Social Network Analysis and Mining (2022) 12:31
https://doi.org/10.1007/s13278-022-00860-5

ORIGINAL ARTICLE

A local updating algorithm for personalized PageRank via Chebyshev polynomials

Esteban Bautista[1] · **Matthieu Latapy**[1]

Received: 5 October 2021 / Revised: 3 January 2022 / Accepted: 5 January 2022 / Published online: 4 February 2022
© The Author(s), under exclusive licence to Springer-Verlag GmbH Austria, part of Springer Nature 2022

Abstract
The personalized PageRank algorithm is one of the most versatile tools for the analysis of networks. In spite of its ubiquity, maintaining personalized PageRank vectors when the underlying network constantly evolves is still a challenging task. To address this limitation, this work proposes a novel distributed algorithm to locally update personalized PageRank vectors when the graph topology changes. The proposed algorithm is based on the use of Chebyshev polynomials and a novel update equation that encompasses a large family of PageRank-based methods. In particular, the algorithm has the following advantages: (i) it has faster convergence speed than state-of-the-art alternatives for local personalized PageRank updating; and (ii) it can update the solution of recent extensions of personalized PageRank that rely on complex dynamical processes for which no updating algorithms have been developed. Experiments in a real-world temporal network of an autonomous system validate the effectiveness of the proposed algorithm.

Keywords PageRank · Updating algorithms · Local algorithms · Chebyshev polynomials · Graph signal processing · Semi-supervised learning

1 Introduction

1.1 Context

Personalized PageRank is an algorithm to assign a score to the vertices of a network, indicating their importance. The algorithm is one of the great successes in network science. It has been used in a wide amount of applications that include ranking of websites (Page et al. 1999; Ding et al. 2003; Haveliwala 2003), clustering of similar objects (Chung 2009; Andersen et al. 2006; Tabrizi et al. 2013), classifying vertices of a network (Avrachenkov et al. 2012; Merkurjev et al. 2018; Dostal et al. 2014; Avrachenkov et al. 2012), detection of anomalous events (Fontugne et al. 2019; Yoon et al. 2019; Yao et al. 2012), or recommender systems (Al Janabi and Kadiam 2019; Zhang et al. 2009; Nguyen et al. 2015), to name a few. Additionally, personalized PageRank possesses numerous properties (Langville and Meyer 2004; Ipsen and Wills 2006; Brezinski and Redivo-Zaglia 2006; Pretto 2002) that have recently inspired, in the semi-supervised learning context, a large family of PageRank generalizations that allow to tackle challenging settings such as signed graphs (Bautista et al. 2019), anomalous diffusion processes (De Nigris et al. 2017), infinite dimensional spaces (Mai and Couillet 2017), Sobolev spaces (Zhou and Belkin 2011), or time-graph dual spaces (Girault et al. 2014). However, despite the success of all these PageRank-based algorithms, they still pose problems from the algorithmic point of view. In particular, they do not adapt well to networks that evolve over time, a situation that we aim to address in this work.

1.2 Related works

While personalized PageRank is formally defined as the solution of a fixed point equation (see Section 2.1), it can be equivalently interpreted as the stationary distribution of a random walk diffusion process with restart. In this process, random walkers are placed in a subset of vertices of the network according to some initial distribution, given by the so-called personalization vector, and then are diffused according to the following rule: at each step, a random walker can either transition to an adjacent vertex with probability α or restart to its initial position with

✉ Esteban Bautista
esteban.bautista-ruiz@lip6.fr

[1] Sorbonne Université, CNRS, LIP6, F-75005 Paris, France

probability $(1-\alpha)$ (Haveliwala 2003; Page et al. 1999). This interpretation of Personalized PageRank has been exploited by numerous works to develop fast algorithms for computing PageRank (Haveliwala 1999; Kamvar et al. 2004; Fujiwara et al. 2012; Bahmani et al. 2011; Maehara et al. 2014; Avrachenkov et al. 2007; Andersen et al. 2006; Berkhin 2006). However, in cases where the graph evolves over time, such stationary distribution drifts and needs to be updated. While the algorithms above allow to fastly recompute a personalized PageRank vector, proceeding in this way becomes impractical for very large networks that constantly evolve, such as the web graph that has been reported to have 60×10^{12} nodes and evolve at a rate of 600×10^3 new pages created every second (Ohsaka et al. 2015). Additionally, several works (Ipsen and Wills 2006; Brezinski and Redivo-Zaglia 2006; Pretto 2002) have studied the impact of graph perturbations to PageRank vectors, showing that (i) the magnitude of the change to a PageRank vector is upper bounded by the size of the perturbation (magnitude of the change to the graph transition matrix); and (ii) the PageRank scores only change significantly for the nodes that are close to the perturbed area. This implies that if a perturbation is small, then it is wasteful to recompute the PageRank vector from scratch. A better alternative consists in only updating the scores of nodes close to the perturbation. In the literature, this challenge is commonly referred to as local PageRank updating. There are three reference methods. Firstly, (Bahmani et al. 2010) proposes a Monte Carlo approach where multiple random walkers are run to estimate the entries of the PageRank vector based on how frequently walkers visit nodes. Secondly, the work of (Yoon et al. 2018) exploits the random walk with restart interpretation of personalized PageRank to show that the stationary distribution of this walk can be updated by running a local diffusion process in the affected area. Then, (Yoon et al. 2018) runs this local process in a distributed fashion via the power method. Thirdly, the work of (Ohsaka et al. 2015) proposes a centralized updating algorithm based on the use of a residual and an approximation vector, sequentially pushing mass from the residual vector into the approximation vector using Gauss-Southwell update rules. While these works have been subject of deep theoretical studies (Zhang et al. 2016) and successfully applied in practice (Yoon et al. 2020), they still suffer from two main limitations: (i) they have slow convergence rates, particularly when trying to attain very small approximation errors; and (ii) they fully rely on the random walk interpretation of PageRank and thus cannot be used to update the novel generalizations of PageRank used in semi-supervised learning (Bautista et al. 2019; De Nigris et al. 2017; Mai and Couillet 2017; Zhou and Belkin 2011; Girault et al. 2014) which rely on more complex dynamical processes.

1.3 Goals, contributions and outline

In this work, we address the two limitations listed above. We propose a novel local updating algorithm based on Chebyshev polynomials. These polynomials have already been used to fastly approximate personalized PageRank vectors from scratch (Bautista Ruiz 2019), showing faster convergence speed than Gauss-Southwell and power iteration based methods (the building blocks of Ohsaka et al. (2015) and Yoon et al. (2018), respectively). However, Chebyshev polynomials have not yet been considered in the updating setting because they lack the ability to set an initial guess, which all the updating algorithms described above rely upon. Thus, the core of our proposal relies on a novel updating equation that is tailored for the large family of PageRank methods mentioned above. Concretely, it allows us to (i) use a previous personalized PageRank vector as an initial guess in the Chebyshev context; (ii) cast the updating challenge as the task of running a local diffusion process that we can efficiently compute (i.e. with just a few message exchanges) via the Chebyshev polynomials; (iii) update any of the recent generalized formulations of PageRank (Bautista et al. 2019; De Nigris et al. 2017; Mai and Couillet 2017; Zhou and Belkin 2011; Girault et al. 2014) among which classical personalized PageRank arises as a special case.

