DE GRUYTER OPEN

*Int. J. Appl. Math. Comput. Sci.*, 2015, Vol. 25, No. 2, 281–293
DOI: 10.1515/amcs-2015-0022

amcs

# LOCAL DEPENDENCY IN NETWORKS

MILOŠ KUDĚLKA [a,*], ŠÁRKA ZEHNALOVÁ [a], ZDENĚK HORÁK [b], PAVEL KRÖMER [a], VÁCLAV SNÁŠEL [a]

[a] Department of Computer Science
VŠB—Technical University of Ostrava, 17. listopadu 15, 708 33, Ostrava, Czech Republic
e-mail: {milos.kudelka,sarka.zehnalova.st}@vsb.cz
{pavel.kromer,vaclav.snasel}@vsb.cz

[b] Inflex, s.r.o., Legionářská 1085/8, 779 00, Olomouc, Czech Republic
e-mail: horak@inflex.cz

Many real world data and processes have a network structure and can usefully be represented as graphs. Network analysis focuses on the relations among the nodes exploring the properties of each network. We introduce a method for measuring the strength of the relationship between two nodes of a network and for their ranking. This method is applicable to all kinds of networks, including directed and weighted networks. The approach extracts dependency relations among the network's nodes from the structure in local surroundings of individual nodes. For the tasks we deal with in this article, the key technical parameter is locality. Since only the surroundings of the examined nodes are used in computations, there is no need to analyze the entire network. This allows the application of our approach in the area of large-scale networks. We present several experiments using small networks as well as large-scale artificial and real world networks. The results of the experiments show high effectiveness due to the locality of our approach and also high quality node ranking comparable to PageRank.

**Keywords:** complex networks, graphs, edge weighting, dependency.

## 1. Introduction

The network representation of complex systems provides a useful model for studying many real-world processes, including biological, technological and social networks (Barabási and Frangos, 2002). Most of the previously studied networks were unweighted binary ones, although an extensive area of network measures has already been extended to the notion of weighted networks (Abdallah, 2011; Barrat *et al.*, 2004a; Opsahl *et al.*, 2010) as well as clustering methods (Farkas *et al.*, 2007; Opsahl and Panzarasa, 2009). Edge weighting can be based on different approaches, which are always related to the type of network. Node weighting (ranking) is usually understood as directly related to the edges that nodes have with their neighbors and the weight of those edges.

Due to the enormous and sustained growth of real world networks, the current trend in analyzing networks is to focus on local methods and also on weighted networks (we can assign weights to the edges as well as the nodes). This is related to the fact that some tasks as originally formulated are difficult to solve in very large networks. Other tasks were formulated only for unweighted networks, which, however, does not quite correspond to the development of real networks where, for example, the relationship between the nodes changes over time. Nevertheless, it appears that the same or modified tasks become significant when looking at a relatively small part of the network.

In this article we will consider only unweighted undirected networks, despite the fact that the following approach is directly applicable even to directed and weighted networks. The reason is to simply illustrate our approach. The first task that the article addresses is the ranking of the nodes in the network. We introduce a dependency centrality measure, which is similar to the simple degree centrality but includes in its calculation the relationships between common neighbors. For the experiments we use both artificial and real world

[*] Corresponding author

networks and compare the results with measurements of PageRank. Due to the fact that in the experiments we use large-scale networks, we describe the details of parallel implementation of the algorithms. The second task is to transform an unweighted undirected network into a weighted network, and in the experiment we show that applying a dependency measure may help to uncover interdependent groups of nodes.

The rest of this article is organized as follows. In Section 2, we discuss the related work. The dependency measure is presented in Section 3. Section 4 addresses the proposed method for ranking the nodes. In Section 5, we focus on an experiment with large-scale networks and its results. Measuring the dependency in an unweighted network is presented in Section 6. Section 7 presents an experiment with transforming unweighted networks. Section 8 concludes the article.

## 2. Related work

Weighted networks have previously been studied in many papers and applications. Many of them emphasize the advantages of weighted networks over classical (binary) ones (Ghazalpour *et al*., 2006; Zhang and Horvath, 2005). There are many approaches to assigning weights to edges in networks. The most natural ones usually arise from reality. For example, for a social network of a telephone company, customers we may assign weight to an edge based on the total length of conversation between the nodes (Onnela *et al*., 2007). Edge weights in a network of air transport between cities can be assigned on the basis of the total number of passenger (or seats) on particular flights (Barrat *et al*., 2004a). The genes network may also benefit from assigning weights to edges based on a similar function of particular genes (Ghazalpour *et al*., 2006). It is quite common that more than one weighting approach based on reality exists.

Community extraction is another area that has been studied extensively (Newman, 2006; Fortunato, 2010), because analyzing interconnected groups provides important information about how they function (Newman, 2008). One of the first works on this topic (Newman, 2004) illustrates the problem of community detection within a group of monkeys. In this particular case, the unweighted network was unable to separate the monkeys based on their grooming habits. When considering the amount of grooming as a weight of an edge within the network, the problem became solvable.

A weighted network might be also considered a result of network evolution (Abdallah, 2011; Barrat *et al*., 2004b). An interesting area of research with many potential applications is the field of link prediction, particularly link weight prediction (Kahanda and Neville, 2009). Zhang and Horvath (2005) described an approach similar to that presented in this article. A method of assigning weight to an edge as a measure of topological overlap between two nodes was used. Han *et al*. (2009) introduced the concept of supportiveness, which captures co-authorship relations in a non-symmetric way and derived a supportiveness-based author ranking scheme.

It makes sense to assign weights not only to edges, but also to nodes (Wiedermann *et al*., 2013). The centrality of nodes, or the identification of which nodes are more ‘central’ than others, has been a key issue in network analysis. Freeman (1979) formalized three different measures of node centrality: the degree, closeness and betweenness. The degree is the number of nodes that a selected node is connected to, and measures the involvement of the node in the network. Based on our dependency measure, we propose a novel degree centrality measure, which provides a ranking of nodes in the network from the most independent to the most dependent ones.

The term ‘dependency’ used in this article has already been used in conjunction with social networks, but this approach is based on probability and is suited to collaborative filtering (Heckerman *et al*., 2001) or modeling influence (Leenders, 2002). Also, when constructing partial correlation networks (Kenett *et al*., 2010), the ‘dependency’ of one node on another is calculated for the entire network. The dependency measure used in this article was also used by Zehnalova *et al*. (2013).

## 3. Local dependency

We understand dependency as a generally asymmetric measure describing a relationship between two nodes of a network. In a network or graph $G = (V, E)$, $V$ is a set of nodes and $E$ is a set of edges.

The computation of the dependency $D(x, y)$ of node $x$ on node $y$ is done locally, only in the immediate surroundings of the two nodes. From the surroundings of node $x$, only edges that lead to the neighbors of $y$ are taken into account. This means that only relations between a ‘friend of a friend’ have some significance.

**3.1. Motivation.** We assume that the dependency between two nodes is influenced not only by the relation between them, but also by the relationships in their surroundings. Let us show an example with two adjacent nodes, $x$ and $y$, in an unweighted undirected network. Consider the situation shown in Figs. 1(a) and (b), where nodes $x$ and $y$ share an edge. The nodes in the first figure have no additional neighbors. In the second figure, node $y$ is adjacent to three additional nodes. The intuition behind the term *dependency* says that the relation between nodes in Fig. 1(a) is balanced, while the situation in Fig. 1(b) is different—node $y$ is less dependent on node $x$ than vice versa. Figure 1(c) contains two additional edges between node $x$ and two different nodes. In this situation, node $x$ is no longer so highly dependent on node $y$ because of the two new edges. When thinking about dependency in Fig. 1(c), we should also consider that the new edges include common neighbors of nodes $x$ and $y$ (which transmit some part of the dependency on node $y$). It is evident that dependency is an asymmetric measure since $D(x, y) = D(y, x)$ may not always hold.

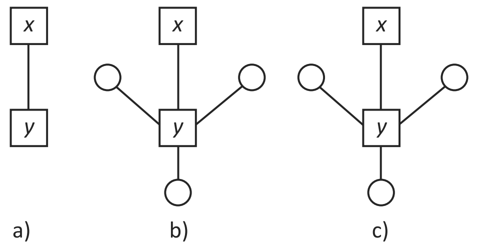

Fig. 1. Examples of dependency between two nodes.

