# Heuristical Top-$k$: Fast Estimation of Centralities in Complex Networks

Erwan Le Merrer[a], Nicolas Le Scouarnec[a], Gilles Trédan[b]

[a] *Technicolor, Rennes, France*
[b] *LAAS/CNRS, Toulouse, France*

**Abstract**

Centrality metrics have proven to be of a major interest when analyzing the structure of networks. Given modern-day network sizes, fast algorithms for estimating these metrics are needed. This paper proposes a computation framework (named Filter-Compute-Extract) that returns an estimate of the top-$k$ most important nodes in a given network. We show that considerable savings in computation time can be achieved by first filtering the input network based on correlations between cheap and more costly centrality metrics. Running the costly metric on the smaller resulting filtered network yields significant gains in computation time. We examine the complexity improvement due to this heuristic for classic centrality measures, as well as experimental results on well-studied public networks.

*Keywords:* Information Retrieval, Network Analysis, Centralities, Complex Networks.

## 1. Introduction

Centrality measures such as Pagerank [1], betweenness [2] and closeness [3] have attracted a lot of attention, mainly because of their capacity to capture the impact of the network structure on numerous problems such as congestion [4], network resilience [5], worm propagation [6], or distributed network monitoring [7].

As many centralities have a global scope (as opposed to local metrics such as degree-based centrality), their complexity is often superlinear in the size of the network. Therefore, their computation is often considered prohibitive for current network sizes. To circumvent this cost, researchers have sought approximation algorithms to save computation time. For example, in order

to approximate betweenness, one can replace the $n$ invocation of the SSSP algorithm by a sample of $k < n$ invocations [8]. Yet in practice, the efficiency of this approach remains limited since relatively large $k$ may be required for results to be accurate [8].

Meanwhile, the size of input networks keeps on increasing, reaching billions of nodes and edges. This calls for fast heuristics that allow practical analysis of current networks. Moreover, we observe that such analyses do not require the exact score of each node but aim at identifying the top nodes (*i.e.*, the most important nodes). For instance, such top nodes of a social network are priority targets for efficient marketing campaigns [9], defense against worm propagation [6], or recommended to users as items of interest for them [10].

In this work, we propose *Filter-Compute-Extract*, a general computation framework leveraging this observation. The core of this framework is a near linear heuristic that does not compute a score for all nodes but instead focuses on identification of the top nodes of the network. Such a research direction was raised but left unanswered since [3]. We first study whether the correlations between different centrality measures allow identifying the most important nodes of an expensive centrality by incurring only the price of a cheap centrality (Section 2). We then use the correlations observed to build a two pass approach that refines the approximation provided by a cheap centrality using the expensive centrality on a reduced graph (Section 3).

## 2. Correlation and emulation in top-$k$

In previous work [11], it has been reported that centralities correlate on some networks. More specifically, paper [12] has shown that Pagerank is strongly correlated to in-degree on Web link graphs. However, these correlations where observed by ranking all nodes in the graph. In this section, we depart from these studies by focusing on the top-$k$ nodes reported by the centrality metrics, and studying an increased variety of complex networks (work [13] specifically studied top "stable" nodes for which Pagerank and degree also strongly correlate in real networks). Furthermore, being qualitative studies, previous works have not emphasized the complexity gains that arise from restricting computation to only the top-$k$.

In the following, we consider undirected graphs of $n$ nodes and $m$ edges. The top-$k$ problem consists in finding the $k \ll n$ most important nodes with respect to a given centrality. We consider the four most widespread

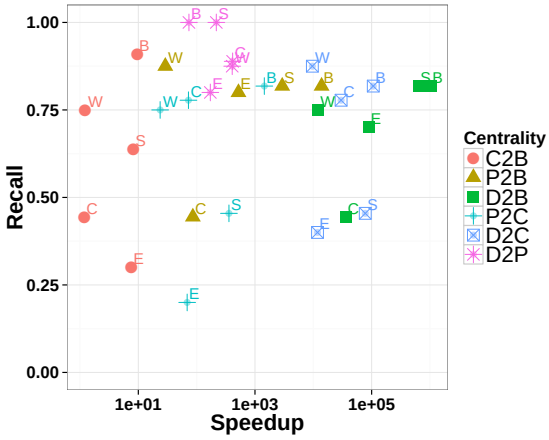


Figure 1: Emulation of a centrality by another one for top-log $n$: results in recall and speedup.