The paper is organized as follows: Section 2 sets definitions and reviews local updating methods. Section 3 presents Chebyshev polynomials and introduces the proposed algorithm. Section 4 numerically evaluates the algorithm. Section 5 concludes the work.

2 Definitions and state-of-the-art

2.1 Definitions

Let $\mathcal{G}(\mathcal{V}, \mathcal{E}, W)$ be an undirected weighted graph where W refers to the adjacency matrix, $\mathcal{E}$ is the set of edges and $\mathcal{V}$ is the set of vertices. By $D = diag(d_1, \cdots, d_N)$ we denote the diagonal matrix of degrees, where $d_i = \sum_j W_{ij}$. The matrices $L = D - W$ and $P = D^{-1}W$ refer to the combinatorial Laplacian and random walk transition matrices of $\mathcal{G}$, respectively. The notation $u \sim v$ implies that u is adjacent to v. Additionally, we denote the personalized PageRank vector of $\mathcal{G}$ by $\mathrm{pr}_\alpha(y)$. Recalling the interpretation of PageRank as a random walk process with restart, the parameter α refers to the restarting probability of the random walkers (also called teleportation parameter) and y is a probability vector with the initial distribution of the random walkers. The personalized PageRank vector is thus defined as the solution to the following fixed point equation

$$\mathrm{pr}_\alpha(y) = (1-\alpha)y + \alpha P^T \mathrm{pr}_\alpha(y). \tag{1}$$

Let $\widetilde{\mathcal{G}}(\widetilde{\mathcal{V}}, \widetilde{\mathcal{E}}, \widetilde{W})$ denote an undirected weighted graph which is an evolved or perturbed version of $\mathcal{G}$. In analogy to the definitions for $\mathcal{G}$, the matrices $\widetilde{D}$, $\widetilde{L}$ and $\widetilde{P}$ refer to the degree, Laplacian and random walk transition matrices of $\widetilde{\mathcal{G}}$, respectively. Additionally, we let $\widetilde{\mathrm{pr}}_\alpha(y)$ be the personalized PageRank vector of $\widetilde{\mathcal{G}}$ with restarting probability α and initial distribution y. In this paper, we address the problem of estimating $\widetilde{\mathrm{pr}}_\alpha(y)$ from $\mathrm{pr}_\alpha(y)$, which can be seen as the problem of updating $\mathrm{pr}_\alpha(y)$ to obtain the PageRank of the evolved network (under the same α and y). To have consistent sized matrices, we model nodes joining/leaving the network as isolated nodes that get connected/disconnected. By definition, isolated nodes correspond to zero rows and columns in the graph matrices and their degree and inverse degree are zero.

2.2 State-of-the-art approaches for local PageRank updating

In this section, we briefly review the state-of-the-art methods of Yoon et al. (2018) and Ohsaka et al. (2015) for local PageRank updating. We focus on these two approaches because they are deterministic methods that can update a PageRank vector up to any desired accuracy; hence, they directly compare to our proposed algorithm that we describe in Section 3.

Random Walk with Restart (RWR) (Yoon et al. 2018). This work is rooted in the power iteration method, which guarantees that the recursive formula

$$\widetilde{p}^{(t)} = (1-\alpha)y + \alpha \widetilde{P}^T \widetilde{p}^{(t-1)} \tag{2}$$

converges to $\widetilde{\mathrm{pr}}_\alpha(y)$ when $t \to \infty$. Since it is guaranteed that a PageRank vector does not substantially change under small perturbations, the authors of Yoon et al. (2018) exploit the fact that Eq. (2) offers the possibility to set an initial guess $\widetilde{p}^{(0)}$, which they set to $\widetilde{p}^{(0)} = \mathrm{pr}_\alpha(y)$. This procedure of using $\mathrm{pr}_\alpha(y)$ as an initial guess to compute $\widetilde{\mathrm{pr}}_\alpha(y)$ is known in the literature as a warm restart. Clearly, the warm restart sets the trajectory of the recursive equation (2) very close to $\widetilde{\mathrm{pr}}_\alpha(y)$ from the start, thus helping to drastically reduce the number of iterations needed to obtain very precise approximations. Notice however that, even though this reduces the number of iterations, the initial guess $\widetilde{p}^{(0)}$ is a dense vector (most entries are nonzero). As a result, all the vertices in the graph must be involved in the computation of the new PageRank vector, albeit only the nodes of the perturbed area must be updated. The authors of Yoon et al. (2018) address this issue by showing that (2), under a warm restart, can be rewritten as:

$$\widetilde{\mathrm{pr}}_\alpha(y) = \mathrm{pr}_\alpha(y) + \frac{1}{(1-\alpha)} \widetilde{\mathrm{pr}}_\alpha(r), \tag{3}$$

where $r = \alpha[\widetilde{P}^T - P^T]\mathrm{pr}_\alpha(y)$. Eq. (3) shows that updating an existing PageRank vector amounts to computing another PageRank vector with an initial seed r that is completely localized (only nonzero) in the 1-hop vicinity of the perturbed nodes. The vector $\widetilde{\mathrm{pr}}_\alpha(r)$ can then simply computed by setting $y = r$ and $\widetilde{p}^{(0)} = r$ in (2), showing that running the recursion for a few iterations is essentially equivalent to locally diffusing r to the affected nodes and updating their values.

Push method (Ohsaka et al. 2015). This work is rooted in the Gauss-Southwell recursive formula which uses two vectors to approximate $\widetilde{\mathrm{pr}}_\alpha(y)$: an approximation vector $\widetilde{p}$ and a residual vector $\widetilde{r}$ coding the difference between $\widetilde{\mathrm{pr}}_\alpha(y)$ and $\widetilde{p}$. The method starts with an initial guess $\widetilde{p}^{(0)}$ and then, at iteration t, transfers mass from $\widetilde{r}^{(t-1)}$ into $\widetilde{p}^{(t)}$ using a set of update equations which ensure that the following equation holds

$$\widetilde{\mathrm{pr}}_\alpha(y) = \widetilde{p}^{(t)} + \frac{1}{1-\alpha} \widetilde{\mathrm{pr}}_\alpha(\widetilde{r}^{(t)}). \tag{4}$$

Clearly, minimizing the entries of $\widetilde{r}$ implies that $\widetilde{p}$ converges to $\widetilde{\mathrm{pr}}_\alpha(y)$. To attain this, if at iteration t the largest entry of $\widetilde{r}^{(t)}$ corresponds to vertex u, then the state of the algorithm at iteration $t+1$ is determined by the following set of update equations (δ_u is the indicator vector of vertex u):

$$\widetilde{p}^{(t+1)} = \widetilde{p}^{(t)} + \widetilde{r}_u^{(t)} \delta_u. \tag{5}$$

$$\widetilde{r}^{(t+1)} = \widetilde{r}^{(t)} - \widetilde{r}_u^{(t)} \delta_u + \alpha \widetilde{P}^T \widetilde{r}_u^{(t)} \delta_u. \tag{6}$$

This procedure is repeated until all entries from the residual diminish below some user-defined threshold. The authors of Ohsaka et al. (2015) show that Eq. (4) still holds when setting $\widetilde{p}^{(0)} = \mathrm{pr}_\alpha(y)$ and $\widetilde{r}^{(0)} = \alpha[\widetilde{P}^T - P^T]\mathrm{pr}_\alpha(y)$. While $\widetilde{r}^{(0)}$ is similar to the residual of the RWR method, in the push method, it is used differently: the locality of $\widetilde{r}^{(0)}$ implies that only a few entries can surpass the tolerance threshold, meaning that only a few push operations are needed to drive them below the threshold again and obtain a good approximation of the evolved PageRank vector.

