Graph Community Discovery Algorithms in Neo4j with a Regularization-based Evaluation Metric

Andreas Kanavos, Georgios Drakopoulos and Athanasios Tsakalidis
Computer Engineering and Informatics Department, University of Patras, Achaia 26504, Greece

Keywords: CNM Algorithm, Community Discovery, Graph Databases, Graph Mining, Graph Signal Processing, Louvain Algorithm, Newman-Girvan Algorithm, Neo4j, Regularization, Walktrap Algorithm.

Abstract: Community discovery is central to social network analysis as it provides a natural way for decomposing a social graph to smaller ones based on the interactions among individuals. Communities do not need to be disjoint and often exhibit recursive structure. The latter has been established as a distinctive characteristic of large social graphs, indicating a modularity in the way humans build societies. This paper presents the implementation of four established community discovery algorithms in the form of Neo4j higher order analytics with the Twitter4j Java API and their application to two real Twitter graphs with diverse structural properties. In order to evaluate the results obtained from each algorithm a regularization-like metric, balancing the global and local graph self-similarity akin to the way it is done in signal processing, is proposed.

1 INTRODUCTION

Twitter is currently the most popular microblogging platform and the stage for ongoing political, financial, and cultural conversations with a vast amount of tweets being posted on a daily basis. Decomposing a Twitter social graph to communities yields a deeper insight to these seemingly chaotic interactions. However, community discovery is by no means a trivial task. Besides the large volume of accounts, tweets, retweets, and hashtags that need to be examined, necessarily implying parallel or distributed processing, the question of what constitutes a community, although posed in easily understood terms, remains to be definitively answered. This does not imply that no formal community definition exists. Quite the contrary, a plethora of such definitions has been in fact proposed, for instance in (Carrington et al., 2005), (Fortunato, 2010), (Newman, 2010), which successfully capture crucial aspects of human social organization. However, they differ in key aspects and, therefore, lead to different community detection algorithms.

Similarly, there are a number of ways to assess the clustering quality, namely community coherence. However, most of the existing coherence metrics are either prohibitively expensive, such as the maximum distance between vertices, or are prone to outliers, such as the diameter-based metrics (Drakopoulos et al., 2015b) (Drakopoulos et al., 2016). To this end, a coherence metric balancing global and local self-similarity properties with a rationale similar to the signal processing *regularization* criterion

$$K = \|\mathbf{x}-\mathbf{A}\mathbf{s}\|^2+\mu_0\|\mathbf{B}\mathbf{s}\|^2, \quad \mu_0>0 \tag{1}$$

which given a data vector $\mathbf{x}$, possibly with noise and outliers, computes a smoother version $\mathbf{s}$ thereof by combining global and local patterns coded in matrices $\mathbf{A}$ and $\mathbf{B}$ respectively. μ_0 (Drakopoulos and Megalooikonomou, 2016) controls their contribution to $\mathbf{s}$.

Graph databases such as Neo4j[1], GraphDB[2], and BrightstarDB[3] provide production grade front- or back-end graph storage. In addition, they also offer graph analytics such as link prediction and minimum spanning trees (Panzarino, 2014) (Robinson et al., 2013). Higher order analytics, such as community discovery, constitute a significant addition as they offer deeper insight in the graph structure.

The primary contribution of this paper is twofold. Four community discovery algorithms, namely the Newman-Girvan, the Walktrap, the Louvain, and the CNM were implemented in Java over Neo4j. Moreover, the results of these algorithms applied to two Twitter graphs created with Twitter4j[4] are evalu-

[1] www.neo4j.com
[2] www.ontotext.com
[3] www.brightstardb.com
[4] http://twitter4j.org/en/index.html

Kanavos, A., Drakopoulos, G. and Tsakalidis, A.
Graph Community Discovery Algorithms in Neo4j with a Regularization-based Evaluation Metric.
DOI: 10.5220/0006382104030410
In *Proceedings of the 13th International Conference on Web Information Systems and Technologies (WEBIST 2017)*, pages 403-410
ISBN: 978-989-758-246-2

ated with a regularization-like criterion which is efficiently computed and relies on the fundamental self-similarity property of scale-free graphs.