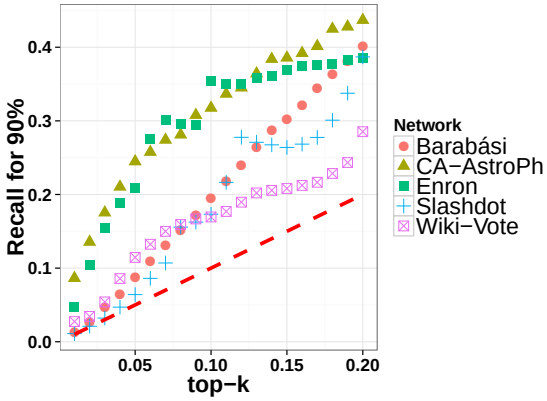


Figure 2: Amount of nodes needed to reach a $90th$ percentile in the emulation of betweenness by degree (D2B).

centralities: betweenness (noted B), closeness (C), Pagerank (P) and degree (D)[1]. Degree is the cheapest of these, with a cost linear in the graph size. Betweenness and closeness are the most expensive ones, with polynomial cost $\Theta(nm + n^2 \log n)$ [2, 3]. Pagerank [1] has a lower cost $\Theta(m \log(1/\epsilon))$ (with $\epsilon$ a parameter controlling convergence/cost); in experiments, then default NetworkX Pagerank implementation is used [14], with $\alpha = 0.7$ and no personalization.

Our core idea is, considering the computational cost of centrality metrics, to define an emulation framework in which a centrality measure with a low cost can be used in place of one with a higher cost, for top-$k$ extraction (knowing that in order of increasing cost, we have $D < P < C \approx B$). Our metric to measure the centrality emulation quality is recall: let $K$ be the $k$ most important nodes for a given centrality, and $K'$ the result obtained using emulation. The resulting recall of this emulation is $\frac{|K \cap K'|}{k}$.

We analyze well studied and publicly available graphs [15]: Slashdot,

[1]Closeness for a node $i$ is computed by averaging the distance between $i$ and all other nodes $v \in V$ in the graph: $C(i) = \frac{1}{\Sigma_{j \in V} d(i,j)}$, where $d(i,j)$ denotes the distance in hops (shortest-path) between nodes $i$ and $j$. Betweenness is computed as $B(i) = \Sigma_{j \neq i \neq k} \frac{\sigma_{jk}(i)}{\sigma_{jk}}$, with $\sigma_{jk}(i)$ the number of shortest-paths from node $j$ to node $k$ that passes on node $i$.

Wiki-vote, Enron, CA-AstroPh (with $n \approx 82k, 7k, 36k, 19k$ and $m \approx 500k$, $100k$, $367k$, $396k$ respectively) and a Barabási-Albert graph of $10k$ nodes and on average 6 edges per node. Note that we use relatively small graphs, as the algorithms we use as references for betweenness [2] and closeness [3] would not terminate in reasonable time on larger graphs, thus making exact comparison of our approach to those algorithms impossible.

The first experiment explores the recall/speedup trade-off that can be obtained by emulating a centrality by another one (*i.e.*, executing a cheap centrality in place of an expensive one, and observing the results). Figure 1 shows how centralities correlate on top-$k$ nodes, with $k$ chosen so that it grows slowly with network input size ($k = \log n$). On the $x$-axis, we plot the speedup in computation time measured on one Intel Xeon X5680 3.3 GHz core using the NetworkX graph package [14]. The recall is plotted on the $y$-axis. Letters S, W, E, C and B within the plot stand respectively for the Slashdot, Wiki-vote, Enron, CA-AstroPh and Barabási-Albert networks. Emulation results are compared on a pairwise fashion: C2B meaning that closeness is used to emulate betweenness; D and P then stand for degree and Pagerank.

