and SCNT, where M is the number of directed edges in the network, $\langle k\rangle$ the average degree of the network, $\langle L\rangle$ the average path length, $\langle C\rangle$ the average clustering coefficient, d the network diameter, $\langle I\rangle$ the average influence of the network, and $\langle D\rangle$ the average dependence of the network. It can be seen that $\langle L\rangle$ of three networks decreases clearly as the network size N decreases, indicating that the function call efficiency is enhanced as more nodes are removed. One can also see that, for all three networks, the largest $\langle L\rangle$ value is 4.132 and the smallest $\langle C\rangle$ value is clearly as the network size N decreases, indicating that the function call efficiency is enhanced as more nodes are removed. One can also see that, for all three networks, the largest $\langle L\rangle$ value is 4.132 and the smallest $\langle C\rangle$ value is 0.012, showing the characteristic of small average path lengths and high clustering coefficient. The small-world property describes the fact that some networks are of high clustering coefficient, like regular lattices, yet have small average path lengths, like random graphs, and these networks are often called as "small-world" networks[11]. The small-world property appears to characterize most complex networks: The actors in Hollywood are on average within three co-stars from each other, or the chemicals in a cell are typically separated by three reactions[12]. Consequently, FCNT, WCNT, and SCNT are of small-world properties, i.e., they are also

"small-world" networks. The values of $\langle C \rangle$ under WCNT and SCNT is smaller than that of FCNT, meaning that there exists less triangular motifs in two coupled networks. Thus removing non-coupling nodes will decrease the aggregation of software network. Interestingly, for six directed software networks, the values of $\langle I \rangle$ and $\langle D \rangle$ is the same (Table 1 and Table 2). For a directed network, there are two kinds of degrees: The out-degree and the in-degree. Here, the out-degree equals to the number of outgoing edges, and the in-degree corresponding to the number of incoming edges. The out-degree and the in-degree for each node are generally different but the average in-degree and out-degree of the network are the same[43]. These software networks are all directed networks, and the average influence, the average dependence correspond to the average in-degree, the average out-degree, respectively. Therefore, it is reasonable that the average influence and the average dependence are the same, although the influence and the dependence with respect to each node are commonly different.

Table 1 Network properties of three kinds of networks of Tar

Network	N	M	$\langle k \rangle$	$\langle L \rangle$	$\langle C \rangle$	d	$\langle I \rangle$	$\langle D \rangle$
FCNT	1 204	3 285	5.384	4.132	0.087	11	35.322	35.322
WCNT	695	1 677	4.722	2.445	0.021	9	7.256	7.256
SCNT	109	163	2.840	1.330	0.012	3	1.917	1.917

Table 2 Network properties of three kinds of networks of MySQL

Network	N	M	$\langle k \rangle$	$\langle L \rangle$	$\langle C \rangle$	d	$\langle I \rangle$	$\langle D \rangle$
FCNM	4 598	16 018	6.514	4.294	0.119	11	1 176.345	1 176.345
WCNM	2 682	8 790	6.016	3.667	0.026	13	15.761	15.761
SCNM	393	1 288	5.715	1.870	0.048	5	6.440	6.440

Let us now consider the structural properties of three types of networks of MySQL (Table 2). One can see that the largest value of $\langle L \rangle$ for three networks is 4.294, and the smallest value of $\langle C \rangle$ is 0.026. This reflects that these networks are of a feature of small average path lengths and high clustering coefficient. Hence, according to the concept of "small-world" networks[11-12], it is obvious that FCNM, WCNM, and SCNM are "small-world" networks. From Table 2, one can see that the value of the average dependence (influence) of FCNM is 1 176.345, which greatly higher than that of WCNM and SCNM. Counterintuitively, the value of the network diameter of FCNM d is only 11, while d value of WCNM is prolonged to 13. Since some peripheral non-coupling nodes are removed from the function call network of MySQL, it is possible that the longest path in the weakly coupled network could be prolonged. Moreover, the value of the average path length of SCNM is only 1.870, implying that the strongly coupled network owns a very high function call efficiency. In particular, the value of $\langle C \rangle$ SCNM is larger than that of WCNM, reflecting that the coupled network owns a higher aggregation when non-coupling nodes are all dismissed.