The rest of this paper is structured as follows. Section 2 provides an overview of community detection algorithms. The main characteristics of graph databases are described in section 3. The inherent high order nature of graph communities and four implemented algorithms are outlined in section 4. Finally, section 5 describes the datasets used in this paper and the results obtained from executing the community detection algorithms in Neo4j, whereas section 6 concludes by recapitulating the main findings and exploring future research directions.

Table 1: Paper notation.

Symbol	Meaning
$\triangleq$	Definition or equality by definition
$\deg(v_k)$	Degree of vertex v_k
K_n	Complete graph with n vertices
$(v_1, \ldots, v_n)$	Path with vertices $v_1, \ldots, v_n$
$\{s_k\}$	Set containing elements s_k
$\lvert S\rvert$ or $\lvert\{s_k\}\rvert$	Cardinality of set S or $\{s_k\}$
$\langle s_k \rangle$	Sequence of items s_k

2 RELATED WORK

Community detection is related mostly to graph clustering (Scott, 2000), Web retrieval (Newman, 2010), and user influence (Carrington et al., 2005). Concerning graph clustering, it can be performed either structurally or spectrally. In the former case partitioning is based on the properties of the graph adjacency matrix (Kernighan and Lin, 1970) (Shi and Malik, 2000), whereas in the latter connectivity patterns such as edge density or modularity (Newman, 2004b) (Newman, 2004a) play a primary role with notable examples being (Blondel et al., 2008), (Girvan and Newman, 2002). Vertex ranking, computed for instance with PageRank (Brin and Page, 1998) including its variants (Langville and Meyer, 2006) or HITS (Kleinberg, 1998), can be used to build communities with vertices which share common topics. Authority estimation can also be used to construct graph communities. In (Agichtein et al., 2008) several graph features as well as hub and authority scores are used to model the relative importance of a given user. Alternatively, in the expertise ranking model (Jurczyk and Agichtein, 2007), authorities are derived by performing link analysis to the graph induced from interactions between users. Moreover, in (Weng et al., 2010) authors employ Latent Dirichlet Allocation and a PageRank variant to cluster the graph according to topics and subsequently the authorities for each topic are identified. This was extended in (Pal and Counts, 2011) with additional features, advanced clustering and real-time capabilities. In addition, a previous work regarding influential communities identification is presented in (Kafeza et al., 2014). Finally, an overall and extensive overview of the community discovery field is (Fortunato, 2010).

Signal regularization is a common technique aiming at deriving a smoother or cleaner version of a data vector without altering the regions of interest. It has numerous applications in signal processing (Drakopoulos and Megalooikonomou, 2016), machine learning (Girosi et al., 1995), system identification (Johansen, 1997), and inverse problem theory (Vogel, 2002), while it also has connections to Sobolev space theory (Adams and Fournier, 2003) and to reproducible kernel Hilbert space theory (Attouch and Azé, 1993).

The interest in the graph processing field has been invigorated with the advent of open source graph databases such as Neo4j, GraphDB and BrightStar. Graph processing is usually implemented with the use of massive distributed graph computing systems like Google Pregel and graph based machine learning frameworks like GraphLab. In these systems, graphs play a twofold role as the data flow model and as the learning model.

3 ARCHITECTURE AND SOFTWARE

Graph databases such as Neo4j constitute one of the four major database technologies collectively known as NoSQL. RDBMSs assume that data can be represented in a structured and tabular manner. However, the modern Web and the IoT generate unstructured or semistructured, higher order, linked data which cannot be easily described by a schema. The primary properties of Neo4j include (Robinson et al., 2013) (Panzarino, 2014) (Drakopoulos et al., 2015a)

Property 1. *Neo4j is schemaless.*

Property 2. *Neo4j conforms to BASE requirements.*

Property 3. *The property graph model is the primary conceptual data model of Neo4j.*

Property 4. *Neo4j supports SPARQL, a W3C RDF query language, and Gremlin, a path query language (Drakopoulos et al., 2015a). However, queries to a Neo4j system are mostly submitted in Cypher, an ASCII art, pattern based, declarative language. The basic Cypher query has the form*

```
[start <pattern>]
match <pattern>
[with <pattern> [as <pattern>]]
where <pattern>
return <expression>
[order by <function> [desc]]
```

Cypher queries can be submitted directly in Neo4j console or, most frequently, through an application over a Neo4j API. For Java the Neo4j API is included in the Neo4j NetBeans extension library.

