Ensemble Enabled Scholarly Article Ranking

Shuai Ma, Chen Gong, Renjun Hu, Dongsheng Luo, Chunming Hu, Jinpeng Huai

SKLSDE Lab, Beihang University, China
Beijing Advanced Innovation Center for Big Data and Brain Computing, China
{mashuai, gongchen, hurenjun, lds1995, hucm, huaijp}@buaa.edu.cn

ABSTRACT

Evaluating the query independent importance of scholarly articles is a critical and challenging task, due to the heterogeneous, evolving and dynamic nature of entities involved in scholarly articles. To do this, we first propose an ensemble enabled approach by Time-Weighted PageRank, which extends traditional PageRank with a time decaying factor, and assembling the importance of heterogeneous entities involved in scholarly articles. We then propose both batch and incremental algorithms for efficient ranking computation. We finally experimentally show that our approach is both efficient and effective for ranking scholarly articles.

1. INTRODUCTION

Query independent ranking of scholarly articles has drawn significant attentions from both academia [9, 13, 16, 19, 25, 27, 30, 35] and industry [2, 3, 26]. Generally speaking, a ranking is *a function that assigns each entity a numerical score*. Query independent ranking aims to give a static ranking based on the scholarly article data only, and is independent of how well articles match a specific query. Such a ranking plays a key role in literature recommendation systems, especially in the *cold start* scenario.

Scholarly articles are involved with multiple entities such as authors, venues, dates and references. Hence, scholarly article ranking is essentially a problem of assessing the importance of nodes in a heterogeneous network. However, effective and efficient ranking of nodes in such a large complex network is a challenging task since entities are heterogeneous, evolving and dynamic [7, 20].

First, even if we are only to rank one type of entities (*e.g.,* scholarly articles), other types of entities (*e.g.,* venues and authors) are closely involved, and, moreover, different types of entities may have different impacts on the ranking of scholarly articles. Second, entities are evolving, and the importance of an article varies with time [18]. Newly published articles are very likely to have increasing impacts in the next few years, and those published many years ago tend to have decreasing impacts, as researchers potentially have more interests in recent reported results. As indicated by the statistics of the Microsoft Academic Graph (MAG) [26] in Fig. 1, the citations of articles indeed increase in the first two years after their publication, and then decrease accordingly. Finally, academic data is dynamic and continuously growing. Indeed, the number of articles in MAG has exceeded 126 million, and keeps increasing at around 5.7 million per year [26].

Query independent ranking of scholarly articles is challenging [5], although there exists quite a few work on scholarly article ranking, *e.g.,* [9, 13, 16, 19, 25, 30, 35]. Further, to our knowledge, little concern has been paid to dynamic scholarly article ranking except [14] with a strong and impractical assumption that there are no citations between papers published in the same year.

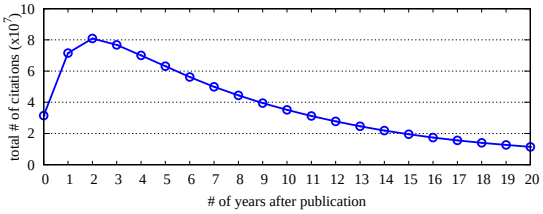


Figure 1: Citation statistics of scholarly articles

Contributions & Organization. To this end, we propose an effective and efficient ensemble enabled approach for query independent scholarly article ranking in a dynamic environment.

(1) We first develop an ensemble method, referred to as ERank, to combine the importance of three entities (articles, venues and authors) for scholarly article ranking, such that the importance is defined as a combination of *prestige* and *popularity* (Section 2).

The prestige of articles and venues is computed by introducing a novel *Time-Weighted PageRank* with a time decaying factor (based on citation statistics), while the one of authors is the average prestige of their published articles. The popularity of an article is the sum of all its citations' freshness (how close to the current year), while the one of venues and authors is the average popularity of their associated articles. Observe that (a), intuitively, prestige favors articles with many citations soon after their publication, and popularity favors those with recent citations, and (b) both prestige and popularity capture the evolving nature of entities.

(2) We then propose an efficient batch algorithm, which exploits the *temporal order* of scholarly data, *i.e.,* articles only cite previously published ones, to speed-up the computation (Section 3). The key of our ensemble approach ERank is a good solution for Time-Weighted PageRank, whose time complexity is significantly reduced with the usage of the temporal order.

(3) We also propose an incremental algorithm that further speeds up scholarly article ranking in a dynamic environment (Section 4). The incremental algorithm is based on the division of original graphs into *affected and unaffected areas*, and different updating strategies are designed for nodes in affected and unaffected areas.