Network robustness is the ability of a network to maintain its connectivity when a fraction of its nodes or edges is removed[39]. By studying the effect of various attack strategies, the key nodes corresponding to network robustness can be identified[40-42]. In this paper, we compare the efficiency of eight attacking strategies to identify key nodes in software networks. Attacking key nodes will heavily damage the network, which also means that key nodes have weakness with respect to the network robustness. The next part we will use the weakly connected indicator W and the strongly connected indicator S to evaluate the robustness of three kinds of software networks. Fig.4(a) shows the relationship between W and the attack rate $f=N'/N$ for FCNT. This shows that the value of W decreases with the increment of f, and the whole network is collapsed under HODS when $f=0.33$. We can see that the efficiency of HODS is better than that of other attacking strategies, implying that high out-degree nodes are important for the robustness of FCNT. Fig.5(a)

plots the relation between the strongly connected indicator S and f for FCNT under eight attack strategies. When the value of f is small, the effectiveness of HODS is better than that of other attacking strategies. However, HIDS outperforms other attacking strategies when the value of f is large. These results indicate that nodes with high out-degree or high in-degree are of important influence on the robustness of the function call network of Tar.

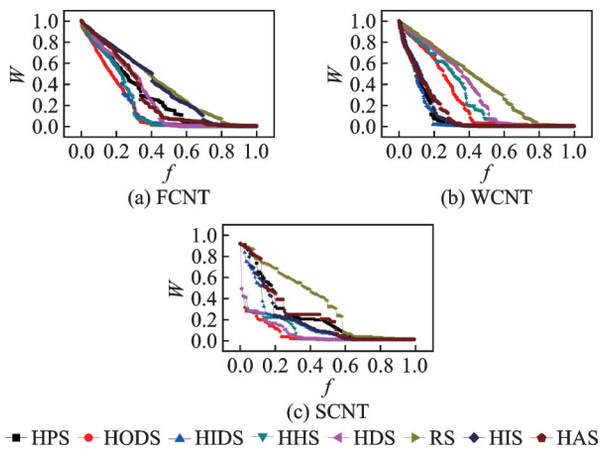

Fig.4 Weakly connected indicator W for three kinds of software networks of Tar

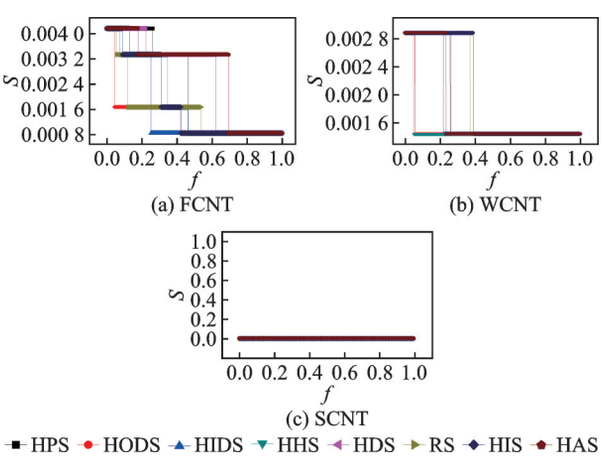

Fig.5 Strongly connected indicator S for three kinds software networks of Tar

For WCNT, we plot the relationship between W and f in Fig.4(b). One can see that the value of W decreases as f increases. HIDS outperforms other attacking strategies, and the network is completely destroyed under HIDS when f=0.27. Fig.5(b) displays the relationship between S and f for WCNT. This shows that the performance of HHS is better than that of other attacking strategies, and WCNT could be completely disintegrated under HHS when 5 percent nodes are removed, reflecting that nodes with high-hub value play an important role on the robustness of WCNT when the strongly connected indicator is adopted. In the case of SCNT, we show the relationship between W and f in Fig.4(c). This indicates that HODS outperforms other attacking strategies, and the network is entirely destroyed when f=0.28. Fig.5(c) displays the relationship between S and f for SCNT. One can see that S=0 whatever the value of f is, which indicates that SCNT is only a weakly connected network rather than a strongly connected one.

Finally, we discuss the robustness of three kinds of software networks with respect to MySQL. For FCNM, we depict the relationship between W and f in Fig.6(a). This shows that the value of W decreases with the increment of f, and HIDS is the best attacking strategy, reflecting that high in-degree nodes play a crucial role on the robustness of FCNM when the weakly connected indicator is used. Fig.7(a) displays the relation between S and f. In this case, however, HODS is the best attacking strategy instead of HIDS. In the case of WCNM, the relationship between W and f is plotted in Fig.6(b). One can see that HIDS outperforms other attacking strategies and the network is completely disintegrated when f=0.36. Fig.7(b) depicts the strongly connected indicator S as a function of f for WCNM. This indicates that the efficiency of HODS is better than that of seven other attacking strategies. For SCNM, we plot the relation between W and f in Fig.6(c). This shows that the performance of HIDS is better than that of other attacking strategies. Fig.7(c) displays the relationship between S and f for SCNM, we can see that the value of S is zero all the time, indicating that SCNM is only a weakly connected network instead of a strongly connected one which is in good accordance with the result of Fig.5(c).

