Article

Fuzzy Random Walkers with Second Order Bounds: An Asymmetric Analysis

Georgios Drakopoulos [1], Andreas Kanavos [1,*] and Konstantinos Tsakalidis [2]

[1] Computer Engineering and Informatics Department, University of Patras, 26504 Patras, Greece; drakop@ceid.upatras.gr
[2] Cheriton School of Computer Science, University of Waterloo, Waterloo, ON N2L3G1, Canada; ktsakali@uwaterloo.ca
* Correspondence: kanavos@ceid.upatras.gr

Academic Editors: Katia Lida Kermanidis, Christos Makris, Phivos Mylonas and Spyros Sioutas
Received: 22 December 2016; Accepted: 27 March 2017; Published: 30 March 2017

Abstract: Edge-fuzzy graphs constitute an essential modeling paradigm across a broad spectrum of domains ranging from artificial intelligence to computational neuroscience and social network analysis. Under this model, fundamental graph properties such as edge length and graph diameter become stochastic and as such they are consequently expressed in probabilistic terms. Thus, algorithms for fuzzy graph analysis must rely on non-deterministic design principles. One such principle is Random Walker, which is based on a virtual entity and selects either edges or, like in this case, vertices of a fuzzy graph to visit. This allows the estimation of global graph properties through a long sequence of local decisions, making it a viable strategy candidate for graph processing software relying on native graph databases such as Neo4j. As a concrete example, Chebyshev Walktrap, a heuristic fuzzy community discovery algorithm relying on second order statistics and on the teleportation of the Random Walker, is proposed and its performance, expressed in terms of community coherence and number of vertex visits, is compared to the previously proposed algorithms of Markov Walktrap, Fuzzy Walktrap, and Fuzzy Newman–Girvan. In order to facilitate this comparison, a metric based on the asymmetric metrics of Tversky index and Kullback–Leibler divergence is used.

Keywords: Bernoulli distribution; binomial distribution; Chebyshev inequality; first order statistics; fuzzy graphs; graph analytics; higher order data; Jensen inequality; Markov inequality; Poisson distribution; Random Walker principle; second order statistics; Walktrap algorithm

1. Introduction

The Random Walker principle is the algorithmic cornerstone for building a number of heuristics for large graphs, namely for those with the fundamental property that neither their vertex nor their edge set fits in main memory. Such heuristics are efficient in terms either of computation time or memory requirements or often both. Under this principle, a virtual entity usually named the Random Walker visits the vertices. Within the scope of this article, the probabilistic strategy followed by the Random Walker to decide which vertex will visit next is of paramount importance, although, depending on the problem under study, other properties of the Random Walker may be of interest.

Virtual or ideal entities play an important role in science and engineering, mainly as a means to prove a theorem, to establish ideal performance limits, and to provide grounds for rejecting a conjecture based on a *reductio ad absurdum* methodology. Consider, for instance, the particle sorting demon of Maxwell [1,2] with its connections to algorithmic information theory and the steam engine of Heron of Alexandria [3]. In addition, the Random Walker principle itself has been applied to a number of graph analytics such as vertex similarity [4] and graph cuts [5] as well as to image processing [6].

A central finding of the theory of scale-free graphs, corroborated by significant evidence from a broad range of fields such as genomics, computational neuroscience, combinatorics, and social network analysis, states that they strongly tend to exhibit modularity. Namely, scale free graphs are recursively composed of vertex communities. The latter is a crucial factor for simultaneously achieving scalability, low-delay local information propagation, and structural robustness [7,8]. For instance, a large online social media group for Roman history afficionados can be composed of overlapping but to a great extent distinct specialized communities about Roman politics, law, and military, whereas the latter may be further subdivided into smaller communities regarding weaponry, tactics, legion standards, key battles, distinguished commanders, and the Praetorian guard.

Even though knowledge about communities offers a deep insight to graph structure, locating them is intractable. Consequently, numerous community discovery algorithms have been developed. Two of the most prominent ones are the Newman–Girvan [8] and the Walktrap [9] algorithms. The former is deterministic and is based on local edge density, while the latter is heuristic and relies on an edge crossing Random Walker. In [10], variants of both algorithms for edge-fuzzy graphs were proposed. Furthermore, in [11], a random walk approach for community detection in complex networks is introduced.

This article extends the work of [12]. Initially, the implementation of the Markov Walktrap algorithm, which is an extended version of Fuzzy Walktrap, was firstly proposed in [10]. Fuzzy Walktrap in turn has been based on deterministic Walktrap algorithm [9]. Furthermore, a method for algorithmically assessing the performance of Markov Walktrap relative to Fuzzy Walktrap and Fuzzy Newman–Girvan, being another fuzzy community detection technique [10], are thoroughly presented. Finally, one very important aspect to be mentioned is the expression of Markov Walktrap in Cypher, which composes the main query language for Neo4j [13].

The primary contribution of this article is the implementation over Neo4j of Chebyshev Walktrap, a community discovery algorithm designed for edge-fuzzy graphs, a class of fuzzy graphs [14] used among others in [10], which is based on the Random Walker algorithmic principle. Additionally, Chebyshev Walktrap relies on the competitive factors of second order statistics though the Chebyshev inequality and on an optional relocation capability in order to bound unnecessarily costly walks and, thus, remaining inside a community and being trapped for too long within the boundaries of a community, respectively. The relocation aspect was also backported to the Markov Walktrap algorithm first proposed in [15]. The effect of relocation on the community coherence was evaluated based on the asymmetric Tversky index using the Fuzzy Newman–Girvan algorithm from [15] as baseline, while its effect on the output distribution was assessed with the asymmetric Kullback–Leibler divergence.

The remainder of this article is structured as follows. Scientific literature regarding community discovery is reviewed in Section 2. The fuzzy graph model is outlined in Section 3 while Markov Walktrap and Fuzzy Walktrap algorithms are presented in Section 4. Experimental results are discussed in Section 5. Finally, the main findings of this paper as well as future research directions are discussed in Section 6. Table 1 summarizes the symbols used in this paper.

Table 1. Paper notation.

Symbol	Meaning
$\stackrel{\triangle}{=}$	Definition or equality by definition
$\otimes$	Kronecker tensor product
$\{s_1, \ldots, s_n\}$	Set with elements $s_1, s_2, \ldots, s_n$
$\lvert\cdot\rvert$	Set cardinality or path length (depending on the context)
$(e_1, \ldots, e_m)$	Path comprised of edges $e_1, \ldots, e_m$
K_n	Complete graph with n vertices and $\binom{n}{2}$ edges

Table 1. *Cont.*

Symbol	Meaning
$\mathrm{E}[X]$	Mean value of random variable X
$\mathrm{Var}[X]$	Variance of random variable X
$\tau_{T,V}$	Tanimoto similarity coefficient for sets T and V
$\nu_{T,V}$	Asymmetric Tversky index for sets T and V
$S_1 \setminus S_2$	Asymmetric set difference S_1 minus S_2
$\tilde{S}$	Fuzzy set S
$\langle s_k \rangle$	Sequence of elements s_k
$\mathcal{H}(s_1, \ldots, s_n)$	Harmonic mean of elements $s_1, \ldots, s_n$
$\mathcal{H}(s_1, \ldots, s_n; \tau_0)$	Thresholded or effective harmonic mean of $s_1, \ldots, s_n$
$\mathbf{1}_n$	$n \times 1$ vector with ones
$\mathbf{e}_n^k$	$n \times 1$ zero vector with a single one at the k-th entry
$f^{(n)}(x)$	n-th order derivative of $f(x)$
$\langle p \,\|\, q \rangle$	Kullback–Leibler divergence between distributions p and q

2. Related Work

From a system perspective, graph analytics play a crucial role in data mining systems such as Google Pregel or in massive machine learning frameworks such as GraphLab (https://turi.com/). Moreover, recently, there is strong interest for scalable, production grade graph databases [13,16] such as BrightStar (https://brightstardb.com), Neo4j (www.neo4j.com), Titan (https://github.com/thinkaurelius/titan), Sparksee (www.sparsity-technologies.com) and GraphDB (www.ontotext.com).

Traditionally, from an algorithmic viewpoint, analytics include structural [17,18] and spectral [19,20] partitioning, where a graph is split according to some functional constraints such as flow or edge density. Efficient information diffusion in large graphs is also of interest [21,22], especially for online political campaigns and digital marketing. The Random Walker principle has been also applied to two other important metrics, namely vertex similarity [4] and heuristic minimum cuts [5]. Both metrics can also be treated deterministically, especially in the context of social network analysis [23,24]. Community structure discovery provides insight to the inner workings of a particular graph [7,8,17], while metrics such as those in [25] control the discovery process quality. Persistent graphs can be instrumental in designing rollback capabilities in graph databases [26].

