# Higher order graph centrality measures for Neo4j

Georgios Drakopoulos*, Aikaterini Baroutiadi†, and Vasileios Megalooikonomou*
* *Multidimensional Data Analysis and Knowledge Management (MDAKM) Lab*
*Computer Engineering and Informatics Department*
*University of Patras*
{*drakop, vasilis*}*@ceid.upatras.gr*
† *Informatics of Life Sciences Graduate Programme*
*University of Patras*
*kbaroutiadi@upatras.gr*

***Abstract*—Graphs are currently the epicenter of intense research as they lay the theoretical groundwork in diverse fields ranging from combinatorial optimization to computational neuroscience. Vertex centrality plays a crucial role in graph mining as it ranks them according to their contribution to overall graph communication. Specifically, within the social network analysis context centrality identifies influential indivduals, whereas in the bioinformatics field centrality locates dominant proteins in protein-to-protein interaction. In recent years graph databases, part of the rising NoSQL movement, have been added to the graph analysis toolset. An implementation of eigenvector centrality, a prominent member of the broad class of spectral centrality, in Java and NetBeans designed for use with Neo4j, a major schemaless graph database, is outlined and the findings resulting from its application to a real world social graph are discussed.**

***Keywords*-Graph databases; Graph mining; Social network analysis; Centrality metrics; High order data; Neo4j; NetBeans**

## I. Introduction

A connected age is starting. The driving factors behind this trend are many. Graphs form the algorithmic cornerstone of several recently developed fields including, among others, the semantic web and ontology [23][2], machine learning and artificial intelligence [13], big data [4] and distributed computing [14].

Social network analysis also relies heavily on graphs, as their analysis allows conclusions to be reached to regarding social coherence [10], expansion [15], or information flow [22]. Among the foundational concepts of social network analysis is that of centrality, which aims to discover influential individuals in the network according to certain prespecified criteria. In turn, those criteria depend on features of the graph representing the social network.

Graph databases such as Neo4j, a prominent member of the NoSQL database movement, are often the preferred front- or back-end media where actual information regarding social graphs is stored to. Moreover, not only do graph databases provide a natural means for graph storage, but they also offer graph analytics such as link prediction and minimum spanning trees [21][19].

Corresponding author

The main contribution of this work is the implementation of eigenvalue centrality, an established vertex centrality metric based on the spectral characteristics of the adjacency matrix associated with a graph, in Java using NetBeans for Neo4j and the validity of this approach is confirmed by comparing the results with those obtained by MATLAB. Moreover, a brief overview of the rising trend of graph databases and their properties is given. A third contribution is the synopsis of two prominent subclasses of spectral vertex centrality metrics, namely the power series subclass and the eigenvector subclass. Finally, there are two theoretical notes regarding graphs. The first pertains to the inherent high order nature of graphs in terms of their information content and the second involves the important role of the adjacency matrix, which allows their representation and handling through linear algebra in addition to combinatorics.

Table I contains the notation used throughout this paper, regarding mostly vectors and matrices. It should be noted that $K_n$ refers to undirected graphs. The rest of this paper is structured as follows. Section II summarizes the brief history of graph databases and social network analysis. An overview of the main characteristics of graph databases is provided in section III. The high order nature of graphs and its relation to vertex centrality metrics, including the eigenvalue metric, are outlined in section IV. Section V describes certain implementation aspects and the results of applying eigenvector centrality to established datasets. Finally this work is concluded in section VI where its main points are recapitulated and future research directions are explored.