Figure 1 illustrates its components, including the social crawler, Neo4j, and the graph analytics, as well as the data flow between them.

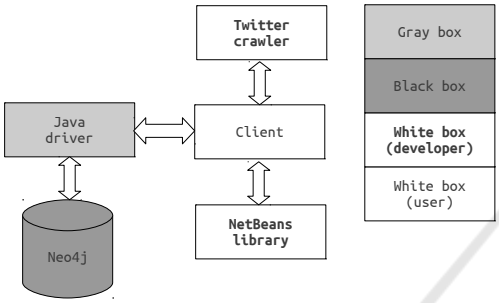


Figure 1: System architecture.

The social crawler has been implemented in Java using the Twitter4j API for collecting Twitter data and NetBeans for interfacing with Neo4j. Concerning system configuration, the Twitter crawler is currently inaccessible from the client, excluding thus any loops with user feedback in subsequent Twitter crawlings. The Neo4j version is 2.2.5, the latest available version at the beginning of development.

4 COMMUNITY DISCOVERY

This section outlines four popular community detection algorithms. It should be noted that these community detection algorithms rely on the inherently higher order information found as the graph structure. The latter is expressed in terms of the number of vertices or edges that need to be visited or traversed respectively in order to compute a graph function. Typical examples include the diameter or the number of shortest paths connecting two given vertices. This can be at least partly attributed to the linked graph nature which balances local and global information. Therefore, graph processing systems should be of similar nature, if useful information is to be extracted.

A manifestation of the higher order nature of the graph community detection problem is that the smallest community is a triangle. In terms of vertices, it can be considered as a third order quantity. If a triangle is closed, then it is a third order quantity in terms of edges as well. This stems from the fact that single relationships between individuals, namely edges in social graphs, do not qualify as communities. Thus, in a group there has to be at least one common acquaintance connecting the individuals in this group. This is reflected by the fact that successful community detection algorithms rely on higher order metrics directly or indirectly. For instance, graph clustering or spectral graph partitioning algorithms exploit higher order constructs such as the primary eigenvector or the resolvent of the graph adjacency matrix (Benzi and Boito, 2010).

4.1 Louvain Algorithm

Louvain or *multilevel* algorithm (Blondel et al., 2008) is a hierarchical clustering algorithm operating on weighted graphs. Initially each vertex is a single community. Then, communities are progressively merged with neighboring ones based on the local edge density change. The objective is to create communities where edge density is high, while intercommunity density remains low.

Louvain algorithm expresses the intuitive notion of edge density with modularity, a scalar m ranging from -1 to $+1$ is defined as

$$m \triangleq \begin{cases} \frac{1}{2|E|}\sum_{(i,j)}\left(w_{i,j}-\frac{\deg(v_i)\deg(v_j)}{2|E|}\right), & v_i \in c_i \wedge v_j \in c_j \\ 0, & v_i, v_j \in c_i \end{cases} \quad (2)$$

In (2) c_i and c_j denote the communities v_i and v_j belong to and $w_{i,j}$ is the weight of (i, j). Although the Louvain algorithm can be applied to unweighted graphs, the result is always a weighted graph where weights are proportional to local edge density. An unweighted graph is treated as a weighted graph with initial weights equal to one.

Modularity is maximized through a sequence of two alternating steps. In the first step, each v_i is merged with each of its neighbors into a single community C and the modularity change Δm is computed as the difference between the new modularity minus the old one. Finally, v_i is assigned to the c_j yielding the bigger Δm. In the second step, a new graph is constructed where all vertices belonging to the same community are merged into a single vertex. All edges connecting two communities form a single edge whose weight is the sum of the individual weights.

Algorithm 1: Louvain (multilevel) algorithm.