Among the numerous applications of graphs or linked data, one can find Web searching and ranking [27] with established algorithms such as PageRank [28] and HITS [29]. Bibliometric and scientometric data analysis [30] can boost collaboration between researchers, while image segmentation [6] is central to computer vision and robotics. Social network analysis has greatly benefited from structural [31] or functional [32,33] community detection algorithms. Additionally, influence and perceived social status in online social media have been tied to the participation in communities [34]. Message diffusion within a social graph is studied [34,35], while [36–38] deal with emotional modeling with respect to user influence [36]. Finally, random walkers have served as models for the propagation of computer viruses both in single systems and in networks, including LANs and the Internet [39,40]. Within this context, the strategy or the mix of strategies followed by the random walker is of paramount importance as it affects the entity and number of resources susceptible to infection.

First and second order statistics are used across a number of fields. In [41], a channel estimation methodology based on first order statistics is proposed. Methods for blind source separation using second order statistics include [42,43]. A comprehensive approach about the applications of higher order methods is given in [44] signal processing and in [45] for biomedical engineering. In [46], a third order method was presented for adaptively scheduling biosignal processing applications at the operating system level. Independent Component Analysis (ICA), a powerful signal processing technique, is based on higher order spectra [47]. Among the multitude of ICA applications is source separation in EEG waveforms [48].

3. Edge-Fuzzy Graphs

3.1. Definitions

Within the scope of this paper, the edge-fuzzy graphs are probabilistic and combinatorial hybrids comprised of a fixed set of vertices V and a fuzzy set of edges $\tilde{E}$. Formally,

Definition 1. *A homogeneous edge-fuzzy graph is the ordered triplet*

$$G \stackrel{\triangle}{=} (V, \tilde{E}, h), \tag{1}$$

where $V = \{v_k\}$ is the set of vertices, $\tilde{E} = \{e_k\} \subseteq V \times V$ is the fuzzy set of edges, and h is the edge membership function $h : E \to (0, 1]$, which quantifies the degree of participation of each e_k to G [10]. Moreover:

- *the vertex set V is fixed, namely they belong to G with probability one,*
- *the distribution h is the same for each edge,*
- *the existence probability of e_k is drawn independently for each edge.*

A subtle point is that h does not affect the structural properties of the graph in the sense that no edges are added or deleted, except when for a particular e_k holds that $h(e_k) = 0$. If h is continuous, the probability of this ocurring is zero. However, if h is discrete, then depending on h a potentially non-negligible portion of the edges may be deleted. In this article, h was chosen so that only at most an exponentially small proportion of the edges would be assigned to a zero weight. Consequently, the underlying graph preserved its original structure along with any associated connectivity patterns. Otherwise, if a considerable fraction of edges were to be deleted, then the resulting graph would behave more like an Erdös-Rényi graph. The latter are known to be easily constructed by randomly sampling a graph space or, equivalently, the edges of K_n but their properties deviate in a significant way from those of real world, large graphs.

Observe that there is no fuzziness whatsoever regarding vertices as by definition they always exist with probability one. In scientific literature, the existence of fuzzy graph classes is prominent. Vertices are fuzzy as well as their fuzziness interacts with that of the edges, mostly by long product chains. Such graphs are beyond the scope of this article.

Definition 2. *Under the fuzzy graph model of Defintion 1, the cost $\delta(e_k)$ of traversing e_k is*

$$\delta(e_k) \stackrel{\triangle}{=} \frac{1}{h(e_k)} \in [1, +\infty), \quad h(e_k) \in (0, 1], \tag{2}$$

which expresses the intuitive requirement that edges which are less likely to belong to the graph are also harder to cross.

Depending on the application, δ may well be connected through another non-linear transform to h as long as edges with high $h(e_k)$ are easy to cross and edges with low h are difficult to cross such as

$$\begin{aligned} \delta'(e_k) &\stackrel{\triangle}{=} \frac{1}{1 + h^2(e_k)} \in \left[\frac{1}{2}, 1\right], \\ \delta''(e_k) &\stackrel{\triangle}{=} \ln\left(\frac{1}{h(e_k)}\right) = -\ln h(e_k) \in [0, +\infty). \end{aligned} \tag{3}$$

Definition 3. *The cost $\Delta(p_j)$ of a fuzzy path $p_j = (e_1, \ldots, e_m)$ is the sum of the cost of its individual edges*

$$\Delta(p_j) \stackrel{\triangle}{=} \sum_{e_k \in p_j} \delta(e_k) = \sum_{k=1}^{m} \frac{1}{h(e_k)} = \frac{m}{\mathcal{H}(h(e_1), \ldots, h(e_m))}, \quad h(e_k) \neq 0, 1 \leq k \leq m, \tag{4}$$

where $\mathcal{H}\left(h(e_1),\ldots,h(e_m)\right)$ is the harmonic mean of $h(e_k)$ defined as

$$\mathcal{H}\left(s_1 \ldots s_n\right) \triangleq \frac{1}{\frac{1}{n}\sum_{k=1}^{n}\left(\frac{1}{s_k}\right)} = \frac{n}{\frac{1}{s_1}+\ldots+\frac{1}{s_n}}, \quad s_k \neq 0, 1 \leq k \leq n. \tag{5}$$

By construction $\Delta\left(p_j\right)$ is bounded as follows

$$\frac{1}{\max_{1\leq k\leq m}\left\{h(e_k)\right\}} \leq \frac{\Delta\left(p_j\right)}{\left|p_j\right|} \leq \frac{1}{\min_{1\leq k\leq m}\left\{h(e_k)\right\}}. \tag{6}$$

Whether the above bounds are loose depends on the variability of the actual values $h(e_k)$. That is, if the latter are drawn from a distribution which favors extreme values, $\Delta\left(p_j\right)$ will tend to be close to these bounds. It should also be noted that the variance $\Delta\left(p_j\right)$ is strongly dependent on that of $h(e_k)$. Moreover, $\Delta\left(p_j\right)$ is prone to outliers, which might lead to an unrealistically high average fuzzy path length. This can be remedied by taking into account the variance of the fuzzy path length. Finally, a low $\Delta\left(p_j\right)$ tends to contain almost exclusively low edge costs $\delta(e_k)$ or, equivalently, edges with high probability of belonging to the graph, an argument which agrees with the weak law of large numbers. In turn, this suggests the intuitive corollary that low cost paths are comprised almost exclusively of edges that are less likely to be fuzzy. This corollary can be used in order to design efficient hybrid probabilistic and combinatorial algorithms based on dynamic programming for finding and enumerating low cost paths akin to the way a similar observation has led to the development of shortest paths relying on dynamic programming in deterministic graphs.

From a probabilistic viewpoint, the sum $\sum_{k=1}^{m}\delta(e_k)$ is interesting by itself as a finite but possibly large sum of inverse random variables. Notice that the central limit theorem may not be applied in such a setting since the variance of the $h(e_k)$ might be infinite. If this is not the case, different bounds can be computed depending on the distribution of $h(e_k)$ such as the abovementioned central limit theorem, a Poisson bound, a power law bound, or finally approximations based on Markov or Chebyshev inequalities, the Chernoff bound, or on the Gnedenko extreme value theorem. Notice that the effect of a single edge which is exceedingly difficult to cross can be instrumental in shaping graph communities.

The numerical properties of the above sum are also of interest. As values of possibly uneven orders of magnitude may be added, catastrophic cancellation may occur resulting in the loss of meaningful information. This might happen if the summation is executed in an order left to the implementation. On the other hand, the summation order dictated by the Priest algorithm [49] results in the least possible loss of significant decimal digits by adding only numbers of comparable magnitude. Another option would be to substitute the harmonic mean with its thresholded counterpart

$$\mathcal{H}\left(s_1,\ldots,s_m;\tau_0\right) \triangleq \frac{m}{\sum_1^m \frac{1}{\max\{h(e_k),\tau_0\}}} = \frac{m}{\frac{1}{\max\{h(e_1),\tau_0\}}+\ldots+\frac{1}{\max\{h(e_m),\tau_0\}}}. \tag{7}$$

For other uses of the thresholded harmonic and geometric means, see [50], while, for the effect of finite precision arithmetic to long biosignals, see [51].

An alternative for long paths would be to substitute, under certain conditions, the finite sum with an appropriate integral. Assuming with no loss of generality that $h(e_1) = \min_{1\leq k\leq m}\left\{h(e_k)\right\} \neq 0$ and $h(e_m) = \max_{1\leq k\leq m}\left\{h(e_k)\right\} \neq 0$, then

$$\Delta\left(p_j\right) \triangleq \sum_{k=1}^{m}\frac{1}{h(e_k)} \approx \rho_0 + \int_{h(e_1)}^{h(e_m)}\frac{1}{x}dx = \ln h(e_m) - \ln h(e_1) + \rho_0 = \ln\left(\frac{h(e_m)}{h(e_1)}\right) + \rho_0, \tag{8}$$

where ρ_0 is an optional correction factor. A finer approach requiring more probabilistic information about the longer paths of a given graph would be to partition such a path p_j so that

$$\Delta(p_j) \stackrel{\triangle}{=} \sum_{k=1}^{m} \frac{1}{h(e_k)} \approx \sum_{i=1}^{n} \left(\frac{\ln h(e_{u_i})}{\ln h(e_{\ell_i})} + \rho_i \right) = \sum_{i=1}^{n} \left(\frac{\ln h(e_{u_i})}{\ln h(e_{\ell_i})} \right) + \sum_{i=1}^{n} \rho_i = \sum_{i=1}^{n} \left(\frac{\ln h(e_{u_i})}{\ln h(e_{\ell_i})} \right) + \rho_0'. \quad (9)$$

Selecting n and forming the sets $\{h(e_{u_i})\}_{i=1}^{n}$, $\{h(e_{\ell_i})\}_{i=1}^{n}$, and $\{\rho_i\}_{i=1}^{n}$ is not a trivial task. Instead, choosing such an approach might be a viable solution only for certain combinations of h and $|p_j|$. Techniques for estimating the variability as well as the cardinality of large sets such as [52] can be useful while pursuing this approach.