3 Proposed method

3.1 PageRank computation via Chebyshev polynomials

In this subsection, we present the Chebyshev polynomials, which are a general technique to approximate matrix functions and form the basis of our algorithm derived in Section 3.2. As discussed in the introduction, the two main drawbacks of current local PageRank updating algorithms are that (i) they are slow to converge; and (ii) they are not

adapted to update the novel generalizations of personalized PageRank which rely on more complex dynamical processes than random walks. Notably, the Chebyshev polynomials carry the potential to address these limitations because (i) they can be used to approximate the effect of general diffusion operators, thus covering the recent generalizations of PageRank in the literature; and (ii) they have been shown to converge faster than methods based on power-iteration and Gauss-Southwell rules when computing standard personalized PageRank from scratch (Bautista Ruiz 2019).

Algorithm 1 Shuman et al. (2011) Distributed application of a matrix function to a vector via Chebyshev polynomials

Input at node u: λ_{max}, y_u, μ, $\mathcal{R}_{uv}$ $\forall$ $v \sim u$, K, and $\{c_t : t = 0, \ldots, K\}$.
Output at node u: $f_u^{(K)}$
$(\bar{T}_0(\mathcal{R})y)_u = y_u$
Transmit y_u to all neighbours $v \sim u$
Receive y_v from all neighbours $v \sim u$
$(\bar{T}_1(\mathcal{R})y)_u = \sum_{v=\{v\sim u\}\cup u} \frac{1}{\phi}\mathcal{R}_{uv}y_v - y_u$
for $t = 2 : K$ **do**
Transmit $(\bar{T}_{t-1}(\mathcal{R})y)_u$ to all neighbours $v \sim u$
Receive $(\bar{T}_{t-1}(\mathcal{R})y)_v$ from all neighbors $v \sim u$
$(\bar{T}_t(\mathcal{R})y)_u = \sum_{v=\{v\sim u\}\cup u} \frac{2}{\phi}(\bar{T}_{t-1}(\mathcal{R})y)_v - 2(\bar{T}_{t-1}(\mathcal{R})y)_u - (\bar{T}_{t-2}(\mathcal{R})y)_v$
end for
Return $f_u^{(K)} = \frac{1}{2}c_0 y_u + \sum_{t=1}^{K} c_t(\bar{T}_t(\mathcal{R})y)_u$

In the context of signal processing on graphs, the Chebyshev polynomials were introduced in Shuman et al. (2011) as a mean to approximate functions of a graph matrix, achieving considerable success in the contexts of graph signal filtering (Cheng et al. 2019; Tseng and Lee 2021; Tian et al. 2014) and graph neural networks (Defferrard et al. 2016; Yan et al. 2021). They operate as follows: let $\mathcal{R}$ denote a graph matrix and $h(\mathcal{R})$ be a function of it, which is defined as the function being applied to its eigenvalues (spectrum). Then, Shuman et al. (2011) shows that $h(\mathcal{R})$ can be approximated by means of the truncated series:

$$h(\mathcal{R}) \approx \frac{1}{2}c_0 + \sum_{t=1}^{K} c_t \bar{T}_t(\mathcal{R}), \tag{7}$$

where

$$\bar{T}_t(\mathcal{R}) = \begin{cases} 1, & t = 0 \\ \frac{\mathcal{R}-\phi}{\phi}, & t = 1 \\ 2\left(\frac{\mathcal{R}-\phi}{\phi}\right)\bar{T}_{t-1} - \bar{T}_{t-2}, & t \geq 2 \end{cases} \tag{8}$$

$\phi = \lambda_{max}/2$, $c_t = \frac{2}{\pi}\int_0^{\pi} cos(t\theta)h(\phi(cos(\theta)+1))d\theta$, and λ_{max} is an upper bound on the spectrum of $\mathcal{R}$. Eq. (7) is known as the Chebyshev polynomial approximation of $h(\mathcal{R})$. The strength of the Chebyshev approximation is its coefficients c_t: they are optimal in the ∞-norm sense (Tremblay et al. 2018), meaning that for a given computational budget K they approximate $h(\mathcal{R})$ with the least ∞-norm error. In contradistinction, the coefficients of Power Iteration are ruled by the spectral gap of $\mathcal{R}$, resulting in a polynomial that requires a large K for an equally good approximation (thus, it converges slowly). While Gauss-Southwell is not a polynomial expression, its convergence speed is inversely proportional to the target error, thus being too penalizing for small errors. Another asset of (7) is that it allows to approximate the result of multiplying $h(\mathcal{R})$ with a vector y in a distributed fashion. To see this, let us recall that y can be interpreted as a signal that lives on the vertices of the graph encoded by $\mathcal{R}$. Thus, if nodes are given communication and computation capabilities, each node can compute its own value of the matrix-vector product $h(\mathcal{R})y$ by transmitting and receiving messages (with the values of y) to and from their neighbors. This distributed algorithm is detailed in Algorithm 1.

It is important to stress that even though (Bautista Ruiz 2019) shows that Chebyshev polynomials can converge to personalized PageRank vectors significantly faster than power iteration and Gauss-Southwell methods from scratch, they have not been considered in the updating scenario because they do not offer the possibility to give them an initial guess.

3.2 Local PageRank updating via Chebyshev polynomials

In this subsection, we detail our main contribution: an algorithm based on Chebyshev polynomials that allows to locally update a large family of PageRank formulations. We start by noticing that classical personalized PageRank and its novel generalizations used in semi-supervised learning (Bautista et al. 2019; De Nigris et al. 2017; Mai and Couillet 2017; Zhou and Belkin 2011; Girault et al. 2014) can all be framed under one general equation where the only difference among methods lies in the matrix encoding the graph. To show this, let us introduce the change of variable $\alpha = \frac{1}{\mu+1}$. From Eq. (1), it is easy to see that personalized PageRank can be rewritten as the solution to: $LD^{-1}\mathrm{pr}_\mu(y) + \mu\mathrm{pr}_\mu(y) = \mu y$, where $LD^{-1} = (\mathbb{I} - P^T)$ is the so-called random walk Laplacian. This expression, which is a discrete partial differential equation,

implies that classical PageRank can alternatively be interpreted as the equilibrium state of a dynamical process in which the dynamics are ruled by the operator LD^{-1}. Notably, several of the novel generalizations of personalized PageRank admit the same interpretation, allowing their solutions to be expressed in the following general form:

$$\mathcal{R}\mathrm{pr}_{\mu}(y) + \mu\mathrm{pr}_{\mu}(y) = \mu y, \tag{9}$$

where $\mathcal{R}$ denotes a generalized reference operator associated to graph $\mathcal{G}$. Thus, the only difference among several personalized PageRank generalizations is the choice of the operator $\mathcal{R}$. In Table 1, we list some of the possible choices of $\mathcal{R}$ and the methods associated to them.

