A Fast Algorithm for Ranking Users by their Influence in Online Social Platforms

Nouamane Arhachoui, Esteban Bautista, Maximilien Danisch, Anastasios Giovanidis

2022 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)

A Fast Algorithm for Ranking Users by their Influence in Online Social Platforms

Nouamane Arhachoui, Esteban Bautista, Maximilien Danisch, Anastasios Giovanidis
Sorbonne Université, CNRS – LIP6, F-75005 Paris, France
{nouamane.arhachoui, esteban.bautista-ruiz, anastasios.giovanidis}@lip6.fr

***Abstract*—Measuring the influence of users in social networks is key for numerous applications. A recently proposed influence metric, coined as ψ-score, allows to go beyond traditional centrality metrics, which only assess structural graph importance, by further incorporating the rich information provided by the posting and re-posting activity of users. The ψ-score is shown in fact to generalize PageRank for non-homogeneous node activity. Despite its significance, it scales poorly to large datasets; for a network of N users, it requires to solve N linear systems of equations of size N. To address this problem, this work introduces a novel scalable algorithm for the fast approximation of ψ-score, named *Power*-ψ. The proposed algorithm is based on a novel equation indicating that it suffices to solve one system of equations of size N to compute the ψ-score. Then, our algorithm exploits the fact that such a system can be recursively and distributedly approximated to any desired error. This permits the ψ-score, summarizing both structural and behavioral information for the nodes, to run as fast as PageRank. We validate the effectiveness of the proposed algorithm, which we release as an open source Python library, on several real-world datasets.**

***Index Terms*—Social Networks, Influence, Algorithms, Scalability, PageRank.**

I. Introduction

A. Context

Online Social Platforms (OSPs) have become ubiquitous in society. Their proliferation, as well as their massive number of users, has made the quantification of the influence of users in OSPs a task of utmost importance. For example, spotting the so-called influencers of a network is crucial for social scientists to better understand the dynamics that determine new trends [1] or the roots of political polarization [2]. It is also key for companies in order to develop a better marketing of their products [3]. Machine learning algorithms are another example, where identifying a reduced set of users whose features are present over the entire network is essential to reduce the number of parameters and to avoid the curse of dimensionality [4], [5].

Social platforms usually provide a user with a wall and a news-feed. In the wall, users can post messages. In the news-feed, users see messages that other users, called their leaders (or followees), have posted on their wall. A user can then re-post a message from its news-feed into its wall, allowing posts to diffuse through the network [6]. A fundamental problem is then to quantify the influence of users. Namely, the degree to which their posts spread in the network and are visible to others.

IEEE/ACM ASONAM 2022, November 10-13, 2022
978-1-6654-5661-6/22/$31.00 © 2022 IEEE

In order to address the aforementioned problem, classical centrality metrics [7] permit to assign a ranking to the nodes of the network according to their structural importance. However, such metrics are unsatisfactory to assess influence in OSPs, as they do not consider the posting and sharing activity of users in their scores. The relevance of this information is highlighted in [8], where it is shown that users with the highest number of followers (in-degree) may not necessarily be the ones that generate the highest number of retweets or impressions. This means that the in-degree, which is correlated with the popularity of a user, is not the only factor that determines influence. Similarly, the authors in [9] address the question "Are social links valid indicators of real user interaction?" and analyze a large user trace from Facebook to showcase the difference. As a result, it is necessary to go beyond pure structural metrics.

To overcome this limitation, a new metric, coined as ψ-score, has been recently introduced in [10].

The ψ-score results from an Online Social Platform model, also presented in [10], in which the diffusion of information is taken into account with activity features for each user. In the general case where users post and re-post content at a different frequency from each other, the OSP model accurately incorporates this user activity to the structural information related to the graph, thus providing a more meaningful ranking metric of user influence in OSPs. This metric was shown in [10, Theorem 5] to equal PageRank in the special case of homogeneous activity, where all users create the same amount of posts in a given time interval and share with the same rate within this time interval.

B. Goal and contributions

The problem we address in this paper is that the ψ-score calculation from [10] does not scale to large graphs since it needs the resolution of N linear systems of N equations each, N being the number of users in the network, whereas the PageRank would only require to solve one system with N unknowns.

We thus propose a new algorithm based on an equation that revamps these N system into a single one, still with N equations. Our algorithm, which accepts a recursive and distributed implementation, allows to approximate the ψ-score of the OSP with a similar speed as the traditional approximation for PageRank via the power method. Hence, the ψ-score can

be used as a more expressive alternative to PageRank in the task of ranking users by their influence.

Finally, we provide an open source software library[1] in order to allow anyone to implement the ψ-score in their own projects.

C. Related Works

Centrality measures have been systematically used in various fields of application of network science such as social networks [11], transportation networks [12], communication networks [13], biological networks [14] or even political networks [15]. The most used metrics are degree, betweenness, closeness, eigenvector centrality, and PageRank, among others [7], [16].

These metrics, which are practically used to rank nodes of a network, only consider the network topology. For example, PageRank [16] is derived through a random walk with teleportation on the studied graph, taking at each step uniform decisions about which node to consider next.

There are several existing algorithms such as Push [17] and the use of Chebyshev polynomials [18] that exploit the random walk interpretation of PageRank in order to accelerate its calculation. The Push method [17] loops on the nodes by allowing each of them to update the approximation of its own PageRank value and updates its neighbors' values. The method proposed in [18] uses the Chebyshev polynomials to approximate the PageRank vector in a more efficient way compared to the known power-method introduced with the metric [16]. Although the literature to accelerate graph eigenvalue-based measures is rich, in this paper we will need to focus in what follows on the specific structure of the ψ-score with its current algebraic limitations in order to propose an original way to make it scale as fast as PageRank does. Having established in this paper a power-iteration inspired algorithm that scales, this opens future perspectives to study other methods inspired by the Push method [17] or Chebyshev polynomials [18] to achieve even better performance.

D. Outline of the paper

The paper is structured as follows. Section II presents the recently proposed ψ-score and discusses its relation to classical centrality metrics. Section III contains our main contribution: a new algorithm for the fast approximation of the ψ-score that scales to massive datasets. We show that the algorithm runtime is comparable to PageRank, thus providing a more-diverse alternative. Section IV briefly presents our software contribution: an open source Python library for the ψ-score computation. Section V evaluates the performance of our new algorithm on several real-world networks. Conclusions and future work are discussed in Section VI.