It should be emphasized that the class of fuzzy graphs of Definition 1 can be well considered as a typical example of higher order data. This is attributed to the inherently distributed way information is stored in a graph, in this particular case as edge existence probabilities. In order for meaningful information regarding path costs to be mined, a non-negligible fraction of edges must be crossed and, thus, the interplay of a number of edges must be considered.

3.2. Reciprocal Random Variables

Because of the Definitions 2 and 3 for $\delta(e_k)$ and $\Delta(p_j)$, respectively, the properties of an inverse random variable gain more interest. The following definition is straightforward.

Definition 4. *The inverse distribution of a mass distribution function of a random variable X is defined as the mass distribution function of $\frac{1}{X}$ [53].*

Property 1. *In the continuous case, the distributions of X and $\frac{1}{X}$ are linked as*

$$f_{\frac{1}{X}}(y) = \frac{1}{y^2} f_X\left(\frac{1}{y}\right), \quad y \neq 0. \quad (10)$$

Proof. The cumulative distribution of $\frac{1}{X}$ is defined as

$$F_{\frac{1}{X}}(y) \stackrel{\triangle}{=} \text{prob}\left\{\frac{1}{X} \leq y\right\} = \text{prob}\left\{X \geq \frac{1}{y}\right\} = 1 - \text{prob}\left\{X < \frac{1}{y}\right\} = 1 - F_X\left(\frac{1}{y}\right). \quad (11)$$

By differentiating the last relationship, the stated result follows. □

For instance, if X is the continuous uniform random variable in $[\alpha_1, \alpha_2]$, where $\alpha_1, \alpha_2 \neq 0$, then the distribution of $\frac{1}{X}$ is

$$f_{\frac{1}{X}}(y) = \frac{1}{y^2(\alpha_2 - \alpha_1)}, \quad y \in \left[\frac{1}{\alpha_2}, \frac{1}{\alpha_1}\right]. \quad (12)$$

Despite its simplicity, in certain scenaria, relationship (10) cannot be used. For instance, only the first moments of X may be known or $y = 0$ might be a legitimate value, in which case there is a singularity in the inversion of $f_X(x)$. Instead, bounds are sought for the first moments of $\frac{1}{X}$, which makes more sense from a programming viewpoint in the case of large graphs.

Jensen inequality provides a straightforward way to bound the expected value $\mathrm{E}\left[\frac{1}{X}\right]$ by using the expected value $\mathrm{E}[X]$ of the non-zero random variable X.

Theorem 1. *(Jensen inequality) For any random variable and any convex function $g(x)$ provided that both the domains of X and $\mathrm{E}[X]$ are subsets of the domain of $g(\cdot)$*

$$g(\mathrm{E}[X]) \leq \mathrm{E}[g(X)]. \quad (13)$$

Corollary 1. *The mean value of the strictly positive random variable $\frac{1}{X}$ has a lower bound of*

$$\frac{1}{\mathrm{E}\left[X\right]} \leq \mathrm{E}\left[\frac{1}{X}\right], \quad X \neq 0. \tag{14}$$

Property 2. *Function $g(x) = \frac{1}{x}$ is convex when $x > 0$.*

Proof. Notice that the second derivative $g^{(2)}(x) = \frac{2}{x^3}$ is positive when x is positive. An alternative way to prove this claim is to apply the standard convexity definition. For every $\alpha_0 \in [0,1]$, $x_1 > 0$, and $x_2 > 0$

$$\begin{aligned} g(\alpha_0 x_1 + (1-\alpha_0)x_2) \; &\leq \; \alpha_0 g(x_1) + (1-\alpha_0)g(x_2) \Rightarrow \\ \frac{1}{\alpha_0 x_1 + (1-\alpha_0)x_2} \; &\leq \; \frac{\alpha_0}{x_1} + \frac{1-\alpha_0}{x_2} \Leftrightarrow \\ \alpha_0 x_2(\alpha_0 x_1 + (1-\alpha_0)x_2) + (1-\alpha_0)x_1(\alpha_0 x_1 + (1-\alpha_0)x_2) - x_1 x_2 \; &\geq \; 0 \Leftrightarrow \\ \alpha_0(1-\alpha_0)(x_1-x_2)^2 \; &\geq \; 0. \end{aligned} \tag{15}$$

□

In order to derive realistic upper bounds for the path lengths of a given fuzzy graph, certain probabilistic inequalities can be employed. The first is Markov inequality, which establishes a first order bound for the probability of X taking very large values by stating that

Theorem 2. *(Markov inequality) The probability of a strictly positive random variable X exceeding γ_0 is bounded by*

$$\mathrm{prob}\left\{X \geq \gamma_0\right\} \leq \frac{\mathrm{E}\left[X\right]}{\gamma_0}, \quad \gamma_0 > 0, X > 0. \tag{16}$$

Second order bounds can be derived by the Chebyshev inequality. The latter provides tighter bounds while lifting the positivity assumption.

Theorem 3. *(Chebyshev inequality) The probability of an arbitrary random variable X exceeding its expected value by a certain fraction γ_0 of its standard deviation as*

$$\mathrm{prob}\left\{|X - \mathrm{E}\left[X\right]| \geq \gamma_0\sqrt{\mathrm{Var}\left[X\right]}\right\} \leq \frac{1}{\gamma_0^2}, \quad \gamma_0 > 0. \tag{17}$$

The Chebyshev inequality is generic enough to be applied in a number of scenaria, including those in the present article. Still, it should be noted that other techniques may provide sharper bounds in certain cases. For instance, when X is normally distributed, then specialized methods exist for evaluating the integral under its tail.

Estimating the variance of a transformed random variable can be done through the delta method.

Theorem 4. *(Delta method) Let X be a random variable whose expected value* $\mathrm{E}\left[X\right]$ *and variance* $\mathrm{Var}\left[X\right]$ *are known. For an analytic $g(x)$, the variance of $g(X)$ can be estimated as*

$$\mathrm{Var}\left[g(X)\right] \approx \left(g^{(1)}(\mathrm{E}\left[X\right])\right)^2 \mathrm{Var}\left[X\right]. \tag{18}$$

Proof. The first order approximation of the Taylor expansion of $g(\cdot)$ around $\mathrm{E}\left[X\right]$ is

$$g(X) = \sum_{k=0}^{+\infty} g^{(k)}(\mathrm{E}\left[X\right])\frac{X^k}{k!} \approx g(\mathrm{E}\left[X\right]) + g^{(1)}(\mathrm{E}\left[X\right])\left(X - \mathrm{E}\left[X\right]\right). \tag{19}$$

Taking the variance of both sides along with the identities,

$$\begin{aligned} \mathrm{Var}\left[\alpha_0 + \alpha_1 X\right] &= \alpha_1^2 \mathrm{Var}\left[X\right], \\ \mathrm{Var}\left[X - \mathrm{E}\left[X\right]\right] &= \mathrm{Var}\left[X\right], \end{aligned} \tag{20}$$

yields the stated result. □

Corollary 2. *For $g(x) = \frac{1}{x}$, the delta method yields*

$$\mathrm{Var}\left[\frac{1}{X}\right] \approx \frac{\mathrm{Var}\left[X\right]}{\mathrm{E}\left[X\right]^4} = \frac{\mathrm{E}\left[X^2\right] - \mathrm{E}\left[X\right]^2}{\mathrm{E}\left[X\right]^4} = \frac{\mathrm{E}\left[X^2\right]}{\mathrm{E}\left[X\right]^4} - \frac{1}{\mathrm{E}\left[X\right]^2}, \quad \mathrm{E}\left[X\right] \neq 0. \tag{21}$$

The Markov and Chebyshev are but two of the probabilistic inequalities collectively known as concentration inequalities, the latter bound the deviation of a random variable or a sequence of random variables from a known value. Other such inequalities include the Talagrand, the Efron–Stein, and the Dvoretzky–Kiefer–Wolfowitz inequalities.