## II. Related Work

Sociologists at least as early as the 1950s at least had also similar interests in social network analysis as outlined in the semimal work of Katz [8] and others. It was not however before the advent of computer networks and their widespread adoption on behalf of the general population that allowed the study of massive and reliable data collected from daily, ordinary interaction between users in social media.

| Symbol | Meaning |
|---|---|
| $\triangleq$ | Equality by definition |
| $\mathbf{I}_n$ | Identity $n \times n$ matrix |
| $\{x_k\}$ | Set comprised by $x_k$ |
| $\langle \underline{\mathbf{g}} \rangle$ | Sequence of vectors $\underline{\mathbf{g}}$ |
| $\underline{\mathbf{g}}^{[k]}$ | $k$-th vector of an iterative process |
| $\underline{\mathbf{g}}[k]$ | $k$-th component of vector $\underline{\mathbf{g}}$ |
| $\underline{\mathbf{g}}_1 \odot \underline{\mathbf{g}}_2$ | Elementwise multiplication of $\underline{\mathbf{g}}_1$ and $\underline{\mathbf{g}}_2$ |
| $\sqrt[\circ]{\underline{\mathbf{g}}}$ | Elementwise square root of vector components |
| $K_n$ | Complete graph with $n$ vertices and $\binom{n}{2}$ edges |

Table I
SYMBOLS USED IN THIS PAPER.

Even more, it has been gradually realized that the Web is evolving into a mirror of our own society.

This sparkled an interest in social network analysis from both sociologists and computer scientists alike. The latter could build upon their own experience derived from earlier research on fundamental graph problems such as persistence [9] and lazy evalutation in functional progamming languages such as Haskel [6] among others. Moreover, Thus, onwards since circa 2008 there has been rising and intese activity in this crossdisciplinary area which has already bore fruit in a multitude of forms including outling large graph properties [12], graph modeling and synthetic graph generation [11], opinion mining and trolling detection [22], as well as information diffusion and dissemination [10].

## III. Graph Databases

This section is dedicated to a brief overview of NoSQL and graph databases in general in subsection III-A and of Neo4j in particular in subsection III-B.

### A. Beyond or Along Relational Databases?

Relational database management systems (RDBMS) or simply relational databases are based on the pioneering work of Codd [5] and today are operational mainstays in a typical business environment. However, the advent of modern Web with the almost continuous, unstructured or semistructured, and high order data has highlighted the need for new approaches in the database world. The recent movement of NoSQL databases aspires to fill this gap. It has four primary independent technologies, namely [21]

- Graph databases. Data is stored in property graphs, allowing interconnections between features to be represented in a natural way.
- Key-value databases. Data is stored in associative arrays, allowing the construction of complex maps between sets of objects.
- Document databases. Data is stored in JSON documents, allowing large document collections to be handled efficiently.
- Column family databases. Data is stored in tables with very long rows which are groupped into column families, allowing efficient row operations.

The above NoSQL branches all share the schemaless property, a major departure from the relational world. Moreover, the four ACID requirements (**A**tomicity, **C**onsistency, **I**solation, **D**urability) have been replaced by the three BASE requirements [21]:

- **B**asic **A**vailability. The database is operational most of the time. The percentage of downtime depends on the local operational requirements.
- **S**oft state. The database does not have to be write consistent. Also replicas do not have to be mutually consistent sometimes.
- **E**ventual consistency. The database does become consistent at later point.

Notice that NoSQL databases like every software tool are not panacea. Therefore, although originally thought as a replacement for relational databases (”No SQL”), they are expected to coexist with the latter, at least in the foreseeable future (”Not only SQL”).

### B. Property Graph Model and Query Languages

The property graph model is the primary conceptual data model supported by Neo4j and offered to a high level user typically through the Cypher querying language [21][19]. Its main characteristics are:

- Graph comprises of vertices, edges, and properties.
- Vertices represent objects.
- Vertices contain properties in key-value pairs.
- Edges represent connections.
- Edges have directions and labels.
- Edges have a start and an end vertex.
- Edges contain properties in key-value pairs.

Notice that these characteristics can be extended in a number of ways. For instance, an edge can be allowed to connect a vertex to two or more vertices resulting in a property multigraph. The latter can serve as a data model in situations where relationships can be clustered or when a higher level but lower resolution view of the original property graph is desired. This paper addresses simple property graphs only.