(4) Using three real-life scholarly datasets (AAN, DBLP and MAG), we conduct an extensive experimental study (Section 5). We find that our ensemble enabled approach ERank is both effective and efficient, compared with PRank [8], FRank [24] and HRank [19]. (a) Using benchmarks RECOM and FCITA, ERank improves the pairwise accuracy [23] over (PRank, FRank, HRank) by (11.5%, 6.5%, 4.3%) and (14.5%, 2.5%, 3.8%) on AAN, (6.1%, 1.2%, 0.7%) and (9.1%, 5.2%, 3.6%) on DBLP, and (6.3%, 2.1%, 1.8%) and (7.0%, 4.2%, 1.5%) on MAG, on average, respectively. (b) Our batch algorithm batERank is on average (1.3, 2.5, 348) times faster

than (powERank, FRank, HRank) on the large dataset MAG. (c) Our incremental algorithm incERank is consistently faster than its batch counterpart batERank, *e.g.,* on average 22% faster on MAG.

Related work. Scholarly article ranking has shifted from citation count analysis [13,15] to graph analysis [9,16,18,19,21,24,29,31, 32,35]. Based on the information used, these methods are divided into four categories: (a) using citation information only [9, 13, 15, 21], (b) using the citation and temporal information [18,29], (c) using the citation information and other heterogeneous information, *e.g.,* authors and venues of articles [16, 19, 35], and (d) combining citation, temporal and other heterogeneous information [24,31,32]. Our work belongs to the last category aiming at fully employing information available for scholarly article ranking.

PageRank [8] and its extensions have been extensively used for citation analyses [30]. While PageRank equally propagates scores along outlinks, Weighted PageRank [34] extends PageRank by distributing scores based on the popularity of pages. Different from previous work, the Time-Weighted PageRank proposed in this work discriminately propagates scores in terms of citation statistics.

Dynamic algorithms have proven useful for various tasks by avoiding computing from scratch [22]. To our knowledge, little concern has been paid to dynamic scholarly article ranking except [14] considering PageRank in dynamic citation networks. However, its solution is based on a strong and impractical assumption that there are no citations for articles in the same year. Further, although there exist several studies on incremental PageRank computation [6,11,33], none of them has been applied for scholarly article ranking.

Different from previous work, we study scholarly article ranking in a dynamic environment by exploiting the temporal order that articles only cite previously published ones, which has never been exploited for efficient scholarly article ranking before.

Ensemble methods use multiple learners to obtain better performance than could be obtained from a constituent learner alone [36]. In this work, we leverage ensembles to produce better and robust results for scholarly article ranking [5, 12, 36].

2. ENSEMBLE RANKING MODEL

In this section, we introduce our ensemble enabled model ERank for ranking scholarly articles, such that the importance is defined as a combination of prestige and popularity.

2.1 Time-Weighted PageRank

We first introduce Time-Weighted PageRank (TWPageRank), a key for ERank to compute prestige, as the direct use of PageRank for ranking scholarly articles is problematic as follows.

(1) Scholarly articles typically have different impacts in practice, and there is a need to differentiate the impacts of different articles, while PageRank essentially assumes equal impacts.

(2) Citation relationships are time-evolving, which has been successfully exploited for scholarly article ranking [18,32], while PageRank does not consider temporal information at all.

Time-Weighted PageRank (TWPageRank). In most previous work, temporal information is simply exploited in the form of exponential decay [18,24,29,32]. We rethink the usage of time information in terms of the impacts of scholarly articles. Recall that Fig. 1 illustrates the total number of citations *w.r.t.* the number of years after publication. As we can see, the number of citations reaches a peak in two years, and gradually decreases after that, which also conforms to our perception of the impacts of articles.

According to Fig. 1, the impact of an article does depend on time, but not simply in the form of exponential decay. Specifically, if an article is cited after the citation peak, its impact should decay with time. Otherwise, its impact is fixed as a constant number. Moreover, considering that articles might reach their citation peaks in different time, we compute *the peak time for each individual article*, rather than using the same citation peak for all articles.

Inspired by the citation statistics, we propose TWPageRank that evaluates the prestige of nodes in a directed graph, such that each node is attached with time information. It differs from PageRank by weighting the influence propagation using the *impact weights on edges*, which represent the relative amounts of prestige that should be propagated from the edge sources to targets, and which also depend on the time information on nodes, following the same temporal tendency as scholarly article citations discussed above.