**3.2. Dependency in a weighted network.** In a weighted network, the weight of edges usually reflects a similarity between the nodes, while the strength of an edge describes the level of cooperation or traffic between the nodes. These weights must be taken into consideration when calculating dependency. To illustrate how 'friend of a friend' relations affect the dependency between two nodes, look at the situation in Fig. 2, where we want to assess the weight from node $V_i$ to node $V_j$ across node $V_x$. When $w_{xj}$ is weak, $w_{ij}$ must also be weak, e.g., $\lim_{w_{xj}\to 0}(V_iV_j)_{V_x} = 0$; alternatively, when $w_{xj}$ is very strong (in terms of similarity), meaning that nodes $V_x$ and $V_j$ are almost identical, then $w_{ij}$ is identical to $w_{ix}$, e.g., $\lim_{w_{xj}\to\infty}(V_iV_j)_{V_x} = w_{ix}$.

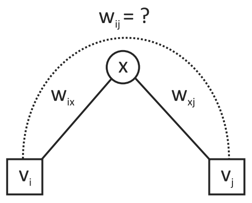

Fig. 2. Example of weight derived from a path.

We define dependency as follows. Let $E(x)$ be the set of all non-zero weight edges adjacent to node $x$. Let Adj$(x, y)$ be the set of all edges between node $x$ and any of the neighbors of node $y$. Clearly, Adj$(x, y) \subseteq E(x)$ and Adj$(x, y)$ does not contain an edge between node $x$ and node $y$. Let $W(e)$ be the weight of edge $e$ and $W(v_1, v_2)$ the weight of an edge between nodes $v_1$ and $v_2$ ($W(v_1, v_2) = 0$, if there is no such edge).

Let $x$ be a non-isolated node of the network. The dependency $D(x, y)$ of node $x$ on node $y$ is defined as

$$D(x, y) = \frac{W(x, y) + \sum_{e_i\in \mathrm{Adj}(x,y)} W(e_i)\cdot R(e_i)}{\sum_{e_i\in E(x)} W(e_i)}, \quad (1)$$

$$R(e_i) = \frac{W(y, v_i)}{W(e_i) + W(y, v_i)}, \quad (2)$$

where $R(e_i)$ is the coefficient of the dependency of node $x$ on node $y$ via the common neighbor $v_i$; therefore, $v_i \in e_i$.

This dependency describes the relation of one node to another node from the point of view of their surroundings. Using this concept, the dependency of one node on another is calculated for the entire network (see Algorithm 1). We obtain a directed weighted adjacency matrix representing the fully connected network, which is capable of uncovering hidden relationships between the nodes. Once the adjacency matrix has been constructed, it is possible to reconstruct the network; several algorithms may be used, such as the minimum spanning tree (MST), or just some sort of threshold.

The presented equations infer $D(x, y) \in [0, 1]$. The dependency being equal to zero means that vertices $x$ and $y$ have no common edge or neighbor. Full dependency (dependency equal to one) describes the situation where node $x$ has only one common edge with node $y$.

**Remark 1.** The dependency is non-zero if at least one of the following conditions holds:

1. There exists an edge between vertices $x$ and $y$.

2. Vertices $x$ and $y$ share at least one common neighbor.

**Algorithm 1.** Calculating dependencies.

```
Require: a graph $G = (V, E)$, $w : E \to \mathbb{R}$
 1: for all node $v \in V$ do
 2:   for all node $x$ from Adj$(v, y)$ do
 3:     if $x$ is adjacent to $v$ then
 4:       $sum_{nom} += w_{v,x}$
 5:     end if
 6:     $sum_{nom} += w_{x,y} \cdot R(x, y)$
 7:   end for
 8:   $sum_{den} += w_{v,y}$
 9:   $D(v, y) = \frac{w_{v,y} + sum_{nom}}{sum_{den}}$
10: end for
11: return $D = |V| \times |V|$ {Returns directed weighted
    adjacency matrix}
```

## 4. Node ranking

In this section we present two methods for node ranking. The dependency centrality measure is introduced first. The second one is a well-known method, PageRank, which was selected in order to compare it with the dependency centrality measure. The results of the application of both methods will be compared in Section 5.

**4.1. Dependency degree.** We define the dependency degree of a node as the sum of all calculated dependency weights of incoming edges from its neighbors. Let $N(x)$ be the set of all nodes adjacent to node $x$. Let $d_{ij}$ be the dependency weight of a connection from $i$ to $j$,

$$Deg_x = \sum_{y \in N(x)} d_{yx}. \tag{3}$$

Since dependency between the nodes $D(x,y) \in [0,1]$, the sum of dependencies will be smaller than the classical degree centrality of the node. The higher the dependency degree, the more nodes in the surroundings of a focal node depend on it. In this manner we are able to create a natural ranking of nodes based on their importance in the network. For reference, we calculated some of the known centrality measures for the unweighted karate club network of Zachary (Zachary, 1977). In Table 1, each measure has a different scale of values, but it is possible to cross-reference them and find differences in proportionality. In Table 1, only half of the nodes (those with the highest node degree) listed were (see Figs. 3 and 4) with the whole network, where the size of the nodes corresponds to the dependency degree and PageRank, respectively (the values here are scaled to a proper interval). Nodes with a noticeable visible difference are highlighted.

**Remark 2.** The dependency degree centrality is by definition suitable for weighted networks.

Table 1. Centrality measures for Zachary's karate club network.

| Vertex | Degree | Dependency | Betweenness | Closeness | PageRank |
|---|---|---|---|---|---|
| 34 | 17 | 9,267 | 160,552 | 0,017 | 3,431 |
| 1 | 16 | 9,108 | 231,071 | 0,017 | 3,298 |
| 33 | 12 | 6,403 | 76,690 | 0,016 | 2,438 |
| 3 | 10 | 3,785 | 75,851 | 0,017 | 1,941 |
| 2 | 9 | 4,456 | 28,479 | 0,015 | 1,798 |
| 4 | 6 | 2,727 | 6,288 | 0,014 | 1,219 |
| 32 | 6 | 1,805 | 73,010 | 0,016 | 1,263 |
| 9 | 5 | 1,120 | 29,529 | 0,016 | 1,012 |
| 14 | 5 | 1,160 | 24,216 | 0,016 | 1,004 |
| 24 | 5 | 1,522 | 9,300 | 0,012 | 1,072 |
| 6 | 4 | 1,875 | 15,833 | 0,012 | 0,990 |
| 7 | 4 | 1,875 | 15,833 | 0,012 | 0,990 |
| 8 | 4 | 1,101 | 0,000 | 0,013 | 0,833 |
| 28 | 4 | 0,822 | 11,792 | 0,014 | 0,872 |
| 30 | 4 | 1,464 | 1,543 | 0,012 | 0,894 |
| 31 | 4 | 0,795 | 7,610 | 0,014 | 0,836 |

**4.2. PageRank.** The PageRank (Brin and Page, 1998; Langville and Meyer, 2006) algorithm was conceived in order to rank linked documents and is inseparably linked to the area of Web searches. However, from a more general perspective, it can be seen as a weighting algorithm that assigns numerical scores (weights) to the nodes of a network, similarly as LocalDependency does. In contrast to LocalDependency, which utilizes local information to infer node weights, PageRank is a global measure that uses structural information about the whole network to weight nodes. Some local approximations have been considered: Fortunato *et al.* (2008) estimated PageRank from local knowledge of in-degree (in the specific case of the Web network) while Bar-Yossef and Mashiach (2008) studied the effectiveness of Reverse PageRank. The basics of PageRank are outlined in this section in order to clarify the similarities and differences with LocalDependency.

The PageRank algorithm, originally inspired by bibliometric measures used for ranking scientific papers, is an objective estimation of the relevance of hypertext documents on the basis of their position within a hypertext network. The main idea behind PageRank is very simple. It assumes that documents that are linked from other trusted pages (i.e., pages with a high PageRank score) are trusted. The contribution of an incoming link is proportional to the trustworthiness of its origin.

