P2P Network Based on Neighbor-neighbor Lists

Hao Zhao¹, Chenyang Wang², Yuefei Zhu¹ and Wei Lin¹

¹State Key Laboratory of Mathematical Engineering and Advanced Computing, Zhengzhou, Henan, 450001, China
²Information Engineering University Information Technology Section, Zhengzhou, Henan, 450001, China

*Corresponding author: 754094629@qq.com

Abstract. In order to solve the vulnerability of the network structure in the real world, this paper constructs a P2P network model based on neighbor-neighbor lists. We propose two algorithms, network repairing and pruning, to improve the effectiveness and robustness of the network. Simulation experiments show that the proposed model and algorithms can effectively enhance the self-healing of P2P network.

1. Introduction

The central structure of the network has the inherent vulnerability of "single point failure". Although technologies such as Domain-Flux [1] and Fast-fluxing [2] can play a role as a protection center server, it does not fundamentally change its concentration. Therefore, the P2P network has become a research hotspot in academia and industry. Its distributed topology and the characteristics of peer nodes have well avoided the defects of a single central server.

Robustness is an important indicator for evaluating network structure. A large number of related papers elaborate and analyze the robustness of networks from different aspects. Because all the nodes controlled by the network are distributed in different physical locations, they may be ordinary machines or servers. Therefore, there is uncertainty in their online time. Some nodes may join or leave the network frequently, and some nodes may be online during the day and offline at night. Whether it is a temporary exit of a node or a permanent killing, the performance of the network is offline. In P2P networks, robustness can be equated with the loss of some nodes and the impact of offline nodes on the entire network.

Existing P2P networks generally use the known P2P network protocol directly, and this method has the advantage of being relatively simple to implement. However, P2P control networks are quite different from ordinary P2P networks. A normal P2P network does not require global information, such as the number of all nodes, which is important for P2P control networks. On the other hand, network protocols need to be more robust because all nodes may be removed at any time. Based on the above points, this paper proposes an improved P2P network topology with self-healing characteristics. Compared with common P2P structures, it enhances self-healing and robustness, and adds repairing and pruning algorithms to effectively improve anti-detection capabilities.

2. P2P network overview
Although the pure central structure network has the advantages of simple implementation, high efficiency, and good synergy, its control process has a single node failure. Once the attack personnel close the central command control server, the node will not be able to continue to acquire commands and messages. The master will also lose control of the nodes. The non-central structure P2P (peer-to-peer) mode is used as its channel topology to improve the robustness of the network and the defense against the shutting down [3]. As shown in Figure 1, each node host in the network acts as both a client and a server, and the communication process does not depend on the public network to reach server resources. Unlike traditional networks, which are vulnerable to domain names or servers being seized, P2P networks create a relatively fragmented network environment. The idea is that all node hosts are connected to each other and communicate. Although the P2P network command delivery delay is higher than the central structure, it does not depend on the characteristics of the vulnerable central node [4], making it difficult to hijack, measure, and shut down.

3. Self-repairing network design and implementation

3.1. Neighbor-neighbor graph design
The bootstrap mechanism commonly used in P2P networks mainly includes a random scan mode and a peer list (Peer-list) mode. The network based on random scanning has inherent vulnerabilities of traffic anomalies. Communication is based on Peer-list exchange. Each node host needs to maintain a list of node information, that is, a list of nodes and periodically select some of them. Communicating, obtaining master’s commands and updating node information, such P2P networks are flexible, scalable, and robust, and are used in many popular P2P networks [5].

Figure 2 is a P2P network built on a node list with 12 node nodes. The node list information maintained by each node is as shown in Figure 3. For example, node 1 knows the addresses of nodes 2, 3, and 7, so that communication communication is established with the three nodes in the initial stage of the network, and the commands of the node controller are received. Dynamically update network node information.

In actual operation, node nodes have the possibility of being invalidated by pollution hijacking, and due to the parity between nodes in the P2P network, the loss of the number of nodes can easily cause the embarrassment of the entire network. Therefore, how to ensure the effectiveness of networks as much as possible in the case of losing some node nodes is a problem worth studying. On the one hand, it is necessary to make the remaining nodes establish communication connections to ensure the maximum operation of the network; on the other hand, when the number of nodes is reduced, the communication between the node nodes is more frequent, which is easier to be discovered by traffic analysis.

Aiming at the above actual situation and theoretical analysis, this paper proposes a distributed dynamic self-healing network topology. Literature [6] studied NoN greedy routing in P2P networks, which can reduce the length of the route and proved to be asymptotically optimal. In this article, we discuss how to use the concept of NoN to create a self-healing network.