II. The ψ-score

Recently, a new score of users' influence over their social network (platform) has been introduced in [10]. The calculation of this score involves information not only about the users' positions inside the graph structure, but additionally their posting and sharing activity. These are two extra important features that enrich previous graph-based metrics: when a user posts more frequently he/she should have a higher influence because his/her posts appear more often on the newsfeed of others; furthermore, when a user shares content rarely, very few posts (other than of his/her own creation) pass through to reach his/her followers.

[1]https://github.com/NouamaneA/psi-score

TABLE I
NOTATIONS USED IN THE PAPER

Notation	Description
ψ_i	ψ-score of user i
$\boldsymbol{\psi}$	ψ-score column-vector
$p_i^{(n)}$	impressions of i on the Newsfeed of n
$q_i^{(n)}$	(influence) impressions of i on the Wall of n
$\mathbf{p}_i$	vector with every $p_i^{(n)}$
$\mathbf{q}_i$	vector with every $q_i^{(n)}$
$\mathbf{P}$	matrix with $\mathbf{p}_i$ as column i
$\mathbf{Q}$	matrix with $\mathbf{q}_i$ as column i
$\mathbf{b}_i$	input vector of normalised posting activity
$\mathbf{d}_i$	vector with the posting ratio of a user i
$\mathbf{B}$	matrix with $\mathbf{b}_i$ as column i
$\mathbf{D}$	diagonal matrix with $\mathbf{d}_i$ as column i
$\mathbf{1}$	column-vector of $\mathbb{R}^N$ full of ones, i.e. $(1\ 1\ \cdots\ 1)^T$

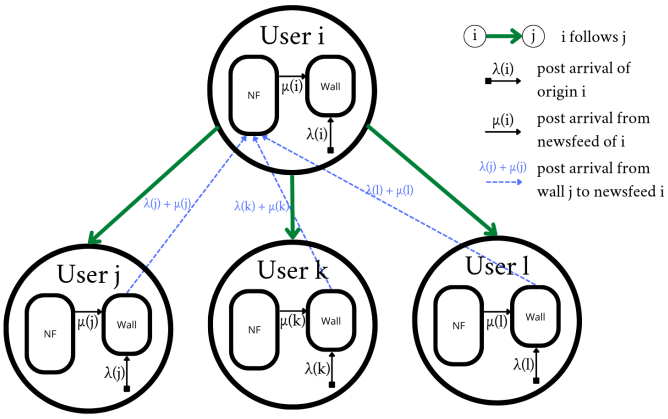

Fig. 1. A small example of the Online Social Platform from the point of view of user i.

More formally, let $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ be an unweighted and directed graph where $\mathcal{N}$ denotes the set of vertices of cardinality $|\mathcal{N}| = N$ and $\mathcal{E}$ refers to the set of edges of cardinality $|\mathcal{E}| = M$. Vertices model the users of social platforms and edges the follower-leader relations: $(i, j) \in \mathcal{E}$ encodes that user i follows user j. Let λ be the vector of posting activity over the set of vertices. That is, $\lambda^{(n)}$ is the number of posts per unit of time (posting frequency) that user n creates on his/her wall. Similarly, let μ be the vector of re-posting/sharing activity over the set of vertices. Then, $\mu^{(n)}$ refers to the frequency with which user n visits his/her news-feed and chooses one of the current entries uniformly at random to re-post it on his/her wall.

From now on in this section we will focus on posts of origin i. The authors of [10] show that the aforementioned graph and

activity features can be used to compute the following two vectors associated with the influence of user $i \in \mathcal{N}$:

- $\mathbf{p}_i = (p_i^{(1)}\ p_i^{(2)}\ \cdots\ p_i^{(N)})^T$ where, for $n \in \mathcal{N}$, $p_i^{(n)}$ is the expected percentage of posts originating from user i on the news-feed of user n.
- $\mathbf{q}_i = (q_i^{(1)}\ q_i^{(2)}\ \cdots\ q_i^{(N)})^T$ where, for all $n \in \mathcal{N}$, $q_i^{(n)}$ is the expected percentage of posts originating from user i on the wall of user n.

A central point in [10] is that $q_i^{(n)}$ can be interpreted as the influence of user i on user n, meaning that the more posts of origin i appear on the wall of n, the larger the influence that i has on n. This is used to define the influence of user i over the entire network as the average influence of i on all users. This notion is formalized by the so-called ψ-score of user i which is defined as

$$\psi_i = \frac{1}{N} \sum_{n \in \mathcal{N}} q_i^{(n)}. \tag{1}$$

The expression in (1) involves all entries of the vector $\mathbf{q}_i$ for the calculation of the influence score ψ_i of user i. However, $\mathbf{q}_i$ is not a given information and to find $q_i^{(n)}\ \forall n \in \mathcal{N}$ we need to resolve a system of equations. Specifically, each $q_i^{(n)}$ can be derived from the $p_i^{(n)}$ thanks to the following equations:

$$\text{for } n = j \neq i,\ (\lambda^{(j)} + \mu^{(j)}) q_i^{(j)} = \mu^{(j)} p_i^{(j)} \tag{2}$$

$$\text{for } n = i,\ (\lambda^{(i)} + \mu^{(i)}) q_i^{(i)} = \lambda^{(i)} + \mu^{(i)} p_i^{(i)} \tag{3}$$

Moreover, the vector $\mathbf{p}_i$ is the solution of a fixed-point problem (see [10], section III). For a post of origin i in the Newsfeed of j we have the following balance equation:

$$\sum_{k \in \mathcal{L}^{(j)}} (\lambda^{(k)} + \mu^{(k)}) p_i^{(j)} = \lambda^{(i)} \mathbb{1}_{\{i \in \mathcal{L}^{(j)}\}} + \sum_{k \in \mathcal{L}^{(j)}} \mu^{(k)} p_i^{(k)} \tag{4}$$

Since there are no self-loops and a user cannot be its own leader or follower, we have the following balance equation for user i's Newsfeed:

$$\sum_{k \in \mathcal{L}^{(i)}} (\lambda^{(k)} + \mu^{(k)}) p_i^{(i)} = \sum_{k \in \mathcal{L}^{(i)}} \mu^{(k)} p_i^{(k)} \tag{5}$$