4. Family of Walktrap Heuristics

4.1. Deterministic Walktrap

The original Walktrap algorithm [9] simulates an edge crossing random walker in order to estimate the stationary distribution of a homogeneous Markov chain. The walker can commence from any vertex and cross edges by randomly selecting destination vertices, systematically moving to vertices with high edge density as vertices are selected with probability proportional to their degree. Since vertices can be visited an arbitrary number of times, unlike algorithms like BFS and DFS, eventually some patterns in the vertex visiting sequence will emerge. As a community is from a structural perspective, essentially a locally dense graph segment, the walker is more likely to move along vertices belonging to the same community for a large time interval before moving to another community. Thus, analysis of the vertex sequence generated by the random walker can reveal the underlying graph community structure. The Walktrap algorithm is outlined in Algorithm 1.

Algorithm 1: Deterministic Walktrap

```
Require: graph $G(V,E)$, termination criterion $\tau_0$
Ensure: vertex pair sequence $\langle s_k, s_k^* \rangle$ is generated
1: pick a random vertex $v$
2: repeat
3:     pick a neighboring vertex $v^*$ with probability proportional to its degree as in (23)
4:     store current vertex in $v$ and move the walker to the new vertex $v^*$
       and $\langle s_{k+1}, s_{k+1}^* \rangle \leftarrow (v, v^*)$
5: until $\tau_0$ is true
6: return vertex sequence $\langle s_k, s_k^* \rangle$
```

The degree of any neighboring vertex can be determined by the graph adjacency matrix **A** defined as

$$\mathbf{A}[i,j] \triangleq \begin{cases} 1, & i = j \vee (i,j) \in E \\ 0, & i \neq j \wedge (i,j) \notin E \end{cases} \in \{0,1\}^{|V| \times |V|}. \tag{22}$$

Specifically, the degree of v_k is the sum of the k-th column of $\mathbf{A}$. The probability that from vertex v_p a neighbor v_q is selected at the next step is directly proportional to

$$\frac{\deg\left(v_q\right)}{\sum_{\left(v_p,v_j\right)\in E}\deg\left(v_j\right)} = \frac{\underline{\mathbf{1}}_{|V|}^T\,\mathbf{A}\,\underline{\mathbf{e}}_{|V|}^q}{\sum_{\left(v_p,v_j\right)\in E}\underline{\mathbf{1}}_{|V|}^T\,\mathbf{A}\,\underline{\mathbf{e}}_{|V|}^j}. \tag{23}$$

In contrast to many graph algorithms, each vertex may be visited more than once. In fact, vertices must be visited many times in order for meaningful patterns to emerge regarding community structure. Typically, for deterministic graphs, a constant number of visits per vertex may suffice resulting in a total of $\mathrm{O}\left(|V|\right)$ visits, though techniques exploiting the self-similarity nature of large, scale free graphs may yield somewhat lower bounds of $\mathrm{O}\left(\log^{1+\epsilon}|V|\right)$, $\epsilon > 0$. For fuzzy graphs, the linear bound is as of yet unknown as to whether it can be improved.

In a distributed setting such as Hadoop, the Deterministic Walktrap can be scaled up since graph segments can be distributed to the nodes. The map part will be the parallel random walkers crossing edges. If such a walker must cross a segment, it can either bounce back or be transferred to the appropriate node. The reduce part will be the frequency count of a large number of vertex pairs.

Once the random walker has finished crossing the graph, the communities are discovered by means of hierarchical clustering using the frequency of pairs (s, s^*) as weights. It should be noted that other methods such as Hidden Markov Models and text mining techniques dealing with missing values [54] may be used for discerning community patterns in the sequence $\left\langle s_k, s_k^*\right\rangle$.

4.2. Fuzzy Walktrap

The Fuzzy Walktrap algorithm has been proposed and analyzed in [10]. Similarly to Algorithm 2, given a vertex v_p, each neighbor v_q is a candidate for being visited by the Random Walker with probability proportional to its probability of belonging to the fuzzy graph, namely proportional to

$$\frac{h\left(\left(v_p,v_q\right)\right)}{\sum_{\left(v_p,v_j\right)\in\tilde{E}}h\left(\left(v_p,v_j\right)\right)} = \frac{\underline{\mathbf{1}}_{|V|}^T\,\mathbf{A}^F\,\underline{\mathbf{e}}_{|V|}^q}{\sum_{\left(v_p,v_j\right)\in E}\underline{\mathbf{1}}_{|V|}^T\,\mathbf{A}^F\,\underline{\mathbf{e}}_{|V|}^j}, \tag{24}$$

where the fuzzy adjacency matrix $\mathbf{A}^F$ defined as

$$\mathbf{A}^F[i,j] \triangleq \begin{cases} 1, & i = j, \\ h(e_{ij}), & e_{ij} = \left(v_i,v_j\right)\in\tilde{E} \\ 0, & \left(v_i,v_j\right)\notin\tilde{E}. \end{cases} \quad \in [0,1]^{|V|\times|V|}, \tag{25}$$

Fuzzy Walktrap is outlined in Algorithm 2.

Algorithm 2: Fuzzy Walktrap

```
Require: fuzzy graph $G\left(V,\tilde{E},h\right)$, termination criterion $\tau_0$
Ensure: vertex pair sequence $\left\langle s_k, s_k^*\right\rangle$ is generated
1: pick a random vertex $v$
2: repeat
3:     pick a neighboring vertex $v^*$ with probability proportional to its cost as in (24)
4:     store current vertex in $v$ and move the walker to the new vertex $v^*$
       and $\left\langle s_{k+1}, s_{k+1}^*\right\rangle \leftarrow \left(v, v^*\right)$
5: until $\tau_0$ is true
6: return $\left\langle s_k, s_k^*\right\rangle$
```

4.3. Markov Walktrap and Chebyshev Walktrap

Both the Markov Walktrap, proposed in [10], and Chebyshev Walktrap, introduced in this article, algorithms improve Fuzzy Walktrap in two ways. The first is that during the walking phase the Random Walker has two optional safeguards against being confined inside a community for too long. Both of these safeguards are common for Markov Walktrap and Chebyshev Walktrap. The second improvement is that, during the clustering phase, two communities may not merge if the path lengths within the resulting community exceed a certain threshold. The latter is based on first order statistics for the Markov Walktrap and on second order statistics for the Chebyshev Walktrap.

Regarding the control of community merge, for community V_k, the mean path cost π_k is the sum of the individual edge costs

$$\pi_k \stackrel{\triangle}{=} \frac{1}{|V_k|} \sum_{e_j \in V_k} \delta(e_j), \tag{26}$$

and, therefore, is also a random variable. By linearity of expectation,

$$\mathrm{E}[\pi_k] = \mathrm{E}\left[\frac{1}{|V_k|} \sum_{e_j \in V_k} \delta(e_j)\right] = \frac{|E_k|}{|V_k|} \mathrm{E}\left[\delta(e_j)\right]. \tag{27}$$

Moreover, as $\delta(e_j)$ are independent,

$$\mathrm{Var}[\pi_k] = \mathrm{Var}\left[\frac{1}{|V_k|} \sum_{e_j \in V_k} \delta(e_j)\right] = \frac{|E_k|}{|V_k|^2} \mathrm{Var}\left[\delta(e_j)\right]. \tag{28}$$

In the general case, the distribution of $\delta(e_j)$ is unknown, for specific choices of h, it can be computed or estimated. Alternatively, since π_k is a sum of random variables, its distribution may be known for certain special cases. For instance, the sum of independent Poisson random variable is another Poisson random variable. In addition, the sum of independent binomial random variables is also a binomial random variable. Finally, the sum of a large number of independent random variables with finite variance is a normal random variable according to the Central Limit Theorem. Nonetheless, in this article, no such assumptions were made and the expected value and the variance of π_k were approximated by the Jensen inequality and the delta method, respectively, in Equations (13) and (18).

Since π_k is by construction positive, the Markov inequality can be applied. Therefore,

$$\mathrm{prob}\left\{\pi_k \geq \gamma_0\right\} \leq \frac{\mathrm{E}[\pi_k]}{\gamma_0} \tag{29}$$

or, equivalently,

$$\mathrm{prob}\left\{\pi_k \geq \frac{\mathrm{E}[\pi_k]}{\gamma_0}\right\} \leq \gamma_0. \tag{30}$$

If, for a threshold $\alpha_0 \in (0,1)$ π_k exceeds $\alpha_0\gamma_0$, then that community is excluded from merging for an iteration provided it has at least ξ_0 vertices. This is a first order probabilistic safeguard preventing almost formed communities from losing their coherence.