Formally, the impact weight on directed edge (u, v), *i.e.,* edge from u to v, is defined as:

$$w(u,v) = \begin{cases} 1 & T_u < Peak_v \\ e^{\sigma * (T_u - Peak_v)} & T_u \geq Peak_v, \end{cases} \tag{1}$$

where T_u is the time information on node u, $Peak_v$ is the peak time of node v using the time information of all nodes connecting to v, and σ is a negative number which controls the decaying speed of influence of articles. By default, Eq. (1) uses years as its time granularity. Note that the time decaying factor σ is introduced to provide flexibility for TWPageRank in various applications, and its value is typically within a small interval, *e.g.,* $[-2, 0]$, such that $w(u, v)$ does not decay when $\sigma = 0$ and already decays more than half per year when $\sigma = -1$. For the sake of completeness, we further set $w(u, v)$ to 0 if these does not exist an edge from u to v.

The update rule of TWPageRank is:

$$PR(v) = \frac{1-d}{n} + d \cdot \sum_{(u,v)\in E} \frac{w(u,v) \cdot PR(u)}{W(u)}, \tag{2}$$

where $PR(u)$ and $PR(v)$ are the TWPageRank scores of u and v, respectively, E is the set of edges, $W(u) = \Sigma_v w(u, v)$ is the sum of impact weights on all edges from u, n is the number of nodes and d is a damping parameter in $(0, 1)$. From Eq. (2) we can see that prestige is based on the impact weights, not equally distributed.

Correspondingly, the matrix form of the update rule is:

$$PR^{(t)} = d \cdot M^T \cdot PR^{(t-1)} + \frac{1-d}{n} \cdot e. \tag{3}$$

Here $PR^{(t)}$ and $PR^{(t-1)}$ are two TWPageRank vectors after t and $t-1$ iterations, respectively, M is the transition matrix such that $M_{u,v} = w(u, v)/W(u)$ and e is a n-dimentional vector $[1]_{n\times 1}$.

We next present the convergence of TWPageRank as follows.

Proposition 1: *TWPageRank converges to a unique TWPageRank vector on any graph, regardless of the initial vector.* □

Proof: It is known that a sequence of vectors $x^{(k)}$ such that $x^{(k+1)} = A \cdot x^{(k)} + b$ (where $k = 0, 1, \dots$) converges to a unique vector x^*, regardless of the initial vector x^0, if and only if $\rho(A) < 1$ [17], where $\rho(A)$ is the spectral radius of matrix A. Hence, it suffices to show $\rho(d \cdot M^T) < 1$ by Eq. (3).

As the column sums of $d \cdot M^T$ are all less than or equal to d, $||d \cdot M^T||_1 \leq d$ where $||d \cdot M^T||_1$ is the 1-norm of matrix $d \cdot M^T$ and is defined as the maximum absolute column sum of $d \cdot M^T$. We then have $\rho(d \cdot M^T) \leq ||d \cdot M^T||_1 \leq d < 1$, as the spectral radius of a matrix is always no more than its consistent matrix norms [4], *e.g.,* $||\cdot||_1$, which gives the conclusion. □

Remarks. Note that here Eq. (2) is indeed a more general update rule than Weighted PageRank [34], and the name of Time-Weighted PageRank comes from the use of time information in the initial impact weight $w(u, v)$ of Eq. (1).

2.2 Ensembles

Our ensemble enabled model ERank combines the importance of articles, venues and authors for scholarly article ranking, corresponding to the citation ensemble, venue ensemble and author ensemble, respectively, as shown in Fig. 2. The importance of an ensemble is defined as a combination of the prestige and popularity of its associated entities. Intuitively, prestige favors those with many citations soon after the publication of articles for citation ensemble or associated articles for venue and author ensembles, and popularity favors those with recent citations. Both prestige and popularity capture the evolving nature of entities.

We next introduce the details of ensembles.

Citation ensemble. The first ensemble computes the importance of articles using citation information.

A *citation graph* $G^c(V^c, E^c)$ is firstly constructed using the citation information such that (a) a node in V^c denotes an article, (b) a directed edge (u, v) in E^c denotes that u cites v, and (c) each node is associated with two types of time information: the publication year and the latest year having the largest number of citations.

(1) The prestige of articles is derived by TWPageRank on the citation graph G^c, and each article v is assigned its prestige $Prs_c(v)$.

(2) The popularity of an article is the sum of all its citations' freshness, *i.e.,* the closeness to the current year, as follows:

$$Pop_c(v) = \sum e^{\sigma \cdot (T_0 - T_u)},\ (u, v) \in E^c. \tag{4}$$

Here T_0 is the current year, *i.e.,* the latest T_u among all articles in V^c, σ is the negative decaying factor used in Eq. (1), and $e^{\sigma \cdot (T_0 - T_u)}$ represents the freshness of citation (u, v).

Intuitively, the more an article has recent citations, the higher its popularity is, no matter how long it has been published. Note that the popularity is also normalized such that the sum of all articles is equal to 1, similar to the prestige produced by TWPageRank.