Clearly, the advantage of Eq. (9) is that any algorithm that we derive based on $\mathcal{R}$ automatically covers a large family of PageRank methods. Therefore, the updating algorithm we propose in this work is an algorithm to update the solution of Eq. (9). To derive our updating algorithm, we start by noticing that the solution of Eq. (9), for an evolved graph $\widetilde{\mathcal{G}}$, can be expressed as a matrix function of $\widetilde{\mathcal{R}}$ in the following way

$$\widetilde{pr}_{\mu}(y) = \mu\left(\widetilde{\mathcal{R}} + \mu\mathbb{I}\right)^{-1} y, \tag{10}$$

where the function is thus given by

$$h(\widetilde{\mathcal{R}}) = \mu\left(\widetilde{\mathcal{R}} + \mu\mathbb{I}\right)^{-1}. \tag{11}$$

While Eq. (10) can be leveraged to efficiently compute $\widetilde{pr}_{\mu}(y)$ from scratch (for instance via the Chebyshev polynomials), it is not useful in an updating scenario because it does not allow to set an initial guess. Therefore, our first goal is to derive a recursive equation that converges to (9), which we can then use to set $\mathrm{pr}_{\mu}(y)$ as an initial guess. We stress that the natural approach of developing Eq. (10) in its geometric series is not useful for our purposes, as such approach results in a recursive expression $\widetilde{p}^{(t)} = y + \frac{1}{\mu}\widetilde{\mathcal{R}}\widetilde{p}^{(t-1)}$ that only converges to $\widetilde{\mathrm{pr}}_{\mu}(y)$ when $\mu > \lambda_{max}$. To obtain a recursive equation that converges for all $\mu > 0$, hence for all $\alpha \in (0, 1]$, we perform two changes: firstly, we transform operator $\widetilde{\mathcal{R}}$ into an equivalent operator $\widetilde{\mathcal{S}}$ that has eigenvalues normalized to the range $[-1, 1]$; secondly, we apply the same transformation to $h(\widetilde{\mathcal{R}})$ in order to find its equivalent $h(\widetilde{\mathcal{S}})$. The operator $\widetilde{\mathcal{S}}$ can be simply obtained by means of the following transformation:

$$\widetilde{\mathcal{S}} = \left(2/\lambda_{max}\right)\widetilde{\mathcal{R}} - \mathbb{I}. \tag{12}$$

Now, let us refer to the diagonal matrices of eigenvalues of $\widetilde{\mathcal{R}}$ and $\widetilde{\mathcal{S}}$ by Λ and s, respectively. Then, by expressing (11) in terms of Λ and applying the same transformation to Λ as in (12), we obtain the expression for $h(s)$, and hence $h(\widetilde{\mathcal{S}})$, as follows:

$$h(\Lambda) = \frac{\mu}{\Lambda + \mu} \tag{13}$$

$$= \frac{\mu}{\left(\lambda_{max}/2\right)(s+1) + \mu} \tag{14}$$

$$= \frac{2\mu/\lambda_{max}}{s + 1 + \left(2\mu/\lambda_{max}\right)} = h(s), \tag{15}$$

where, for the sake of clarity, we have expressed matrix inversion in the form of division. The normalized spectrum of $\widetilde{\mathcal{S}}$ allows us to then develop the geometric series of $h(\widetilde{\mathcal{S}})$ to obtain the following recursive expression that converges for all $\mu > 0$:

$$h(\widetilde{\mathcal{S}}) = \left(\frac{2\mu}{2\mu + \lambda_{max}}\right)\sum_{t=0}^{\infty}\left(-\frac{\lambda_{max}}{2\mu + \lambda_{max}}\right)^{t}\widetilde{\mathcal{S}}^{t} \tag{16}$$

Eq. (16) highlights our restriction to undirected graphs: if the spectrum is complex, then the recursion is not guaranteed to converge. We thus leave the extension of (16) to directed graphs as future work. Finally, by setting

$$\rho = \frac{2\mu}{2\mu + \lambda_{max}}, \tag{17}$$

$$\psi = -\frac{\lambda_{max}}{2\mu + \lambda_{max}}, \tag{18}$$

and applying (16) to y, we obtain the following recursive equation

$$\widetilde{p}^{(t)} = \rho y + \psi\widetilde{\mathcal{S}}\widetilde{p}^{(t-1)} \tag{19}$$

that converges to $\widetilde{\mathrm{pr}}_{\mu}(y)$ as $t \to \infty$. Clearly, Eq. (19) allows us to set $\widetilde{p}^{(0)} = \mathrm{pr}_{\mu}(y)$ and drive the trajectory of the recursion close to the exact evolved PageRank, reducing the number of iterations towards convergence.

However, Eq. (19) is not fully satisfactory. Firstly, using it to update a PageRank vector involves sending messages

Table 1 Possible choices of the reference operator $\mathcal{R}$ reported in the semi-supervised learning literature

Method	Std. PageRank (Page et al. 1999)	L^{γ}-PageRank (Bautista et al. 2019)	Iter. PageRank (Zhou and Belkin 2011)	Recentered kernel (Mai and Couillet 2017)	Time-graph dual (Girault et al. 2014)
$\mathcal{R}$	LD^{-1}	$L^{\gamma}D_{\gamma}^{-1}$	$(LD^{-1})^{m}$	$-PWP$, $P = \mathbb{I} - \frac{1}{N}\mathbb{1}\mathbb{1}^{T}$	$D^{-\sigma}LD^{\sigma-1}$

across the entire network due to the fact that $\widetilde{p}^{(0)} = \mathrm{pr}_{\mu}(y)$ is dense, even though the update mostly takes place in the perturbed graph region. Secondly, it follows power-iteration convergence speed, which is slow. To amend these issues, we extend the result of Yoon et al. (2018) in Eq. (3) to a large family of PageRank methods by means of the following Lemma. For the sake of notation clarity, we refer to the convergent state of (19) by $\widetilde{\mathrm{pr}}_{\rho,\psi}(y) = \widetilde{p}^{(\infty)}$.