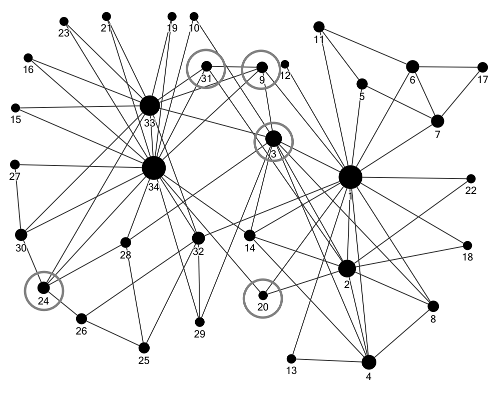

Fig. 3. Dependency degree.

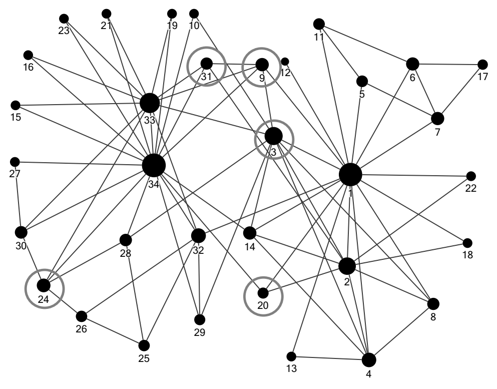

Fig. 4. PageRank.

The highest contribution comes from links that are incoming from prestigious documents. In Google's search algorithm, PageRank is utilized for the prioritization of documents in the result set retrieved by a keyword-based search. The PageRank of document $A$, denoted by $PR(A)$, can be evaluated using (Brin and Page, 1998; Langville and Meyer, 2006)

$$PR(A) = 1 - d + d \sum_{k=1}^{n} \frac{PR(T_k)}{C(T_k)}. \tag{4}$$

The variable $d$ in Eqn. (4) represents a dumping factor (Brin and Page, 1998; Langville and Meyer, 2006) (or teleportation probability (Witten *et al.*, 2006)) from the interval [0,1]. The function $C(A)$ evaluates the number of hyperlinks leading from document $A$. PageRank is based on the so-called random surfer model. The random surfer starts his or her session at a random Web document. In every document, the surfer can either follow any of the hyperlinks linking the page to the rest of the World Wide Web or, with probability $d$, teleport to another randomly chosen Web document (including documents not linked from the current one). The concept of teleportation allows documents to be reached that are not linked from any other document on the Web (Witten *et al.*, 2006).

Practical computation of the page rank vector is realized by applying power iteration to the sparse stochastic primitive matrix $\boldsymbol{G}$, which is a twice-modified row-normalized hyperlink matrix $\boldsymbol{H}$. The relationship between $\boldsymbol{G}$ and $\boldsymbol{H}$ is illustrated as follows:

$$\boldsymbol{G} = \alpha \boldsymbol{S} + (1 - \alpha)\boldsymbol{E}. \tag{5}$$

Google matrix $\boldsymbol{G}$ depends on the parameter $\alpha$, stochastic matrix $\boldsymbol{S}$ and teleportation matrix $\boldsymbol{E}$. Stochastic matrix $\boldsymbol{S}$ is used to deal with sinks in the hyperlink graph. Once the user enters a node (opens a document) that has no outlinks, she or he is offered the possibility of jumping to any other document in the collection with uniform probability. In other words, stochastic matrix $\boldsymbol{S}$ replaces all 0s rows of $\boldsymbol{H}$ with rows containing $1/n$,

$$\boldsymbol{S} = \boldsymbol{H} + \frac{1}{n}\boldsymbol{a}\boldsymbol{e}^T, \tag{6}$$

where vector $\boldsymbol{a}$ is a dangling vector having $a_i = 1$ if page $i$ is a dangling node (has no outlinks); otherwise, $a_i = 0$. Matrix $\boldsymbol{E}$ represents a teleportation matrix. It gives the surfer the possibility to teleport to any document in the collection at any time,

$$\boldsymbol{E} = \frac{1}{n}\boldsymbol{e}\boldsymbol{e}^T. \tag{7}$$

In Eqn. (7), vector $\boldsymbol{e}^T$ is the row vector of all 1s. The elements of $\boldsymbol{H}$ are set according to

$$\boldsymbol{H}_{ij} = \begin{cases} \frac{1}{|P_i|} & \text{if } i \text{ links to } j, \\ 0 & \text{otherwise.} \end{cases} \tag{8}$$

The non-zero elements of row $i$ represent outlinks from $i$, non-zero elements of column $j$ represent inlinks to $j$. The parameter $\alpha \in [0,1]$ in Eqn. (5) is a scaling parameter controlling the priority between link-following and random teleportation.

The random surfer is bound to the assumptions and simplifications made by Page and Brin in order to be able to compute PageRank. A real Web surfer, however, does not seem to follow all outlinks from a page with uniform probability or teleport to any document in the collection randomly. One approach to the improvement of PageRank is the personalization of hyperlink matrix $\boldsymbol{H}$ and teleportation matrix $\boldsymbol{E}$.

#### 4.2.1. Distributed PageRank computation.

Since PageRank processes primarily the Web matrix and other very large networks, distribution is necessary to obtain results in a reasonable time. Google uses MapReduce (Lin and Dyer, 2010) based distributed implementation of the power iteration method. However, many other studies have dealt with different types of distributed PageRank computations in the past.

The design of the fully distributed PageRank for P2P networks is due to Sankaralingam *et al.* (2003). The authors based their solution on the chaotic iterative solution of linear systems and developed a complex model of the computation. The method allowed incremental PageRank updates as documents to be added and was intended for a ranking and search application designed for P2P networks. The evaluation of the proposed algorithm was only theoretical (i.e., no implementation details were considered). It was, however, cited as an inspiration for another work dealing with distributed PageRank computation (de Jager, 2004). The study proposed three distributed algorithms based on the original PageRank, which incrementally mitigated distribution- and communication-related issues of the distributed PageRank computation.

A practical approach to the distributed PageRank computation in the environment of a real-word cluster was presented by Rungsawang and Manaskasemsak (2003). The proposal included a design of data structures and used the message passing interface (MPI), namely, the MPICH library, for communication. The authors performed computational experiments and showed that their method scales. Practical issues related to data distribution were discussed as well.

The communication overhead of the previous method was addressed by Manaskasemsak and Rungsawang (2004) and an optimized algorithm was proposed. Another improvement involving a hybrid MPI–multi-threaded implementation which very well fits the architecture of compute nodes of real-world clusters was presented and evaluated by Manaskasemsak *et al.* (2006).

The communication overhead which exists when executing the traditional power iteration method in a distributed way was also the focus of the work of Zhu *et al*. (2005). The authors proposed a divide-and-conquer algorithm for iterative aggregation and disaggregation utilizing the block Jacobi smoothing method to reduce the communication.

An MPI-based library for MapReduce (Lin and Dyer, 2010) was presented by Plimpton and Devine (2011) and PageRank was used as one of the examples of graph algorithms implemented efficiently and in a portable way using their framework. The framework isolated application developers from distribution and communication details, but remained transparent, i.e., allowed the client code to call the MPI directly. The work pointed out and solved a number of issues found during the implementation of MapReduce in the MPI but, in contrast to other MapReduce libraries, did not provide fault-tolerance and redundancy.

A distributed Monte Carlo-based approach to the PageRank computation was presented by Das Sarma *et al*. (2013). The work emphasized the suitability of random-walk based Monte Carlo methods for scalable distributed PageRank computation (especially in contrast to power iteration, which is hard to perform in a distributed context and is sensitive to the volume of communication involved). The study proposed two PageRank algorithms with good time complexity $O(\sqrt{\log(n)/\epsilon})$ and $O(\sqrt{\log(n)}/\epsilon)$ for directed and undirected graphs, respectively. Moreover, a proof of the probability of convergence of the proposed algorithms was presented as well.