Neo4j supports a querying language named Cypher operating at the highest conceptual database level. Moreover, it supports three programmatic user-facing APIs. From higher to lower abstraction level these are the traverser, the core, and the kernel API. The proposed approach of this paper was based on the core API, which is fully integrated into the Neo4j NetBeans extension library.

## IV. Spectral Centrality

This section focuses on an area of vertex centrality termed spectral centrality because the associated algorithms rely on the spectral properties of the graph adjacency matrix.

Specifically, subsection IV-A discusses the need for high order vertex centrality metrics, subsection IV-B outlines a popular scheme based on adjacency matrix power series, based indirectly on the adjacency matrix eigenpairs, and subsection IV-C describes the eigenvector centrality method which depends directly on the adjacency matrix primary eigenvector. Throughout this section graphs are assumed to be simple, unweighted, and directed.

### *A. A High Order Metric*

A basic concept in social network analysis is vertex centrality. As its name suggests, it identifies influential individuals in a particular network in two distinct yet parallel senses. From a structural viewpoint the centrality of a given vertex $v_k$ quantifies its contribution to graph connectivity. Alternatively, from a spectral perspective centrality is a measure of information that $v_k$ reveals about the graph.

Literature abounds with vertex centrality metrics. Historically among the first ones was Katz centrality model [8]. It is worth mentioning that most known successful centrality metrics, either structural or spectral, rely heavily on the connectivity properties not only of the vertices under consideration but also on the connectivity properties of their neighbors at various distances. The families of algorithms in subsequent subsections illustrate this point.

A direct consequence is that graphs contain information of inherently high order, in terms of edges that need to be crossed, partly at least due to their distributed and connection-oriented nature. Therefore, graph processing should be of analogous nature, if useful information is to be extracted from a graph.

A manifestation of higher order graph nature in the particular yet significant case of vertex centrality is that degree centrality, a first order metric, identifies only some of the important vertices and its success depends on the topology of the graph under consideration [16][3]. On the contrary, the number of closed triangles and the number of open ones a given vertex participates to, both third order metrics, combine to yield clustering coefficient, a reliable centrality indicator [17][26]. For undirected graphs, a closed triangle is one where all three edges exist between its three vertices, in other words an instance of $K_3$, whereas in an open triangle any of these three edges is absent. These notions generalize in more than one ways for directed graphs. From the viewpoint of higher order processing, the above make perfect senese.

### *B. Power Series Centrality*

A broad class of vertex centrality metrics which take into account the higher order nature employ the fact that by its very definition the adjacency matrix is square, hinting to the possibility of building matrix power series [7]. Computationally, if $\mathbf{A}$ denotes the adjacency matrix of a any given graph, then the first step is to select a member of the power series family

$$\gamma_0\mathbf{I}+\gamma_1\mathbf{A}+\gamma_2\mathbf{A}^2+\gamma_3\mathbf{A}^3+\ldots = \sum_{k=0}^{+\infty}\gamma_k\mathbf{A}^k = f(\mathbf{A}) \quad (1)$$

Then the centrality of $v_k$ is defined as the $k$-th diagonal element of $f(\mathbf{A})$. The rationale behind this approach is that the $k$-th diagonal element of $\mathbf{A}^\ell$ is the number of closed paths of length $\ell$ originating from $v_k$. In general $\mathbf{A}^\ell[i,j]$ contains the number of paths of length $\ell$ from $v_i$ to $v_j$. Thus progressive powers of $\mathbf{A}$ codify concisely the number of paths of various lengths without having actually to traverse these paths in the actual stored graph. Although matrix multiplication is by no means a cheap operation especially for large graphs, $\mathbf{A}$ can be and most often is sparse and this sparsity is to be taken advantage of [12].