On similar grounds, a second order such safeguard can be built on the Chebyshev inequality

$$\mathrm{prob}\left\{|\pi_k - \mathrm{E}[\pi_k]| \geq \gamma_1 \sqrt{\mathrm{Var}[\pi_k]}\right\} \leq \frac{1}{\gamma_1^2}. \tag{31}$$

4.4. Escape Strategies

Although the purpose of the Random Walker is to discover communities by repeatedly visiting neighboring vertices and crossing low cost edges, it is possible to be trapped inside a community if the latter is connected only through very high cost edges from the remaining graph. To this end, the Random Walker has the option as in [15] to reverse its strategy and select neighboring vertices with probability *inversely* proportional to their probability of existence if a random flag is triggered. The latter was implemented as a Bernoulli random variable with success probability q_0, which can be set to zero if so desired in order to disable the weight inversion strategy. Recommended values are typically $\mathrm{O}\left(|V|^{-\epsilon}\right)$, $\epsilon \geq 2$. Therefore, as in [15], when weight inversion is enabled, the distance d between communities implicitly depends on terms of the form

$$d \propto \frac{\prod_{e_k \in \tilde{E}} \delta(e_k)}{\prod_{e_j \in \tilde{E}} \delta(e_j)}, \tag{32}$$

instead of terms of the form

$$d \propto \prod_{e_k \in \tilde{E}} \delta(e_k). \tag{33}$$

Alternatively, a probabilistically triggered restart of the Random Walker akin to the PageRank teleportation [28], the random mutation operator in a genetic algorithm [55,56], or the restart strategy in GMRES iterative solver for linear systems [57,58] was also considered. The relocation probability q_1 has a Bernoulli distribution and is evaluated independently at each step. The number of steps before such a relocation takes place is finite however small q_1 may be as long as it remains strictly positive. The number of steps N to the first relocation has a geometric distribution with success probability equal to q_1 with mass distribution function

$$\mathrm{prob}\left\{N = k\right\} = q_1 \left(1 - q_1\right)^k, \quad k \geq 0. \tag{34}$$

Therefore, the expected value and variance of N, respectively, are

$$\mathrm{E}\left[N\right] = \frac{1}{q_1} \quad \text{and} \quad \mathrm{Var}\left[N\right] = \frac{1 - q_1}{q_1^2}. \tag{35}$$

Even though the relocation modification clearly violates the inherent locality of the Walktrap family of heuristics, if properly calibrated, it happens infrequently enough so as not to severely degrade time performance. Moreover, in a distributed system, a simple move of the random walker to the appropriate graph segment suffices and its cost is certainly affordable. Moreover, since relocation is a rare event, the total number of relocations can be modeled by a Poisson distribution.

5. Analysis

5.1. Data

In order to experimentally evaluate the performance of Markov Walktrap, a Kronecker synthetic graph [59–61] has been created. Kronecker graphs are recursively constructed from an original generator graph with the following model

$$\begin{aligned} \mathbf{A}_0 &= \mathbf{A}, \\ \mathbf{A}_{k+1} &= \mathbf{A}_k \otimes \mathbf{A}, \quad n \geq 1, \end{aligned} \tag{36}$$

where $\mathbf{A}_0$ is the generator graph and $\otimes$ denotes the Kronecker tensor product.

The generator matrix was

$$\mathbf{A}_0 = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 & 1 & 1 \end{bmatrix}, \tag{37}$$

which has $p = 7$ vertices numbered from 0 to 6 and each v_k is connected to v_{k_1}, v_{k_2}, and v_{k_3}, where

$$\begin{aligned} k_1 &= (k-1) \bmod p, \\ k_2 &= (k+1) \bmod p, \\ k_3 &= (k+2) \bmod p. \end{aligned} \tag{38}$$

The generator graph is shown in Figure 1.

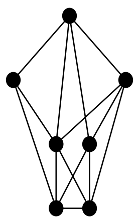

Figure 1. The generator graph.

The Kronecker model of Equation (36) has been executed six times aiming to obtain a large graph Υ whose properties are summarized in Table 2.

Definitions 5 and 6 outline four important structural graph metrics.

Definition 5. *The (log)density of a fuzzy graph is the ratio of the (logarithm of the) number of its edges to the (logarithm of the) number of its vertices.*

$$\begin{aligned} \rho_0 &\triangleq \frac{|E|}{|V|} = \frac{|V|^{1+\epsilon}}{|V|} = |V|^{\epsilon}, \quad 0 \le \epsilon \le 1, \\ \rho_0' &\triangleq \frac{\log|E|}{\log|V|} = \frac{\log|V|^{1+\epsilon}}{\log|V|} = 1+\epsilon. \end{aligned} \tag{39}$$

Definition 6. *The (log)completeness of a fuzzy graph is the ratio of the (logarithm of the) number of its edges to the (logarithm of the) number of the edges of the complete graph with the same number of vertices.*

$$\begin{aligned} \sigma_0 &\triangleq \frac{|E|}{\binom{|V|}{2}} = \frac{2|E|}{|V|(|V|-1)} \approx 2\,\frac{|E|}{|V|^2} = 2\,|V|^{\epsilon-1}, \\ \sigma_0' &\triangleq \frac{\log|E|}{\log\binom{|V|}{2}} = \frac{\log|E|}{\log\left(\frac{|V|(|V|-1)}{2}\right)} = \frac{\log|E|}{\log|V| + \log(|V|-1) - 1} \approx \frac{\log|E|}{2\log|V|} = \frac{\rho_0'}{2} = \frac{1+\epsilon}{2}. \end{aligned} \tag{40}$$

Table 2. Structural properties of graph Y.

Property	**Value**	**Property**	**Value**	**Property**	**Value**	**Property**	**Value**
Vertices	117,649	σ_0	7.68×10^{-4}	Triangles	37127	Squares	1981
Edges	5,315,625	σ'_0	0.6631	min cost	5.1891	max cost	10.7232
ρ_0	45.1821	Diameter length	19	avg cost	72.1149	avg cost	106.0012
ρ'_0	1.3263	Diameter cost	124.4021	max cost	143.2716	max cost	198.2221

Observe that Y is highly connected as it has a low diameter of a relatively low cost, high average degree, and a high number of triangles and squares.

5.2. Time and Memory Requirements

In Table 3, the total execution time for Chebyshev Walktrap (CW), Markov Walktrap (MW), Fuzzy Walktrap (FW), and Fuzzy Newman–Girvan (FN-G) is shown. The last two algorithms are outlined in [10], whereas Fuzzy Markov was proposed in [15]. The effect of the escape mechanisms of weight inversion (I) and relocation (R) are also shown for Markov Walktrap and Chebyshev Walktrap. The Fuzzy Newman–Girvan is an exhaustive algorithm that will serve as baseline both for the requirements and the clustering quality. For the Walktrap algorithms, the time for the two phases, namely random walking (RW) and community building (CB), are recorded separately, while for the Fuzzy Newman–Girvan case only, the total time is recorded as there is only a single phase. In addition, the last column of Table 3 lists the number of the vertices visited by the random walker.

Table 3. Performance in terms of time (sec) and vertex visits.

Algorithm	**Distribution**	**RW (s)**	**CB (s)**	**Total (s)**	**Visits**
CW	Poisson	128.1381	2000.9831	2129.1212	471,858
CW	Binomial	131.2182	2000.0304	2131.2486	470,342
CW + I	Poisson	144.4752	1911.9118	2056.3870	477,423
CW + I	Binomial	139.9916	1904.0216	2044.0132	477,216
CW + R	Poisson	157.0104	1840.0013	1997.0117	476,997
CW + R	Binomial	157.6633	1850.1003	2007.7636	476,957
CW + IR	Poisson	164.3779	2002.7745	2167.1524	477,762
CW + IR	Binomial	162.0222	1911.8664	2073.8886	477,002
MW	Poisson	157.5248	2104.9918	2262.5166	475,308
MW	Binomial	156.9924	2099.5256	2256.5180	475,121
MW + I	Poisson	165.3374	1908.6612	2073.9986	478,313
MW + I	Binomial	163.0015	1902.4319	2065.4334	478,102
MW + R	Poisson	173.9016	1850.0971	2023.9987	477,916
MW + R	Binomial	171.0017	1840.0054	2011.0071	477,831
MW + IR	Poisson	181.0013	2000.9917	2181.9930	478,514
MW + IR	Binomial	178.0017	2000.8585	2178.8602	478,333
FW	Poisson	191.9989	2425.1121	2617.1110	515,444
FW	Binomial	184.0451	2417.3376	2601.3827	514,312
FN-G	Poisson	–	–	9322.9514	–
FN-G	Binomial	–	–	9344.7778	–

Fuzzy Newman–Girvan is considerably slower than any member of the Walktrap family of algorithms. This can be attributed to the exhaustive nature of Fuzzy Newman–Girvan as well as to the extensive use of locality by the Walktrap family. Moreover, the probabilistic constraints of Markov Walktrap and Fuzzy Walktrap resulted in the acceleration of both phases of the respective algorithms, with the second order constraints yielding the lowest times in each case. Concerning the escape strategy of the random walker, the relocation option resulted in a slower walking phase but in an accelerated community building phase, with that combination being more efficient than both weight inversion and the combination of the two escape strategies. Omitting an escape strategy is not advisable. Therefore, it is not recommended to activate both escape strategies at the same time. At any rate, the Chebyshev Walktrap with relocation (CW+R) had the best overall performance tagged along in a close manner by the Markov Walktrap with relocation (MW+R). The original Fuzzy Markov being the tardiest member of the family.