Require: Graph $G(V,E)$
Ensure: G is partitioned into communities
if G is unweighted **then**
  for all $(v_i, v_j) \in E$ **do**
    $w_{i,j} \leftarrow 1$
  end for
end if
$k \leftarrow 0$ **and** $V_0 \leftarrow V$ **and** $E_0 \leftarrow E$
for all $v_i \in V_0$ **do**
  v_i becomes a separate community
end for
Compute m as in (2)
repeat
  for all $v_i \in V_k$ **do**
    for all $v_j \neq v_i \,|\, (i,j) \in E_k$ **do**
      Assign temporarily v_i to c_j.
      Compute Δm.
    end for
  end for
  Assign v_i to c_j with the biggest Δm
  Merge vertices of c_i to a single vertex
  Merge edges within c_i to a single loop
  Merge edges between c_i and c_j to a single edge
  $k \leftarrow 1$ **and** update V_k, E_k
until no Δm can occur.
return

4.2 Newman-Girvan Algorithm

Newman-Girvan or edge betweeness algorithm (Girvan and Newman, 2002) relies on betweeness centrality, an edge centrality metric which counts the fraction of the number of the shortest paths connecting two vertices v_i and v_j a given edge e_k is part of, denoted by $\zeta^k_{i,j}$, to the total number of shortest paths connecting v_i and v_j, denoted by $\zeta_{i,j}$. Then the betweeness centrality for e_k, denoted by B_k, is computed by averaging over each vertex pair

$$B_k \triangleq \begin{cases} \frac{1}{\binom{|V|}{2}} \sum_{(v_i,v_j)\in V\times V} \frac{\zeta^k_{i,j}}{\zeta_{i,j}}, & v_i \neq v_j \\ 1, & v_i = v_j \end{cases} \quad (3)$$

In (Girvan and Newman, 2002) a process for computing B_k for each e_k, in a manner resembling breadth-first search, is described. The rationale is that vertices belonging to different communities should rely on edges connecting communities for information exchange. However, note that the converse does not need to be true. Moreover, depending on graph topology, some of the community connecting edges may not be high ranked in terms of betweeness centrality, as other edges may be more preferable. Therefore, the edge e^* with the highest betweeness centrality should be removed and subsequently the process should be again applied to the new graph. Eventually, all edges connecting communities will be identified. Intuitively, the edge sequence $\langle e^* \rangle$ should contain the graph bridges as well, which are a subset of the community connecting edges. In case the graph becomes disconnected, then the process is repeated for each of the connected components.

Algorithm 2: Newman-Girvan algorithm.

Require: Graph $G(V,E)$; Termination criterion τ_0
Ensure: G is partitioned into communities
while $E \neq \varnothing$ **and** τ_0 **not** satisfied **do**
  Compute B_k as in (3)
  $e^* \leftarrow \text{argmax}_k\{B_k\}$
  $E \leftarrow E \setminus \{e^*\}$
end while
return

4.3 Walktrap Algorithm

Walktrap algorithm is based on the principle of random walker. Starting from any random vertex the random walker will eventually spend more time steps in densely interconnected graph segments, as it is more probable for a randomly picked edge to lead to another vertex inside the segment than to a vertex outside it. Since such densely connected segments intuitively correspond to communities, random walks based metrics for community detection has been proposed in (Pons and Latapy, 2005). The probability that the walker moves from v_i to v_j is

$$p_{i,j} = \frac{\mathbf{A}[i,j]}{\deg(v_i)} \quad (4)$$

where $\mathbf{A}$ denotes the adjacency matrix

$$\mathbf{A}[i,j] \triangleq \begin{cases} 1, & i = j \vee (i,j) \in E \\ 0, & i \neq j \wedge (i,j) \notin E \end{cases} \in \{0,1\}^{|V|\times|V|} \quad (5)$$

As the probability that the random walker reaches v_j from v_i through a path of length ℓ, is denoted by $p^\ell_{i,j}$, then if v_i and v_j belong to the same community, then $p^\ell_{i,j}$ should be large for at least large values of ℓ. Note that the converse is not always true, depending on graph topology

$$p^\ell_{i,j} = \sum_{\pi_k |\pi|=\ell} \prod_{(v_i,v_j)\in\pi_k} p_{i,j} \quad (6)$$

where

$$\pi = (v_{k_1}, \ldots, v_{k_{\ell+1}}), \quad |\pi| = \ell \quad (7)$$

Equations lay the groundwork for defining the distance $d_{i,j}$ between v_i and v_j as

$$d_{i,j}^{\ell} \triangleq \sqrt{\sum_{k=1}^{|V|} \frac{\left(p_{i,k}^{\ell} - p_{j,k}^{\ell}\right)^2}{\deg(v_k)}} \tag{8}$$