Obviously, different choices of coefficients $\{\gamma_k\}$ in equation (1) essentially assign different weights to yield a different centrality measure. However, notice that the convergence of 1 to a matrix function does not depend on $\{\gamma_k\}$ alone but also on the eigenpairs of $\mathbf{A}$. To see why, let us consider the case where the geometric and the algebraic multiplicity of each of the eigenvalues of $\mathbf{A}$ coincide. Then, the spectral eigendecomposition of $\mathbf{A}$ is

$$\mathbf{A} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^H \quad (2)$$

where matrix $\mathbf{U}$ is unitary and contains the eigenvectors of $\mathbf{A}$ while $\mathbf{\Lambda}$ is diagonal containing the corresponding and possibly distinct eigenvalues of $\mathbf{A}$. It is easy to verify, assuming function $f$ is analytic, that

$$f(\mathbf{A}) = \mathbf{U}\, f(\mathbf{\Lambda})\,\mathbf{U}^H \quad (3)$$

where $f(\mathbf{\Lambda})$ denotes that function $f$ is now applied to each of the eigenvalues of $\mathbf{A}$ separately resulting in a new diagonal matrix containing the transformed spectrum.

The role of spectrum to series convergence is illustrated in the Neumann metric defined as

$$\mathbf{W}_n \stackrel{\triangle}{=} \mathbf{I}+\mathbf{A}+\mathbf{A}^2+\ldots = \sum_{k=0}^{+\infty}\mathbf{A}^k = (\mathbf{I}-\mathbf{A})^{-1} \quad (4)$$

since (4) converges if and only if the spectral radius of $\mathbf{A}$ is strictly less than one. Note that this is a strict requirement placed on $\mathbf{A}$.

On the other hand, the Estrada index [7] is based on the series

$$\mathbf{W}_e \stackrel{\triangle}{=} \mathbf{I}+\mathbf{A}+\frac{1}{2!}\mathbf{A}^2+\frac{1}{3!}\mathbf{A}^3+\ldots = \sum_{k=0}^{+\infty}\frac{1}{k!}\mathbf{A}^k = e^{\mathbf{A}} \quad (5)$$

which exists for every $\mathbf{A}$. $e^{\mathbf{A}}$ denotes the matrix exponential of $\mathbf{A}$. In this metric longer paths contribute little to vertex centrality because of the inverse factorial weight. In practice $e^{\mathbf{A}}$ is not computed through (5) as power series are expensive and rather inaccurate in this sort of computations [18]. A viable approach for matrix exponentials relies on the property

$$e^{\mathbf{A}} = \mathcal{L}^{-1}\left[(s\mathbf{I}-\mathbf{A})^{-1}\right] \tag{6}$$

where $\mathcal{L}^{-1}[\cdot]$ denotes the inverse Laplace transform.

In a similar manner, the Estrada centrality can be modified in order to exclude the effect of even sized closed paths yielding

$$\mathbf{W}_s \triangleq \mathbf{A}+\frac{1}{3}\mathbf{A}^3+\ldots = \sum_{k=0}^{+\infty}\left(\frac{1}{2k+1}\right)\mathbf{A}^{2k+1} = \sinh(\mathbf{A}) \tag{7}$$

which is known as the odd Estrada centrality. The grounds for ignoring even lengths are that they mostly contain multiple crossings of a single edge. Similar to series (5), (7) always converges to matrix hyperbolic sinus.

The Mercator centrality moves a step further and assigns negative weights to the paths of even length, leading to

$$\mathbf{W}_m \triangleq \mathbf{A}-\frac{1}{2}\mathbf{A}^2+\frac{1}{3}\mathbf{A}^3-\ldots = \sum_{k=1}^{+\infty}\frac{(-1)^k}{k}\mathbf{A}^k = \ln(\mathbf{A}+\mathbf{I}) \tag{8}$$

### C. Eigenvector and Gell-point Centrality

Another common vertex centrality metric whose success highlights the high order nature of graphs is eigenvector centrality. As its name suggests, it is based on the computation of the adjacency matrix primary eigenvector $\underline{\mathbf{g}}_1$, namely the eigenvector corresponding to the largest eigenvalue. Then, $\underline{\mathbf{g}}_1[k]$ is assigned as a centrality indicator to vertex $v_k$ and sorting $\underline{\mathbf{g}}_1$ in descending order provides a vertex ranking [15]. It is of interest to notice that, according to empirical studies [12][4], the components of $\underline{\mathbf{g}}_1$ follow a Zipf law.