(3) The prestige and popularity are finally seamlessly combined to produce the importance of an article. As important articles are both prestigious and popular, the importance of an article is defined as the geometric mean of its prestige and popularity.

$$Imp_c(v) = \sqrt{Prs_c(v) \cdot Pop_c(v)}. \tag{5}$$

The rationals behind Eq. (5) are as follows. (a) Prestigious articles with many recent citations are ranked at the top, as researchers are very willing to find these articles; (b) Prestigious articles with rare current citations are ranked lower, as researchers may lose interests in these old articles; And (c) articles with many recent citations are ranked higher, as researchers have potential interests.

Here the prestige and popularity are equally weighted to produce the importance of an article, as we focus on query independent ranking. They may be properly weighted when the query information is available, which is beyond the scope of this work.

Venue ensemble. The second ensemble computes the importance of venues with their associated articles. As a venue evolves with time, we treat the venue in each year individually, and the importance of a venue is the sum of its importance in all individual years.

A *venue graph* $G^v(V^v, E^v)$ is firstly constructed using the citation information among venues such that (a) a node in V^v represents a venue in a specific year, (b) a direct edge (s, t) in E^v denotes that there exist articles published in venue s citing articles published in venue t, and (c) we use *impact weights* to denote the weight $w_v(s, t)$ from venues s to t, which is the sum of the impact weights from articles published in s to t, *i.e.,*

$$w_v(s, t) = \sum_{u \in C(s), v \in C(t)} w(u, v). \tag{6}$$

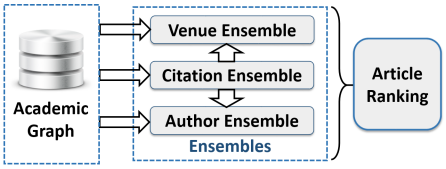


Figure 2: Ranking model ERank

Here, $C(s)$ and $C(t)$ are the collections of articles published in s and t, respectively, and $w(u, v)$ is the impact weight of edge (u, v) produced in the citation ensemble.

The prestige of a venue in a specific year is computed using the impact weights and the update rule in Eq. (2), and the popularity of a venue in a specific year is defined as the average popularity of its articles. Finally, the prestige and popularity are combined to derive the importance of a venue in the same way as the citation ensemble, which is treated as the venue ensemble score for all articles published in this venue.

Author ensemble. The author ensemble computes the importance of authors with their published articles.

Similar to the venue ensemble, we could evaluate the importance of each author, and treat the average importance of the authors associated with an article as its author ensemble score.

However, the resulting graph to compute the prestige is typically too large to handle. Hence, we evaluate the prestige of an author, by using the average prestige of all articles published by an author. Similar to the venue ensemble, the popularity of an author is also defined as the average popularity of her/his published articles. Finally, the prestige and popularity are combined to derive importance in the same way as the citation ensemble.

2.3 Ranking with Ensembles

The aforementioned ensembles are finally assembled to produce the final ranking, as illustrated in Fig. 2. Before assembling, each ranking is properly scaled such that the average scores of different rankings are the same. Let the scaled ranking scores of article v be $R_c(v)$, $R_v(v)$, and $R_a(v)$ from the citation, venue and author ensembles, respectively. The final ranking score $R(v)$ of an article v is aggregated as follows:

$$R(v) = \alpha \cdot R_c(v) + \beta \cdot R_v(v) + (1 - \alpha - \beta) \cdot R_a(v). \tag{7}$$

Here parameters α and β are used to regularize the contributions of the citation, venue and author information. Intuitively, these values indicate the intensity of the correlation between the importance of articles and the specific information.

3. ENSEMBLE COMPUTATION

As described in Section 2, each ensemble needs to compute the prestige and popularity of its corresponding entities. The popularity can be computed by scanning through all citations once. While the prestige of entities is based on TWPageRank, which is usually computed in an iterative manner [8]. Hence, the key of our ensemble approach computation is a good solution for TWPageRank.

3.1 Citation Ensemble Computation

We first present the computation of the citation ensemble.

It is easy to see that the popularity of articles can be computed by scanning all citations once and adding the freshness of citations to their corresponding articles, by Eq. (4). The prestige of articles is computed with TWPageRank on citation graphs, which occupies most computation cost of the citation ensemble. Nevertheless, we speed-up TWPageRank computation on citation graphs by exploiting the temporal order of scholarly data.

The main result here is stated as follows.

Input: citation graph $G^c(V^c, E^c)$.
Output: the TWPageRank vector of G^c.
1. O := the topological order of G^c;
2. **for** each node v following O **do**
3. update $PR(v)$ using Eq. (2);
4. **return** PR.