## 5. Experiment 1: Large-scale networks

For the use in this work, we implemented a simple MPI-based distributed implementation of the iterative version of PageRank as defined by Eqn. (4) for execution on the Anselm cluster. The cluster represents the first phase ("small cluster") of the Czech National Supercompting Centre IT4Innovations. It consists of 209 compute nodes, totaling 3344 compute cores with 15 TB RAM and giving over 94 Tflop/s theoretical peak performance. Each node is an x86-64 computer with 16 cores and 64+ GB RAM. The nodes are interconnected by fully non-blocking fat-tree Infiniband networks. A few nodes are also equipped with an NVIDIA Kepler GPU or Intel Xeon Phi MIC accelerators. However, GPUs and MIC accelerators were not used in our computations. Anselm uses the PBS Pro (Nitzberg *et al*., 2004) resource allocation manager in version 12.

**5.1. Distributed PageRank and LocalDependency computation.** The distributed implementation was meant to be simple, intuitive, well portable (e.g., to clusters of commodity PCs) and, last but not least, easy to extend by other graph computations that cannot be expressed in a vector × matrix form like PageRank such as LocalDependency. The main aim was to enable computation of iterative graph algorithms in a distributed and memory-efficient way. The MPI code quite literally corresponds to PageRank as defined by eq. (4) and LocalDependency. Each MPI process in this approach computes the PageRank (or LocalDependency) score for a portion of nodes of the original graph. It knows the local topology (i.e., the edges among its assigned nodes) and neighborhood connections (i.e., the edges connecting local nodes to remote nodes), and no other information is stored in the MPI process. Each PageRank update between local nodes is handled locally with no communication overhead, and each PageRank update between remote nodes causes a message to be sent and received. The communication is handled in an asynchronous manner using the *MPI_Isend* and *MPI_Irecv* MPI functions for message passing and *MPI_Waitall* for synchronization. The MPI library used for our computations was OpenMPI 1.6.5.

**5.2. Results.** We computed PageRank and LocalDependency (as defined in Section 4.1) on four different large datasets, two artificial networks with 100,000 and 1,000,000 nodes, and on two co-authorship networks constructed from a DBLP database[1]. We chose the PageRank measure to compare with LocalDependency, because the results on small networks (as seen in Table 1) suggested some similarity between them. Different scales of values of PageRank and LocalDependency were normalized for the following comparison. Figures 5(a) and (b) depict cumulative distributions of PageRank and LocalDependency values for artificial networks with 100,000 and 1,000,000 nodes, respectively. Likewise, Figs. 5(c) and (d) show those distributions for unweighted and reduced DBLP, respectively. The reduced DBLP network was constructed using the forgetting function, which takes into account the frequency and regularity of publishing (for details, see Kudělka *et al*., 2012), so it became a weighted network.

It is clear that the distribution of LocalDependency values is similar to PageRank. For the difference in actual values, see Fig. 6, where in the case of the DBLP networks more than $95\%$ of values fall in the 0–$5\%$ range. Since both PageRank and LocalDependency provide orderings of the data based on the importance of the nodes, we compare orders given by PageRank and LocalDependency. We use Kendall's $\tau$ (Christensen, 2005) as a measure of concordance between the two ranked lists[2]. Kendall's $\tau$ is a number in the interval

[1]`http://www.informatik.uni-trier.de/~ley/db/`.

[2]There are several definitions of $\tau$ based on how ties should be treated; we used the one known as $\tau_b$.

Table 2. Kendall's $\tau$ of two orderings.

| Graph | No. of nodes | $\tau$ |
|---|---|---|
| artificial 100k | 100,000 | 0.841 |
| artificial 1M | 1,000,000 | 0.774 |
| DBLP unweighted | 1,216,515 | 0.822 |
| DBLP reduced | 318,971 | 0.969 |

$[-1, 1]$, where two orders induced by the ranks that are the same have the value of 1; conversely, two orders that are opposite of each other have the value of $-1$, while value $0$ can be interpreted as a lack of correlation. High values of $\tau$ measured on our four datasets (see Table 2) suggest that most pairs of values are in the same order in both lists.

**5.3. Note on performance.** The performance of the employed implementation on two artificial data sets and on two variants of DBLP is presented in this section. However, we note that the experiments were executed in a shared production environment which was utilized up to 90% before and during our experiments (i.e., only limited resources were available). The results of the experiments in terms of LocalDependency (LD) and PageRank (PR) execution times are shown for all data sets in Table 3 and illustrated in Fig. 7. Table 3 also shows the speedup (or slowdown) in execution times for 40, 64, 80, 100, and 120 MPI processes when compared with execution using 32 MPI processes.

The results of the performance measurements clearly confirm the expected: a naïve parallel implementation of PageRank and LocalDependency does not yield a significant performance boost and scales poorly. The computation of LocalDependency, which in contrast to PageRank requires only a single iteration, even slows down due to the growing communication overhead, especially for smaller graphs. The speedup on graphs with a larger number of nodes and edges is superlinear, but the execution time of the application using 32 MPI processes was actually slower than a fine-tuned single-process implementation for small graphs. It can also be seen that the performance is data bound and depends on the structure of the graph because the execution times for *1M nodes*, *DBLP unweighted*, and *DBLP reduced* are different despite the similar number of nodes.

## 6. Dependency in an unweighted network

There are many ways of weighting the relations (Barrat *et al.*, 2004a), it could be based on the traffic between the nodes, similarity or the amount of collaboration. We may use the dependency approach to unweighted networks and obtain a weighted version of the network; again a directed adjacency matrix is calculated, which is capable of uncovering hidden relationships between the nodes.

Table 3. Execution time for Anselm.

| Pro- | Time [s] | | Speedup | |
|---|---|---|---|---|
| cesses | LD | PR | LD | PR |
| | *100k nodes* | | | |
| 32 | 1134.57 | 6512.81 | 1.00 | 1.00 |
| 40 | 984.367 | 3650.96 | 1.15 | 1.78 |
| 64 | 1396.18 | 2987.83 | 0.81 | 2.18 |
| 80 | 1984.11 | 2783.59 | 0.57 | 2.34 |
| 100 | 2143.56 | 2972.12 | 0.53 | 2.19 |
| 120 | 3057.69 | 3112.97 | 0.37 | 2.09 |
| | *1M nodes* | | | |
| 32 | 21515.3 | 211747.00 | 1.00 | 1.00 |
| 40 | 14681 | 111701.00 | 1.47 | 1.90 |
| 64 | 5316.6 | 29276.20 | 4.05 | 7.23 |
| 80 | 6431.91 | 16912.90 | 3.35 | 12.52 |
| 100 | 3260.74 | 11623.50 | 6.60 | 18.22 |
| 120 | 3917.37 | 9323.83 | 5.49 | 22.71 |
| | *DBLP unweighted* | | | |
| 32 | 70123.7 | 872217.00 | 1.00 | 1.00 |
| 40 | 48034.5 | 520514.00 | 1.46 | 1.68 |
| 64 | 22204.8 | 147783.00 | 3.16 | 5.90 |
| 80 | 15370.6 | 79201.80 | 4.56 | 11.01 |
| 100 | 12889.6 | 70454.40 | 5.44 | 12.38 |
| 120 | 11217.3 | 49441.00 | 6.25 | 17.64 |
| | *DBLP reduced* | | | |
| 32 | 2324.79 | 23903.40 | 1.00 | 1.00 |
| 40 | 2144.85 | 12874.50 | 1.08 | 1.86 |
| 64 | 1767.31 | 5154.57 | 1.32 | 4.64 |
| 80 | 2461.76 | 3973.94 | 0.94 | 6.02 |
| 100 | 2862.5 | 4274.06 | 0.81 | 5.59 |
| 120 | 3198.01 | 4215.51 | 0.73 | 5.67 |

Dependency degree centrality may be used for assigning weights to the nodes.

Using Eqns. (1) and (2) for an unweighted network, i.e., all weights are equal 1, and measuring the dependency only between the neighbors, we get what follows.

Let $x$ not be a non-isolated node of the network. The dependency $D(x, y)$ of node $x$ on node $y$ is

$$D(x, y) = \frac{1 + \sum_{e_i \in \mathrm{Adj}(x,y)} \frac{1}{2}}{\deg(x)}. \tag{9}$$

For situations depicted in Fig. 1, the following holds:

(a) $D(x, y) = D(y, x) = 1$,

(b) $D(x, y) = 1$, $D(y, x) = 1/4$,

(c) $D(x, y) = 2/3$, $D(y, x) = 1/2$.