The system of equations (2)-(3) and (4)-(5) can be written in matrix form with the following linear system:

$$\mathbf{p}_i = \mathbf{A}.\mathbf{p}_i + \mathbf{b}_i \tag{6}$$

$$\mathbf{q}_i = \mathbf{C}.\mathbf{p}_i + \mathbf{d}_i \tag{7}$$

where,

$\mathbf{A} \in \mathbb{R}_+^{N \times N}:\ a_{ji} = \frac{\mu^{(i)}}{\sum_{\ell \in \mathcal{L}^{(j)}} (\lambda^{(\ell)} + \mu^{(\ell)})} \mathbb{1}_{\{i \in \mathcal{L}^{(j)}\}}$,

$\mathbf{b}_i \in \mathbb{R}_+^N:\ b_{ji} = \frac{\lambda^{(i)}}{\sum_{\ell \in \mathcal{L}^{(j)}} (\lambda^{(\ell)} + \mu^{(\ell)})} \mathbb{1}_{\{i \in \mathcal{L}^{(j)}\}}$,

$\mathbf{C} \in \mathbb{R}_+^{N \times N}:\ c_{ji} = \frac{\mu^{(j)}}{\lambda^{(j)} + \mu^{(j)}} \mathbb{1}_{\{j=i\}}$,

$\mathbf{d}_i \in \mathbb{R}_+^N:\ d_{ji} = \frac{\lambda^{(i)}}{\lambda^{(i)} + \mu^{(i)}} \mathbb{1}_{\{j=i\}}$.

To solve the fixed-point problem of (6), we can use the following iteration (see [10], Theorem 4):

$$\mathbf{p}_i(t) = \mathbf{A}\mathbf{p}_i(t-1) + \mathbf{b}_i \tag{8}$$

Algorithm 1: `Power-NF` from [10]: Power method for the Newsfeed probabilities $\mathbf{p}_i$ related to user i.

input : Origin User i, N number of users, $N \times N$ matrix $\mathbf{A}$, vector $\mathbf{b}_i$, $\mathbf{p}$-tolerance ε
output: vector $\mathbf{p}_i$
$\mathbf{p}_i \leftarrow \mathbf{b}_i$;
$t \leftarrow 0$;
$gap \leftarrow 1$;
while $(gap > \varepsilon)$ **do**
  $\mathbf{p}_i^{old} \leftarrow \mathbf{p}_i$;
  $\mathbf{p}_i \leftarrow \mathbf{A}\mathbf{p}_i^{old} + \mathbf{b}_i$;
  $gap \leftarrow \left\|\mathbf{p}_i - \mathbf{p}_i^{old}\right\|$;
  $t \leftarrow t + 1$;
end
return $\mathbf{p}$;

where t is the current iteration. With any initialization of $\mathbf{p}_i(0)$, the $\mathbf{p}_i(t)$ converges towards the solution $\mathbf{p}_i$ of (6) as $t \to \infty$ because $\mathbf{A}$ is a sub-stochastic matrix. This method is less costly than using matrix inversion to solve the system.

To rank all users, we need to calculate ψ_i for all $i \in \mathcal{N}$, i.e. we need the vector $\boldsymbol{\psi} = (\psi_1,\ \psi_2,\ \cdots,\ \psi_N)^T \in \mathbb{R}_+^N$. Thus, the iterative algorithm in (8) has to be applied N times to solve the linear system (6) for each origin $i \in \mathcal{N}$. After that, we can map each vector $\mathbf{p}_i$ to $\mathbf{q}_i$ through (7) and finally using (1) we can derive each ψ_i.

As shown in [10, Theorem 5], in the homogeneous case when all users in the network have the same posting and re-posting activity, i.e. $\forall n \in \mathcal{N},\ \lambda^{(n)} = \lambda$ and $\mu^{(n)} = \mu$, it holds $\boldsymbol{\psi} = \boldsymbol{\pi}$ where $\boldsymbol{\pi}$ is the PageRank vector with a damping factor of $\alpha = \frac{\mu}{\lambda + \mu}$.

Problem Statement: The current computation of the ψ-score is very slow (compared e.g. to PageRank), as it requires to solve N systems of N equations each, where N is the number of users. This is especially problematic when we want to calculate the ψ-score in real networks with very large number of users.

Hence, given a directed social graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ where each node has information about its activity of posting and sharing, we aim for an algorithm that computes the ψ-score for all nodes in the graph as fast as PageRank, which is the classically used alternative.

III. Proposed Method

In this section, we present our main contribution: a distributed algorithm for the fast approximation of the ψ-score, to any given requirement of error-tolerance. To attain this goal, we first derive a new expression for the ψ-score that reduces its computational complexity from solving N linear systems of size N, to just one linear system of size N summarizing all users. Then, we show that power-iteration rules can be used

to approximate the solution of such linear system, something that can furthermore profit by distributed computations. This way, we obtain an algorithm for the derivation of the ψ-score that matches the empirical complexity of PageRank, while we benefit by the expressiveness of the score due to the incorporation of richer information.

A. Expressing with a single linear system

For the sake of notation clarity, we will express the ψ-score of the network in terms of the matrices $\mathbf{P}, \mathbf{Q}, \mathbf{B}, \mathbf{D}$, which are defined in Table I. This matrix notation allows us to express the vector of ψ-scores as

$$\boldsymbol{\psi}^T = \frac{1}{N}\mathbf{1}^T\mathbf{Q} \tag{9}$$

Thus, the problem of computing the ψ-score amounts to computing a matrix $\mathbf{Q}$, whose columns require knowledge of the vectors $\mathbf{p}_i$. Finding each $\mathbf{p}_i$ amounts to solving a linear system as shown in Eq. (6). Note in (6)-(7) that irrespective of the user origin i, all equations use the same matrices $\mathbf{A}$ and $\mathbf{C}$. The solution of each system can be expressed as (see [10, Lemma 2]):

$$\mathbf{p}_i = (\mathbf{I} - \mathbf{A})^{-1}\mathbf{b}_i = \sum_{t=0}^{\infty} \mathbf{A}^t \mathbf{b}_i \tag{10}$$

The linearity of (10) implies that the solution for all users can be written in a single matrix form as

$$\mathbf{P} = (\mathbf{I} - \mathbf{A})^{-1}\mathbf{B} = \sum_{t=0}^{\infty} \mathbf{A}^t \mathbf{B} \tag{11}$$