For instance, the blue (D2C) point B at $(1e5, 0.8)$ means that on the Barabási-Albert graph, degree centrality provides a recall of 80% when emulating closeness and runs significantly faster ($1e5$ times). In other words, degree identifies 4 of the 5 most important nodes in less than a second whereas closeness took approximately 2 hours to identify all 5 top nodes. Most points are above 75% of recall, while considerably reducing the computation time (often more than $100\times$ except for C2B). More specifically, one can observe that degree emulates Pagerank with a good accuracy, and that using degree for betweenness results in high speedups ($\times 1e3$ to $\times 1e4$) for a reasonable accuracy (*i.e.,* a recall of 80%).

Next, we observe on Figure 2 the amount of nodes needed to obtain the 90th recall percentile, for the emulation of betweenness by degree. For instance, point $(0.2, 0.28)$ means that for a top-$k$ of size 20% of original network size, the first 28% top nodes according to degree centrality contains 90% of the 20% top nodes according to betweenness. The dash line represents a perfect emulation of a metric by a different one. The different curves represent the different networks we considered. We observe that twice the number of nodes is generally required to reach the percentile, with notably three curves with a good fit for low $k$ (Barabási-Albert, Wiki-vote and Slashdot), confirming results for top-$\log n$ on Figure 1.

Those strong correlations highlight a known property of many complex networks, namely a contrasted core/periphery structure [16] in which core (or elite) nodes play major structural roles that translate into top-scores for a broad range of centralities. In the next section, we propose to further leverage this core/periphery structure by designing a framework that $(i)$ identifies the core with a cheap centrality and $(ii)$ focuses the search for top-$k$ nodes on the core. For instance, degree centrality can be computed in near-linear time by partially sorting the nodes according to their degree, while metrics like closeness or betweenness are computed in superlinear time. We build on these observations to design the FCE heuristic in the next section.

## 3. Refining using the "elite"

### *3.1. Filter-Compute-Extract : a fast heuristic*

Since the top-$k$ nodes according to an expensive centrality are often, in practice, in the top nodes reported by other cheaper centralities, we propose to filter nodes using a cheap heuristic prior to performing the expensive centrality computation. More specifically, rather than performing the computation over the whole network, one can perform the computation over a sub-graph (*the core*) containing $n_c$ candidates nodes, the core being formed of $n_c$ elite nodes being selected by a cheap centrality that can emulate the expensive centrality (*e.g.*, $n_c = \Theta(\sqrt{m})$ elite nodes filtered using degree-centrality as proposed in [16]). Given that $k \ll n_c$, the $n_c$ nodes have a great chance of containing most of the top-$k$ nodes according to the targeted expensive metric (as presented in Figure 2). As a consequence, if $n_c \ll n$, this is a fast approximation heuristic with a good accuracy.

When degree centrality is used to extract the core, a partial sorting allows extracting the top-k nodes in an array with an average complexity of $\Theta(n + k \log k)$ [17]. This fast operation, compared to algorithms computing centralities, leverages the observed correlations to reduce computational costs. The FCE heuristic (for Filter-Compute-Extract) we propose consists of the following steps:

1. Filter the $\Theta(\sqrt{m})$ nodes belonging to the network core;
2. Compute centrality over the sub-graph induced by nodes belonging to the core;
3. Extract the top-$k$ nodes according to the centrality computed on the sub-graph.