In general, when the weakly connected indicator W is used, we can see that HIDS is always the best attacking strategy for MySQL whatever the kind of the network is. HODS is a good choice as the strongly connected indicator S is adopted. The

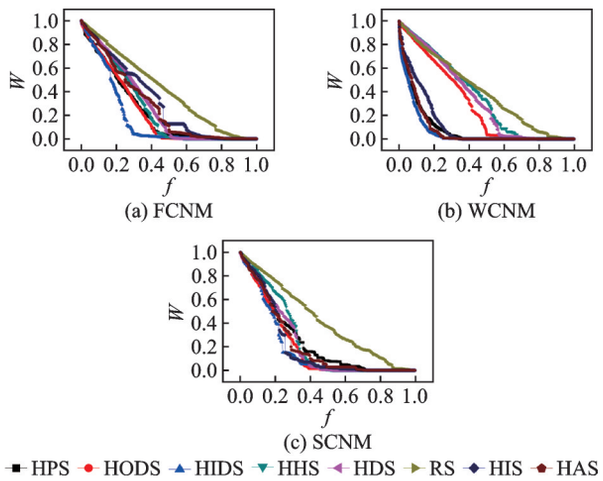

Fig.6 Weakly connected indicator W for three kinds of software networks of MySQL

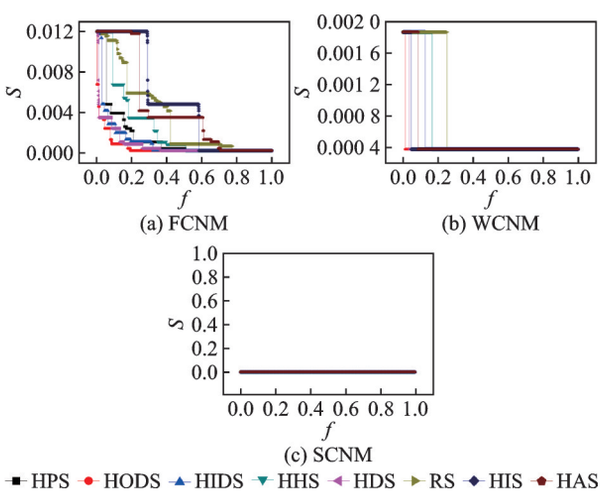

Fig.7 Strongly connected indicator S for three kinds of software networks of MySQL

result indicate that functions with high in-degree or high out-degree are of crucial influence on the robustness of MySQL.

To further understand the differences exhibited by various attack strategies in the robustness evaluation, we carried out an in-depth analysis from the perspective of network structural characteristics and complex network theory. First, regarding the weak connectivity metric, the superior performance of the HIDS is primarily due to the fact that high in-degree nodes which usually serve as convergence points or key receiving nodes in the network, responsible for handling a large amount of information flow and dependencies from other nodes. When these high in-degree nodes are attacked and removed, a significant number of nodes that depend on their inputs lose their primary connection source, leading the entire network to rapidly disintegrate and connectivity to drop. This outcome is consistent with the theoretical expectation of network fragility after targeted attacks on "hub" nodes in complex networks.

When considering the strong connectivity metric, the superiority of the HODS is evident. High out-degree nodes often function as information dissemination centers, responsible for transmitting data and function calls to multiple nodes and forming necessary feedback loops to maintain bidirectional communication within the network. When high out-degree nodes are removed, not only a large number of outgoing connections are disrupted, but the bidirectional paths formed in the network may also fracture, thereby significantly weakening the network's strong connectivity. It is apparent that the impact of different attack strategies on network connectivity is quite different, reflecting the key role of different types of nodes in maintaining the overall connectivity and functional coordination of the system.