An explanation for the time achieved under the relocation strategy is that the teleportation of the random walker results in cache misses, which translates to expensive fetch cycles in the memory hierarchy system. This can be seen in the last two columns of Table 3, as there is not a clear correspondence between the number of total visits and the total walking phase time. When relocation is enabled, the mean visit time is clearly higher. At any rate, the number of visits is linear in the vertex set cardinality.

In addition, the selection of h did not appear to have a significant performance impact, although in most cases the random walker was slower when h was a Poisson random variable both in terms of time and in terms of total visits. This can be attributed to the large number of high cost edges which forced the walker to bounce more times inside a community before eventually moving to another. On the other hand, the symmetric form of the binomial distribution mass function resulted in a larger number of low cost edges, facilitating the movement of the random walker and making the communities easily separable compared to the Poisson case.

The memory requirements were monitored with the Ubuntu Watch administrative tool as presented in Table 4. In contrast to other similar tools such as htop, Watch generates a text output which can be parsed and analyzed. It was periodically ran every 10 s through a bash script resulting in records of several thousand of entries each.

Table 4. Performance in terms of memory (rounded in MBs).

Algorithm	Distribution	min	max	mean	std
CW	Poisson	4128	6742	4892	322
CW	Binomial	4128	6744	4880	324
CW + I	Poisson	4128	6739	4886	335
CW + I	Binomial	4128	6744	4881	338
CW + R	Poisson	4128	8002	5113	427
CW + R	Binomial	4128	8000	5121	422
CW + IR	Poisson	4128	8001	5208	345
CW + IR	Binomial	4128	8002	5214	339
MW	Poisson	4128	6744	4800	331
MW	Binomial	4128	6740	4881	325
MW + I	Poisson	4128	6739	4879	336
MW + I	Binomial	4128	6751	4883	335
MW + R	Poisson	4128	8004	5112	428
MW + R	Binomial	4128	8002	5108	428
MW + IR	Poisson	4128	8002	5200	343
MW + IR	Binomial	4128	8000	5204	341
FW	Poisson	4128	6750	4912	281
FW	Binomial	4128	6742	4910	280
FN-G	Poisson	8192	12,402	11,012	278
FN-G	Binomial	8192	12,464	11,121	280

Fuzzy Newman–Girvan consumes more memory than any other algorithm presented in this article by far. However, it utlilizes the memory constantly and consistently, as denoted by the relatively low standard deviation. This is an important performance feature for operating systems process schedulers [46]. On the other hand, the Walktrap family exploits graph caching. This in turn translates to lower traffic between the disk and the memory, as Neo4j is not a memory-only database, as well as to fewer synchronization and serialization operations. When the relocation strategy is selected, then memory utilization has certain spikes, as it can be inferred from the increased maximum memory occupied and the increased standard deviation. This is a direct result of the random walker teleportation which temporarily annuls any scheduling optimization as well as any caching done at the software or hardware level.

5.3. Community Coherence

The following definition will facilitate further analysis of the experimental results.

Definition 7. *The (log)scree plot of a set S is the plot of the (logarithm of the) values of S versus their sorted frequency.*

Since Y does not contain ground truth communities, the communities obtained by the Fuzzy Newman–Girvan will be used as a baseline reference since their sizes are closer to a power law distribution, which is an essential feature of large, scale-free graphs. The deviation ξ of a set of numbers $\{x\}_{k=1}^{n}$ from a power law

$$f_k = \alpha_0 k^{-\gamma_0}, \quad 1 \leq k \leq n, \alpha_0 > 0, \gamma_0 > 0 \tag{41}$$

is quantified by the formula [62,63]

$$\xi = \sqrt{\frac{1}{n}\sum_{k=1}^{n}\left(\log f_k - (\log \alpha_0 - \gamma_0 \log k)\right)^2}, \tag{42}$$

where parameters α_0 and γ_0 can be estimated by, for instance, a least squares method [25]. Additionally, the estimated value of α_0 serves as a quality indicator, as it should be as close to $[2, 3]$ as possible.

The number of communities for each algorithm are shown in Table 5. Notice that this is not an absolute clustering quality metric, as typically a large number of coherent communities is preferable to a smaller number of sparse ones. Nonetheless, the introduction of the relocation strategy systematically pushes the number of communities towards the reference number, although more evidence is required for determining community coherence. This will be addressed by the two asymmetric indices of this section.

Table 5. Number of communities.

	CW	**CW + I**	**CW + R**	**CW + IR**	**MW**	**MW + I**	**MW + R**	**MW + IR**	**FW**	**FN-G**
Poisson	13,751	137,58	13,332	13,443	13,761	13,789	13,456	13,804	14,127	12,816
Binomial	18,841	18,912	16,090	17,621	18,877	18,801	17,002	18,811	18,891	15,117

In order to evaluate the clustering quality, the Kullback–Leibler divergence between the sorted sizes of the communities generated by the Fuzzy Newman–Girvan and the sorted community sizes of the remaining algorithms was computed. Recall that for two discrete distributions p_k and q_k the Kullback–Leibler divergence is defined as

$$\langle p \,||\, q \rangle \stackrel{\triangle}{=} \sum_{k=1}^{n} p_k \log\left(\frac{p_k}{q_k}\right) = \sum_{k=1}^{n} p_k \log p_k - \sum_{k=1}^{n} p_k \log q_k, \tag{43}$$

where k ranges over the union of discrete events. If p_k and q_k have no events in common, then the result is undefined. If for a single event $p_k = 0$ or $q_k = 0$, then the corresponding summand is zero. Table 6 summarizes the divergence for the Poisson and the binomial cases.

Table 6. Kullback–Leibler divergence.

	CW	CW + I	CW + R	CW + IR	MW	MW + I	MW + R	MW + IR	FW
Poisson	0.6409	0.6311	0.2766	0.4504	0.3826	0.3911	0.3281	0.4519	0.8012
Binomial	0.3885	0.3977	0.3519	0.3700	0.5804	0.5687	0.6140	0.5626	0.6748

Chebyshev Walktrap with relocation outperforms the remaining algorithms, as it has less divergence from the reference distribution.

A question at this point is whether a correspondence between the communities returned by each algorithm can be found. The asymmetric Tversky index between two sets T and V is defined as

$$\nu_{T,V} \triangleq \frac{|T \cap V|}{|T \cap V| + w_1|T \setminus V| + w_2|V \setminus T|}, \quad w_1, w_2 > 0, \tag{44}$$

and it quantifies the distance between the template set T and the variant set V. By the very definition of the index, the template set T and the variant set V are not interchangeable, namely $\nu_{T,V} \neq \nu_{V,T}$. This agrees with intuition, as it makes sense to ask how much the heuristic results differ from the ground truth community, whereas there is no point in asking the inverse question. On the contrary, with a symmetric distance metric such as, for instance, the Tanimoto similarity coefficient

$$\tau_{T,V} \triangleq \frac{|T \cap V|}{|T \cup V|} = \frac{|T \cap V|}{|T| + |V| - |T \cap V|}, \tag{45}$$

no distinction can be made between the template and the variant, which can potentially lead to misleading results.

At this point, it should be highlighted that Fuzzy Newman Girvan was executed only once since it is a deterministic algorithm.

Returning to Label (44), the case $w_1 + w_2 = 1$ is of particular interest in data mining, as it confines the coefficients on the plane which maximizes the minimum distance of T from V. Notice that algebraically this asymmetry stems from both the terms $|T \setminus V|$ and $|V \setminus T|$, which denote the number of elements of T not found in V and vice versa. Both terms signify in their own way how V is different from T. The former corresponds to the part of V which is missing from T, whereas the latter corresponds to any additions to V. As a rule, $|T \setminus V|$ is more important and, consequently, $w_1 > w_2$. As there is no standard rule for selecting w_1 and w_2, the following two schemes have been used, a linear

$$w_1 = \frac{s}{1+s} = \frac{1}{1+\frac{1}{s}}, \quad w_2 = \frac{1}{1+s}, \quad 1 \leq s \leq 5 \tag{46}$$

and an exponential

$$w_1 = \frac{e^s}{1+e^s} = \frac{1}{1+e^{-s}}, \quad w_2 = \frac{1}{1+e^s}, \quad 0 \leq s \leq 2. \tag{47}$$

Observe that in the first case $\frac{w_1}{w_2} = s$, while in the second $\frac{w_1}{w_2} = e^s$, which clearly represents a non-linear scaling of the first case. Furthermore, the second case is considerably biased in favor of $|T \setminus V|$.

Once for each possible pair of the m ground truth communities T_i $1 \leq i \leq m$ and the n estimated ones V_j $1 \leq j \leq n$ the mn Tversky indices have been computed, the similarity score $J(s)$ for a given s is computed

$$J(s) \triangleq \frac{1}{mn} \sum_{i=1}^{m} \sum_{j=1}^{n} \nu_{T_i,V_j}. \tag{48}$$

Summing over the range of s and taking the average, the mean similarity score $\bar{J}$ is obtained

$$\bar{J} \triangleq \frac{1}{|s|} \sum_{s} J(s). \tag{49}$$

The overall similarity scores are shown in Table 7.