Then, by leveraging (7) in matrix form and the series development in (11), we can further develop Eq. (9) as follows:

$$\begin{aligned}
\boldsymbol{\psi}^T &= \frac{1}{N}\mathbf{1}^T\mathbf{Q} \\
&= \frac{1}{N}\mathbf{1}^T(\mathbf{CP} + \mathbf{D}) \\
&= \frac{1}{N}\mathbf{1}^T\left[\mathbf{C}(\mathbf{I} - \mathbf{A})^{-1}\mathbf{B} + \mathbf{D}\right] \\
&= \frac{1}{N}\mathbf{1}^T\left[\mathbf{C}\left(\sum_{t=0}^{\infty}\mathbf{A}^t\right)\mathbf{B} + \mathbf{D}\right] \\
&= \frac{1}{N}\left[\mathbf{1}^T\mathbf{C}\left(\sum_{t=0}^{\infty}\mathbf{A}^t\right)\mathbf{B} + \mathbf{1}^T\mathbf{D}\right] \\
&= \frac{1}{N}\left[\left(\sum_{t=0}^{\infty}\mathbf{1}^T\mathbf{C}\mathbf{A}^t\right)\mathbf{B} + \mathbf{1}^T\mathbf{D}\right]
\end{aligned}$$

Now, if we let $\mathbf{c}^T := \mathbf{1}^T\mathbf{C}$ and $\mathbf{d}^T := \mathbf{1}^T\mathbf{D}$, we have that $\mathbf{c}$ and $\mathbf{d}$ are vectors in $\mathbb{R}_+^N$ (the diagonals of the matrices $\mathbf{C}$ and $\mathbf{D}$). These derivations allow us to state our main theoretical result, which is the following expression for the ψ-score:

$$\boldsymbol{\psi}^T = \frac{1}{N}\left[\left(\sum_{t=0}^{\infty}\mathbf{c}^T\mathbf{A}^t\right)\mathbf{B} + \mathbf{d}^T\right] \tag{12}$$

Eq. (12) indicates that it is not necessary to solve N linear systems of size N in order to compute the ψ-score. Instead, it suffices to solve one linear system of size N which is expressed as an infinite sum of vector matrix products, and can even be calculated distributedly. The drawback is that we lose with this new expression the calculation of intermediate detailed influence quantities $\mathbf{p}_i$ and $\mathbf{q}_i$. These are not necessary to be explicitly known for the calculation of the ψ-score as (12) shows, but they do contain valuable information that could be useful for certain practical applications. Accelerating the detailed calculation of all by-products of the ψ-score is an important challenge for future work.

B. Convergence of the sum

In this subsection, we demonstrate that the infinite sum in Eq. (12) can be evaluated by power-iteration rules. The sum always converges (for the specific expression of $\mathbf{A}$) and can thus be truncated to a finite number of terms to find an approximation of the ψ-score. To show this, let us define:

$$\mathbf{s}_t^T = \sum_{\tau=0}^{t}\mathbf{c}^T\mathbf{A}^\tau, \qquad \mathbf{s} = \lim_{t\to\infty}\mathbf{s}_t, \tag{13}$$

where T denotes (matrix or vector) transpose, t is the iteration index and τ the running summation index. We stress that Eq. (13) converges if the spectral radius bound of $\mathbf{A}$ is in the open unit interval. This convergence is guaranteed by Lemmas 1 and 2 of [10] which use the fact that $\mathbf{A}$ is a sub-stochastic matrix.

C. Truncation of the sum

We now proceed to study the approximation error after truncating the sum to the first t terms. For this, we express the truncation of the sum in the following recursive way:

$$\mathbf{s}_t^T = \mathbf{s}_{t-1}^T\mathbf{A} + \mathbf{c}^T \tag{14}$$

where $\mathbf{s}_0 = \mathbf{c}$. This allows us to define the following gap parameter

$$\varepsilon_t = \left\|\mathbf{s}_t^T - \mathbf{s}_{t-1}^T\right\| \tag{15}$$

which is useful for the termination rule of the algorithm. We do not specify which norm to use in the gap formula because all the work done in this part is valid with any norm. Nevertheless, in our implementation, we choose to use the L^1-norm as it is the common way to deal with series approximation with power-iterations [16]. We stop the recursion if the trajectory towards the fixed point between two successive iterations does not change, i.e. when the following condition is satisfied:

$$\varepsilon_t \leq \varepsilon \tag{16}$$

An important question to address is therefore how the tolerance set to terminate the series summation $\mathbf{s}_t$ in (13) affects the approximation of the ψ-score in (12). To address this point, let us define $\boldsymbol{\psi}_t$ as the approximation of $\boldsymbol{\psi}$ by replacing (14) into (12) using t terms. Then, we can define the gap between two approximations of the ψ-score with different truncations as:

$$\delta_t = \left\|\boldsymbol{\psi}_t^T - \boldsymbol{\psi}_{t-1}^T\right\| \tag{17}$$

Algorithm 2: `Power-`ψ: Power iteration based algorithm for the ψ-score vector.

input : N number of users, $N \times N$ matrices $\mathbf{A}$ and $\mathbf{B}$, two vectors $\mathbf{c}$ and $\mathbf{d}$, $\mathbf{s}$-tolerance ε
output: vector $\boldsymbol{\psi}$ with the ψ-score of all users

$\mathbf{s} \leftarrow \mathbf{c}$;
$B\_norm \leftarrow \|\mathbf{B}\|$;
$t \leftarrow 0$;
$gap \leftarrow 1$;
while $(gap > \varepsilon)$ **do**
  $\mathbf{s}_{old} \leftarrow \mathbf{s}$;
  $\mathbf{s}^T \leftarrow \mathbf{s}_{old}^T\mathbf{A} + \mathbf{c}$;
  $gap \leftarrow B\_norm\,\|\mathbf{s}_{old} - \mathbf{s}\|$;
  $t \leftarrow t + 1$;
end
$\boldsymbol{\psi}^T \leftarrow \frac{1}{N}\left(\mathbf{s}^T\mathbf{B} + \mathbf{d}^T\right)$;
return $\boldsymbol{\psi}$;

Our goal is thus to express δ_t as a function of ε_t or to find an upper-bound relation between them to not only set a convergence condition for the vector $\mathbf{s}_t$ but more importantly for the vector $\boldsymbol{\psi}_t$. We derive this relationship by expressing the difference $\boldsymbol{\psi}_t - \boldsymbol{\psi}_{t-1}$ in terms of $\mathbf{s}_{t-1}$ and $\mathbf{s}_t$ as follows:

$$\begin{aligned}\boldsymbol{\psi}_t^T - \boldsymbol{\psi}_{t-1}^T &= \frac{1}{N}(\mathbf{s}_t^T\mathbf{B} + \mathbf{d}^T) - \frac{1}{N}(\mathbf{s}_{t-1}^T\mathbf{B} + \mathbf{d}^T) \\ &= \frac{1}{N}(\mathbf{s}_t^T - \mathbf{s}_{t-1}^T)\mathbf{B} \end{aligned} \tag{18}$$