The out-degree or in-degree of nodes has a more significant impact on the static robustness of function call networks. In other words, attack strategies based on high out-degree or high in-degree are generally the most effective. Specifically, for the Tar network, when the attack ratio f is small, the high out-degree strategy clearly outperforms the high in-degree strategy. However, when the attack ratio becomes large, the difference between the two is no longer significant. For the MySQL network, under the weak connectivity indicator, the high in-degree strategy is more likely to collapse the network compared to the high out-degree strategy, while under the strong connectivity indicator, the situation is just the opposite. In summary, the high out-degree strategy has a broader range of applicability than the high in-degree strategy. To further ensure the interpretability and generalizability of the above proposed strategies, we have performed some statistical tests on all networks including WCNT, WCNM, SCNT, and SCNM under each strategy. However, since all networks' structures are fixed due to the real-world software systems (Tar and MySQL), and thus the remove or-

der of nodes is identical for a fixed strategy, leading to the statistical results of each case are the same (Figs.4—6).

3 Conclusions

To summarize, based on the source code of two open source software systems (Tar and MySQL), we have proposed a scheme to generate coupled software networks and constructed three kinds of directed software networks: The function call network, the weakly coupled network and the strongly coupled network. The structural features of these networks are extensively discussed. It is found that all of them display small-world characteristic. For directed software networks, the average influence and the average dependence for all nodes are the same although the influence value and the dependence value for each node are generally different. By introducing eight attacking strategies, we investigate the robustness of software networks with two robustness indicators (the weakly connected indicator and the strongly connected indicator). The result show that the strongly coupled network of Tar and MySQL is just a weakly connected network rather than a strongly connected one. For MySQL, high in-degree strategy is always the best attacking strategy whatever the kind of the network is, indicating that high in-degree functions act an important role on the robustness of MySQL when the weakly connected indicator is considered. And high out-degree strategy is a good choice when we adopt the strongly connected indicator. In general, functions with high in-degree or high out-degree are of important influence on the robustness of software systems.

References

[1] MYERS C R. Software systems as complex networks: Structure, function, and evolvability of software collaboration graphs[J]. Physical Review E, 2003, 68(4): 046116.

[2] MAILLART T, SORNETTE D, SPAETH S, et al. Empirical tests of zipf's law mechanism in open source linux distribution[J]. Physical Review Letters, 2008, 101(21): 218701.

[3] SHAN C, MEI S S, HU C Z, et al. Software structure characteristic measurement method based on weighted network[J]. Computer Networks, 2019, 152: 178-185.

[4] PAN W F, LI B, LIU J, et al. Analyzing the structure of Java software systems by weighted K-core decomposition[J]. Future Generation Computer Systems, 2018, 83: 431-444.

[5] LOU Y, WANG L, CHEN G. Structural robustness of complex networks: A survey of a posteriori measures [J]. IEEE Circuits and Systems Magazine, 2023, 23 (1): 12-35.

[6] PAN W F, LI B, MA Y T, et al. Multi-granularity evolution analysis of software using complex network theory[J]. Journal of Systems Science and Complexity, 2011, 24(6): 1068-1082.

[7] CAI K Y, YIN B B. Software execution processes as an evolving complex network[J]. Information Sciences, 2009, 179(12): 1903-1928.

[8] GAO Y C, ZHENG Z, QIN F Y. Analysis of Linux kernel as a complex network[J]. Chaos, Solitons & Fractals, 2014, 69: 246-252.

[9] XIAO G P, ZHENG Z, WANG H Q. Evolution of Linux operating system network[J]. Physica A: Statistical Mechanics and Its Applications, 2017, 466: 249-258.

[10] VALVERDE S, CANCHO R F, SOLÉ R V. Scale-free networks from optimal design[J]. Europhysics Letters, 2002, 60(4): 512-517.

[11] WATTS D J, STROGATZ S H. Collective dynamics of "small-world" networks[J]. Nature, 1998, 393 (6684): 440-442.

[12] ALBERT R, BARABÁSI A L. Statistical mechanics of complex networks[J]. Reviews of Modern Physics, 2002, 74(1): 47-97.

[13] HAN Y, LI D, CHEN G. Analysis on the topological properties of software network at different levels of granularity and its application[J]. Chinese Journal of computers, 2009, 32(9): 1711-1721.

[14] PENG H E, WANG P, BING L I. An evolution analysis of software system based on multi-granularity software network[J]. Acta Electronica Sinica, 2018, 46 (2): 257-267.

[15] MA J, FAN J P, LIU F, et al. The evolution model of objective-oriented software system[J]. Journal of Jilin University, 2018, 48(2): 545-550.

[16] ZHANG B, HUANG G Y, ZHENG Z Q, et al. Approach to mine the modularity of software network based on the most vital nodes[J]. IEEE Access, 2018, 6: 32543-32553.