Table 7. Tversky index.

Linear	**CW**	**CW + I**	**CW + R**	**CW + IR**	**MW**	**MW + I**	**MW + R**	**MW + IR**	**FW**
Poisson	0.6199	0.6126	0.6616	0.6012	0.5881	0.5902	0.6211	0.5577	0.2742
Binomial	0.4323	0.4153	0.6059	0.5853	0.5648	0.5702	0.5814	0.5332	0.2131
Exponential	**CW**	**CW + I**	**CW + R**	**CW + IR**	**MW**	**MW + I**	**MW + R**	**MW + IR**	**FW**
Poisson	0.5505	0.5345	0.7503	0.5462	0.5271	0.5383	0.7354	0.4811	0.1703
Binomial	0.5230	0.4841	0.6992	0.4737	0.5102	0.4890	0.6772	0.4283	0.1523

Again, Chebyshev Walktrap with relocation outperforms the remaining algorithms as it has the highest similarity with the reference communities. Note that the exponential weighting scheme sharpens the difference between the algorithms by raising the maximum scores and lowering the minimum ones.

For the experiments of the section, the termination criterion τ_0 was chosen to be a user supplied number of iterations, namely $|V| \log |V|$. This number of iterations is sufficiently large for generating communities in a reliable way. Moreover, each iteration is very quick, so the overall execution time was kept at an acceptable level despite the large number of iterations.

5.4. Relocations

Analysis is concluded with a summary regarding the relocations made by the Chebyshev Walktrap and the Markov Walktrap.

In Table 8, certain statistics regarding the random walker relocations are shown. Specifically, the first line presents the total number of relocations, whereas the second line shows the number of steps that the random walker makes before being relocated for the first time. Similarly, the last three lines contain the minimum, maximum and average number of steps between two successive relocations, respectively.

Table 8. Relocation summary.

	CW + R	**MW + R**
Number of relocations	18	14
First relocation step	101	44
min between relocations	17,812	32,991
max between relocations	27,099	64,818
mean between relocations	21,002	38,002

6. Conclusions

The primary contribution of this article is the implementation over Neo4j of Chebyshev Walktrap, a community discovery algorithm designed for edge-fuzzy graphs, a class of fuzzy graphs used among others in [10], which is based on the Random Walker algorithmic principle. Additionally, Chebyshev Walktrap relies on the competitive factors of second order statistics though the Chebyshev inequality and on an optional relocation capability in order to bound unnecessarily costly walks and, thus, remaining inside a community and being trapped for too long within the boundaries of a community, respectively. The relocation aspect was also backported to the Markov Walktrap algorithm first proposed in [15]. The effect of relocation on the community coherence was evaluated based on the asymmetric Tversky index using the Fuzzy Newman–Girvan algorithm from [15] as baseline, while its effect on the output distribution was assessed with the asymmetric Kullback–Leibler divergence. The latter was also the basis for evaluating the distance between the community size distribution generated by Fuzzy Newman–Girvan and the one computed by the Makrov Walktrap and the Chebyshev Walktrap. In these cases, the introduction of asymmetry resulted in the clear distinction between the baseline data and their variants.

The test dataset was a large synthetic Kronecker graph whose edge fuzziness was controlled either by a binomial or by a Poisson distribution. In this dataset, our performance metrics showed that Chebyshev Walktrap yields more compact communities whose sizes are more clustered. Additionally, Markov Walktrap is, in many instances, slightly faster at the expense of a somewhat bigger memory footprint.

The experimental results of Section 5 hint at some future research directions. More sophisticated from a probabilistic viewpoint, community discovery algorithms should be able to exploit the asymmetry of the edge fuzziness distribution through higher order concentration inequalities such as the Talagrand inequality, provided their computation is efficient. Moreover, new metrics for community matching, perhaps utilizing functional or semantic information should be developed. Additionally, methodologies for reliably assessing community coherence based on higher order structural or functional interactions should be sought. Finally, more experiments in larger graphs should be conducted in order to determine any inherent scalability limitations.

Author Contributions: Georgios Drakopoulos, Andreas Kanavos, and Konstantinos Tsakalidis conceived of the idea, designed and performed the experiments, analyzed the results, drafted the initial manuscript and revised the final manuscript.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Zurek, W.H. *Algorithmic Information Content, Church-Turing Thesis, Physical Entropy, and Maxwell's Demon*; Technical Report; Los Alamos National Lab.: Los Alamos, NM, USA, 1990.
2. Brillouin, L. Maxwell's demon cannot operate: Information and entropy. I. *J. Appl. Phys.* **1951**, *22*, 334–337.
3. Herrmann, D. Heron von Alexandria. In *Die antike Mathematik*; Springer: Berlin/Heidelberg, Germany, 2014; pp. 257–288.
4. Fouss, F.; Pirotte, A.; Renders, J.M.; Saerens, M. Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation. *IEEE Trans. Knowl. Data Eng.* **2007**, *19*, 355–369.
5. Sinop, A.K.; Grady, L. A seeded image segmentation framework unifying graph cuts and random walker which yields a new algorithm. In Proceedings of the 2007 IEEE 11th International Conference on Computer Vision, Rio de Janeiro, Brazil, 2007; pp. 1–8.
6. Couprie, C.; Grady, L.; Najman, L.; Talbot, H. Power watersheds: A new image segmentation framework extending graph cuts, random walker and optimal spanning forest. In Proceedings of the 2009 IEEE 12th International Conference on Computer Vision, Kyoto, Japan, 27 September–4 October 2009; pp. 731–738.
7. Blondel, V.D.; Guillaume, J.L.; Lambiotte, R.; Lefebvre, E. Fast unfolding of community hierarchies in large networks. *J. Stat. Mech. Theory Exp.* **2008**, doi:10.1088/1742-5468/2008/10/P10008.
8. Girvan, M.; Newman, M. Community Structure in Social and Biological Networks. *Proc. Natl. Acad. Sci. USA* **2002**, *99*, 7821–7826.
9. Pons, P.; Latapy, M. Computing Communities in Large Networks Using Random Walks. Available Online: https://arxiv.org/abs/physics/0512106 (accessed on 28 March 2017).
10. Drakopoulos, G.; Kanavos, A.; Makris, C.; Megalooikonomou, V. On converting community detection algorithms for fuzzy graphs in Neo4j. In Proceedings of the 5th International Workshop on Combinations of Intelligent Methods and Applications (CIMA), Vietri sul Mare, Italy, 9–11 November 2015.
11. Rosvall, M.; Bergstrom, C. *Maps of Information Flow Reveal Community Structure in Complex Networks*; Technical Report; Available online: https://arxiv.org/abs/0707.0609 (accessed on 28 March 2017).