Lemma 1 *Given a fixed set of coefficients ρ, ψ, and initial condition y, we have that*

$$\widetilde{\mathrm{pr}}_{\rho,\psi}(y) = \mathrm{pr}_{\rho,\psi}(y) + \frac{1}{\rho}\widetilde{\mathrm{pr}}_{\rho,\psi}(r) \tag{20}$$

where

$$r = \psi\left[\widetilde{\mathcal{S}} - \mathcal{S}\right]\mathrm{pr}_{\rho,\psi}(y) \tag{21}$$

Proof We start with a warm restart in recursion (19) as follows:

$$\widetilde{p}^{(1)} = \rho y + \psi\widetilde{\mathcal{S}}\widetilde{p}^{(0)} \tag{22}$$

$$= \rho y + \psi\widetilde{\mathcal{S}}\mathrm{pr}_{\rho,\psi}(y) \tag{23}$$

$$= \mathrm{pr}_{\rho,\psi}(y) - \psi\mathcal{S}\mathrm{pr}_{\rho,\psi}(y) + \psi\widetilde{\mathcal{S}}\mathrm{pr}_{\rho,\psi}(y) \tag{24}$$

$$= \mathrm{pr}_{\rho,\psi}(y) + \psi\left[\widetilde{\mathcal{S}} - \mathcal{S}\right]\mathrm{pr}_{\rho,\psi}(y) \tag{25}$$

$$= \mathrm{pr}_{\rho,\psi}(y) + r \tag{26}$$

Then, for the second iteration, we have

$$\widetilde{p}^{(2)} = \rho y + \psi\widetilde{\mathcal{S}}\widetilde{p}^{(1)} \tag{27}$$

$$= \rho y + \psi\widetilde{\mathcal{S}}\left(\mathrm{pr}_{\rho,\psi}(y) + r\right) \tag{28}$$

$$= \mathrm{pr}_{\rho,\psi}(y) - \psi\mathcal{S}\mathrm{pr}_{\rho,\psi}(y) + \psi\widetilde{\mathcal{S}}\left(\mathrm{pr}_{\rho,\psi}(y) + r\right) \tag{29}$$

$$= \mathrm{pr}_{\rho,\psi}(y) + \psi\left[\widetilde{\mathcal{S}} - \mathcal{S}\right]\mathrm{pr}_{\rho,\psi}(y) + \psi\widetilde{\mathcal{S}}r \tag{30}$$

$$= \rho y + \psi\widetilde{\mathcal{S}}\mathrm{pr}_{\rho,\psi}(y) + \psi\widetilde{\mathcal{S}}r \tag{31}$$

$$= \mathrm{pr}_{\rho,\psi}(y) + \psi r + \psi\widetilde{\mathcal{S}}r \tag{32}$$

By successive applications of this procedure, we have that

$$\widetilde{p}^{(\infty)} = \mathrm{pr}_{\rho,\psi}(y) + \sum_{t=0}^{\infty}\psi^t\widetilde{\mathcal{S}}^t r \tag{33}$$

$$= \mathrm{pr}_{\rho,\psi}(y) + \frac{1}{\rho}\widetilde{pr}_{\rho,\psi}(r) \tag{34}$$

□

Lemma 1 has several implications. Firstly, it states that updating a PageRank vector amounts to computing another PageRank vector with an initial distribution r that is completely localized (nonzero) in the 1-hop vicinity of the nodes that changed between $\mathcal{G}$ and $\widetilde{\mathcal{G}}$. Secondly, it shows that the size of the update is upper bounded by the size of the perturbation. This is, it shows that $\|\widetilde{\mathrm{pr}}_{\rho,\psi}(y) - \mathrm{pr}_{\rho,\psi}(y)\|$ is upper bounded by $\|\widetilde{\mathrm{pr}}_{\rho,\psi}(r)\|$ and thus by $\|\widetilde{\mathcal{S}} - \mathcal{S}\|$. Thirdly, since computing $\widetilde{\mathrm{pr}}_{\rho,\psi}(r)$ involves diffusing r through the graph, then the locality of r implies that only a few messages among the nodes in the affected region are enough update the affected nodes. Fourthly, it makes it obvious that it is not necessary to use the slow recursive equation (19) to perform the update. Instead, $\widetilde{\mathrm{pr}}_{\rho,\psi}(r)$ can be more efficiently computed by means of the Chebyshev polynomials. Therefore, our proposed algorithm consists in leveraging Eq. (20) and in approximating $\widetilde{\mathrm{pr}}_{\rho,\psi}(r)$ by means of Chebyshev polynomials. It is summarized in Algorithm 2.

Algorithm 2 Distributed local updating of general PageRank methods via Chebyshev polynomials

Input: μ, $\mathrm{pr}_{\rho,\psi}(y)$, $\mathcal{R}$, $\widetilde{\mathcal{R}}$ and ρ, ψ, ϕ.
Output: $\widetilde{\mathrm{pr}}_{\rho,\psi}(y)$.

$r = \psi\phi[\widetilde{\mathcal{R}} - \mathcal{R}]\mathrm{pr}_{\rho,\psi}(y)$
Compute $\widetilde{pr}_{\rho,\psi}(r)$ via Algorithm 1
Return $\widetilde{pr}_{\rho,\psi}(y) = \mathrm{pr}_{\rho,\psi}(y) + \widetilde{pr}_{\rho,\psi}(r)/\rho$

We conclude the section by highlighting that the generality of the Chebyshev polynomials and our derivations pave to way to explore updating algorithms for other centrality metrics, such as the recently proposed Ψ-score to measure influence in social networks (Giovanidis et al. 2021). Yet, this calls for a deep study of the spectral properties of such centrality metrics, which goes beyond the scope of this work. We thus leave the extension of our results to other centrality measures as future work.

4 Numerical evaluation

Goals. In this section, we evaluate the performance of the proposed algorithm. In particular, our goals are as follows: (*i*) to assess the performance gains obtained by the algorithm with respect to computing PageRank from scratch using the Chebyshev polynomials; (*ii*) to demonstrate that the proposed algorithm can be used to update both standard and generalized PageRank vectors; (*iii*) to assess how the performance of the algorithm degrades as perturbations grow in size; (*iv*) to compare the proposed algorithm with the state-of-the-art alternatives; and (*v*) to evaluate the performance of the algorithm in a tracking scenario where a PageRank vector needs to be updated during a long period of time.

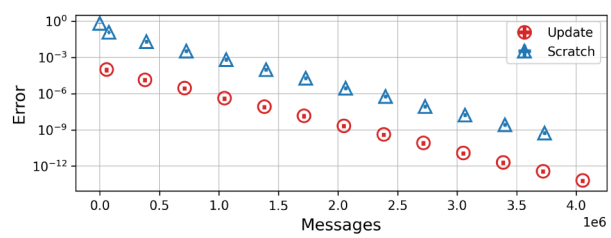
Metrics. We assess performance in terms of the number of messages that need be exchanged in order to approximate the evolved PageRank vector within a specified relative error (ℓ_2-norm sense). Since each node can update its PageRank score in constant time once it receives the neighbor's values, the complexity of our algorithm is governed by the number of messages transmitted and the time it takes to transmit and receive messages. From these variables, only the number of message exchanges is ruled by our algorithm since the time it takes to transmit and receive messages is dependent of the technology where the algorithm is implemented. We thus consider the number of message exchanges a better metric that the routine's running time.