Power method is a straightforward way of generating a sequence $\langle \underline{\mathbf{g}}^{[k]} \rangle$ whose limit under mild conditions is the principal eigenvector $\underline{\mathbf{g}}_1$ of $\mathbf{A}$. As algorithm 1 illustrates, it is an iterative method requiring only the matrix-vector multiplication $\mathbf{A}\underline{\mathbf{g}}_1$ at step 4 in each iteration. Therefore, it belongs to the class of matrix free algorithms, meaning that the explicit matrix-vector product can be replaced by a function computing it indirectly. This is attractive both from computational and memory view as matrices representing social graphs are sparse [10]. Notice that the distribution used in step 1 of algorithm (1) is irrelevant as eventually the vector sequence $\langle \underline{\mathbf{g}}^{[k]} \rangle$ converges to $\underline{\mathbf{g}}_1$, provided that $\underline{\mathbf{g}}^{[0]}$ contains a component of $\underline{\mathbf{g}}_1$. Norm selection is also a matter of choice since $\|\cdot\|_1$, $\|\cdot\|_2$, and $\|\cdot\|_\infty$ are equivalent in $\mathbb{R}^n$ in the sense that for any vector $\underline{\mathbf{w}} \in \mathbb{R}^n$ and for any dimension $n$ there are fixed constants $\eta_0$ and $\eta_1$ independent of $\underline{\mathbf{w}}$ such that

$$\eta_0 \|\underline{\mathbf{w}}\|_X \leq \|\underline{\mathbf{w}}\|_Y \leq \eta_1 \|\underline{\mathbf{w}}\|_X \quad X,Y \in \{1,2,\infty\} \tag{9}$$

Gell-point centrality can be thought of as a concept analogous to eigenvalue centrality for directed graphs. Indeed, it is based in a simple manner on both the primary $\underline{\mathbf{g}}_1$ and the left primary $\underline{\mathbf{q}}_1$ eigenvectors of $\mathbf{A}$ [25], ie the primary eigenvectors of $\mathbf{A}$ and $\mathbf{A}^T$. Contrary to symmetric adjacency matrices, a nonsymmetric one in the general case has complex spectrum and the left eigenvectors do not coincide with their right counterparts.

**Algorithm 1** Power method

```
Require: Valid adjacency matrix $\mathbf{A}$.
Require: Positive threshold $\tau_0$.
Ensure: $\underline{\mathbf{g}}^{[k]}$ is a principal eigenvector approximation.
1: Initialize $\underline{\mathbf{g}}^{[0]}$ with random values.
2: $k \leftarrow 1$
3: repeat
4:     $\underline{\mathbf{g}}^{[k]} \leftarrow \mathbf{A}\underline{\mathbf{g}}^{[k-1]} / \|\mathbf{A}\underline{\mathbf{g}}^{[k-1]}\|_1$
5:     $k \leftarrow k+1$
6: until $\|\underline{\mathbf{g}}^{[k]} - \underline{\mathbf{g}}^{[k-1]}\|_1 \leq \tau_0$
7: return $\underline{\mathbf{g}}^{[k]}$
```

Once $\underline{\mathbf{g}}_1$ and $\underline{\mathbf{q}}_1$ are computed, the elementwise product

$$\underline{\mathbf{w}} = \sqrt[\odot]{\underline{\mathbf{g}}_1 \odot \underline{\mathbf{q}}_1} \tag{10}$$

is formed and in a similar manner to eigenvalue centrality $\underline{\mathbf{w}}[k]$ is used as the score of $v_k$. The square root is also applied elementwise. Ranking then is a simple issue of ordering scores in descending order. According to Perron-Frobenius theorem, since $\mathbf{A}$ is a nonnegative matrix, $\underline{\mathbf{g}}_1$ and $\underline{\mathbf{q}}_1$ can be nonnegative too.