Eq. (18) then allows to see that:

$$\begin{aligned}\delta_t &= \frac{1}{N}\left\|(\mathbf{s}_t^T - \mathbf{s}_{t-1}^T)\mathbf{B}\right\| \\ &\leq \frac{1}{N}\left\|\mathbf{s}_t^T - \mathbf{s}_{t-1}^T\right\|\|\mathbf{B}\| \\ \delta_t &\leq \frac{\varepsilon_t\,\|\mathbf{B}\|}{N}, \end{aligned} \tag{19}$$

With this upper bound, we can thus select a termination tolerance ε that guarantees that the ψ-score trajectory did not change more than δ. Namely, if we terminate the algorithm by satisfying the condition $\varepsilon_t\,\|\mathbf{B}\| \leq \varepsilon$, then we are certain that δ_t is lower than $\frac{\varepsilon}{N}$. We coin our algorithm for the fast approximation of the ψ-score as `Power-`ψ. It is summarized in Algorithm 2.

D. Relationship between ψ-score and PageRank

For the sake of completeness, in this subsection we discuss the relation between equation (14) in the homogeneous activity case and PageRank [16]. Then, we highlight the difference between our proposed method and the standard power-iteration algorithm for PageRank.

Let us define $\boldsymbol{\pi}$ as the PageRank vector and $\mathbf{W} = \mathbf{D}_{out}^{-1}\mathbf{L}$ the random walk transition matrix where $\mathbf{L}$ is the adjacency matrix and $\mathbf{D}_{out}$ is the diagonal matrix in which each diagonal entry (i, i) is the out-degree of node i. It was shown in [10, Theorem 5] that in the homogeneous activity case (where $\lambda^{(i)} = \lambda$ and $\mu^{(i)} = \mu$, $\forall i$) the ψ-score coincides with PageRank. This is, $\boldsymbol{\psi} = \boldsymbol{\pi}$ with $\alpha = \frac{\mu}{\lambda+\mu}$ being PageRank's damping factor.

In the case of homogeneous activity, the vectors and matrices involved in the ψ-score expression of Eq. (12) take the following form:

$$\begin{aligned}\mathbf{A} &= \alpha\mathbf{W} \\ \mathbf{B} &= (1-\alpha)\mathbf{W} \\ \mathbf{c} &= \alpha\mathbf{1} \\ \mathbf{d} &= (1-\alpha)\mathbf{1}. \end{aligned}$$

Then, after substitution in Eq. (14), it becomes:

$$\mathbf{s}_t^T = \alpha\mathbf{s}_{t-1}^T\mathbf{W} + \alpha\mathbf{1}^T \tag{20}$$

By substituting the above expression in (12) using the homogeneous values for $\mathbf{B}$ and $\mathbf{d}$ we get

$$\begin{aligned}\boldsymbol{\psi}_t^T &= \frac{1}{N}\left(\mathbf{s}_t^T(1-\alpha)\mathbf{W} + (1-\alpha)\mathbf{1}^T\right) \\ &= \frac{1}{N}\left((\alpha\mathbf{s}_{t-1}^T\mathbf{W} + \alpha\mathbf{1}^T)(1-\alpha)\mathbf{W} + (1-\alpha)\mathbf{1}^T\right) \\ &= \alpha\left(\frac{1}{N}(\mathbf{s}_{t-1}^T(1-\alpha)\mathbf{W} + (1-\alpha)\mathbf{1}^T)\right)\mathbf{W} + \\ &+ \frac{(1-\alpha)}{N}\mathbf{1}^T \\ &= \alpha\boldsymbol{\psi}_{t-1}^T\mathbf{W} + \frac{(1-\alpha)}{N}\mathbf{1}^T \end{aligned} \tag{21}$$

For comparison, the above iteration actually evolves as the power-method for PageRank

$$\boldsymbol{\pi}_t^T = \alpha\boldsymbol{\pi}_{t-1}^T\mathbf{W} + \frac{1-\alpha}{N}\mathbf{1}^T. \tag{22}$$

Notice that the proposed `Power-`ψ algorithm in the general non-homogeneous case uses power-iteration to first approximate the series $\mathbf{s}$ up to some truncation step t based on the tolerance required; after convergence it multiplies by $\mathbf{B}$ and adds $\mathbf{d}$ at the very end, to avoid unnecessary matrix vector multiplications. This is the main difference in implementation compared to the power-iteration with PageRank. As shown above, both methods essentially converge to the same value in the homogeneous case, but the critical detail is the tolerance, which in the case of ψ-score is on the $\mathbf{s}$-vector, whereas in the case of PageRank is on the PageRank-vector itself.

IV. The `psi-score` Python package

The *psi-score* project[2] is a software library that we have developed in Python. It is open source and licensed under the terms of the MIT license, allowing free usability.

This package is a practical tool for the community to easily use the ψ-score metric in Network Science projects without the need to develop each algorithm individually. Both methods mentioned in this work are accessible.

Inspired by *scikit-network* [19] and *scikit-learn* [20], the project has a similar Application Programming Interface (API) in order to be user-friendly.

[2]https://github.com/NouamaneA/psi-score

V. Numerical Evaluation

In this section, we aim to evaluate the performance of our Power-ψ algorithm (Algorithm 2). We employ the following two metrics to assess complexity: (i) the number of matrix-vector multiplications to reach a targeted tolerance; and (ii) the actual runtime of the algorithms in a machine with the following characteristics: 256GB of RAM, 4 CPUs (with 6 cores) at 3GHz. The former allows to assess and compare the complexity of distributed algorithms irrespective of the technology where the algorithms are implemented. The latter allows to get insights on the scalability of the algorithms in a real-world use case.

We run three series of experiments. The first one aims to validate the ability of Power-ψ to approximate the true ψ-score by comparing the approximation error `Power-`ψ does with the current state-of-the-art alternative. The second one aims to compare the performance of `Power-`ψ with `Power-NF` and `PageRank` in terms of required number of matrix-vector multiplications. The third experiment shows that the proposed `Power-`ψ scales to large graphs. It aims to compare the computation time for the three mentioned methods.