As non-elite nodes are not meant to be part of the computation, they cannot get a result. Nevertheless, a straightforward optimization is to take these into account in the following way: assign to each elite node $i \in E \subset V$ a weight $w_i$ equal to the number of non-elite neighbors it has: $w_i = |\{(i,j) \in E \text{ s.t. } j \in V \backslash E\}|$. Since all centralities presented here have equivalent runtimes on weighted and non-weighted graphs, this optimization is cheap. Various additional optimizations may be tailored depending on the emulated centrality.

*3.2. Complexity improvement, a case study*

An interesting case is to approximate an expensive centrality (*e.g.*, betweenness), by applying a cheap filtering (*e.g.*, degree) at step 1, and by subsequently computing, on the filtered graph, the expensive centrality used at step 2. We now detail the gain in term of computational complexity.

**Step 1**. Given a graph of size $n$, we filter the $h$ nodes belonging to the core according to degree centrality using an efficient partial sorting [17] whose expected complexity is $\Theta(n + h \log h)$. Applied for the core extraction of size $h = \sqrt{m}$, this operation then requires $\Theta(n + \sqrt{m} \log \sqrt{m})$ time. If the graph is sparse, as most real life complex networks, complexity is $\Theta(n)$ (while $\Theta(n \log n)$ if dense).

**Step 2**. The second step applies the betweenness algorithm [2] over the $n_c = \sqrt{m}$ nodes of the core, which are connected by $m_c$ edges. The computational complexity is $\Theta(n_c\, m_c + n_c^2 \log n_c)$. As each core node is connected on average to a constant fraction of other core nodes [16] (*i.e.*, the core is dense), $m_c$ is upper and lower bounded by $(\sqrt{m})^2 = m$. The resulting complexity is $\Theta(m^{\frac{3}{2}} + m \log \sqrt{m}) = \Theta(n^{\frac{3}{2}})$ for sparse graphs (*i.e.*, graphs where $m = O(n)$).

**Step 3**. Similarly to Step 1, the extraction algorithm for selecting the top-$k$ value from the result of the previous step has a complexity of $\Theta(n_c + k \log k)$.

Given that $k \ll \sqrt{m}$, the heuristic computation time is dominated by Step 2, for a complexity of $\Theta(n^{\frac{3}{2}})$ for sparse graphs, which significantly improves the $\Theta(n^2)$ bound for classic betweenness on the same sparse input graph.

These results generalize for non-betweenness evaluation. Consider a generic X2Y emulation: emulate centrality $Y$ of cost $Y(n)$ using centrality $X$ of cost $X(n)$. The cost of this emulation is at step 1: $\Theta(X(n))$ plus sorting $\Theta(n + \sqrt{m} \log \sqrt{m})$, at step 2: $\Theta(Y(\sqrt{m}))$, and at step 3: $\Theta(\sqrt{m} + k \log k)$. This sums up to $\Theta(X(n) + n + \sqrt{m} \log \sqrt{m} + Y(\sqrt{m}) + \sqrt{m} + k \log k)$. Assume that $X(n) = \Theta(n^a)$ and $Y(n) = \Theta(n^b)$ are the respective complexities of $X$