## V. Experiments

### A. Datasets

Table II summarizes some basic information regarding the dataset used in this paper, namely the anonymized Facebook egonet collection from [24]. It is undirected, so that the primary eigenvector can be used as a centrality indicator, and of average size, three orders of magnitude smaller compared to real, large scale graphs. In table II density is the ratio of edges to vertices, an approximation to the average degree, and completeness is the ratio of the edges to the number of edges of $K_n$, where $n$ is the number of vertices.

| Feature | Yeast |
|---|---|
| Type | Undirected |
| Vertices | 4039 |
| Edges | 88234 |
| Density | 21.3127 |
| Completeness | 0.0108 |
| Triangles | 1612010 |

Table II
Overview of the dataset.

## B. Results

Figure 1 shows the scree plot of the primary eigenvector components, groupped in ten bins. This eigenvector has been computed entirely in Java and it has been compared to the results obtained by the MATLAB function `eig`. The vertex rankings were the same and this demonstrates the capability of integrating at least some graph analytics at the application level.

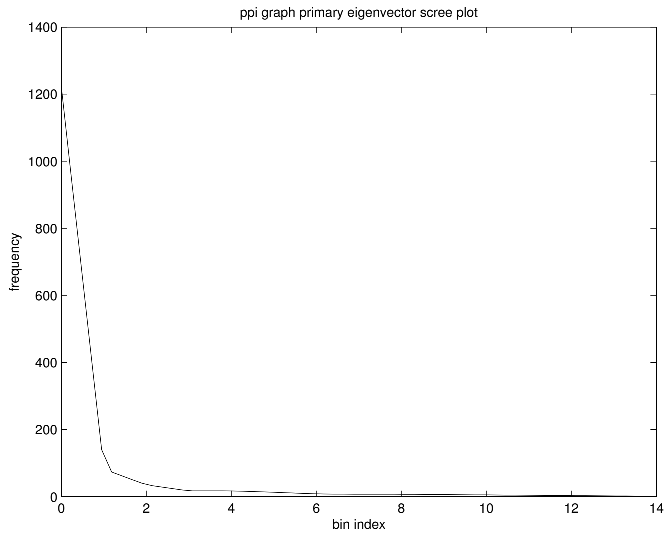

Figure 1. Scree plot

In a similar manner, figure 2 shows the logarithmic scree plot of primary eigenvector components as well as the ideal line connecting the ends of the log scree plot. The rationale behind that is that in log scale a power law becomes a linear relationship.