Data. We perform our experiments in the Tech-AS-Topology temporal network (Rossi and Ahmed 2015), which is a real-world network from an autonomous system with 34.8K nodes and 171.4K edges organized in 32.8K graph snapshots. This network contains both sparse and dense regions, meaning that perturbations can affect small or large regions. For the experiments, we pre-process the data by turning the snapshots into undirected graphs, resulting in a total of 215.4K timestamped edges. The first graph snapshot in the sequence contains 32K nodes and 111.6K edges. Then, in successive snapshots, new edges adhere into the network branching nodes already present or new nodes joining the graph. The dataset only contains edge additions; therefore, we simulate an edge deletion setting (see Experiment 4) by reversing the time axis: we consider the originally last snapshot as the new first one and the originally first snapshot as the new last one. From this perspective, a new snapshot causes edges to disappear or nodes to leave.

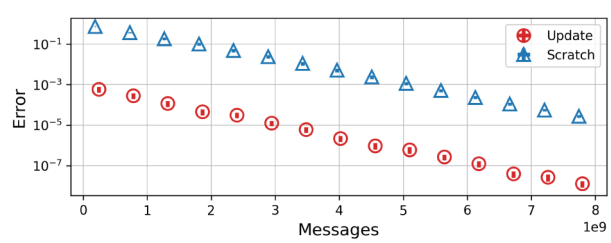
4.1 Experiment 1

In our first experiment, we address goals (*i*) and (*ii*). For this, we fix a small graph perturbation. Then, as we vary the allowed number of messages, we measure how well Algorithm 2 approximates the true PageRank of the evolved graph. For comparison purposes, we apply the same test to a computation from scratch using Algorithm 1. To show that the proposed algorithm can update generalized PageRank propositions as well as standard PageRank, we employ it to update standard PageRank vectors and the recently proposed L^γ-PageRank vectors from Bautista et al. (2019). For this experiment, we use the first snapshot from the Tech-AS-Topology network as initial graph. Then, we use the second snapshot of the network as the perturbation: it contains 120 new edges and 1 new node joining the graph. We choose a vertex at random and use its indicator function as the initial distribution y (a common setting in local graph clustering). We use $\alpha = 0.5$, measure relative error in the ℓ_2 sense, and repeat the experiment for 20 realizations of y.

Results of Experiment 1 are displayed in Figure 1. Figure 1a shows the result of updating standard PageRank, while Figure 1b depicts the result of updating the generalized L^γ-PageRank (Bautista et al. 2019). They show that the proposed algorithm successfully updates both standard and generalized PageRank vectors. In both cases, the proposed algorithm offers significant approximation improvements compared to computing from scratch. We additionally verify that our updating algorithm converges at the same rate than the Chebyshev polynomials from scratch. This amends a tradeoff that needs to be made with current updating algorithms: they are a good option if only few iterations are



(a) Standard PageRank



(b) Generalized PageRank (L^γ-PageRank)

Fig. 1 Experiment 1. Performance of proposed algorithm vs a computation from scratch. We assess approximation error as larger computational budgets are given to the algorithms. We update both standard PageRank and a Generalized PageRank used in semi-supervised learning (Bautista et al. 2019)

allowed but their slow convergence makes them worse than Chebyshev polynomials from scratch if several iterations are needed (Bautista Ruiz 2019). Lastly, we notice that the error bars (standard error) are negligible, indicating that the performance of the algorithm is irrespective of the choice of initial seed y and, consequently, of a particular PageRank vector to update.

4.2 Experiment 2

In our second experiment, we address goal (*iii*). For this, we fix a target approximation error. Then, as we vary the perturbation size, we measure the number of messages required by Algorithm 2 to approximate the true PageRank of the evolved graph within the specified error. For comparison purposes, we apply the same test to a computation from scratch using Algorithm 1. Since larger perturbations imply larger updates, our proposed algorithm should be highly sensitive to the size of perturbations. On the other hand, a computation from scratch should only augment messages proportionally to $|\widetilde{\mathcal{E}}| - |\mathcal{E}|$. Therefore, we aim to empirically spot the point where the update needed is so large that our updating algorithm offers no benefit over a computation from scratch. For this experiment, we use the first snapshot from the Tech-AS-Topology network as initial graph. Then, we control the size of the perturbation by aggregating an increasingly larger number of subsequent snapshots. We set y as the indicator function of a random vertex and update its associated standard PageRank vector. We use $\alpha = 0.5$ and set the error at 1×10^{-13}.

Results of Experiment 2 are displayed in Figure 2. For small perturbations, the number of messages needed by our algorithm to attain the desired error is small. This number increases as the perturbation grows in size, reaching a point where the updating algorithm does not provide any advantage with respect to a computation from scratch. For the Tech-AS-Topology network, this operational limit occurs for a perturbation of around 4000 new edges, which corresponds to roughly 3% of the edges from the initial graph. This confirms that our updating algorithm should preferably be used when the changes in the graph are small.

4.3 Experiment 3

In our next third experiment, we address goal (*iv*). For this, we fix a graph perturbation. Then, as we vary the target approximation error, we measure the number of message exchanges our proposed Algorithm 2 and the state-of-the-art alternatives (Yoon et al. 2018; Ohsaka et al. 2015) require to approximate the true PageRank of the evolved graph within the specified error. We stress that our algorithm and the RWR one (Yoon et al. 2018) are distributed and can thus be assessed in terms of transmitted messages. Yet, the push algorithm (Ohsaka et al. 2015) is centralized and normally studied in terms of push operations rather than messages. We notice that the complexity of each push operation is dominated by a matrix-vector multiplication that can be interpreted as transmitted messages; thus, we track this quantity for the Push method. For this experiment, we use the first snapshot from the Tech-AS-Topology network as initial graph. Then, we use the second snapshot of the network as the perturbation. We set y as the indicator function of a random vertex and update its associated standard PageRank vector. We use $\alpha = 0.5$.

Results of Experiment 3 are displayed in Fig. 3. The proposed algorithm outperforms state-of-the-art alternatives for local PageRank updating, being able to attain any desired relative approximation error with significantly less messages. Indeed, for approximation errors in the order of 10^{-14}, which are the best approximations we can obtain using Python's float64 data types, the proposed Algorithm requires roughly 35% less messages than the second best method of RWR. We notice that the push algorithm is not competitive for very precise approximations, as the number of message operations it requires quickly becomes large.