## 7. Experiment 2: Network edge weighting

In this section we focus on applications of our method on real-world network data and we analyze two well-known networks in the literature: Zachary's karate club network

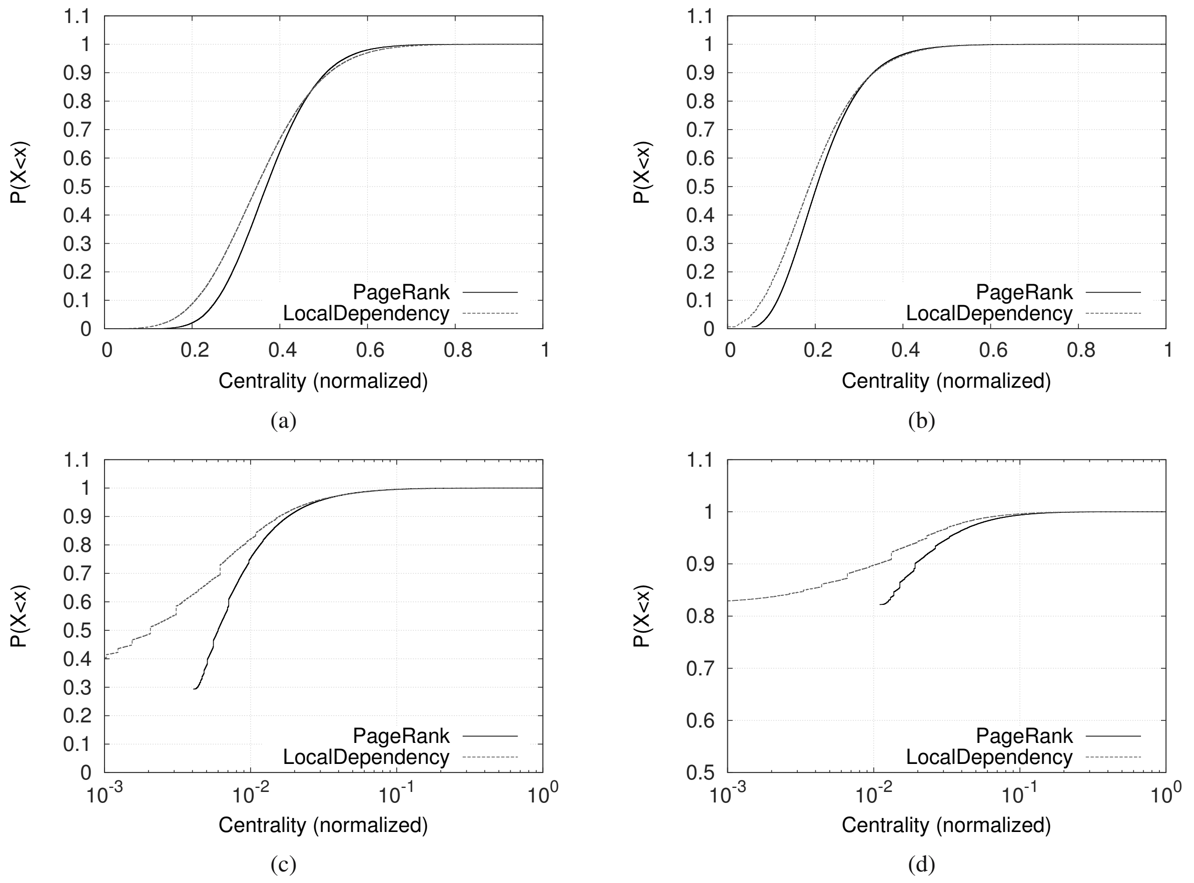

Fig. 5. Cumulative distributions (normalized): 100k nodes (a), 1M nodes (b), DBLP unweighted (c) and DBLP reduced (d).

(Zachary, 1977) and a social network of a community of bottlenose dolphins (Lusseau, 2003). We took the original unweighted data and, using our method from the previous section, transformed those networks into weighted ones. From the obtained directed adjacency matrix, we retained only the edges between the nodes that were adjacent in the original networks. Then, from a pair of dependency edges between two nodes, we took the value of a maximum of them. We assigned weights to the nodes according to the dependency degree centrality. The resulting networks are presented in Figs. 8 and 12.

**7.1. Transforming unweighted networks.** We may perform a simple reduction of this network and filter some of the weakest edges. We define a binary dependency $D(V_i, V_j) \in \{0, 1\}$ and say that nodes are dependent when $D(V_i, V_j) = 1 : d_w(V_i, V_j) \geq 0.5$; they are not dependent when $D(V_i, V_j) = 0 : d_w(V_i, V_j) < 0.5$. Using this simple threshold we reduced our weighted networks—see the emerging community structures in Figs. 9 and 13. Also, the remaining parts of the network suggest a strong dependency of nodes with a small dependency degree on the nodes with a high dependency degree. In a co-authorship network this is usually a type of relationship that doctoral students have to their supervisors.

To obtain yet another view of the network, this time from the perspective of independency, we made an inversion of dependency weights $I(V_i, V_j) = 1 - D(V_i, V_j)$; see Figs. 10 and 14 for the results. Here, the strong edges are between the nodes that are not dependent on each other; they represent independent relationships and their removal would affect the network connectivity.

And after another reduction based on binary division with independent nodes defined as $I(V_i, V_j) \in \{0, 1\}$ where $I(V_i, V_j) = 1 : i_w(V_i, V_j) > 0.5$ and $I(V_i, V_j) = 0 : i_w(V_i, V_j) \leq 0.5$, we get the part of the network where the remaining nodes are independent; see Figs. 11 and 15 for the results. In a co-authorship network this is a peer-to-peer type of relationship that professors may have between each other. Removal of any of the nodes that are connected in this reduced network would considerably affect this node's surroundings; conversely, removal of any of the isolated nodes from the network would not affect the network or the surroundings of that node.

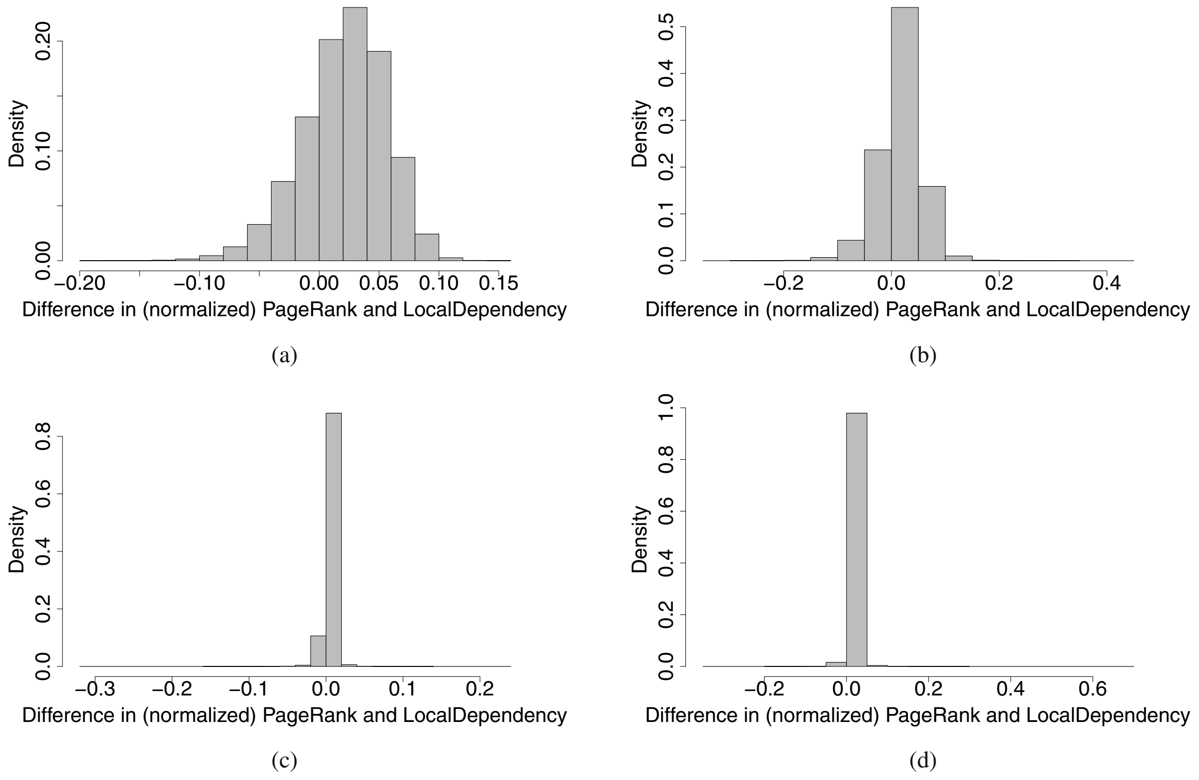

Fig. 6. Difference between PageRank and LocalDependency: 100k nodes (a), 1M nodes (b), DBLP unweighted (c) and DBLP reduced (d).

