Linux Kernels as Complex Networks: A Novel Method to Study Evolution

Lei Wang   Zheng Wang   Chen Yang   Li Zhang   Qiang Ye

School of Computer Science, Beijing University of Aeronautics and Astronautics, Beijing, China
wanglei@buaa.edu.cn, {wangzheng, yangchen}@cse.buaa.edu.cn, lily@buaa.edu.cn

Dept. of Computer Science & IT, UPEI, Charlottetown, Canada
qye@upei.ca
Outline

- Introduction
- Call graphs of Linux kernel modules
- Evolution of nodes and edges
- Features of complex networks
- A method to find major structural changes
- Conclusions
Introduction (1)

- There are all kinds of networks around us:
  - Social networks (e.g. friendship networks)
  - Technological networks (e.g. the Internet)
  - Biological networks (e.g. neural networks)
- Many real networks are neither regular graph nor random graph.
- Complex networks have been used to study these networks.
Code statistics is a widely used method to study software evolution. However, this method can only be used to predict the scale of software systems in terms of code size. It cannot be used to study the internal structures of software system. We use complex networks to study the evolution of operating systems.
A call graph is a directed graph that represents the calling relationship between functions in a program.

Each function in the program is modeled as a graph node.

When a function calls another function, an edge is added to the graph.
  - The edge starts from the caller node and ends at the called node.
Call graphs of Linux kernel modules (2)

- We compiled 223 different Linux kernels that fall in the range of Version 1.1.0 to 2.4.35
- For clarity, we assigned sequence numbers to these kernels according to the chronological order.
- Namely, the kernel that appeared earliest is No. 1 kernel and the most recent kernel is No. 223 kernel.
Evolution of nodes

- Number of nodes: we are focused on the file system and drivers module.
Evolution of edges

- Number of edges
Two problems to solve

- Are the call graphs of Linux kernel complex networks?

- How to use the features of complex networks?
  - Many studies focused on how to generate complex networks
  - They were not carried out from the view of software engineering
Features of complex networks

- Small-world:
  - The average path length is very small.

- Other features:
  - Scale-free
  - Clustering coefficient
  - Preferential attachment
Average path length

- The average path length of a graph is defined as the average distance between any pair of nodes in the graph.

\[
L = \frac{1}{\frac{1}{2} N (N + 1)} \sum_{i \geq j} d_{ij}
\]

where \( d_{ij} \) is the distance between node \( i \) and \( j \), \( N \) is the number of nodes within the graph, and \( L \) is the average path length of the graph.
Small-world feature
A method to find major structural changes

- The theory of small-world model shows that $L$ scales as $\ln(N)$, $N$ is the number of nodes in the graph.

- We checked whether this property holds for the call graphs of Linux kernels using the concept of the "slope" of the average path length.

$$k_{ij} = \frac{L_j - L_i}{\ln(N_j) - \ln(N_i)}$$
Unusual evolitional points
Structural Change Discovery

1. Calculate the slope and find out the unusual evolitional points.

2. Compare the call graph of the kernel that leads to an unusual evolitional point with that of its previous kernel, and find out the edges that are added or deleted in the call graph of the later kernel.

3. Find out the nodes that are associated with the edges that were found in step 2 and count the number of appearances. This might result in a large number of nodes. However, 1 to 3 nodes usually account for more than 50% of the total number of appearances.

4. Finally, we need to analyze the Linux code of the functions corresponding to the nodes that account for more than 50% and find out the structural changes.
The first unusual point

<table>
<thead>
<tr>
<th>Node</th>
<th>edge</th>
</tr>
</thead>
<tbody>
<tr>
<td>smp_processor_id</td>
<td>175</td>
</tr>
<tr>
<td>sys_vhangup</td>
<td>2</td>
</tr>
<tr>
<td>mem_mmap</td>
<td>2</td>
</tr>
<tr>
<td>xiafs_unlink</td>
<td>1</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Four unusual points in fs

- SMP support leads to the first and second unusual point that corresponds to kernel 1.3.39 and 2.1.29.
- The unusual point corresponding to kernel 2.3.48 occurs because of the reconstruction of *hfs* file system and two functions in *vfs*, *block_read_full_page* and *block_write_full_page*.
- Finally, the *jfs* file system reconstruction triggers the last unusual point that corresponds to kernel 2.4.23.
Conclusions

- Generated the call graphs of 223 Linux kernels (Version 1.1.0 to Version 2.4.35)
- The call graphs of the file system and driver module in all Linux kernels under investigation are scale-free small-world complex networks.
- Proposed a generic method that could be used to find major structural changes that occur during the evolution of software systems.
Thanks!