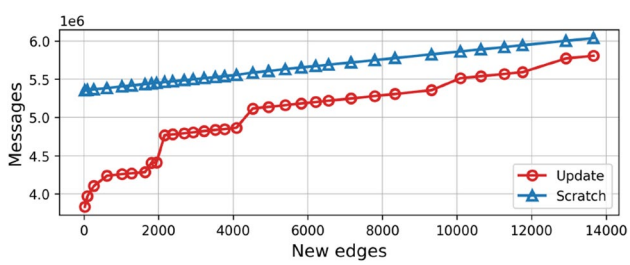


Fig. 2 Experiment 2. Sensitivity of proposed algorithm to perturbations. We assess the number of messages required to update within an error of 1×10^{-13} as the number of added edges grows. The alternative from scratch (same conditions) is used as reference

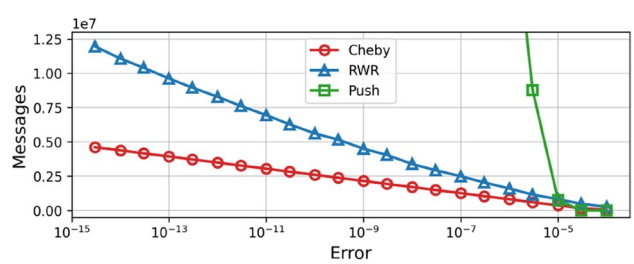
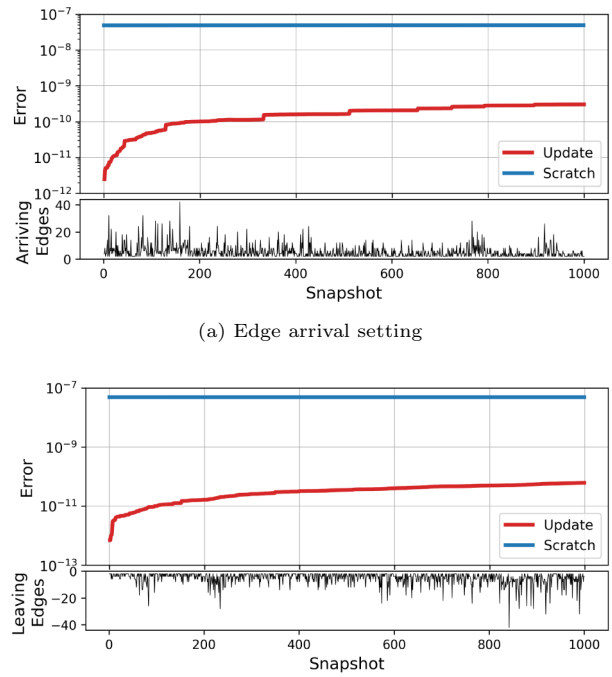


Fig. 3 Experiment 3. Comparison of proposed algorithm with state-of-the-art alternatives. We assess the number of messages required to update a PageRank vector within an increasingly smaller approximation error

4.4 Experiment 4

In our fourth experiment, we address goal (v). For this, we fix a number of communication rounds (K). Then, as new graph snapshots arrive, we estimate the PageRank vector of the current snapshot by updating the PageRank vector estimated for the previous snapshot using Algorithm 2. To demonstrate that our algorithm addresses equally edge additions and deletions, we run the experiment in both settings. Since the data only contain edge additions, we simulate deletions by reversing the time axis, meaning that we start from the evolved network and run backwards to the primitive one. For comparison purposes, we estimate the exact PageRank of the current snapshot via a computation from scratch using Algorithm 1 under the same number of communication rounds (K). We stress that this is an extremely challenging task for the updating algorithm because the vector to update is no longer the exact PageRank vector of the previous snapshot but an approximation of it, thus meaning that errors accumulate over time. Therefore, we aim to empirically spot if our method can maintain a PageRank vector for a long time or if it soon becomes worse than the approximation obtained by the method from scratch. For this experiment, we use the aggregated first 100 snapshots from the Tech-AS-Topology network as initial graph. Then, we track the standard PageRank vector during the following 1000 snapshots (reverse for edge removals). The only exact PageRank vector is given to the initial graph. We set y as the indicator function of a random vertex, $K = 15$ and $\alpha = 0.5$.

Results of Experiment 4 are displayed in Figure 4. The upper panel shows the relative error between the tracked vectors and the true PageRank of each snapshot when edge additions are considered, while the bottom panel depicts the same quantities for the case of edge deletions. The size of the perturbation in each new snapshot is shown as additional information in both panels. In both cases, the proposed algorithm is able to effectively track the PageRank vector during the entire time horizon. For early times, the updating algorithm returns extremely precise approximations: up to five orders of magnitude improvement with respect to a computation from scratch with the same computational budget. Then, we notice that errors steadily accumulate. However, it is at a sufficiently slow rate that, during the 1000 snapshots, the tracked vector is at least two orders of magnitude closer to the exact PageRank than the alternative from scratch.



(a) Edge arrival setting

(b) Edge removal setting

Fig. 4 Experiment 4. Performance of proposed algorithm in a tracking setting. We update a PageRank vector during a long period of time using $K = 15$ fixed communication rounds. We measure error with respect to the exact PageRank of each snapshot. The alternative from scratch (same conditions) is used as reference. We consider both edge arrival and edge removal settings

5 Conclusion

We proposed a Chebyshev polynomial-based distributed algorithm to locally update personalized PageRank vectors. We showed that the proposed algorithm has faster convergence than state-of-the-art alternatives, bringing us closer to the goal of effortlessly maintaining PageRank vectors in real world networks. Additionally, the algorithm can be used to update more general formulations of PageRank recently proposed in the context of semi-supervised learning. These improvements were possible due to a novel updating equation that encompasses a family of PageRank formulations and that makes it direct to employ Chebyshev polynomials to locally solve the updating challenge. Numerical evaluations showed that the proposed algorithm is an effective tool for tracking PageRank vectors for a long period of time when changes in the graph are small. An interesting prospective work would be to extend these results to undirected graphs that have non-real eigenvalues and to the case in which the PageRank initial distribution and teleportation parameters also evolve over time.

Acknowledgements The authors would like to thank P. Gonçalves and P. Abry for helpful discussions.

Funding Information This work is funded in part by the ANR (French National Agency of Research) under the Limass (ANR-19-CE23-0010) and FiT LabCom grants.

Data availability The datasets generated during and/or analyzed during the current study are available in the Network Data Repository, [https://

networkrepository.com/tech-as-topology.php]. The code to replicate the results is available at https://github.com/estbautista/PageRank_Updating_Chebyshev_Paper.

Declarations

Conflict of interest The authors have no conflicts of interest to declare that are relevant to the content of this article.

References

Page L, Brin S, Motwani R, Winograd T (1999) The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab

Ding C, He X, Husbands P, Zha H, Simon H (2003) Pagerank, hits and a unified framework for link analysis. In: Proceedings of the 2003 SIAM International Conference on Data Mining, pp. 249–253. SIAM

Haveliwala TH (2003) Topic-sensitive pagerank: A context-sensitive ranking algorithm for web search. IEEE Trans Knowl Data Eng 15(4):784–796

Chung F (2009) A local graph partitioning algorithm using heat kernel pagerank. Internet Math 6(3):315–330

Andersen R, Chung F, Lang K (2006) Local graph partitioning using pagerank vectors. In: 2006 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06), pp. 475–486. IEEE

Tabrizi SA, Shakery A, Asadpour M, Abbasi M, Tavallaie MA (2013) Personalized pagerank clustering: A graph clustering algorithm based on random walks. Physica A: Stat Mech Appl 392(22):5772–5785

Avrachenkov K, Gonçalves P, Legout A, Sokol M (2012) Classification of content and users in bittorrent by semi-supervised learning methods. In: 2012 8th International Wireless Communications and Mobile Computing Conference (IWCMC), pp. 625–630. IEEE