Figure 3: Algorithm for TWPageRank on citation graphs

Theorem 2: *TWPageRank on a citation graph $G^c(V^c, E^c)$ can be computed in $O(|V^c| + |E^c|)$ time.* □

Topological ordering. The edges of the citation graph strictly follow temporal order, in the sense that an article can only cite the articles published earlier, and, moreover, it is pretty rare that two articles published in the same time mutually cite each other. Hence, the citation graph G^c can be treated as a directed acyclic graph with few side effects. Accordingly, there exits a *topological order* for the nodes in a citation graph, *i.e.,* a sequence of its nodes such that every edge is directed from earlier to later.

We next prove Theorem 2 by providing such an algorithm with the desired property, by using the temporal order on a citation graph.

Algorithm CTWPR. The algorithm is presented in Fig. 3, which takes as input a citation graph G^c and returns its corresponding TWPageRank vector PR. It first computes a topological order of G^c since G^c is acyclic (line 1). By the topological order with nodes in higher oder first, it updates the TWPageRank score of each node with the update rule in Eq. (2) (lines 2–3), and finally it returns the computed TWPageRank vector (line 4).

We now show that CTWPR is indeed the desired algorithm.

Lemma 3: *The TWPageRank vector PR returned by* CTWPR *equals to the convergent TWPageRank vector PR^*.* □

Proof Sketch: We have $PR^* = d \cdot M^T \cdot PR^* + \frac{1-d}{n} \cdot e$, as PR^* is the convergent TWPageRank vector. Hence, we also have

$$PR^*(v) = d \cdot \sum_{(u,v) \in E^c} M_{u,v} \cdot PR^*(u) + \frac{1-d}{n}. \quad (8)$$

Consider a topological order $v_1/\ldots/v_n$ on citation graph $G^c(V^c, E^c)$, we then prove $PR(v_k) = PR^*(v_k)$ ($k \in [1, n]$) by induction, from which we have the conclusion (see [1] for details). □

Lemma 4: *Given a citation graph $G^c(V^c, E^c)$, algorithm* CTWPR *runs in $O(|V^c| + |E^c|)$ time.* □

Proof: The topological order of citation graph G^c can be computed in $O(|V^c| + |E^c|)$ time [10] (line 1), and the updating step (lines 2–3) can be done in $O(|V^c| + |E^c|)$ time as well. □

By Lemmas 3 & 4, we have shown Theorem 2.

Remark. Observe that by employing the topological order, the time complexity of TWPageRank on citation graphs is significantly reduced to $O(|V^c| + |E^c|)$, instead of $O(t \cdot (|V^c| + |E^c|))$ that is closely related to the number t of iterations.

3.2 Venue Ensemble Computation

We next present the computation of the venue ensemble. The popularity of venues in a specific year, is computed by averaging the popularity of the corresponding articles (given by the citation ensemble) published in the venues.

The prestige of venues in a specific year is also computed by TWPageRank on a venue graph $G^v(V^v, E^v)$. However, mutual citations are common on the venue graphs, *e.g.,* journal articles in a specific year may be published in different issues and/or numbers, and, hence, two journal venues in the same year can cite each other. That is, the venue graph is cyclic, and the linear algorithm on citation graphs does not work on venue graphs.

Input: venue graph $G^v(V^v, E^v)$, iteration threshold ϵ.
Output: the TWPageRank vector of G^v.
1. G' := the converted graph of G^v;
2. O := topological order of G';
3. **for** each node v' following O **do**
4. scc := the corresponding SCC of v';
5. **while** sum of changes of $PR(v)$ where $v \in scc$ exceeds $\frac{|scc|}{|V^v|} \cdot \epsilon$ **do**
6. update $PR(v)$ using Eq. (9) where $v \in scc$;
7. **return** PR.

Figure 4: Algorithm for TWPageRank on venue graphs

Nevertheless, articles typically cite those closely related articles, *e.g.,* from the same field of study. That is, there is a possibility of grouping venues. It is known that a graph becomes acyclic by replacing its strongly connected components (SCCs) with single nodes [10]. This motivates us to revise the algorithm by processing an SCC, instead of a single node, at each time, by which we can exploit the temporal order of scholarly data again. More specifically, the edges of venue graph G^v is first partitioned into two disjoint sets E^v_w and E^v_b with $E^v = E^v_w \cup E^v_b$, where E^v_w and E^v_b are sets of edges inside and across SCCs, respectively. Further, the update rule is revised by treating E^v_b and E^v_w differently as follows.

$$PR(v) = d \cdot \sum_{(u,v) \in E^v_w} M_{u,v} \cdot PR(u) + d \cdot \sum_{(u,v) \in E^v_b} M_{u,v} \cdot PR(u) + \frac{1-d}{n}. \quad (9)$$

The main result here is stated as follows.