## 8. Conclusions

In our article we described the dependency relationship between two nodes of a network. We understand dependency as a local non-symmetrical property of a pair of nodes. We used this dependency measure when proposing a new degree centrality measure that is well suited for weighted networks. Dependency centrality is computationally similar to classical degree centrality; however, it is able to capture more precisely the importance of a node in a network. The advantage of both degree and dependency centralities is that knowledge of the entire graph structure is not required; nevertheless, dependency centrality naturally evaluates the local surroundings of a node. Thanks to this, dependency centrality is applicable in the analysis of large-scale weighted or unweighted complex networks.

Furthermore, we used the dependency measure for transformation of an originally unweighted network to a weighted network. In the resulting network we interpret the edges as a dependency of a pair of nodes on each other. We may understand dependency also as a binary property. In such a case it is possible to detect strongly dependent or independent subgroups in a network.

## Acknowledgment

This work was supported by the European Regional Development Fund within the IT4Innovations Centre of Excellence project (CZ.1.05/1.1.00/02.0070) and by SGS, VSB—Technical University of Ostrava, under the grant no. SP2014/110.

## References

Abdallah, S. (2011). Generalizing unweighted network measures to capture the focus in interactions, *Social Network Analysis and Mining* **1**(4): 255–269.

Bar-Yossef, Z. and Mashiach, L.-T. (2008). Local approximation of PageRank and reverse PageRank, *Proceedings of the 17th ACM Conference on Information and Knowledge Management, Napa Valley, CA, USA*, pp. 279–288.

Barabási, A.-L. and Frangos, J. (2002). *Linked: The New Science of Networks Science Of Networks*, Basic Books, New York, NY.

Barrat, A., Barthelemy, M., Pastor-Satorras, R. and Vespignani, A. (2004a). The architecture of complex weighted networks, *Proceedings of the National Academy of Sciences of the United States of America* **101**(11): 3747–3752.

Barrat, A., Barthélemy, M. and Vespignani, A. (2004b). Weighted evolving networks: coupling topology and weight dynamics, *Physical Review Letters* **92**(22): 228701.

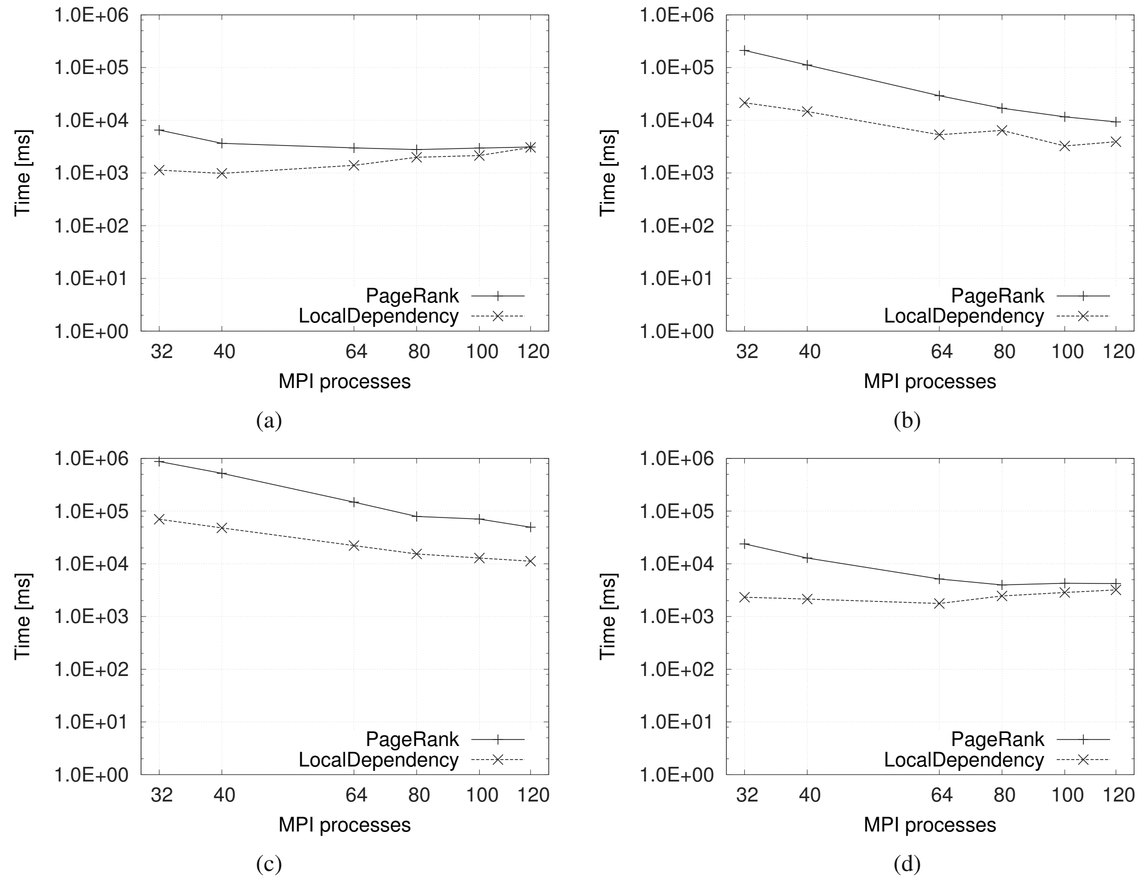

Fig. 7. Execution time for Anselm: 100k nodes (a), 1M nodes (b), DBLP unweighted (c) and DBLP reduced (d).

Brin, S. and Page, L. (1998). The anatomy of a large-scale hypertextual web search engine, *Proceedings of the 7th International Conference on World Wide Web, Brisbane, Australia*, pp. 107–117.

Christensen, D. (2005). Fast algorithms for the calculation of Kendall $\tau$, *Computational Statistics* **20**(1): 51–62.

Das Sarma, A., Molla, A., Pandurangan, G. and Upfal, E. (2013). Fast distributed PageRank computation, *in* D. Frey, M. Raynal, S. Sarkar, R. Shyamasundar and P. Sinha (Eds.), *Distributed Computing and Networking*, Lecture Notes in Computer Science, Vol. 7730, Springer, Berlin/Heidelberg, pp. 11–26.

de Jager, D. (2004). *PageRank: Three Distributed Algorithms*, Master's thesis, Imperial College London, London, `pubs.doc.ic.ac.uk/pagerank-algorithms/`.

Farkas, I., Ábel, D., Palla, G. and Vicsek, T. (2007). Weighted network modules, *New Journal of Physics* **9**(6): 180.

Fortunato, S. (2010). Community detection in graphs, *Physics Reports* **486**(3): 75–174.

Fortunato, S., Boguñá, M., Flammini, A. and Menczer, F. (2008). Approximating PageRank from in-degree, *in* W. Aiello, A. Broder, J. Janssen and E. Milios (Eds.), *Algorithms and Models for the Web-Graph*, Springer, Berlin/Heidelberg pp. 59–71.

Freeman, L.C. (1979). Centrality in social networks conceptual clarification, *Social Networks* **1**(3): 215–239.