The transition probability $p_{C,k}^{\ell}$ from any vertex belonging to a community C to v_k in ℓ steps, is defined as

$$p_{C,k}^{\ell} = \frac{1}{|C|} \sum_{i \in C} p_{i,k}^{\ell} \tag{9}$$

Generalizing (8), the distance r_{C_i,C_j} between the communities C_i and C_j is defined as

$$r_{C_i,C_j} = \sqrt{\sum_{k=1}^{|V|} \frac{\left(p_{C_i,k}^{\ell} - p_{C_j,k}^{\ell}\right)^2}{\deg(v_k)}} \tag{10}$$

Algorithm 3: Walktrap algorithm.

Require: Graph $G(V,E)$; Termination criterion τ_0
Ensure: G is partitioned into communities
1: **for all** $v_k \in V$ **do**
2: Assign v_k to a separate community
3: **end for**
4: **repeat**
5: **for all** distinct community pairs C_i and C_j **do**
6: Compute r_{C_i,C_j} as in (10).
7: **end for**
8: Merge communities which minimize r_{C_i,C_j}.
9: **until** one community remains.
10: **return**

4.4 CNM Algorithm

The CNM algorithm is also a hierarchical vertex partitioning algorithm. As such, initially each vertex constitutes a separate community. Then, neighboring communities are progressively merged to larger ones until no more merging is feasible according to the structurality criterion a. For a single vertex v_i, a_i is defined as

$$a_i \triangleq \frac{\deg(v_i)}{2|E|} \tag{11}$$

For two neighboring vertices, $\Delta a_{i,j}$ is defined as

$$\Delta a_{i,j} \triangleq \frac{1}{2|E|} - \frac{\deg(v_i)\deg(v_j)}{4|E|^2} \tag{12}$$

and it is zero for non-neighboring vertices. $\Delta a_{i,j}$ corresponds to the structural changes incurred from adding (i,j) to a community. In order to keep track of $\Delta a_{i,j}$, they are stored in a sparse matrix. Also the communities are stored in a binary tree where the leaves are the individual vertices. Each time two communities are merged, the resulting community is their parent at the tree. Moreover, the two corresponding columns of the $\Delta a_{i,j}$ sparse matrix are merged and their elements are updated according to the following rules (assuming communities i and j are to be fused):

- If community k is linked with communities i and j, then
$$\Delta a_{j,k} = \Delta a_{i,k} + \Delta a_{j,k} \tag{13}$$
- If community k is linked with community i but not with j, then
$$\Delta a_{j,k} = \Delta a_{i,k} - 2a_j a_k \tag{14}$$
- Finally, if community k is linked with community j but not with i, then
$$\Delta a_{j,k} = \Delta a_{j,k} - 2a_i a_k \tag{15}$$

Algorithm 4: CNM algorithm.

Require: Graph $G(V,E)$; Termination criterion τ_0
Ensure: G is partitioned into communities
1: Assign each vertex to a separate community
2: **for all** discrete pairs $(v_i, v_j) \in V \times V$ **do**
3: Compute pairwise $\Delta a_{i,j}$ as in (12)
4: **end for**
5: **repeat**
6: **for all** remaining communities **do**
7: Compute pairwise $\Delta a_{i,j}$ as in (12)
8: **end for**
9: Find $\max \Delta a_{i,j}$ **and** fuse communities
10: Update binary tree **and** a_i **and** matrix $\Delta a_{i,j}$
11: **until** one community is left.
12: **return**

5 Results

5.1 Data Synopsis

Definition 1. *The (log)completeness σ_0 (σ_0') of a graph is defined as the ratio of the (log)number of edges to the (log)number of edges of* K_n.

$$\sigma_0 \triangleq \frac{|E|}{\binom{|V|}{2}} = \frac{2|E|}{|V|(|V|-1)} \approx \frac{2|E|}{|V|^2}$$
$$\sigma_0' \triangleq \frac{\log|E|}{\log\binom{|V|}{2}} \approx \frac{\log|E|}{2\log|V|} \tag{16}$$