Each experiment has been run twice: (i) one time to compute the ψ-score with heterogeneous activity for the nodes, i.e. $\lambda^{(i)} \neq \lambda^{(j)}$ and $\mu^{(i)} \neq \mu^{(j)}$ for at least one pair of nodes i, j (in practice, the activity frequencies have been chosen uniformly at random within the interval $(0, 1)$); (ii) the second time we run the experiments to compute the ψ-score algorithm with homogeneous activity $\lambda^{(n)} = \lambda = 0.15$ and $\mu^{(n)} = \mu = 0.85$ for all n (case that reduces to PageRank as discussed in Section III-D) and compare it to the standard power-iteration for PageRank with $\alpha = \frac{\mu}{\lambda+\mu} = 0.85$ for consistency in our comparisons.

In the experiments we vary the quantity ε related to tolerance. Note here that each one of the algorithms (`Power-`ψ, `Power-NF` and `PageRank`'s power-method) have a different notion of tolerance, depending on the criterion of convergence each algorithm applies. Specifically, for `PageRank` we refer to π-tolerance $\left\|\boldsymbol{\pi}_t^T - \boldsymbol{\pi}_{t-1}^T\right\| \leq \varepsilon$ because the convergence is based on the change in PageRank values; for `Power-NF` we refer to p-tolerance $\|\mathbf{p}_i(t) - \mathbf{p}_i(t-1)\| \leq \varepsilon$ because the convergence is based on the change in the news-feed p-values; for `Power-`ψ we refer to s-tolerance $\left\|\mathbf{s}_t^T - \mathbf{s}_{t-1}^T\right\|$ because as shown above the convergence is based on the change in the series s-values. Note in the case of `Power-NF` and `Power-`ψ further processing needs to be done after convergence in order to eventually derive the resulting ψ-scores. Hence, when we set some value ε for the x-tolerance in a specific method, we are interested to evaluate the resulting relative error in the result $\boldsymbol{\psi}_\varepsilon$ of this method compared to the true value $\boldsymbol{\psi}_{true}$, defined as follows

$$error = \frac{\|\boldsymbol{\psi}_{true} - \boldsymbol{\psi}_\varepsilon\|_2}{\|\boldsymbol{\psi}_{true}\|_2} \tag{23}$$

Table II lists the datasets used for our experiment, which are available in the Konect repository [21]. Twitter and Facebook graphs, where users have posting and re-posting activity, are natural application scenarios for our method. The two citation datasets (DBLP and HepPh arXiv) are also relevant application cases for our algorithm. This is because paper publications per author can be seen as equivalent to posting activity, and paper citations by author can be seen as equivalent to reposting activity. In all datasets, only the graph topology is available, thus user activity has been generated either at random for (i) or chosen for (ii) in order to have $\frac{\mu}{\lambda+\mu} = 0.85$.

In order to be able to assess the error with respect to a true ψ-score, for experiments 1 and 2 we focus entirely on the DBLP citation network [22] that has a moderate size and it is thus realistic to exactly solve the linear system. Experiment 3 addresses the remainder of datasets from Table II.

Our implementation is available on GitHub[3] as well as the code for numerical evaluation.

TABLE II
Datasets used for evaluation

Dataset name	Type	#Nodes	#Edges	References
`DBLP`	Citation Network	12 591	49 743	[21], [22]
`Twitter`	Social Network	465 017	834 797	[21], [23]
`Facebook`	Social Network	63 731	817 035	[21], [24]
`HepPh arXiv`	Citation Network	34 546	421 578	[21], [25]

A. Experiment 1

In the first experiment, we assess the approximation error made with `Power-`ψ. We compare it with the approximation errors made with the other methods `Power-NF` and `PageRank`'s power-method. This experiment seeks to show that our proposed algorithm can approximate the ψ-score vector (also the PageRank vector in the homogeneous case) at least as well as existing methods.

We compute the true ψ-score vector: once for the heterogeneous case (i) and a second time for the homogeneous case. Then, we run each algorithm with several targeted tolerance from 10^{-9} to 10^{-1}. Finally we calculate the approximation error with equation (23) for each realization of each method.

The results are shown for heterogeneous activity (i) in Fig. 2 and for homogeneous activity (ii) in Fig. 3. We fix the same level of $\mathbf{p}$- and $\mathbf{s}$-tolerance for both algorithms `Power-NF` and `Power-`ψ as well as PageRank power iteration and derive the error from (23). We see in Fig. 2 and Fig. 3, for the same targeted tolerance, that the error made with `Power-`ψ method is lower than the approximation error made with `Power-NF` and `PageRank`, which validates the convergence of our method to the true value.

B. Experiment 2

The second experiment evaluates the number of matrix-vector multiplications that are required to compute the ψ-score and compare the methods with the same precision.

We run the proposed algorithm and the state-of-the-art algorithm presented in [10] with different tolerance parameters from 10^{-9} from 10^{-1}. We measure the number of matrix-vector multiplications done for each method as well as the

[3]https://github.com/NouamaneA/fast-algorithm-psiscore

relative error calculated with (23) in order to compare the algorithms within a given precision. Figure 4 shows the result in the case where we compute the ψ-score with (i) heterogeneous activity and Figure 5 shows the result when computing ψ-score in the specific case of (ii) homogeneous activity that gives the PageRank vector.

We see clearly that the proposed algorithm `Power-`ψ outperforms the old one `Power-NF` by several orders of magnitude. We can also observe that the difference in the number of matrix-vector multiplications compared to PageRank is very small, which renders the two methods computationally equivalent, at least for the power-iteration method.

C. Experiment 3

In the third experiment, we seek to demonstrate that our proposed algorithm scales to real-world networks with several hundreds of thousands of edges, at least as well as PageRank does. For this purpose, we chose one targeted tolerance (for space reasons) and we measured the computation time after running each method. Table III shows the results we obtained in the case where we compute the ψ-score with heterogeneous activity (i) and Table IV shows the results in the case of homogeneity (ii) when the ψ-score vector equals the PageRank vector.

Here again, the proposed `Power-`ψ outperforms by far the state-of-the-art `Power-NF`. We can also conclude that the proposed algorithm is close to PageRank's power-method in terms of computation time. In some cases, `Power-`ψ takes longer but that is because the algorithm needs to perform other operations after the convergence of the approximated series.