**Definition 1** Neighbor-neighbor graph: Consider graph $G$ with $n$ nodes, the node set is denoted as $V$, and the node is denoted as $u_i$ ($0 \leq i < n$). The set of neighbor nodes of $u_i$ is denoted as $N(u_i)$, and $u_i$ knows the node information contained in $N(u_i)$, that is, each node has the identity and address (such as IP) of the neighbor node of its neighbor.

Taking the network connection diagram in Figure 2 as an example, the node list of node 1 in the neighbor-neighbor graph is shown in Figure 4. The neighbor nodes of node 1 are nodes 2, 3 and 7, wherein the neighbor nodes of node 2 are 1, 10 and 11, the neighbor nodes of node 3 are 1, 4 and 6 ..., and so on, that is, the neighbors of neighbor nodes The node information exists in the node list of node 1, which is the design idea of the neighbor-neighbor graph, which is the basis for the construction of the distributed self-repairing topology proposed in this paper. The next section focuses on specific algorithms for implementing self-healing topologies on neighbor-neighbor graph models.
3.2. Algorithm design and description

The previous section has pointed out two points to improve the robustness of P2P networks. The following two problems are solved. Two targeted algorithms are proposed under the neighbor-neighbor graph model.

The first consideration is how the remaining nodes spontaneously form new communication connections when a node is removed to continue to play the network’s performance. In a P2P network, each node is both a client and a server, so once it fails, the impact is enormous. Based on the neighbor-neighbor graph, this paper proposes a network repairing algorithm, which uses the information of the neighbor nodes in the neighbor list to form a new connection between the neighbor nodes of the removed node, and “replaces” the relay function of the removed node. It is possible to maintain connectivity throughout the network. The specific operation of the repairing is: when a node $u_i$ is deleted, it is judged whether the neighbor nodes $u_j$ and $u_k$ are reachable, and if not, the edge formed by $u_i$, $(u_j, u_k)$ is added to the existing edge set $E$. The flow of the algorithm is described as follows:

**Algorithm 1 Repairing**

1. for each deleted node $u_i$
2. for neighbors of $u_i$: $u_j, u_k$
3. if $u_j$ and $u_k$
4. connect $(u_j, u_k)$;
5. renew node lists;
6. end for
7. end for
8. finish

The specific process of network repairing can be seen in Figure 5. The neighbor nodes of node 7 are 0, 1, and 4. When node 7 is removed, nodes 0, 1, and 4 traverse their neighbor list to determine whether they are connected to each other, and then form a new one. The edges $(0,1)$, $(1,4)$ and $(0,4)$. Thereafter, when node 6 is removed, the same process is repeated, forming new edges $(0, 3)$, $(3, 8)$ and $(8, 0)$.
Next consider the problem on the other hand. For the graph $G$, when the node $u_i$ is deleted, each of its nodes starts the repairing process as described above. However, this causes the degree of neighbors of $u_i$ (which can be simply understood as the number of neighbors of the node) to increase significantly after $t$ steps, such as node 0 in Figure 5, after undergoing the repairing process in which nodes 6 and 7 are removed, its degree increased from 3 to 5. This can significantly increase the likelihood that node 0 will be detected. Therefore, the range $[d_{\text{min}}, d_{\text{max}}]$ of a node degree is determined, so that each neighbor node of the node $u_i$ deletes the node with the highest degree in the node list (i.e., disconnects the communication connection with it) until the degree range of the neighbor node meets the condition. Details are described in Algorithm 2, pruning.

4. Establishment of evaluation indicators

When studying the robustness of existing networks, it is common to observe the change of the connection rate after deleting a certain number of nodes by simulation. The less the impact of the connection rate, the less deleted nodes in the network have little effect on the remaining nodes, so the better the robustness. For real networks, however, it is impossible to remove certain nodes to study their robustness. In fact, networks can be regarded as complex networks in nature, and for the study of complex networks, the relevant knowledge of graph theory is usually used [7]. Therefore, this paper selects the attributes and indicators of graph theory to evaluate the robustness of networks. However, existing attributes such as degrees and maximum distances in graph theory cannot be directly used as indicators of P2P network performance.

In order to test the effectiveness of the proposed neighbor network model and the proposed two algorithms, and to evaluate its ability to improve the robustness of P2P networks, this paper proposes two attributes of centrality and connectivity as evaluation indicators.

