THE DEPENDENCY NETWORK IN FREE OPERATING SYSTEM*

Tomasz M. Gradowski[a†], Maciej J. Mrowinski[a]
Robert A. Kosinski[a,b]

[a]Warsaw University of Technology, Faculty of Physics
Koszykowa 75, 00-662 Warszawa, Poland
[b]Central Institute for Labour Protection — National Research Institute
(CIOP — PIB), Czerniakowska 16, 00-701 Warszawa, Poland

(Received December 12, 2011)

Phenomena observed in many complex systems can be attributed to their network structure. In this paper we present an analysis of the dependency network of a large software project — Debian Operating System (GNU/Linux distribution) and show its properties, like power-law degree distribution, modularity and hierarchical organization.

DOI:10.5506/APhysPolBSupp.5.47
PACS numbers: 89.20.–a, 89.75.Fb, 64.60.ah

1. Introduction

Free operating systems are becoming more and more popular nowadays. Many computer users choose these systems as an alternative for commercial systems because of their low price (actually no price), flexibility and reliability. In this case *free* means not only *free of charge*, but first of all *libre*. Free software is written and maintained by communities of open source developers and based on free licenses like GPL (General Public License). It means that it can be used, studied and modified without restriction, and it can be copied and redistributed in modified or unmodified form either without restriction or with minimal restrictions. Free software powers not only home computers but also large servers and data centers as well as mobile devices. In this paper we focus on the Debian Operating System, which is one of the most popular and influential GNU/Linux distributions [1].

* Presented at the Summer Solstice 2011 International Conference on Discrete Models of Complex Systems, Turku, Finland, June 6–10, 2011.
† Corresponding e-mail address: tomgrad@if.pw.edu.pl

Such a project is a collection of a large number of software packages. Each package can contain a compiled program, library, documentation, source files or other data, such as images and sound files. The distribution defines relations between these packages — dependencies. Each package has a list of dependencies — other packages required for its installation. A small portion of this system — the dependency subgraph of one particular package — is presented in Fig. 1. The whole distribution can be described as a large complex network of dependencies between packages. A proper structure of the dependency graph implies stability of the whole operating system and its susceptibility to random damages or attacks.

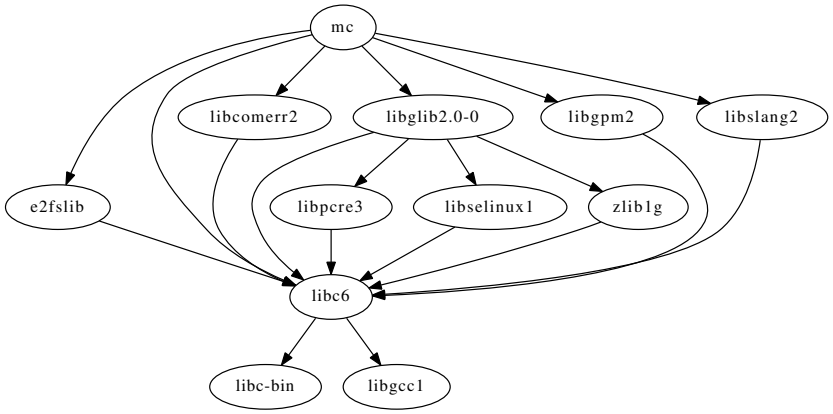

Fig. 1. Dependency graph for one of the packages from the Debian distribution (mc — Midnight Commander — a very popular file manager). Each node represents one package from the distribution. Relations are represented by arrows pointing from one package to all its dependencies. Top package mc is the root of this subgraph and the core package libc6 is present with 9 incoming links.

In this paper we study the dependency network of the Debian 6.0 Operating System. We conduct research on the scaling of the degree distribution. We analyze the nodes correlations in order to describe the assortativity of the network. By studying the relation between the degree and the average degree of nearest neighbors, we find the disassortative mixing, which is typical for technical networks, as well as biological networks like food-webs.

A paper on the same matter was published recently by de Sousa *et al.* [2]. The authors presented interesting results of the research devoted to the stability and modularity of the Debian Operating System. In our paper we study this issue further and deliver more results and comments concerning modularity and node correlations.

2. The network

In this section we present some basic properties of the dependency network. In our research we used the up-to-date stable release of Debian 6.0 with only the main repository active. Network consists of 28234 nodes and 116783 edges. The fully connected component (giant cluster) contains 25775 vertices and 116644 edges. The edges represent the dependencies between packages, therefore there are no cycles and no parallel links in the network.