Ghazalpour, A., Doss, S., Zhang, B., Wang, S., Plaisier, C., Castellanos, R., Brozell, A., Schadt, E.E., Drake, T.A., Lusis, A.J. and Horvath, S. (2006). Integrating genetic and network analysis to characterize genes related to mouse weight, *PLoS Genetics* **2**(8): e130.

Han, Y., Zhou, B., Pei, J. and Jia, Y. (2009). Understanding importance of collaborations in co-authorship networks: A supportiveness analysis approach, *Proceedings of the 9th SIAM International Conference on Data Mining, Sparks, NV, USA*, pp. 1111–1122.

Heckerman, D., Chickering, D. M., Meek, C., Rounthwaite, R. and Kadie, C. (2001). Dependency networks for inference, collaborative filtering, and data visualization, *The Journal of Machine Learning Research* **1**: 49–75.

Kahanda, I. and Neville, J. (2009). Using transactional information to predict link strength in online social networks, *Proceedings of the 3rd International Conference on Weblogs and Social Media (ICWSM), San Jose, CA, USA*, pp. 74–81.

Kenett, D.Y., Tumminello, M., Madi, A., Gur-Gershgoren, G., Mantegna, R.N. and Ben-Jacob, E. (2010). Dominating clasp of the financial sector revealed by partial correlation analysis of the stock market, *PLoS One* **5**(12): e15032.

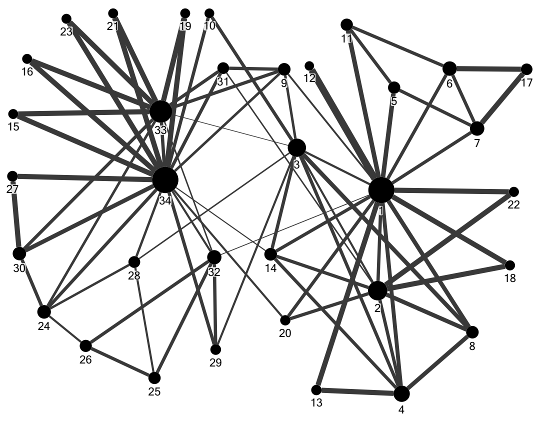

Fig. 8. Dependency weights.

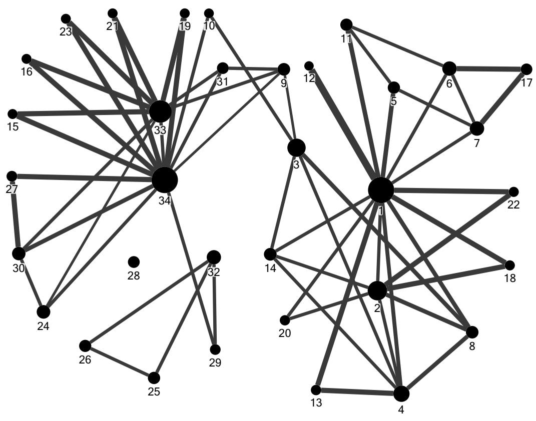

Fig. 9. Reduced network.

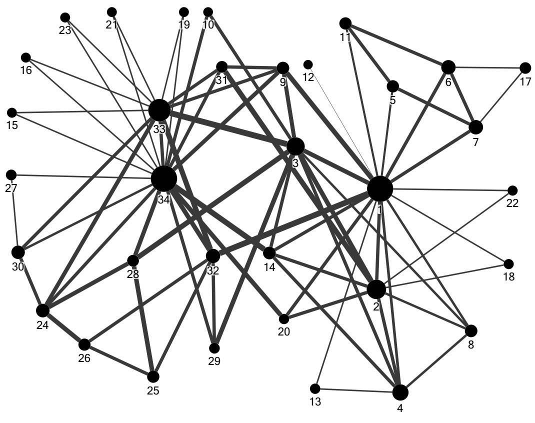

Fig. 10. Inverted dependency.

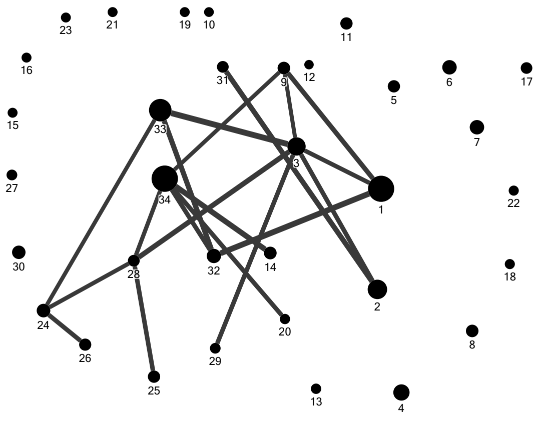

Fig. 11. Reduced network.

Kudělka, M., Horák, Z., Snášel, V., Krömer, P., Platoš, J. and Abraham, A. (2012). Social and swarm aspects of co-authorship network, *Logic Journal of IGPL* **20**(3): 634–643.

Langville, A.N. and Meyer, C.D. (2006). *Google's PageRank and Beyond: The Science of Search Engine Rankings*, Princeton University Press, Princeton, NJ.

Leenders, R.T.A. (2002). Modeling social influence through network autocorrelation: Constructing the weight matrix, *Social Networks* **24**(1): 21–47.

Lin, J. and Dyer, C. (2010). *Data-intensive Text Processing with MapReduce*, Synthesis Lectures on Human Language Technologies, Morgan & Claypool, San Rafael, CA.

Lusseau, D. (2003). The emergent properties of a dolphin social network, *Proceedings of the Royal Society of London, Series B: Biological Sciences* **270**(Suppl 2): S186–S188.

Manaskasemsak, B. and Rungsawang, A. (2004). Parallel PageRank computation on a gigabit PC cluster, *18th International Conference on Advanced Information Networking and Applications, AINA 2004, Fukuoka, Japan*, Vol. 1, pp. 273–277.

Manaskasemsak, B., Uthayopas, P. and Rungsawang, A. (2006). A mixed MPI-thread approach for parallel page ranking computation, *in* R. Meersman and Z. Tari (Eds.), *Proceedings of the 2006 Confederated International Conference on On the Move to Meaningful Internet Systems: CoopIS, DOA, GADA, and ODBASE, Part II*, Springer-Verlag, Berlin/Heidelberg, pp. 1223–1233.

Newman, M. (2008). The physics of networks, *Physics Today* **61**(11): 33–38.

Newman, M.E. (2004). Analysis of weighted networks, *Physical Review E* **70**(5): 056131.

Newman, M.E. (2006). Finding community structure in networks using the eigenvectors of matrices, *Physical Review E* **74**(3): 036104.

Nitzberg, B., Schopf, J. and Jones, J. (2004). PBS Pro: Grid computing and scheduling attributes, *in* J. Nabrzyski, J. Schopf and J. Węglarz (Eds.), *Grid Resource Management*, International Series in Operations Research and Management Science, Vol. 64, Springer, New York, NY, pp. 183–190.

Onnela, J.-P., Saramäki, J., Hyvönen, J., Szabó, G., De Menezes, M. A., Kaski, K., Barabási, A.-L. and Kertész, J. (2007).

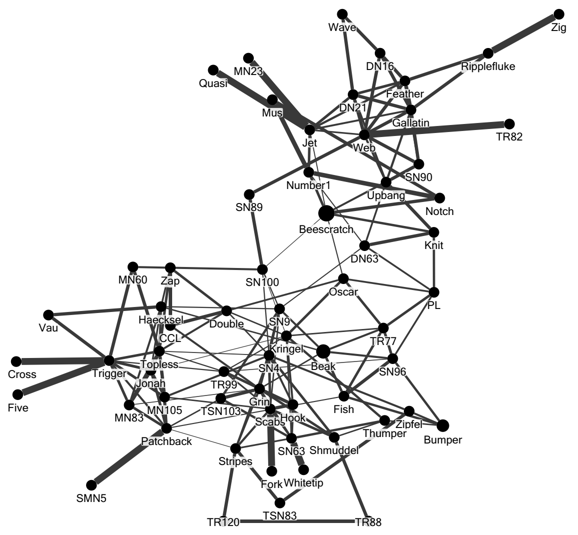

Fig. 12. Dependency weights.