12. Drakopoulos, G.; Kanavos, A. Tensor-based Document Retrieval over Neo4j with an Application to PubMed Mining. In Proceedings of the 7th International Conference of Information, Intelligence, Systems, and Applications (IISA 2016), Chalkidiki, Greece, 13–15 July 2016.
13. Panzarino, O.P. *Learning Cypher*; PACKT Publishing: Birmingham, UK, 2014.
14. Rosenfeld, A. Fuzzy Graphs. *Fuzzy Sets Appl.* **1975**, *513*, 77–95.
15. Drakopoulos, G.; Kanavos, A.; Tsakalidis, A. A Neo4j implementation of fuzzy random walkers. In Proceedings of the 9th Hellenic Conference on Artificial Intelligence (SETN 2016), Thessaloniki, Greece, 18–20 May 2016.
16. Robinson, I.; Webber, J.; Eifrem, E. *Graph Databases*; O'Reilly: Sebastopol, CA, USA, 2013.
17. Fortunato, S. Community Detection in Graphs. *Phys. Rep.* **2010**, *486*, 75–174.
18. Ng, A.Y.; Jordan, M.I.; Weiss, Y. On Spectral Clustering: Analysis and an algorithm. In Proceedings of the Advances in Neural Information Processing Systems (NIPS 2001), Vancouver, BC, Canada, 3–8 December 2001.
19. Kernighan, B.; Lin, S. An Efficient Heuristic Procedure for Partitioning Graphs. *Bell Syst. Tech. J.* **1970**, *49*, 291–307.
20. Shi, J.; Malik, J. Normalized Cuts and Image Segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2000**, *22*, 888–905.
21. Lancichinetti, A.; Fortunato, S. Community Detection Algorithms: A Comparative Analysis. *Phys. Rev. E* **2009**, *80*, 056117.
22. Leskovec, J.; Lang, K.J.; Mahoney, M.W. Empirical Comparison of Algorithms for Network Community Detection. In Proceedings of the 19th International Conference on World Wide Web (WWW 2010), Raleigh, NC, USA, 26–30 April 2010; pp. 631–640.
23. Carrington, P.J.; Scott, J.; Wasserman, S. *Models and Methods in Social Network Analysis*; Cambridge University Press: Cambridge, UK, 2005.
24. Scott, J. *Social Network Analysis: A Handbook*; SAGE Publications: Thousand Oaks, CA, USA, 2000.
25. Drakopoulos, G.; Kanavos, A.; Tsakalidis, A. Evaluating Twitter Influence Ranking with System Theory. In Proceedings of the 12th International Conference on Web Information Systems and Technologies (WEBIST), Rome, Italy, 23–25 April 2016.
26. Kontopoulos, S.; Drakopoulos, G. A space efficient scheme for graph representation. In Proceedings of the 26th International Conference on Tools with Artificial Intelligence (ICTAI 2014), Limassol, Cyprus, 10–12 November 2014; pp. 299–303.
27. Langville, A.; Meyer, C. *Google's PageRank and Beyond: The Science of Search Engine Rankings*; Princeton University Press: Princeton, NJ, USA, 2006.
28. Page, L.; Brin, S.; Motwani, R.; Winograd, T. *The PageRank Citation Ranking: Bringing Order to the Web*; Stanford InfoLab: Stanford, CA, USA, 1999.
29. Kleinberg, J.M. Authoritative Sources in a Hyperlinked Environment. In Proceedings of the Symposium of Discrete Algorithms (SODA), San Francisco, CA, USA, 25–27 January 1998; pp. 668–677.
30. Newman, M. *Networks: An Introduction*; Oxford University Press: Oxford, UK, 2010.
31. Newman, M.E. Fast Algorithm for Detecting Community Structure in Networks. *Phys. Rev. E* **2004**, *69*, 066133.
32. Kafeza, E.; Kanavos, A.; Makris, C.; Chiu, D. Identifying Personality-based Communities in Social Networks. In Proceedings of the Legal and Social Aspects in Web Modeling (Keynote Speech) in Conjunction with the International Conference on Conceptual Modeling (ER) (LSAWM), Hong Kong, China, 11–13 November 2013.
33. Kafeza, E.; Kanavos, A.; Makris, C.; Vikatos, P. Predicting Information Diffusion Patterns in Twitter. In Proceedings of the Artificial Intelligence Applications and Innovations (AIAI), Rhodes, Greece, 19–21 September 2014; pp. 79–89.
34. Kanavos, A.; Perikos, I. Towards Detecting Emotional Communities in Twitter. In Proceedings of the 9th IEEE International Conference on Research Challenges in Information Science (RCIS), Athens, Greece, 13–15 May 2015; pp. 524–525.
35. Kafeza, E.; Kanavos, A.; Makris, C.; Vikatos, P. T-PICE: Twitter Personality based Influential Communities Extraction System. In Proceedings of the IEEE International Congress on Big Data, Anchorage, AK, USA, 27 June–2 July 2014; pp. 212–219.

36. Zamparas, V.; Kanavos, A.; Makris, C. Real Time Analytics for Measuring User Influence on Twitter. In Proceedings of the 27th IEEE International Conference on Tools with Artificial Intelligence (ICTAI), Vietri sul Mare, Italy, 9–11 November 2015.
37. Kanavos, A.; Perikos, I.; Vikatos, P.; Hatzilygeroudis, I.; Makris, C.; Tsakalidis, A. Conversation Emotional Modeling in Social Networks. In Proceedings of the 26th IEEE International Conference on Tools with Artificial Intelligence (ICTAI), Limassol, Cyprus, 10–12 November 2014; pp. 478–484.
38. Kanavos, A.; Perikos, I.; Vikatos, P.; Hatzilygeroudis, I.; Makris, C.; Tsakalidis, A. Modeling Retweet Diffusion using Emotional Content. In Proceedings of the Artificial Intelligence Applications and Innovations (AIAI), Rhodes, Greece, 19–21 September 2014; pp. 101–110.
39. Kephart, J.O.; White, S.R. Directed-graph epidemiological models of computer viruses. In Proceedings of the 1991 IEEE Computer Society Symposium on Research in Security and Privacy, Oakland, CA, USA, 20–22 May 1991; pp. 343–359.
40. Ren, J.; Yang, X.; Yang, L.X.; Xu, Y.; Yang, F. A delayed computer virus propagation model and its dynamics. *Chaos Solitons Fractals* **2012**, *45*, 74–79.
41. Tugnait, J.K.; Luo, W. On channel estimation using superimposed training and first-order statistics. In Proceedings of the 2003 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP'03), Hong Kong, China, 6–10 April 2003; pp. 4–624.
42. Tong, L.; Xu, G.; Kailath, T. Blind identification and equalization based on second-order statistics: A time domain approach. *IEEE Trans. Inf. Theory* **1994**, *40*, 340–349.
43. Belouchrani, A.; Abed-Meraim, K.; Cardoso, J.F.; Moulines, E. A blind source separation technique using second-order statistics. *IEEE Trans. Signal Process.* **1997**, *45*, 434–444.
44. Mendel, J.M. Tutorial on higher-order statistics (spectra) in signal processing and system theory: Theoretical results and some applications. *Proc. IEEE* **1991**, *79*, 278–305.
45. Chua, K.C.; Chandran, V.; Acharya, U.R.; Lim, C.M. Application of higher order statistics (spectra) in biomedical signals—A review. *Med. Eng. Phys.* **2010**, *32*, 679–689.
46. Drakopoulos, G.; Megalooikonomou, V. An adaptive higher order scheduling policy with an application to biosignal processing. In Proceedings of the 2016 Symposium Series on Computational Intelligence (SSCI 2016), Athens, Greece, 6–9 December 2016; pp. 921–928.
47. Comon, P. Independent component analysis, a new concept? *Signal Process.* **1994**, *36*, 287–314.
48. Delorme, A.; Sejnowski, T.; Makeig, S. Enhanced detection of artifacts in EEG data using higher-order statistics and independent component analysis. *Neuroimage* **2007**, *34*, 1443–1449.
49. Priest, D.M. Algorithms for arbitrary precision floating point arithmetic. In Proceedings of the 10th IEEE Symposium on Computer Arithmetic, Grenoble, France, 26–28 June 1991; pp. 132–143.
50. Drakopoulos, G. Tensor fusion of affective Twitter metrics in Neo4j. In Proceedings of the 7th International Conference of Information, Intelligence, Systems, and Applications (IISA 2016), Chalkidiki, Greece, 13–15 July 2016.
51. Drakopoulos, G.; Megalooikonomou, V. Regularizing Large Biosignals with Finite Differences. In Proceedings of the 7th International Conference of Information, Intelligence, Systems, and Applications (IISA 2016), Chalkidiki, Greece, 13–15 July 2016.
52. Drakopoulos, G.; Kontopoulos, S.; Makris, C. Eventually consistent cardinality estimation with applications in biodata mining. In Proceedings of the 31st Annual ACM Symposium on Applied Computing, Pisa, Italy, 4–8 April 2016; pp. 941–944.
53. Hamming, R.W. On the distribution of numbers. *Bell Syst. Tech. J.* **1970**, *49*, 1609–1625.
54. Aggarwal, C.C.; Zhai, C. *Mining Text Data*; Springer Science and Business Media: Berlin/Heidelberg, Germany, 2012.
55. De Jong, K. Learning with genetic algorithms: An overview. *Mach. Learn.* **1988**, *3*, 121–138.
56. De Jong, K.A.; Spears, W.M. Using Genetic Algorithms to Solve NP-Complete Problems. In Proceedings of the ICGA, Fairfax, VA, USA, 4–7 June 1989; pp. 124–132.
57. Saad, Y.; Schultz, M.H. GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. Stat. Comput.* **1986**, *7*, 856–869.
58. Morgan, R.B. Implicitly restarted GMRES and Arnoldi methods for nonsymmetric systems of equations. *SIAM J. Matrix Anal. Appl.* **2000**, *21*, 1112–1135.

59. Leskovec, J.; Chakrabarti, D.; Kleinberg, J.; Faloutsos, C.; Ghahramani, Z. Kronecker graphs: An approach to modeling networks. *J. Mach. Learn. Res.* **2010**, *11*, 985–1042.
60. Leskovec, J.; Kleinberg, J.; Faloutsos, C. 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), Chicago, IL, USA, 21–24 August 2005; pp. 177–187.
61. Tsourakakis, C.E. Fast counting of triangles in large real networks without counting: Algorithms and laws. In Proceedings of the ICDM, Pisa, Italy, 15–19 December 2008; pp. 608–617.
62. Drakopoulos, G.; Kanavos, A.; Makris, C.; Megalooikonomou, V. Finding fuzzy communities in Neo4j. In *Smart Innovation, Systems, and Technologies*; Howlett, R.J., Jain, L.C., Eds.; Springer: Berlin/Heidelberg, Germany, 2016.
63. Drakopoulos, G.; Baroutiadi, A.; Megalooikonomou, V. Higher order graph centrality measures for Neo4j. In Proceedings of the 6th International Conference of Information, Intelligence, Systems, and Applications (IISA 2015), Corfu, Greece, 6–8 July 2015.