The degree distribution for the giant cluster is depicted in Fig. 2, separately for incoming and outcoming links. The maximum indegree is $k_{\text{in}} = 12364$. Package with such a large number of incoming links is labeled libc6, it is the most important library, since the most used language in the GNU/Linux ecosystem is *C*. The distribution follows the power law $k_{\text{in}}^{-\gamma}$ with the γ exponent of 2.

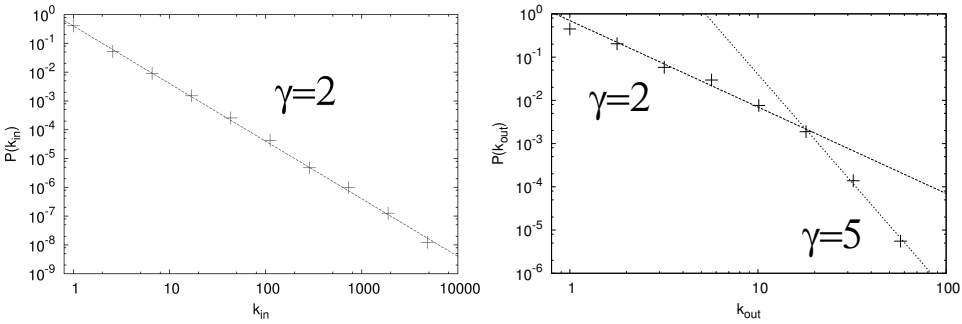

Fig. 2. Scaling properties of the dependency network. Degree distributions for incoming (left) and outcoming links (right). Dashed line shows the $k^{-\gamma}$ slope with $\gamma = 2$ for indegrees, $\gamma = 2$ and $\gamma = 5$ for outdegrees.

Distribution of outdegrees is more interesting because of the double scaling. For small degrees, under 20, we observe scaling with exponent $\gamma = 2$, the same as for ingoing links. But above that value the situation drastically changes and the probability obeys k_{out}^{-5} scaling. There is rather simple explanation of this phenomenon. Outdegree is a number of dependencies of a packages. The more dependencies a package has, the harder it is to maintain such a package. The developers and distribution maintainers try to keep the number of dependencies as low as possible. Sometimes it is better to split one packages into two or more packages with less dependencies to simplify the maintenance.

Results presented in this section partially agree with those by de Sousa *et al.* [2], however in our calculations we used logarithmic bin sizes for histograms to obtain degree distributions, instead of constant size bins, which is a more proper method for scale free systems. With constant binning it is impossible to describe the double scaling of outdegree phenomenon.

3. Hierarchical organization

Clustering coefficient of a node indicates how close its nearest neighbors are to a complete graph (clique). For a directed graph it is defined as follows [3] (simplified version of the original definition due to unweighted network)

$$C_i = \frac{1}{k_i(k_i-1)} \sum_{j,k} a_{ij} a_{ik} a_{jk} \,, \tag{1}$$

where a_{ij} equals 1 if there is a directed edge from i-th to j-th node and 0 otherwise. The clustering coefficient is based on triplets (triangles) $a_{ij}a_{ik}a_{jk}$ and because the edges represent software dependencies its maximum value can be 0.5. If there is a link from node i to j and from i to k, there can be only one directed link between j and k. Otherwise there would be a cycle, which is forbidden in network of dependencies. The global clustering coefficient averaged over 15819 nodes with $k_{\text{out}} \geq 2$ is $\langle C \rangle = 0.306$.

The network of dependencies is not only scale-free, but also modular and hierarchical. Modularity is a tendency of groups of nodes to cluster with each other. Modularity is a common property of large software projects as well. Qt and GTK+ libraries, KDE and GNOME desktop environments, Xorg graphical system, Open Office suite, Tex Live distribution, just to name the most popular, all these projects consist of many programs or libraries which have similar purpose and share their dependencies. For instance, Open Office is a collection of programs like Writer, Calc, Impress, Draw which are tools for writing, spreadsheets, presentations and drawing. On the other hand modules often depend on each other, *i.e.* GNOME environment is build with the GTK+ library, which works on top of the Xorg system.

It was shown previously [4] that networks with hierarchical organization exhibit power-law dependency between the average clustering coefficient and degree. This property is common for networks without strong geographical constrains (*i.e.* where cost of the link is insignificant) and was proven for few real networks: actors collaboration networks, semantic web, WWW, Internet on the autonomous system level and metabolism networks. We found similar property in Debian network of dependencies (Fig. 3). Although the $\langle C(k_{\text{out}}) \rangle$ relation follows the scaling law k^{-2} as opposed to k^{-1} in cited results, it proves the existence of different degrees of modularity in the network. This result is opposite to random networks, where clustering coefficient is k-independent.