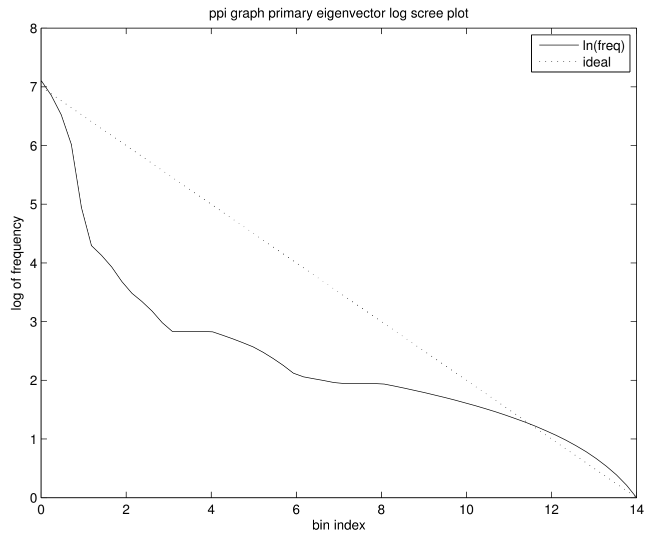

Figure 2. Logarithmic scree plot

The *scree plot* of a countable set $S$ is the plot of the frequency of the various elements of $S$ verus their rank. In other words, a scree plot is generated by identifying the unique elemets of $S$, computing their frequency of appearance, sorting these frequencies, and plotting the result. In a log scree plot, the logarithms of the sorted frequences appear instead. By its very definition, a scree plot is decreasing. However, in social network analysis and in graph mining in general, the way the scree plot and the log scree plot decrease is important. It has been empirically shown that for large, scale-free graphs scree plot follows a power law and, therefore, log scree plot is linear.

Figures 1 and 2 have been generated by groupping the elements of the primary eigenvector for the given dataset using the power method. There were $1430$ eigenvector components in total groupped in $16$ bins. A relatively small number of bins has been selected in order to yield more robust estimates of the way these plots decay. For the specific dataset it appears that the average decay of the logscree plot is very close to $0.5$, indicating a square root average decay of the components of $\underline{\mathbf{g}}_1$.

## VI. Conclusions and Future Work

This paper presented the Java implementation utilizing the Neo4j low level API in the NetBeans environment of the eigenvector centrality metric for undirected graphs using the power method. The anonymized Facebook egonet collection from SNAP [24] has been used to verify the correctness of the proposed implementation. The rankings computed by the Java code have been compared to those produced by MATLAB and no mismatches have been found. The time required in Java was within reasonable bounds, implying that for medium size graphs at least some analytics can be computed at the application level, offering a first insight to the final user without the need for external applications. Notice that this approach can be used in graphs from other fields.

Research on efficient graph tools remains to be done. One possible research path at the system level lies in the direction of combining a graph database with a streaming system. Thus the underlying graph would need to be updated in almost real time. In turn graph analytics would have to take this fact into account and adjust accordingly, perhaps sacrificng a degree of accuracy in favor of speed. Such scenaria are expected to be commonplace in the emerging IoT.

Graph modeling is another aspect of current analysis strategies. In situations where real graphs are inacessible, synthetic ones are employed in their place. Synthetic graphs obey certain laws that their real counterparts do, in the sense that they have similar structural and spectral characteristics [12]. Structural models include the Aiello family [1], while spectral models include the Kronecker model [11]. A suit of synthetic graph generators could be a useful part of Neo4j API.

One last possible research alternative would be to extend existing graph analytics and graph models to fuzzy graphs. In a fuzzy graph edges exist with a certain probability, paving the way for new definitions of path length and, consequently, of vertex centrality. Very recently a fuzzy version of Cypher has been developed based on a fuzzy alebra for graph querying [20], indicating research interest towards this direction.

## References

[1] W. Aiello, F. Chung, and L. Lu. A random graph model for massive graphs. In *Proceedings of the Thirty-second Annual ACM Symposium on Theory of Computing*, STOC '00, pages 171–180, New York, NY, USA, 2000. ACM.

[2] G. Antoniou. *A semantic web primer*. The MIT Press, 2004.

[3] M. Benzi and P. Boito. Quadrature rule-based bounds for functions of adjacency matrices. *Linear Algebra and its Applications*, 433(3):637–652, 2010.

[4] Z. Bi, C. Faloutsos, and F. Korn. The "dgx" distribution for mining massive, skewed data. In *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '01, pages 17–26, New York, NY, USA, 2001. ACM.

[5] E. F. Codd. Relational database: A practical foundation for productivity. In *Turing award lectures*. ACM, November 1981.

[6] M. Erwig. Fully persistent graphs: Which one to choose? In C. Clack, K. Hammond, and T. Davie, editors, *Implementation of Functional Languages*, volume 1467 of *Lecture Notes in Computer Science*, pages 123–140. Springer Berlin Heidelberg, 1998.