| | Network / top-$k$ | 5 | log | sqrt |
|---|---|---|---|---|
| **D2B** | | | | |
| 1 | web-NotreDame$^{\dagger}$ | 0.800 (100.0%) | 0.583 (75.0%) | 0.281 (-16.7%) |
| 2 | com-dblp.ungraph$^{\dagger}$ | 0.600 (50.0%) | 0.667 (33.3%) | 0.586 (33.6%) |
| 3 | web-Stanford$^{\dagger}$ | 0.600 (0.0%) | 0.417 (25.0%) | 0.215 (15.2%) |
| 4 | Gowalla$^{\dagger}$ | 1.000 (0.0%) | 0.833 (0.0%) | 0.359 (5.3%) |
| 5 | soc-Epinions1 | 0.800 (-20.0%) | 0.909 (11.1%) | 0.756 (12.4%) |
| 6 | Slashdot | 1.000 (0.0%) | 0.909 (11.1%) | 0.853 (3.0%) |
| 7 | loc-brightkite | 0.800 (0.0%) | 0.900 (12.5%) | 0.809 (25.8%) |
| 8 | Enron | 0.400 (0.0%) | 0.700 (0.0%) | 0.759 (9.8%) |
| 9 | cit-HepTh | 0.400 (-33.3%) | 0.800 (-11.1%) | 0.578 (20.0%) |
| 10 | CA-AstroPh | 0.600 (200.0%) | 0.556 (25.0%) | 0.584 (14.3%) |
| 11 | Wiki-Vote | 0.800 (0.0%) | 0.750 (0.0%) | 0.833 (6.1%) |
| 12 | Barabási-Albert$^{\star}$ | 0.960 (11.0%) | 0.933 (0.3%) | 0.904 (1.8%) |
| 13 | *Erdős-Rényi*$^{\star}$ | 0.000 (-100.0%) | 0.000 (-100.0%) | 0.262 (-67.6%) |
| **D2P** | | | | |
| 14 | web-NotreDame | 0.200 (0.0%) | 0.417 (66.7%) | 0.646 (1.1%) |
| 15 | com-dblp.ungraph | 0.600 (50.0%) | 0.500 (20.0%) | 0.696 (41.5%) |
| 16 | web-Stanford | 0.400 (0.0%) | 0.500 (0.0%) | 0.638 (6.6%) |
| 17 | Gowalla | 1.000 (25.0%) | 0.917 (10.0%) | 0.883 (5.1%) |
| 18 | soc-Epinions1 | 0.800 (0.0%) | 0.818 (28.6%) | 0.818 (11.9%) |
| 19 | Slashdot | 0.800 (0.0%) | 0.909 (0.0%) | 0.850 (4.3%) |
| 20 | loc-brightkite | 0.800 (0.0%) | 0.900 (12.5%) | 0.830 (15.6%) |
| 21 | Enron | 0.600 (0.0%) | 0.800 (0.0%) | 0.838 (8.8%) |
| 22 | cit-HepTh | 0.800 (0.0%) | 0.900 (0.0%) | 0.783 (4.8%) |
| 23 | CA-AstroPh | 1.000 (25.0%) | 0.778 (16.7%) | 0.759 (13.0%) |
| 24 | Wiki-Vote | 0.800 (0.0%) | 0.750 (20.0%) | 0.810 (4.6%) |
| 25 | Barabási-Albert$^{\star}$ | 1.000 (0.0%) | 0.978 (0.0%) | 0.982 (0.2%) |
| 26 | *Erdős-Rényi*$^{\star}$ | 0.760 (-12.0%) | 0.800 (-10.0%) | 0.812 (-3.1%) |
| **P2B** | | | | |
| 27 | web-NotreDame$^{\dagger}$ | 0.800 (100.0%) | 0.667 (300.0%) | 0.475 (22.1%) |
| 28 | com-dblp.ungraph$^{\dagger}$ | 0.600 (0.0%) | 0.667 (33.3%) | 0.625 (11.7%) |
| 29 | web-Stanford$^{\dagger}$ | 0.600 (0.0%) | 0.500 (20.0%) | 0.274 (6.6%) |
| 30 | Gowalla$^{\dagger}$ | 1.000 (25.0%) | 0.833 (-9.1%) | 0.361 (0.6%) |
| 31 | soc-Epinions1 | 0.800 (0.0%) | 0.909 (-9.1%) | 0.793 (-3.1%) |
| 32 | Slashdot | 1.000 (25.0%) | 0.909 (11.1%) | 0.867 (-1.6%) |
| 33 | loc-brightkite | 0.800 (33.3%) | 0.900 (0.0%) | 0.817 (2.1%) |
| 34 | Enron | 0.400 (-33.3%) | 0.800 (0.0%) | 0.791 (-6.2%) |
| 35 | cit-HepTh | 0.600 (0.0%) | 0.800 (-11.1%) | 0.657 (2.8%) |
| 36 | CA-AstroPh | 0.600 (50.0%) | 0.667 (50.0%) | 0.650 (4.7%) |
| 37 | Wiki-Vote | 0.800 (0.0%) | 0.875 (0.0%) | 0.821 (-2.8%) |
| 38 | Barabási-Albert$^{\star}$ | 0.920 (6.0%) | 0.956 (6.3%) | 0.908 (3.7%) |
| 39 | *Erdős-Rényi*$^{\star}$ | 0.000 (-100.0%) | 0.044 (-94.3%) | 0.224 (-71.3%) |