Theorem 5: *TWPageRank on a venue graph $G^v(V^v, E^v)$ can be done in $O(|V^v| + |E^v_b| + t \cdot |E^v_w|)$ time.* □

We next prove Theorem 5 by providing such an algorithm with the desired property, which uses the temporal order, and processes an SCC, instead of a single node, at each time on a venue graph.

Algorithm VTWPR. The SCC-based algorithm to compute the TWPageRank vector for venue graph G^v is shown in Fig. 4.

It first computes the converted graph G' by treating SCCs in G^v as single nodes (line 1), and then derives a topological order O of G' (line 2). It then iteratively updates the TWPageRank scores of nodes in each SCC in the topological order of G' using Eq. (9) (lines 3–6), and finally returns the TWPageRank vector (line 7). More specifically, the iteration continues until the sum of TWPageRank score changes is less than $\frac{|scc|}{|V^v|} \cdot \epsilon$ (lines 5–6).

Lemma 6: *The TWPageRank vector PR returned by* VTWPR *converges such that $||PR - PR^*||_1 < \epsilon$, where PR^* is the convergent TWPageRank vector.* □

Proof Sketch: We first prove that the sum of changes after another iteration from PR, *i.e.,* $||d \cdot M^T \cdot PR + \frac{1-d}{n} \cdot e - PR||_1$, is smaller than ϵ, and then prove that $||PR^* - PR||_1$ is smaller than the sum of changes (see [1] for details). □

Lemma 7: *Given a venue graph $G^v(V^v, E^v)$, algorithm* VTWPR *runs in $O(|V^v| + |E^v_b| + t \cdot |E^v_w|)$ time, where t is the maximum number of iterations among all SCCs.* □

Proof: Observe the following. (1) The converted graph G' and topological order O can be computed in $O(|V^v| + |E^v|)$ [10] (lines 1,2), and (2) updating the TWPageRank scores takes $O(|V^v| + |E^v_b| + t \cdot |E^v_w|)$ where edges in $|E^v_b|$ are only scanned once (lines 3–6). Note that $|E^v_w|$ is typically much smaller than $|E^v|$ for scholarly article data obeying a temporal citation order in practice. □

By Lemmas 6 & 7, we have shown Theorem 5.

3.3 Author Ensemble Computation

As the prestige (popularity) of authors is defined as the average prestige (popularity) of their published articles, it suffices to scan

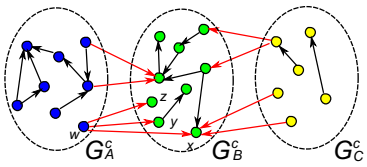


Figure 5: An example of affected and unaffected areas

through all author-article relationships for computing both the prestige and the popularity of authors.

3.4 The Complete Batch Algorithm

We finally present the complete batch algorithm batERank, which combines the computation of the citation, author and venue ensembles with Eq. (7). It takes as input academic graph data D and an iteration threshold ϵ and returns the scholarly article ranking of D. It first constructs the citation and venue graphs $G^c(V^c, E^c)$ and $G^v(V^v, E^v)$. Then it computes the prestige and popularity of citation, venue and author ensembles. Finally, it combines the prestige and popularity of three ensembles to produce the final ranking.

Time & space complexity analyses. By the analyses above and Lemmas 4 & 7, it is easy to verify that algorithm batERank runs in $O(|V^c|+|E^c|+|V^v|+|E_b^v|+t\cdot|E_w^v|+|PA|)$, where $|E_w^v|$ is the number of edges inside SCCs of G^v and $|PA|$ is the total number of author-article relationships.

Compared with the traditional power method, algorithm batERank uses (a) $|V^c|$ extra space to store the topological order of G^c, (b) $(2\cdot|V'|+|E'|)$ extra space to store the converted graph $G'(V', E')$ of G^v and its topological order, and (c) $|V^c|$ space is saved for the TWPageRank vector of G^c. To conclude, algorithm batERank only uses limited extra space to speed-up computation.

4. INCREMENTAL COMPUTATION

Scholarly article data is dynamic and continuously growing, and, hence, it is impractical to recompute the ranking from scratch for each update. In this section, we propose an incremental algorithm for our ranking model ERank to deal with the dynamic scenario.

4.1 Incremental Citation Ensemble

We first present the incremental computation of prestige and popularity for the citation ensemble. Consider an update $\Delta^c = \Delta V^c \cup \Delta E^c$ added to citation graph $G^c(V^c, E^c)$, and the resulting citation graph is $G^{c,+}(V^c \cup \Delta V^c, E^c \cup \Delta E^c)$, where ΔV^c is a set of nodes with $\Delta V^c \cap V^c = \emptyset$, and ΔE^c is a set of directed edges on ΔV^c and from ΔV^c to V^c only, by the temporal order of citations.