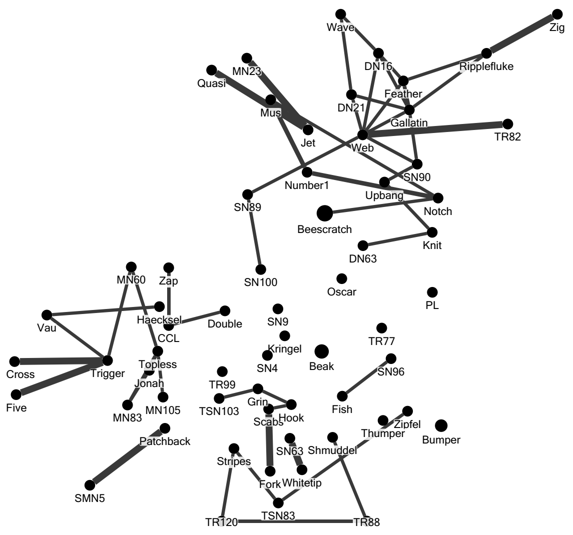

Fig. 13. Reduced network.

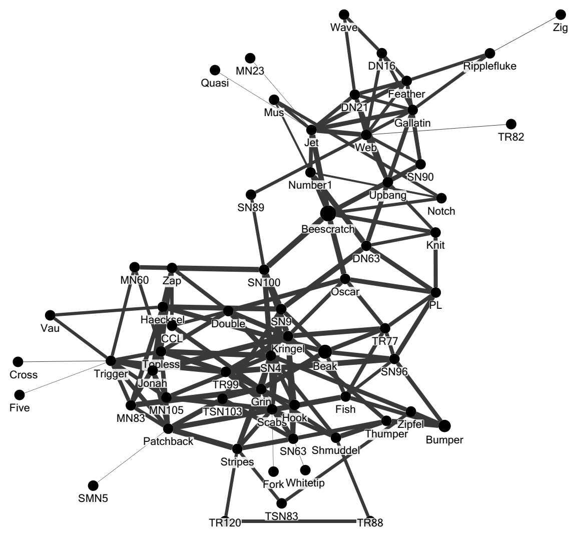

Fig. 14. Inverted dependency.

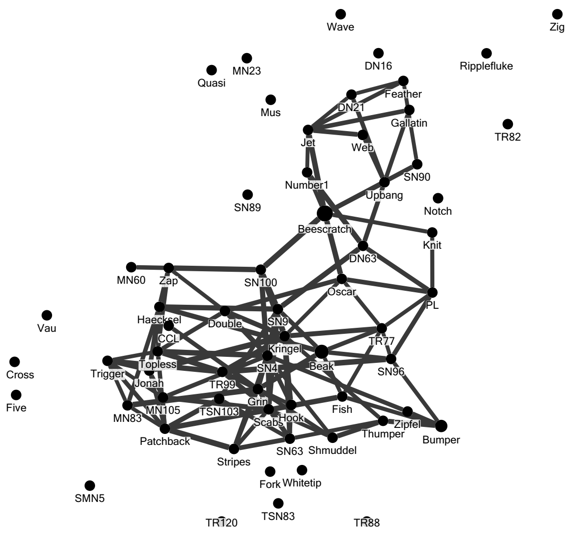

Fig. 15. Reduced network.

Analysis of a large-scale weighted network of one-to-one human communication, *New Journal of Physics* **9**(6): 179.

Opsahl, T., Agneessens, F. and Skvoretz, J. (2010). Node centrality in weighted networks: Generalizing degree and shortest paths, *Social Networks* **32**(3): 245–251.

Opsahl, T. and Panzarasa, P. (2009). Clustering in weighted networks, *Social Networks* **31**(2): 155–163.

Plimpton, S.J. and Devine, K.D. (2011). MapReduce in MPI for large-scale graph algorithms, *Parallel Computing* **37**(9): 610–632.

Rungsawang, A. and Manaskasemsak, B. (2003). PageRank computation using PC cluster, *in* J. Dongarra, D. Laforenza and S. Orlando (Eds.), *Recent Advances in Parallel Virtual Machine and Message Passing Interface*, Lecture Notes in Computer Science, Vol. 2840, Springer Berlin/Heidelberg, pp. 152–159.

Sankaralingam, K., Sethumadhavan, S. and Browne, J. (2003). Distributed PageRank for P2P systems, *12th IEEE International Symposium on High Performance Distributed Computing, 2003, Seattle, WA, USA*, pp. 58–68.

Wiedermann, M., Donges, J.F., Heitzig, J. and Kurths, J. (2013). Node-weighted interacting network measures improve the representation of real-world complex systems, *Europhysics Letters* **102**(2): 28007.

Witten, I.H., Gori, M. and Numerico, T. (2006). *Web Dragons: Inside the Myths of Search Engine Technology*, Morgan Kaufmann, San Francisco, CA.

Zachary, W.W. (1977). An information flow model for conflict and fission in small groups, *Journal of Anthropological Research* **33**(4): 452–473.

Zehnalova, S., Horak, Z., Kudelka, M. and Snael, V. (2013). Local dependency in networks, *5th International Conference on Intelligent Networking and Collaborative Systems (INCoS), Xi'an, China*, pp. 250–254.

Zhang, B. and Horvath, S. (2005). A general framework for weighted gene co-expression network analysis, *Statistical Applications in Genetics and Molecular Biology* **4**(1): 1128.

Zhu, Y., Ye, S. and Li, X. (2005). Distributed PageRank computation based on iterative aggregation-disaggregation methods, *Proceedings of the 14th ACM International Conference on Information and Knowledge Management, CIKM'05, Bremen, Germany*, pp. 578–585.

**Miloš Kudělka** works as an associate professor at the CS Department of VSB—Technical University of Ostrava and as a researcher at the IT4Innovations Center of Excellence at VSB. He received the Ph.D. degree in computer science and applied mathematics in 2009. His research and development experience includes over 20 years at the academia and industry. He works in a multi-disciplinary environment involving software engineering, data mining, and social network analysis. He is also interested in teaching methods in computer science education. He has authored/co-authored several refereed journal/conference papers and book chapters, published more than 60 papers, and served as a PC member of many international conferences.

**Šárka Zehnalová** received the M.Sc. degree in computer science from Palacky University in Olomouc, Czech Republic, in 2007. She is currently a Ph.D. student at the VSB Technical University of Ostrava and is a member of the project *SGS* as a young researcher. Her main research interests are social network analysis, information retrieval, data mining and knowledge discovery.

**Zdeněk Horák** received the M.Sc. degree in computer science from Palacky University in Olomouc, Faculty of Science, in 2006. In 2012 he finished his Ph.D. in computer science and applied mathematics at VSB—Technical University of Ostrava. His area of interest includes information retrieval, data mining, and mobile computing.

**Pavel Krömer** received the Master's degree in computer science from VSB—Technical University of Ostrava in 2006 and the Ph.D. in computer science and applied mathematics from the same institution in 2010. His area of interest includes computational intelligence, information retrieval, the World Wide Web, data mining, and parallel computing. He has focused on development of innovative techniques as well as deployment of intelligent optimization methods in different application domains. He joined the Department of Computer Science at the VSB Technical University of Ostrava, Czech Republic, as a full time research associate and assistant professor of data mining in 2010. Since 2011, he has worked as a junior researcher in the IT4Innovations Center of Excellence at VSB. Currently, he also holds a postdoc fellow position at the Department of Electrical and Computer Engineering of the University of Alberta, Edmonton, Canada.

**Václav Snášel** is a professor of computer science. He is the dean of the Faculty of Electrical Engineering and Computer Science and also works as a researcher at the IT4Innovations Center of Excellence at VSB. His research and development experience includes over 30 years in the industry and academia. He works in a multi-disciplinary environment involving artificial intelligence, social network, conceptual lattice, information retrieval, semantic web, knowledge management, data compression, machine intelligence, neural network, web intelligence, nature and bio-inspired computing, data mining, and applications to various real world problems. He has given more than 20 plenary lectures and conference tutorials in these areas. He has authored/co-authored several refereed journal/conference papers, books and book chapters, published more than 500 papers, and served as a PC member and an organizer of many international conferences.

Received: 4 February 2014
Revised: 8 August 2014