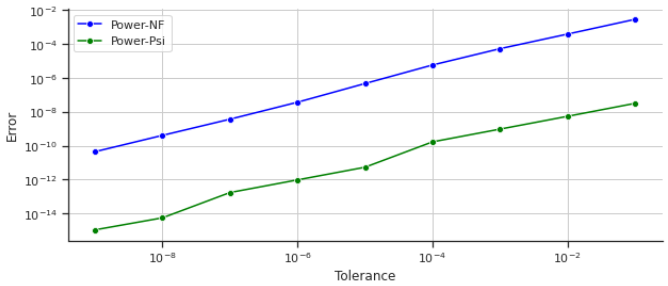

Fig. 2. Experiment 1 (i) with `DBLP`: Precision assessment of the proposed algorithm compared to the state-of-the-art method. Heterogeneous case where users do not necessarily have the same activity.

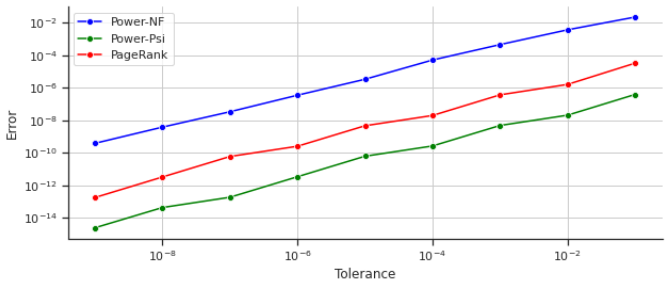

Fig. 3. Experiment 1 (ii) with `DBLP`: Precision assessment of the proposed algorithm compared to the state-of-the-art method. Homogeneous case where all users have the same activity and ψ-score=PageRank.

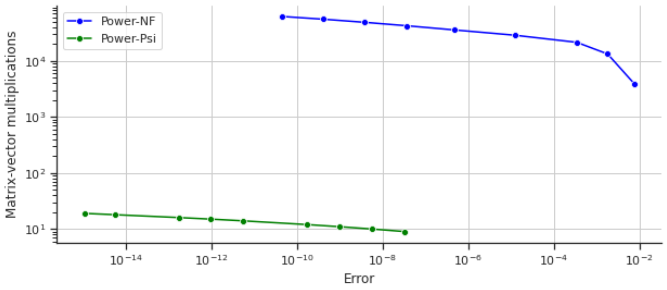

Fig. 4. Experiment 2 (i) with `DBLP`: Comparison of the proposed algorithm with the state-of-the-art method in terms of number of matrix-vector multiplications. Heterogeneous case where users do not necessarily have the same activity.

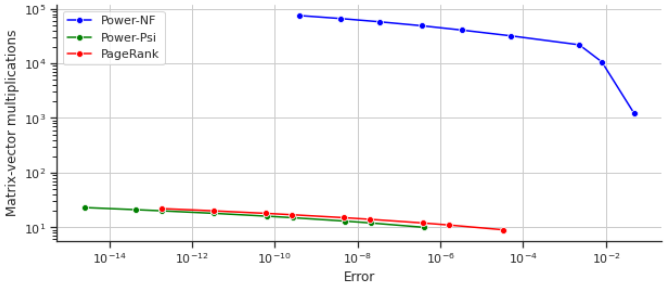

Fig. 5. Experiment 2 (ii) with `DBLP`: Comparison of the proposed algorithm with the state-of-the-art method and PageRank's power-method in terms of number of matrix-vector multiplications. Homogeneous case where all users have the same activity and ψ-score=PageRank.

TABLE III
EXPERIMENT 3 (i): COMPUTATION TIME EVALUATION WITH ε SET TO 10^{-9}. HETEROGENEOUS CASE WHERE USERS DO NOT NECESSARILY HAVE THE SAME ACTIVITY.

Dataset	Power-NF [10]	Power-ψ
`DBLP`	17.805 sec	0.029 sec
`Facebook`	1764.226 sec	0.307 sec
`Twitter`	14526.039 sec	0.634 sec
`HepPh`	272.358 sec	0.622 sec

TABLE IV
EXPERIMENT 3 (ii): COMPUTATION TIME EVALUATION WITH ε SET TO 10^{-9}. HOMOGENEOUS CASE WHERE ALL USERS HAVE THE SAME ACTIVITY AND ψ-SCORE=PAGERANK.

Dataset	PageRank [16]	Power-NF [10]	Power-ψ
`DBLP`	0.023 sec	20.775 sec	0.034 sec
`Facebook`	0.308 sec	2253.302 sec	0.454 sec
`Twitter`	0.584 sec	17411.146 sec	0.806 sec
`HepPh`	0.361 sec	360.769 sec	0.908 sec

VI. Conclusion and Future Work

We have proposed a new method to compute the ψ-score, which is influence of users in a social network (or in any network with nodes having a posting a sharing activity). We have shown that the proposed algorithm converges faster than the state-of-the-art method. The algorithm enables scalability for real-world datasets for the ψ-score.

We have also provided an open source software library that implements the existing algorithm for the ψ-score calculation as well as the novel approach.

The proposed method gives directly the influence of every user over the entire network. But in some cases, one might need to use the influence of a user on a specific user, which is giver by the OSP model but skipped by the proposed `Power-`ψ to go faster to the point. In the future, we are interested in further study on the ψ-score in order to explore the possibility to get these other information potentially valuable for some applications.

Acknowledgements

The work of N.A. is funded directly by Sorbonne University, CNRS-LIP6 laboratory by an internal project. The research of A.G. is supported by the French National Agency of Research (ANR) through the FairEngine project under Grant ANR-19-CE25-0011. E.B. is funded by the French National Agency of Research (ANR) under the FiT LabCom grant.

References

[1] D. Kempe, J. Kleinberg, and E. Tardos, "Maximizing the spread of influence through a social network," in *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '03. New York, NY, USA: Association for Computing Machinery, 2003, p. 137–146. [Online]. Available: https://doi.org/10.1145/956750.956769

[2] M. S. Alvim, B. Amorim, S. Knight, S. Quintero, and F. Valencia, "A multi-agent model for polarization under confirmation bias in social networks," in *International Conference on Formal Techniques for Distributed Objects, Components, and Systems*. Springer, 2021, pp. 22–41.

[3] K. Bok, Y. Noh, J. Lim, and J. Yoo, "Hot topic prediction considering influence and expertise in social media," *Electronic Commerce Research*, vol. 21, no. 3, pp. 671–687, 2021.

[4] C. Musto, P. Lops, M. de Gemmis, and G. Semeraro, "Context-aware graph-based recommendations exploiting personalized pagerank," *Knowl. Based Syst.*, vol. 216, p. 106806, 2021. [Online]. Available: https://doi.org/10.1016/j.knosys.2021.106806

[5] V. Cabannes, L. Pillaud-Vivien, F. Bach, and A. Rudi, "Overcoming the curse of dimensionality with laplacian regularization in semi-supervised learning," *Advances in Neural Information Processing Systems*, vol. 34, 2021.