Table 1: Recall results for FCE top-$k$ extraction, for three emulations: D2B, D2P and P2B. Percentages in brackets are the relative improvement provided by FCE over the filtering centrality from Step 1. Networks marked with $^{\dagger}$ are too big to allow exact betweenness in our setting, therefore baseline is computed using [8] with $\sqrt{n}$ samples (which may bias plotted results for FCE). Networks marked with $^{\star}$ are synthetic topologies for which results are averaged over five independent constructions.

and $Y$ algorithms such that $b > a \geq 1$, that the target graph is sparse, and that $k \ll \sqrt{m}$. Then the asymptotic cost of FCE is $\Theta(n^a + n^{b/2})$.

### *3.3. Evaluation of the* FCE *heuristic*

Having derived the asymptotic improvement on running time for FCE emulation, we now illustrate in practice the recall and running time of FCE for top-$k$. FCE is applied to the five previously considered networks, plus eight others, representing various network types, also publicly available in [15] (web-NotreDame, com-dblp.ungraph, web-Stanford, Gowalla, soc-Epinions1, loc-brightkite, cit-HepTh, Erdős-Rényi, with respectively $325k$, $317k$, $281k$, $196k$, $75k$, $58k$, $27k$, $10k$ nodes). We use degree-centrality for the filtering in the D2B and D2P emulations, and Pagerank filtering for the P2B emulation. We then apply the optimization described previously to take into account discarded neighbors. We finally run betweenness centrality (D2B and P2B) or Pagerank (D2P) in the second step.

The considered elite size is $3 \times \sqrt{m}$ nodes for each network. We compared the resulting recall according to the exact emulated centrality computation. Results for top-5, top-$\log n$ and top-$\sqrt{n}$ [2] are depicted on Table 1, along with recall of the filtering centrality (*i.e.* the Step 1 centrality), used as a baseline.

Recall is over 75% for the same three networks that have provided the best results in the basic emulation of betweenness by degree (D2B), as seen on Figure 2. FCE results are poor in practice on Erdős-Rényi random graphs ($p = 0.001$) since the filtered nodes do not constitute a connected and representative elite, as noted in paper [16]. These results underline the necessary condition for FCE to provide satisfying results: the existence of a core in the network. Enron, CA-AstroPh, and the two web-graph datasets provide relatively less accurate results. This can be explained by their structural differences: both datasets do not exhibit as tightly connected core as defined in paper [16]; for instance Enron instead exhibits a hierarchical structure with no hub effect (*i.e.*, no power law distribution [18]). No core is then fully extracted from Step 1 of FCE by considering degrees. Step 1 should therefore be adapted for these outlier datasets by using a filter that better fits their structure. Despite some rare degradations of the recall over the recall produced by the filtering centrality (negative percentage in brackets)

[2] Top-$\sqrt{n}$ returns a great part of the nodes selected at Step 1 ($3 \times \sqrt{m}$), demonstrating the potential correlation of the large part of the network elite nodes with regards to the two centralities used jointly.

-highlighting that FCE is a heuristic-, most networks show satisfying emulation results, with consistent improvement over the filtering centrality. D2P and P2B result in particularly good results for most of the tested networks.