[7] E. Estrada and D. J. Higham. Network properties revealed through matrix functions. *SIAM Rev.*, 52(4):696–714, Nov. 2010.

[8] L. Katz. A new status index derived from sociometric analysis. *Psychometrika*, 18(1):39–43, March 1953.

[9] S. Kontopoulos and G. Drakopoulos. A space efficient scheme for graph representation. In *Proceedings of the 26th International Conference on Tools with Artificial Intelligence*, ICTAI 2014, pages 299–303. IEEE, November 2014.

[10] J. Leskovec. Social media analytics: Tracking, modeling and predicting the flow of information through networks. In *Proceedings of the 20th International Conference Companion on World Wide Web*, WWW '11, pages 277–278, New York, NY, USA, 2011. ACM.

[11] J. Leskovec, D. Chakrabarti, J. Kleinberg, C. Faloutsos, and Z. Ghahramani. Kronecker graphs: An approach to modeling networks. *J. Mach. Learn. Res.*, 11:985–1042, Mar. 2010.

[12] J. Leskovec, J. Kleinberg, and C. Faloutsos. Graphs over time: densification laws, shrinking diameters and possible explanations. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, KDD05, pages 177–187, New York, NY, USA, 2005. ACM.

[13] Y. Low, J. Gonzalez, A. Kyrola, D. Bickson, C. Guestrin, and J. M. Hellerstein. GraphLab: A new parallel framework for machine learning. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, Catalina Island, California, July 2010.

[14] G. Malewicz, M. H. Austern, A. J. Bik, J. C. Dehnert, I. Horn, N. Leiser, and G. Czajkowski. Pregel: a system for large-scale graph processing. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*, SIGMOD '10, pages 135–146, New York, NY, USA, 2010. ACM.

[15] F. D. Malliaros and V. Megalooikonomou. Expansion properties of large social graphs. In *Proceedings of the 16th international conference on Database systems for advanced applications*, pages 311–322. Springer, 2011.

[16] F. D. Malliaros, V. Megalooikonomou, and C. Faloutsos. Estimating robustness in large social graphs. *Knowledge and Information Systems: An International Journal*.

[17] F. D. Malliaros, V. Megalooikonomou, and C. Faloutsos. Fast robustness estimation in large social graphs: Communities and anomaly detection. In *Proceedings of the SIAM International Conference on Data Mining*, 2012.

[18] C. Moler and C. V. Loan. Nineteen dubious ways to compute the exponential of a matrix. *SIAM Review*, 20:801–836, 1978.

[19] Onofrio Panzarino. *Learning Cypher*. PACKT publishing, May 2014.

[20] O. Pivert, V. Thion, H. Jaudoin, and G. Smits. On a fuzzy algebra for querying graph databases. In *Proceedings of the 26th International Conference on Tools with Artificial Intelligence*, ICTAI 2014, pages 748–755. IEEE, November 2014.

[21] I. Robinson, J. Webber, and E. Eifrem. *Graph Databases*. O'Reilly, June 2013.

[22] M. A. Russell. *Mining the social web: Analyzing data from Facebook, Twitter, LinkedIn, and other social media sites*. O'Reilly, 2nd edition, October 2013.

[23] N. Shadbolt, W. Hall, and T. Berners-Lee. The semantic web revisited. *Intelligent Systems, IEEE*, 21(3):96–101, 2006.

[24] SNAP, February 2015.

[25] H. Tong, B. A. Prakash, T. Eliassi-Rad, M. Faloutsos, and C. Faloutsos. Gelling and melting large graphs by edge manipulation. In *Proceedings of the 21st ACM international conference on Information and knowledge management*, CIKM12, pages 245–254, New York, NY, USA, 2012. ACM.

[26] C. E. Tsourakakis. Fast counting of triangles in large real networks without counting: Algorithms and laws. In *ICDM*, pages 608–617, 2008.