Definition 2. *The (log)density ρ_0 (ρ'_0) of a graph is defined as the ratio of the (log)number of edges to the (log)number of vertices.*

$$\rho_0 \triangleq \frac{|E|}{|V|} = \frac{|V|\sigma_0}{2}$$
$$\rho'_0 \triangleq \frac{\log|E|}{\log|V|} \approx 2\sigma'_0 \quad (17)$$

Notice that the base of the logarithm affects neither σ'_0 nor ρ'_0 since

$$\log_{b_1} x = \frac{\log_{b_2} x}{\log_{b_2} b_1}, \quad x \neq 0 \quad (18)$$

It follows from (16) and (17) that

$$\frac{\rho'_0}{\rho_0} = \frac{4}{|V|}\frac{\sigma'_0}{\sigma_0} \quad (19)$$

implying a balance between density, which connects the number of vertices and edges of the same graph, and completeness, which relates the number of edges of a graph to those of $K_{|V|}$.

In order to demonstrate the differences between the algorithms of section 4, two social graphs with anonymized Twitter users were constructed. Twitter4j retrieved users as well as information regarding who follows whom using a topic sampling approach. A keyword search query collected the users whose tweets or retweets contained #*Grexit*, a trendy and politically highly controversial topic, whereas a second query used the hashtag #*SocialNetwork*, a generic and by no means inciendiary topic. Subsequently, users following each other or having a common follower were connected with an edge as in (Kanavos et al., 2014). Tables 2 and 3 review graphs #*SocialNetwork* and #*SocialNetwork* respectively. Both seem to have similar properties on a macroscopic scale, however the seemingly subtle differences correspond to significant structural differences at the community level stemming from the nature of the two topics.

Table 2: #*Grexit* graph sunopsis.

Feature	Value	Feature	Value
Directed	True	Weighted	False
$\|V\|$	3696	$\|E\|$	8225
ρ_0	2.2313	ρ'_0	1.0973
σ_0	0.0012	σ'_0	0.5486

5.2 Analysis

Table 2 outlines the size of each community, expressed as a percentage of the total number of vertices, as generated by the four aforementioned algorithms. Louvain and CNM algorithms yield fewer communities than Newman-Girvan and Walktrap. Another observation is that communities tend to be clustered in size.

Table 3: #*SocialNetwork* graph sunopsis.

Feature	Value	Feature	Value
Directed	True	Weighted	False
$\|V\|$	4246	$\|E\|$	12054
ρ_0	2.8387	ρ'_0	1.1249
σ_0	0.0013	σ'_0	0.5624

Table 4: Community sizes (%) of #*Grexit* graph.

id	Edge	Walktrap	Louvain	CNM
1	9.30	10.50	6.50	9.70
2	6.10	12.60	13.60	10.20
3	5.10	7.00	11.20	22.10
4	13.10	7.20	6.40	18.50
5	2.70	11.20	7.90	14.30
6	13.20	10.20	12.50	11.40
7	5.20	8.40	13.60	5.50
8	15.10	6.30	12.20	8.30
9	12.10	5.50	6.30	-
10	11.10	11.40	5.20	-
11	1.10	2.10	4.60	-
12	3.40	3.40	-	-
13	2.50	4.20	-	-

Table 5: Community sizes (%) of #*SocialNetwork* graph.

id	Edge	Walktrap	Louvain	CNM
1	9.10	12.50	6.90	9.70
2	6.60	14.60	18.40	12.00
3	5.10	7.10	13.20	25.10
4	15.10	7.10	6.70	18.50
5	2.80	13.30	7.90	17.30
6	15.20	11.20	13.50	11.90
7	5.40	9.40	18.20	5.50
8	15.10	6.50	15.20	-
9	13.10	5.60	-	-
10	12.50	12.70	-	-