In hierarchical networks the higher level of hierarchy means lower clustering coefficient, and this kind of organization is observed in Debian dependency network for packages with more than 20 dependencies. But this relation is not observed for packages with lower k_{out} due to the existence of a large number of hierarchy-independent modules.

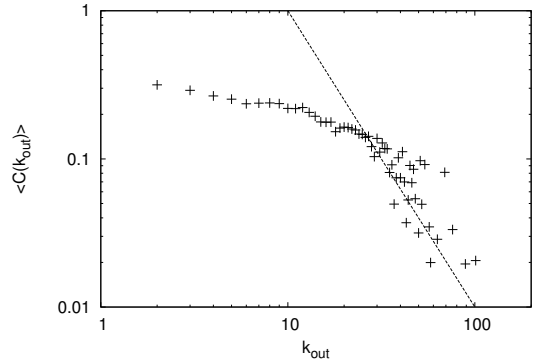

Fig. 3. The scaling of clustering coefficient C with k_{out}. The dashed line has slope -2. The $C(k)$ scaling indicates hierarchical organization in the network.

4. Assortativity

Assortativity is a tendency for vertices in networks to be connected to vertices with certain properties [5]. In particular, in networks with assortative mixing high-degree nodes tend to connect to other high-degree nodes (common for social networks). Such a behavior is opposite to disassortative mixing where high-degree nodes tend to connect only to low-degree nodes (different sorts of technical networks).

Following this definition and focusing on the average connectivity of nearest neighbors [6] we can determine how important the dependencies are. In order to do so, we present the $\langle k_{\text{in}} \rangle_{\text{DEPS}}(k_{\text{out}})$ relation, where $\langle k_{\text{in}} \rangle_{\text{DEPS}}$ is a number of ingoing links averaged over all the dependencies of a node with outgoing degree k_{out} (Fig. 4). Decreasing average reveals the disassortative

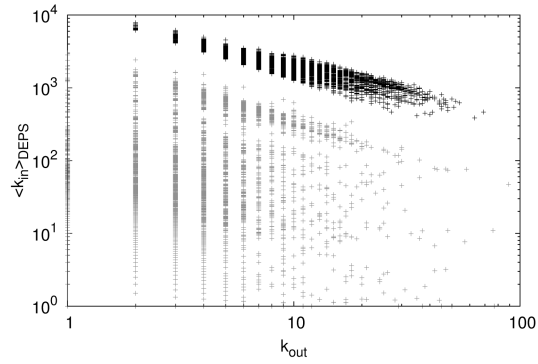

Fig. 4. Average indegree of dependencies of each package with its outdegree. Each point represents one package. Packages depending on the libc6 library are colored darker than the rest of packages.

mixing but does not deliver the full information about correlations in this network. A large number of nodes with high $\langle k_{\text{in}} \rangle_{\text{DEPS}}$ is separated from the rest of nodes. This is a consequence of the existence of a super hub in the network (libc6 library which is a dependency for almost half of the packages) and scale-free nature (most nodes have a low degree).

5. Conclusions

Operating system understood as system kernel, basic tools and user applications, just like in popular GNU/Linux distributions, is a complex system with a structure of a scale-free network. This fact gives us not only the opportunity to describe computer software in the language of complex networks but also opens a lot of possibilities to study such systems with methods known from complex networks field. One of the crucial issues is the stability of operating system and its resistance for random damages as well as planned attacks.

One of the common problems causing instabilities in GNU/Linux distributions, including Debian, are mistakes in the dependency graph. These mistakes, in computer jargon called bugs, often mean missing dependencies, non-existing dependencies, wrong version of dependencies or dependency loops (cycles). All these bugs make it impossible to install damaged package. Though Debian is known for its high quality and stability, we found a small number of cycles in the dependencies.

Of course, determining the stability of the distribution relying on *e.g.* the connectivity analysis is a massive simplification. It would be interesting to investigate the effect of higher order dependencies (that is dependencies of dependencies) on the stability.

REFERENCES

[1] `http://www.debian.org`

[2] O.F. de Sousa, M.A. de Menezes, T.J.P. Penna, *JCIS* **1**, 127 (2009).

[3] A. Barrat, M. Barthelemy, R. Pastor-Satorras, A. Vespignani, *PNAS* **101**, 3747 (2004).

[4] E. Ravasz, A.-L. Barabasi, *Phys. Rev.* **E67**, 026112 (2003).

[5] M.E.J. Newman, *Phys. Rev. Lett.* **89**, 208701 (2002).

[6] R. Pastor-Satorras, A. Vazquez, A. Vespignani, *Phys. Rev. Lett.* **87**, 258701 (2001).