[6] S. Goel, D. J. Watts, and D. G. Goldstein, "The structure of online diffusion networks," in *Proceedings of the 13th ACM Conference on Electronic Commerce*, ser. EC '12. New York, NY, USA: Association for Computing Machinery, 2012, p. 623–638. [Online]. Available: https://doi.org/10.1145/2229012.2229058

[7] Z. Wan, Y. Mahajan, B. W. Kang, T. J. Moore, and J.-H. Cho, "A survey on centrality metrics and their network resilience analysis," *IEEE Access*, vol. 9, pp. 104 773–104 819, 2021.

[8] M. Cha, H. Haddadi, F. Benevenuto, and K. Gummadi, "Measuring user influence in twitter: The million follower fallacy," *Proceedings of the International AAAI Conference on Web and Social Media*, vol. 4, no. 1, pp. 10–17, May 2010. [Online]. Available: https://ojs.aaai.org/index.php/ICWSM/article/view/14033

[9] C. Wilson, A. Sala, K. P. N. Puttaswamy, and B. Y. Zhao, "Beyond social graphs: User interactions in online social networks and their implications," *ACM Trans. Web*, vol. 6, no. 4, nov 2012. [Online]. Available: https://doi.org/10.1145/2382616.2382620

[10] A. Giovanidis, B. Baynat, C. Magnien, and A. Vendeville, "Ranking online social users by their influence," *IEEE/ACM Trans. Netw.*, vol. 29, no. 5, pp. 2198–2214, 2021. [Online]. Available: https://doi.org/10.1109/TNET.2021.3085201

[11] C. Correa, T. Crnovrsanin, and K.-L. Ma, "Visual reasoning about social networks using centrality sensitivity," *IEEE Transactions on Visualization and Computer Graphics*, vol. 18, no. 1, pp. 106–120, 2012.

[12] R. Puzis, Y. Altshuler, Y. Elovici, S. Bekhor, Y. Shiftan, and A. Pentland, "Augmented betweenness centrality for environmentally aware traffic monitoring in transportation networks," *J. Intell. Transp. Syst.*, vol. 17, no. 1, pp. 91–105, 2013. [Online]. Available: https://doi.org/10.1080/15472450.2012.716663

[13] A. Tizghadam and A. Leon-Garcia, "Betweenness centrality and resistance distance in communication networks," *IEEE Network*, vol. 24, no. 6, pp. 10–16, 2010.

[14] M. Ashtiani, A. Salehzadeh-Yazdi, Z. Razaghi-Moghadam, H. Hennig, O. Wolkenhauer, M. Mirzaie, and M. Jafari, "A systematic survey of centrality measures for protein-protein interaction networks," *bioRxiv*, 2017. [Online]. Available: https://www.biorxiv.org/content/early/2017/10/09/149492

[15] P. R. Miller, P. S. Bobkowski, D. Maliniak, and R. B. Rapoport, "Talking politics on facebook: Network centrality and political discussion practices in social media," *Political Research Quarterly*, vol. 68, no. 2, pp. 377–391, 2015. [Online]. Available: https://doi.org/10.1177/1065912915580135

[16] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: Bringing order to the web," 1998.

[17] J. Whang, A. Lenharth, I. Dhillon, and K. Pingali, "Scalable data-driven pagerank: Algorithms, system issues, and lessons learned," 08 2015, pp. 438–450.

[18] E. Bautista and M. Latapy, "A local updating algorithm for personalized pagerank via chebyshev polynomials," *Soc. Netw. Anal. Min.*, vol. 12, no. 1, p. 31, 2022. [Online]. Available: https://doi.org/10.1007/s13278-022-00860-5

[19] T. Bonald, N. de Lara, Q. Lutz, and B. Charpentier, "Scikit-network: Graph analysis in python," *Journal of Machine Learning Research*, vol. 21, no. 185, pp. 1–6, 2020. [Online]. Available: http://jmlr.org/papers/v21/20-412.html

[20] L. Buitinck, G. Louppe, M. Blondel, F. Pedregosa, A. Mueller, O. Grisel, V. Niculae, P. Prettenhofer, A. Gramfort, J. Grobler, R. Layton, J. VanderPlas, A. Joly, B. Holt, and G. Varoquaux, "API design for machine learning software: experiences from the scikit-learn project," in *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, 2013, pp. 108–122.

[21] J. Kunegis, "KONECT – The Koblenz Network Collection," in *Proc. Int. Conf. on World Wide Web Companion*, 2013, pp. 1343–1350. [Online]. Available: http://dl.acm.org/citation.cfm?id=2488173

[22] M. Ley, "The DBLP computer science bibliography: Evolution, research issues, perspectives," in *String Processing and Information Retrieval, 9th International Symposium, SPIRE 2002, Lisbon, Portugal, September 11-13, 2002, Proceedings*, ser. Lecture Notes in Computer Science, A. H. F. Laender and A. L. Oliveira, Eds., vol. 2476. Springer, 2002, pp. 1–10. [Online]. Available: https://doi.org/10.1007/3-540-45735-6_1

[23] M. De Choudhury, Y. Lin, H. Sundaram, K. Candan, L. Xie, and A. Kelliher, "How does the data sampling strategy impact the discovery of information diffusion in social media?" in *ICWSM 2010 - Proceedings of the 4th International AAAI Conference on Weblogs and Social Media*, ser. ICWSM 2010 - Proceedings of the 4th International AAAI Conference on Weblogs and Social Media, Dec. 2010, pp. 34–41, 4th International AAAI Conference on Weblogs and Social Media, ICWSM 2010 ; Conference date: 23-05-2010 Through 26-05-2010.

[24] B. Viswanath, A. Mislove, M. Cha, and K. P. Gummadi, "On the evolution of user interaction in facebook," in *Proceedings of the 2nd ACM Workshop on Online Social Networks*, ser. WOSN '09. New York, NY, USA: Association for Computing Machinery, 2009, p. 37–42. [Online]. Available: https://doi.org/10.1145/1592665.1592675

[25] J. Leskovec, J. Kleinberg, and C. Faloutsos, "Graph evolution: Densification and shrinking diameters," *ACM Trans. Knowl. Discov. Data*, vol. 1, no. 1, p. 2–es, mar 2007. [Online]. Available: https://doi.org/10.1145/1217299.1217301