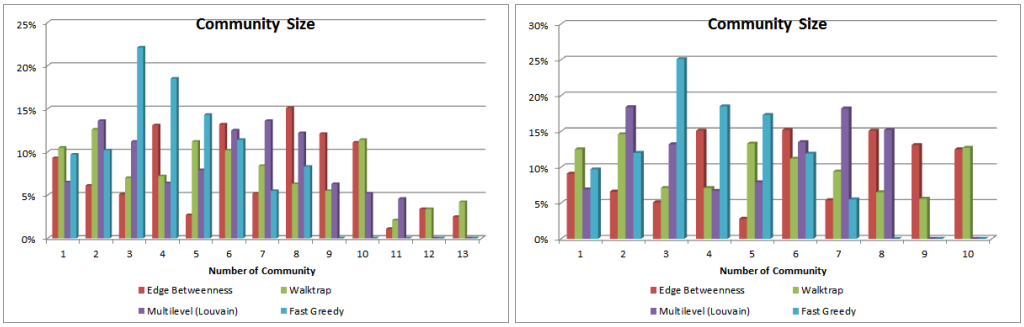
A metric for evaluating the clustering quality is inspired by the regularization cost function from (Drakopoulos and Megalooikonomou, 2016)

$$J(\lambda_0) = J_1 + \lambda_0 J_2, \quad \lambda_0 \in \mathbb{R}^+ \quad (20)$$

where λ_0 is a strictly positive factor expressing the relative importance of J_1 compared to J_2.

The first term measures the combined and weighted relative deviation of k-th community in terms of logdensity and logcompleteness in macroscopic or global scale, namely from the entire graph

$$J_1 \triangleq \sum_{k=1}^{|C_k|} \frac{|V_k|}{|V|}\left(\frac{|\rho'_k - \rho'_0|}{\rho'_0} + \frac{|\sigma'_k - \sigma'_0|}{\sigma'_0}\right) \quad (21)$$



(a) #Grexit (b) #SocialNetwork

Figure 2: Graph community sizes.

where ρ'_k and σ'_k are the logdensity and the logcompleteness of the k-th community whereas C_k is the set of communities. The weight of each community is the ratio of its vertices to the total number of vertices.

The second term quantifies the combined and weighted deviation from the expected scale-free behavior, again expressed in terms of logdensity and logcompleteness as in (19), in microscopic or local scale, namely at the community level

$$J_2 \triangleq \sqrt{\sum_{k=1}^{|C_k|} \frac{|V_k|}{|V|} \left(\frac{\rho'_k}{\rho_k} - \frac{4}{|V_k|} \frac{\sigma'_k}{\sigma_k} \right)^2} \qquad (22)$$

Once communities are derived, computing logdensity and logcompleteness is straightforward. This is an advantage over community metrics such as diameter. Moreover, $J(\lambda_0)$ is less prone to outliers and captures the scale-free behavior of the graph.

Table 6: J score for *#Grexit* graph.

λ_0	Edge	Walktrap	Louvain	CNM
0.1	14.94	15.55	13.67	21.44
0.3	11.61	12.69	12.33	18.12
0.5	09.11	11.07	10.59	18.37
0.7	08.42	11.01	11.12	19.95
0.9	09.73	12.45	13.49	21.17

Table 7: J score for *#SocialNetwork* graph.

λ_0	Edge	Walktrap	Louvain	CNM
0.1	18.42	20.42	20.61	25.34
0.3	17.75	20.11	19.70	24.99
0.5	18.04	19.23	17.44	23.18
0.7	19.78	18.92	18.63	22.34
0.9	20.53	19.53	21.00	21.53

As a general remark, there is no single optimum value for λ_0. Nonetheless, Newman-Girvan is consistently better with Walktrap and Louvain closely following and sharing the second position. CNM has the worst performance, which can be attributed to the fact that it creates fewer communities, which are bound to be heterogeneous. As Newman-Girvan is typically an exhaustive algorithm, it seems that Louvain and Walktrap algorithms are balanced options.

6 CONCLUSIONS AND FUTURE WORK

This paper outlines the implementation of Newman-Girvan, Walktrap, Louvain, and CNM community detection algorithms over Neo4j. Also, a criterion for assessing the compactness of the communities combining global and local scale-free graph behavior is proposed and tested on the results of applying these algorithms to two real Twitter graphs created from a neutral as well as a politically charged topic.

As future work, the scalability properties of community discovery should be considered in parallel or distributed environments. In addition, the proposed criterion should be tested on larger graphs. Finally, regarding λ_0, a scheme for computing its optimum value in finer granularity should be developed.

REFERENCES

Adams, R. A. and Fournier, J. J. (2003). *Sobolev spaces*, volume 140. Academic press.