Merkurjev E, Bertozzi AL, Chung F (2018) A semi-supervised heat kernel pagerank mbo algorithm for data classification. Commun Math Sci 16(5):1241–1265

Dostal M, Nykl M, Ježek K (2014) Exploration of document classification with linked data and pagerank. In: Intelligent Distributed Computing VII, pp. 37–43. Springer

Avrachenkov K, Mishenin A, Gonçalves P, Sokol M (2012) Generalized optimization framework for graph-based semi-supervised learning. In: Proceedings of the 2012 SIAM International Conference on Data Mining, pp. 966–974. SIAM

Fontugne R, Bautista E, Petrie C, Nomura Y, Abry P, Gonçalves P, Fukuda K, Aben E (2019) Bgp zombies: An analysis of beacons stuck routes. In: International Conference on Passive and Active Network Measurement, pp. 197–209. Springer

Yoon M, Hooi B, Shin K, Faloutsos C (2019) Fast and accurate anomaly detection in dynamic graphs with a two-pronged approach. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 647–657

Yao Z, Mark P, Rabbat M (2012) Anomaly detection using proximity graph and pagerank algorithm. IEEE Trans Inf Forens Secur 7(4):1288–1300

Al_Janabi S, Kadiam, N (2019)Recommendation system of big data based on pagerank clustering algorithm. In: International Conference on Big Data and Networks Technologies, pp. 149–171. Springer

Zhang Y, Zhang N, Tang J (2009) A collaborative filtering tag recommendation system based on graph. ECML PKDD discovery challenge, 297–306

Nguyen P, Tomeo P, Di Noia T, Di Sciascio E (2015) An evaluation of simrank and personalized pagerank to build a recommender system for the web of data. In: Proceedings of the 24th International Conference on World Wide Web, pp. 1477–1482

Langville AN, Meyer CD (2004) Deeper inside pagerank. Internet Math 1(3):335–380

Ipsen IC, Wills RS (2006) Mathematical properties and analysis of google's pagerank. Bol Soc Esp Mat Apl 34:191–196

Brezinski C, Redivo-Zaglia M (2006) The pagerank vector: properties, computation, approximation, and acceleration. SIAM J Matrix Analy Appl 28(2):551–575

Pretto L (2002) A theoretical analysis of google's pagerank. In: International Symposium on String Processing and Information Retrieval, pp. 131–144. Springer

Bautista E, Abry P, Gonçalves P (2019) L^{γ}-pagerank for semi-supervised learning. Appl Netw Sci 4(1):1–20

De Nigris S, Bautista E, Abry P, Avrachenkov K, Gonçalves P (2017) Fractional graph-based semi-supervised learning. In: 2017 25th European Signal Processing Conference (EUSIPCO), pp. 356–360. IEEE

Mai X, Couillet R (2017) The counterintuitive mechanism of graph-based semi-supervised learning in the big data regime. In: 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 2821–2825. IEEE

Zhou X, Belkin M (2011) Semi-supervised learning by higher order regularization. In: Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, pp. 892–900. JMLR Workshop and Conference Proceedings

Girault B, Gonçalves P, Fleury E, Mor AS (2014) Semi-supervised learning for graph to signal mapping: A graph signal wiener filter interpretation. In: 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 1115–1119. IEEE

Haveliwala T (1999) Ecient computation of pagerank. Technical report, Stanford

Kamvar S, Haveliwala T, Golub G (2004) Adaptive methods for the computation of pagerank. Linear Algebra Appl 386:51–65

Fujiwara Y, Nakatsuji M, Yamamuro T, Shiokawa H, Onizuka M (2012) Efficient personalized pagerank with accuracy assurance. In: Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 15–23

Bahmani B, Chakrabarti K, Xin D (2011) Fast personalized pagerank on mapreduce. In: Proceedings of the 2011 ACM SIGMOD International Conference on Management of Data, pp. 973–984

Maehara T, Akiba T, Iwata Y, Kawarabayashi K-i (2014) Computing personalized pagerank quickly by exploiting graph structures. Proc VLDB Endowment 7(12):1023–1034

Avrachenkov K, Litvak N, Nemirovsky D, Osipova N (2007) Monte carlo methods in pagerank computation: When one iteration is sufficient. SIAM J Numer Analy 45(2):890–904

Berkhin P (2006) Bookmark-coloring algorithm for personalized pagerank computing. Internet Math 3(1):41–62

Ohsaka N, Maehara T, Kawarabayashi K-i (2015) Efficient pagerank tracking in evolving networks. In: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 875–884

Bahmani B, Chowdhury A, Goel A (2010) Fast incremental and personalized pagerank. Proc. VLDB Endow. 4(3):173–184. https://doi.org/10.14778/1929861.1929864

Yoon M, Jin W, Kang U (2018) Fast and accurate random walk with restart on dynamic graphs with guarantees. In: Proceedings of the 2018 World Wide Web Conference, pp. 409–418

Zhang H, Lofgren P, Goel A (2016) Approximate personalized pagerank on dynamic graphs. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1315–1324

Yoon M, Gervet T, Hooi B, Faloutsos C (2020) Autonomous graph mining algorithm search with best speed/accuracy trade-off. In: 2020 IEEE International Conference on Data Mining (ICDM), 751–760

Bautista Ruiz E (2019) Laplacian Powers for Graph-Based Semi-Supervised Learning. Theses, Université de Lyon. https://tel.archives-ouvertes.fr/tel-02476246

Shuman DI, Vandergheynst P, Frossard P (2011) Chebyshev polynomial approximation for distributed signal processing. In: 2011 International Conference on Distributed Computing in Sensor Systems and Workshops (DCOSS), pp. 1–8. IEEE

Cheng C, Jiang J, Emirov N, Sun Q (2019) Iterative chebyshev polynomial algorithm for signal denoising on graphs. In: 2019 13th International Conference on Sampling Theory and Applications (SampTA), pp. 1–5. IEEE

Tseng C-C, Lee S-L (2021) Minimax design of graph filter using chebyshev polynomial approximation. IEEE Trans Circ Syst II: Express Briefs 68(5):1630–1634

Tian D, Mansour H, Knyazev A, Vetro A (2014) Chebyshev and conjugate gradient filters for graph image denoising. In: 2014 IEEE International Conference on Multimedia and Expo Workshops (ICMEW), pp. 1–6. IEEE

Defferrard M, Bresson X, Vandergheynst P (2016) Convolutional neural networks on graphs with fast localized spectral filtering. Adv Neural Inf Process Syst 29:3844–3852

Yan B, Wang G, Yu J, Jin X, Zhang H (2021) Spatial-temporal chebyshev graph neural network for traffic flow prediction in iot-based its. IEEE Internet Things J

Tremblay N, Gonçalves P, Borgnat P (2018) Design of graph filters and filterbanks. In: Cooperative and Graph Signal Processing, pp. 299–324. Elsevier

Giovanidis A, Baynat B, Magnien C, Vendeville A (2021) Ranking online social users by their influence. IEEE/ACM Transactions on Networking

Rossi RA, Ahmed NK (2015) The network data repository with interactive graph analytics and visualization. In: AAAI. http://networkrepository.com

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.