**Definition 2 Centrality**: The reciprocal of the sum of the shortest paths between node $u$ and all $n-1$ other nodes.

The centrality mainly reflects the speed at which the message spreads from the node $u$ to all other nodes in the network, and is therefore the reciprocal of the sum of the path distances. Considering that the sum of distances depends on the number of nodes, the calculation needs to be normalized. The calculation formula is as shown in equation (1), where $n$ is the number of nodes, and $d(u, v)$ is between nodes $u$ and $v$. The shortest path.

$$C(u) = \frac{n - 1}{\sum_{v \neq u} d(u, v)}$$ (1)
In the experiment, the centrality of a single node can't reflect the whole network. Therefore, the average Centrality of the whole network is taken during the test, which is calculated by (2).

\[ C = \frac{\sum C(u_i)}{n} \]  

(2)

**Definition 3 Connectivity:** The ratio of the degree of the node \( u \) to the maximum possible degree.

The degree of connectivity indicates the probability that a message propagating in the network flows through node \( u \), which actually reflects the intensiveness of network traffic. The connection degree calculation formula is as shown in the formula (3), where \( d(u) \) represents the degree of the node \( u \), and \( k \) represents the maximum degree of the network node.

\[ D(u) = \frac{k}{d(u)} \]  

(3)

In the experiment, the average connectivity of all nodes can be used as a test indicator, and the average connectivity can be calculated by equation (4).

\[ D = \frac{k \cdot n}{\sum d(u_i)} \]  

(4)

5. Experimental testing and analysis

Although theoretical analysis and verification are more rigorous, they are more difficult. Therefore, this paper uses simulation experiments to evaluate the characteristics of networks based on neighbor lists. The experiment simulates the node deletion process in a \( k \)-regular graph of 500 nodes (\( k = 5, 10, 15 \)), with up to 30% (150) nodes being deleted.

![Figure 6 Centrality](a) (b)

Figure 6 shows that the average centrality of nodes in the network as a function of the number of deleted nodes when pruning (Figure 6(b)) and not pruning (Figure 6(a)). As reflected in Figure 7, the node centering is stable and it does not decrease even after the node is deleted. On the other hand, when it is not trimmed, since the effect of the repairing process between the remaining nodes is not canceled, the average Centrality is significantly increased. In fact, the centrality reflects the speed of message propagation, as long as the basic requirements are met, greater than or equal to the initial rate. The experimental results show that the network model proposed in this paper has no negative impact on the network centrality.
Figure 7 Connectivity

Figure 7 reflects the average degree of connectivity of nodes in the network as a function of the number of deleted nodes when pruning (Figure 7(b)) and not pruning (Figure 7(a)). Obviously, if the trimming process is not added after the repairing process, the connection degree of the node will increase greatly, which indicates that the message transmission rate of the entire network is significantly improved. However, a large increase in the communication frequency of the network will result in a significant increase in communication traffic between nodes, with the result that it is easily detected and targeted, causing the network to be exposed and eventually found to be corrupted.

In fact, it is necessary to keep the network low connectivity, especially in Advanced Persistent Attacks (APT). The network keeps the low connection to avoid alarms, so that the chances of detection and removal are reduced. For example, nodes in Stuxnet only infect the other three nodes at most to slow down their spread and reduce the risk of detection [8].

6. Conclusion

The Internet has become an indispensable part of people's work and life. How to improve the stability of network services has become an urgent problem to be solved. This paper studies the improvement of network topology and proposes a distributed and self-healing P2P network architecture. Through network repairing and pruning algorithms, the robustness of the network are enhanced, which can effectively resist the impact of node removal. The next step will be to apply this model to P2P network services such as file transfer and download to verify its reliability under real conditions.

References

[1] Almomani, A. (2016). Fast-flux hunter: a system for filtering online fast-flux botnet. Neural Computing & Applications, 1-11.
[2] Soltanaghaei, E., & Kharrazi, M. (2015). Detection of fast-flux botnets through dns traffic analysis. Scientia Iranica, 22(6).
[3] Grizzard, J. B., Sharma, V., Nunnery, C., Kang, B. H., & Dagon, D. (2007). Peer-to-peer botnets: overview and case study. In USENIX Workshop on Hot Topics in Understanding Botnets (HotBots'07, 1-1.
[4] Feily, M., Shahrestani, A., & Ramadass, S. (2009, June). A survey of botnet and botnet detection. In Emerging Security Information, Systems and Technologies, 2009. SECURWARE'09. Third International Conference on (pp. 268-273). IEEE.
[5] Yin, J., Cui, X., & Li, K. (2017, June). A Reputation-Based Resilient and Recoverable P2P Botnet. In Data Science in Cyberspace (DSC), 2017 IEEE Second International Conference on (pp. 275-282). IEEE.
[6] Manku, G. S., Naor, M., & Wieder, U. (2004). Know thy neighbor's neighbor:the power of lookahead in randomized P2P networks. (pp.54-63).
[7] MacArthur, B. D., Sánchez-García, R. J., & Anderson, J. W. (2008). Symmetry in complex networks. Discrete Applied Mathematics, 156(18), 3525-3531.
[8] Langner, R. (2011). Stuxnet: Dissecting a cyberwarfare weapon. IEEE Security & Privacy, 9(3), 49-51.