[17] WANG J, LIU Y H, LIU X L. Model for cascading

faults in complex software[J]. Chinese Journal of Computers, 2011, 34(6): 1137-1147.

[18] HE H T, REN R, ZHANG B, et al. Analysis onimpact of node failure in software execution network[J]. Journal of Computational Information Systems, 2015, 11(6): 2217-2225.

[19] WANG E S, REN H F, HONG C, et al. Cascading failures in coupled map lattices with sustained attack[J]. International Journal of Modern Physics C, 2021, 32(9): 2150126.

[20] SNARSKII A A. Transitions in growing networks using a structural complexity approach[J]. Physical Review E, 2024, 110(5): 054309.

[21] ARTIME O, GRASSIA M, DE DOMENICO M, et al. Robustness and resilience of complex networks [J]. Nature Reviews Physics, 2024, 6(2): 114-131.

[22] WANG D, REN X Q, WANG X F. Assessing multilayer network robustness under asymmetric coupling using motif entropy[J]. Chaos, Solitons & Fractals, 2025, 194: 116238.

[23] NEWMAN M J. Modularity and community structure in networks[J]. Proceedings of the National Academy of Sciences of the United States of America, 2006, 103(23): 8577-8582.

[24] BOCCALETTI S, LATORA V, MORENO Y, et al. Complex networks: Structure and dynamics[J]. Physics Reports, 2006, 424(4/5): 175-308.

[25] HONG C, ZHANG J, CAO X B, et al. Structural properties of the Chinese air transportation multilayer network[J]. Chaos, Solitons & Fractals, 2016, 86: 28-34.

[26] WANG J W, LI Y, ZHENG Q F. Cascading load model in interdependent networks with coupled strength[J]. Physica A: Statistical Mechanics and Its Applications, 2015, 430: 242-253.

[27] LIU R R, EISENBERG D A, SEAGER T P, et al. The "weak" interdependence of infrastructure systems produces mixed percolation transitions in multilayer networks[J]. Scientific Reports, 2018, 8(1): 2111.

[28] TAN F, XIA Y X, WEI Z. Robust-yet-fragile nature of interdependent networks[J]. Physical Review E, 2015, 91(5): 052809.

[29] JIANG J, XIA Y, XU S, et al. An asymmetric interdependent networks model for cyber-physical systems[J]. Chaos, 2020, 30(5): 053135.

[30] LIU R R, JIA C X, LAI Y C. Asymmetry in interdependence makes a multilayer system more robust against cascading failures[J]. Physical Review E, 2019, 100(5): 052306.

[31] BULDYREV S V, PARSHANI R, PAUL G, et al. Catastrophic cascade of failures in interdependent networks[J]. Nature, 2010, 464(7291): 1025-1028.

[32] TAN F, XIA Y X, ZHANG W P, et al. Cascading failures of loads in interconnected networks under intentional attack[J]. Europhysics Letters, 2013, 102 (2): 28009.

[33] CUI P S, ZHU P D, WANG K, et al. Enhancing robustness of interdependent network by adding connectivity and dependence links[J]. Physica A: Statistical Mechanics and Its Applications, 2018, 497: 185-197.

[34] GAO Y L, CHEN S M, NIE S, et al. Robustness analysis of interdependent networks under multiple-attacking strategies[J]. Physica A: Statistical Mechanics and Its Applications, 2018, 496: 495-504.

[35] YANG Y, XU K J, HONG C. Network dynamics on the Chinese air transportation multilayer network[J]. International Journal of Modern Physics C, 2021, 32 (5): 2150070.

[36] KLEINBERG J M. Authoritative sources in a hyperlinked environment[J]. Journal of the ACM, 1999, 46 (5): 604-632.

[37] PAGE L, BRIN S, MOTWANI R, et al. The pagerank citation ranking: Bringing order to the web: SIDL-WP—1999-0120[R]. [S.l.]: Stanford Infolab, 1999.

[38] COSTA L F, RODRIGUES F A, TRAVIESO G, et al. Characterization of complex networks: A survey of measurements[J]. Advances in Physics, 2007, 56 (1): 167-242.

[39] DEKKER A H. Simulating network robustness: Two perspectives on reality[C]//Proceedings of SimTecT 2004 Simulation Conference. [S.l.]: [s.n.], 2004: 126-131.

[40] ALBERT R, JEONG H, BARABASI A L. Error and attack tolerance of complex networks[J]. Nature, 2000, 406(6794): 378-382.

[41] HOLME P, KIM B J, YOON C N, et al. Attack vulnerability of complex networks[J]. Physical Review E, 2002, 65(5): 056109.