4.1.1 Incremental Prestige Computation

We first present how to incrementally compute the prestige of the citation ensemble, *i.e.,* the TWPageRank vector on the updated citation graph $G^{c,+}$, given an update $\Delta^c = \Delta V^c \cup \Delta E^c$ and the TWPageRank vector on the original citation graph G^c.

Affected and unaffected areas. The TWPageRank vector of the original citation graph G^c is mainly affected in two ways.

(1) Let $V_{B,1}^c \subseteq V^c$ be the set of nodes reachable from the newly added nodes ΔV^c in $G^{c,+}$, $V_{B,2}^c \subseteq V^c$ be the set of nodes with incoming edges having weight changes, and $V_{B,3}^c \subseteq V^c$ be the set of nodes reachable from $V_{B,2}^c$.

Then $V_B^c = V_{B,1}^c \cup V_{B,2}^c \cup V_{B,3}^c$ is obviously the set of nodes in G^c affected by the update Δ^c.

(2) Let $V_A^c = V^c \setminus V_B^c$. As the TWPageRank scores on V_A^c can be simply adjusted by scaling in terms of Lemma 9 (to be seen shortly), V_A^c is essentially not affected by the update Δ^c.

Input: An update $\Delta^c = \Delta V^c \cup \Delta E^c$, TWPageRank vector PR, matrix M, and the topological order O of the original citation graph G^c.
Output: TWPageRank vector PR^+ of the updated citation graph $G^{c,+}$.
1. ΔO := topological order of G_C^c; $O^+ := \Delta O / O$;
2. label all nodes of G^c as A and all nodes of G_C^c as C;
3. **for** each node v following O^+ **do**
4. **if** v is labeled as C **then**
5. update $PR^+(v)$ using Eq. (2);
6. label node w as B such that $w \in V^c$ and $(v, w) \in \Delta E^c$;
7. **else if** v is labeled as B **then**
8. update $PR^+(v)$ using Eq. (10);
9. label node w as B such that $(v, w) \in E^c$;
10. **else** $PR^+(v) := PR(v) \cdot n/n^+$;
11. **if** $W(v)$ is changed **then**
12. label node w as B such that $(v, w) \in E^c$;
13. **return** PR^+.

Figure 6: Algorithm incCTWPR

Let $G_A^c = (V_A^c, E_A^c)$, $G_B^c = (V_B^c, E_B^c)$ and $G_C^c = (V_C^c, E_C^c)$, respectively; Let E_{AB}^c be the set of edges from G_A^c to G_B^c, and E_{CB}^c be the set of edges from G_C^c to G_B^c. Then we have $V_C^c = \Delta V^c$, $E_C^c \cup E_{CB}^c = \Delta E^c$, $V^c = V_A^c \cup V_B^c$, and $E^c = E_A^c \cup E_B^c \cup E_{AB}^c$.

In this way, the updated citation graph $G^{c,+}$ is partitioned into (a) subgraphs $\{G_A^c, G_B^c, G_C^c\}$ and (b) edge sets $\{E_{AB}^c, E_{CB}^c\}$.

Example 1: Figure 5 illustrates an example of affected and unaffected areas, in which subgraphs G_A^c, G_B^c and G_C^c are associated with node sets V_A^c, V_B^c and ΔV^c, respectively. Here the citation peak time of node x is changed, which leads to the changes of the normalized weights of edges (w, x), (w, y) and (w, z). □

We now present the incremental update rule.

Incremental update rule. Here notations without/with superscript '+' are defined on G^c and $G^{c,+}$, respectively. We expand M and PR to the same sizes as M^+ and PR^+, respectively, by filling zeros, and let $\Delta M = M^+ - M$ and $\Delta PR = PR^+ - PR$.

The TWPageRank vector PR^+ on the original part of $G^{c,+}$, *i.e.,* G^c, is incrementally updated as follows.

$$\begin{aligned} PR^+(v) =& \frac{1-d}{n^+} + d \cdot \sum_{(u,v)} (M_{u,v} + \Delta M_{u,v}) \cdot (PR(u) + \Delta PR(u)) \\ =& \frac{n}{n^+} \cdot PR(v) + d \cdot \sum_{(u,v)} M_{u,v} \cdot (\frac{n^+ - n}{n^+} \cdot PR(u) + \Delta PR(u)) \\ &+ \sum_{(u,v)} \Delta M_{u,v} \cdot PR^+(u). \end{aligned} \tag{10}$$

Here $(u, v) \in E^c \cup \Delta E^c$, and $M_{u,v} = 0$ when $(u, v) \in \Delta E^c$.