Agichtein, E., Castillo, C., Donato, D., Gionis, A., and Mishne, D. (2008). Finding high-quality content in social media. In *Web Search and Data Mining conference (WSDM)*, pages 183–194. ACM.

Attouch, H. and Azé, D. (1993). Approximation and regularization of arbitrary functions in Hilbert spaces by the Lasry-Lions method. In *Annales de l'IHP Analyse non linéaire*, volume 10, pages 289–312.

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

Blondel, V. D., Guillaume, J.-L., Lambiotte, R., and Lefebvre, E. (2008). Fast unfolding of community hierarchies in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, P1000.

Brin, S. and Page, L. (1998). The PageRank citation ranking: Bringing order to the web. *Stanford Digital Library*.

Carrington, P. J., Scott, J., and Wasserman, S. (2005). *Models and Methods in Social Network Analysis*. Cambridge University Press.

Drakopoulos, G., Baroutiadi, A., and Megalooikonomou, V. (2015a). Higher order graph centrality measures for Neo4j. In *Conference of Information, Intelligence, Systems, and Applications (IISA)*.

Drakopoulos, G., Kanavos, A., Makris, C., and Megalooikonomou, V. (2015b). On converting community detection algorithms for fuzzy graphs in Neo4j. In *International Workshop on Combinations of Intelligent Methods and Applications*, CIMA 2015.

Drakopoulos, G., Kanavos, A., Makris, C., and Megalooikonomou, V. (2016). Comparing algorithmic principles for fuzzy graph communities over Neo4j. In *Advances in Combining Intelligent Methods*, pages 47–73.

Drakopoulos, G. and Megalooikonomou, V. (2016). Regularizing large biosignals with finite differences. In *International Conference of Information, Intelligence, Systems, and Applications (IISA)*.

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

Girosi, F., Jones, M., and Poggio, T. (1995). Regularization theory and neural networks architectures. *Neural computation*, 7(2):219–269.

Girvan, M. and Newman, M. (2002). Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, 99(2):7821–7826.

Johansen, T. A. (1997). On Tikhonov regularization, bias and variance in nonlinear system identification. *Automatica*, 33(3):441–446.

Jurczyk, P. and Agichtein, E. (2007). Discovering authorities in question answer communities by using link analysis. In *Conference of Information and Knowledge Management (CIKM)*, pages 919–922.

Kafeza, E., Kanavos, A., Makris, C., and Vikatos, P. (2014). T-PICE: Twitter personality based influential communities extraction system. In *IEEE International Congress on Big Data*, pages 212–219.

Kanavos, A., Perikos, I., Vikatos, P., Hatzilygeroudis, I., Makris, C., and Tsakalidis, A. (2014). Conversation emotional modeling in social networks. In *International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 478–484.

Kernighan, B. and Lin, S. (1970). An efficient heuristic procedure for partitioning graphs. *The Bell System Technical Journal*, 49(1):291–307.

Kleinberg, J. M. (1998). Authoritative sources in a hyperlinked environment. In *Symposium of Discrete Algorithms (SODA)*, pages 668–677.

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

Newman, M. E. (2004a). Detecting community structure in networks. *The European Physical Journal B-Condensed Matter and Complex Systems*, 38(2):321–330.

Newman, M. E. (2004b). Fast algorithm for detecting community structure in networks. *Physical Review E*, 69(6).

Newman, M. E. (2010). *Networks: An Introduction*. Oxford University Press.

Pal, A. and Counts, S. (2011). Identifying topical authorities in microblogs. In *Web Search and Data Mining (WSDM)*, pages 45–54.

Panzarino, O. (2014). *Learning Cypher*. PACKT publishing.

Pons, P. and Latapy, M. (2005). Computing communities in large networks using random walks.

Robinson, I., Webber, J., and Eifrem, E. (2013). *Graph Databases*. O'Reilly.

Scott, J. (2000). *Social Network Analysis: A Handbook*. SAGEPublications Ltd.

Shi, J. and Malik, J. (2000). Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905.

Vogel, C. R. (2002). *Computational methods for inverse problems*. SIAM.

Weng, J., Lim, E.-P., Lim, J., and Jiang, Q. H. (2010). Twitterrank: Finding topic-sensitive influential twitterers. In *Web Search and Data Mining (WSDM)*, pages 261–270.