[42] XU K J, HONG C, ZHANG X H, et al. Cascades in coupled map lattices with heterogeneous distribution of perturbations[J]. Physica A: Statistical Mechanics and Its Applications, 2020, 547: 123839.

Acknowledgements This work was supported by the Beijing Education Commission Science and Technology Project (No.KM201811417005), the National Natural Science Foundation of China (No. 62173237), the Aeronautical Science Foundation of China (No.20240055054001), the Open Fund

of State Key Laboratory of Satellite Navigation System and Equipment Technology (No. CEPNT2023A01), Joint Fund of Ministry of Natural Resources Key Laboratory of Spatiotemporal Perception and Intelligent Processing (No. 232203), the Civil Aviation Flight Technology and Flight Safety Engineering Technology Research Center of Sichuan (No. GY2024-02B), the Applied Basic Research Programs of Liaoning Province (No. 2025JH2/101300011), the General Project of Liaoning Provincial Education Department (No. 20250054), and Research on Safety Intelligent Management Technology and Systems for Mixed Operations of General Aviation Aircraft in Low-Altitude Airspace (No. 310125011).

Authors

The first author Prof. **WANG Ershen** received the Ph.D. degree in Communication and Information System from Dalian Maritime University, China, in 2009. From January 2014 to December 2016, he worked as a postdoc in Beihang University, China. He is currently a professor in Shenyang Aerospace University, Shenyang, China. His research interests include BeiDou navigation satellite system / global navigation satellite system and signal processing, integrity monitoring, integrated navigation, target tracking, artificial intelligence and applications to unmanned systems.

The corresponding author Dr. **HONG Chen** received the Ph.D. degree in information and signal processing from Beihang University and conducted postdoctoral work at the National Key Laboratory of Communications, Navigation and Surveillance/Air Traffic Management until 2016. Now, he is an associate professor of College of Robotics at Beijing Union University. His recent research has focused on multi-agent systems and game theory.

Author contributions Prof. **WANG Ershen** conducted the writing-original draft, supervision, resources, and methodology, formal analysis and conceptualization. Mr. **TONG Zeqi** conducted the writing-review & editing, visualization, software, and data curation. Dr. **HONG Chen** conducted the writing-review & editing, writing-original draft, validation, supervision, resources, project administration, methodology, investigation, funding acquisition, formal analysis, and conceptualization. Dr. **WANG Yanwen** conducted the writing-original draft, visualization, software and data curation. Mr. **MEI Sen** conducted the validation, investigation, and formal analysis. Mr. **XU Song** conducted the supervision, investigation, funding acquisition, and conceptualization. Ms. **NA La** conducted the supervision, funding acquisition, and conceptualization.

Competing interests The authors declare no competing interests.

(Executive Editor: WANG Jie)

耦合软件网络的结构特征与鲁棒性

王尔申[1,2], 佟泽奇[1], 宏 晨[3,4], 王延文[5], 梅 森[6], 徐 嵩[1], 娜 拉[1]

(1.沈阳航空航天大学电子信息工程学院,沈阳 110136,中国; 2.智慧中原地理信息技术河南省协同创新中心,郑州 450052,中国; 3.北京联合大学多智能体系统研究中心,北京 100101,中国;4.北京联合大学机器人学院,北京 100101,中国; 5.东北大学计算机科学与工程学院,沈阳 110169,中国; 6.沈阳无距科技有限公司,沈阳 110170,中国)

摘要:软件系统在现代社会中发挥着越来越重要的作用,抵御攻击的能力对关键软件系统具有重要的实际意义,这使得软件系统的结构和鲁棒性在近年来引起了极大的关注。本文基于Tar和MySQL的源代码,提出了一种生成耦合软件网络的方法,并构建了3种有向软件网络:函数调用网络、弱耦合网络和强耦合网络。对这些复杂网络的结构特性进行了广泛的研究。研究发现,所有函数的平均影响力和平均依赖性是相同的。此外,引入了8种攻击策略和两个鲁棒性指标(弱连通指标和强连通指标)来分析软件网络的鲁棒性。研究表明,强耦合网络只是一个弱连通网络,而不是强连通网络。对于MySQL,当使用弱连通指标时,高入度策略优于其他攻击策略。另一方面,当采用强连通指标时,高出度策略是一个很好的选择。此研究将有助于更好地理解软件网络的结构和鲁棒性。

关键词:软件网络;软件结构;软件鲁棒性;软件系统;复杂网络