Those results, consistently comprised from 50 to 100% recall, are produced much faster than the exact computation. For instance for D2B, $\times 10$ to $\times 1e4$ faster (in details, Enron: 74s vs. 5h30s, Slashdot: 337s vs. 2 days, CA-AstroPh: 69s vs. 1h32s, Wiki-Vote: 47s vs. 738s). Those results thus clearly motivates the use of such a framework and heuristic for a fast estimation of centralities, in scenarios where exact computation is not required.

## 4. Conclusion

We have proposed a general framework for fast computation of top-$k$ centralities, based on the observation that restricting computation solely to the core of networks gives good accuracy in practice, and significantly reduces computation times both theoretically and in practice. We expect this work to trigger discussion on the relaxed problem of ranking only top nodes, for it may yield very significant computational gains for the analysis of giant networks, or for the continuous analysis of fast changing topologies.

## Acknowledgement

We thank the anonymous reviewers and the editor Dr. Jef Wijsen for their valuable comments which helped us to improve the manuscript.

## References

[1] M. Bianchini, M. Gori, F. Scarselli, Inside PageRank, ACM Tr. Internet Tech. 5 (1) (2005) 92–128.

[2] U. Brandes, A Faster Algorithm for Betweenness Centrality, J. of Math. Sociology 25 (2001) 163–177.

[3] K. Okamoto, W. Chen, X.-Y. Li, Ranking of Closeness Centrality for Large-Scale Social Networks, in: FAW, 2008.

[4] S. Dolev, Y. Elovici, R. Puzis, Routing betweenness centrality, J. ACM 57 (4) (2010) 25:1–25:27.

[5] J. Sterbenz, E. etinkaya, M. Hameed, A. Jabbar, S. Qian, J. Rohrer, Evaluation of network resilience, survivability, and disruption tolerance: analysis, topology generation, simulation, and experimentation, Telecommunication Systems (2011) 1–32.

[6] S. Wen, W. Zhou, Y. Wang, W. Zhou, Y. Xiang, Locating Defense Positions for Thwarting the Propagation of Topological Worms, Communications Letters, IEEE 16 (4) (2012) 560 –563.

[7] S. Dolev, Y. Elovici, R. Puzis, P. Zilberman, Incremental deployment of network monitors based on group betweenness centrality, Inf. Process. Lett. 109 (20) (2009) 1172–1176.

[8] U. Brandes, C. Pich, Centrality estimation in large networks, in: Int. J. of bifurcation and chaos, 2007.

[9] C. Kiss, M. Bichler, Identification of influencers - Measuring influence in customer networks, Decision Support Systems 46 (1) (2008) 233 – 253.

[10] B. Bahmani, A. Chowdhury, A. Goel, Fast incremental and personalized pagerank, in: VLDB, 2010.

[11] T. W. Valente, K. Coronges, C. Lakon, E. Costenbader, How Correlated Are Network Centrality Measures?, Connect (Tor) 28(1) (2008) 16–26.

[12] S. Fortunato, M. Bogu, A. Flammini, F. Menczer, Approximating pagerank from in-degree, in: W. Aiello, A. Broder, J. Janssen, E. Milios (Eds.), Algorithms and Models for the Web-Graph, Vol. 4936 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2008, pp. 59–71.

[13] G. Ghoshal, A.-L. Barabási, Ranking stability and super-stable nodes in complex networks, Nature Communications 2 (2011) 1–7.

[14] NetworkX, `http://networkx.github.io/`.

[15] Stanford Network Dataset Collection, `http://snap.stanford.edu/data/`.

[16] C. Avin, Z. Lotker, Y. A. Pignolet, On The Elite of Social Networks, CoRR abs/1111.3374.

[17] C. Martínez, Partial Quicksort, in: ANALCO, 2004.

[18] A. Chapanond, M. S. Krishnamoorthy, B. Yener, Graph Theoretic and Spectral Analysis of Enron Email Data, Comput. Math. Organ. Theory 11 (3) (2005) 265–281.