Algorithm incCTWPR. We now present our incremental algorithm for the prestige computation, shown in Fig. 6.

It takes as input an update Δ^c and the previous results on the original citation graph $G^c(V^c, E^c)$, and returns the TWPageRank vector of the updated citation graph $G^{c,+}$. It first derives a topological order ΔO of G_C^c, and concatenates ΔO with O (line 1). After that, it labels the newly added nodes with C and existing nodes with A (line 2). It then updates the TWPageRank score of each node following O^+ (lines 3–12). More specifically, the scores of nodes labeled as C and B are updated according to Eq. (2) and Eq. (10), respectively, while the scores of nodes labeled as A are simply scaled. It finally returns the TWPageRank vector (line 13).

Lemma 8: *O^+ is indeed a valid topological order of $G^{c,+}$.* □

Proof: It suffices to show that for each $(u, v) \in E^c \cup E_C^c \cup E_{CB}^c$, u comes before v in O^+, which obviously holds (1) for $E^c \cup E_C^c$ as O and ΔO are topological orders of G^c and G_C^c, respectively, and (2) for E_{CB}^c as nodes in G_C^c come before nodes in G_B^c. □

Lemma 9: *For nodes v in G_A^c, $PR^+(v) = PR(v) \cdot n/n^+$.* □

Proof Sketch: Assume a topological order $v_1/\ldots/v_{n_A}$ of graph G_A^c with $n_A = |V_A^c|$, and we prove $PR^+(v_k) = n/n^+ \cdot PR(v_k)$ ($k \in [1, n_A]$) by induction (see [1] for details). □

The main result here is stated as follows.

Theorem 10: *Given an update $\Delta^c = \Delta V^c \cup \Delta E^c$, and the citation graph $G^c(V^c, E^c)$, together with its TWPageRank vector PR and topological order O, algorithm* incCTWPR *takes $O(|V^c \cup \Delta V^c| + |E^c \cup \Delta E^c \setminus E_A^c|)$ time.* □

Proof: Observe the following. (1) A topological order of G_C^c can be computed in $O(|V_C^c| + |E_C^c| + |E_{CB}^c|)$ time. (2) Updating the TWPageRank scores of nodes in subgraphs G_B^c and G_C^c costs $O(|V_B^c \cup V_C^c| + |E_B^c \cup E_C^c \cup E_{AB}^c \cup E_{CB}^c|)$ time. Finally, (3) Updating the scores of nodes in G_A^c costs $O(|V_A^c|)$ time. □

By Proposition 1 and Lemmas 3,8 & 9, one can easily verify the correctness of algorithm incCTWPR, and algorithm incCTWPR is always faster than algorithm CTWPR as long as E_A^c is not empty.

4.1.2 Incremental Popularity Computation

As the popularity of articles is defined as the sum of all their citations' freshness, it is very convenient to incrementally maintain. Again, we denote the popularity of node v on G^c and $G^{c,+}$ as $pop_c(v)$ and $pop_c^+(v)$, respectively.

Given an update $\Delta^c = \Delta V^c \cup \Delta E^c$, the updated popularity $pop_c^+(v)$ can be easily computed as:

$$pop_c^+(v) = pop_c(v) \cdot e^{\sigma \cdot (T_0^+ - T_0)} + \sum_{(u,v)} e^{\sigma \cdot (T_0^+ - T_u)}. \quad (11)$$

Here $(u, v) \in \Delta E^c$, and T_0 (respectively T_0^+) is the current time in G^c (respectively $G^{c,+}$).

It is easy to see that the time complexity of incrementally updating popularity is $O(\,|V^c| + |\Delta V^c| + |\Delta E^c|)$. This further shows that Eq. (11) is a desired solution for popularity maintenance.

4.2 Incremental Venue Ensemble

We next present the incremental computation of venue ensemble. More specifically, the popularity computation is exactly the same as batERank, due to the way that the popularity of venues in a specific year is defined, and the popularity of every article is potentially changed. In the following, we focus on incremental prestige computation, *i.e.,* incremental TWPageRank on venue graphs.

We consider an update $\Delta^v = \Delta V^v \cup \Delta E^v$ to the original venue graph $G^v(V^v, E^v)$, and the resulting updated venue graph $G^{v,+}(V^v \cup \Delta V^v, E^v \cup \Delta E^v)$.

Algorithm incVTWPR. We now present algorithm incVTWPR to incrementally compute the TWPageRank vector on venue graphs. It takes as input an update Δ^v and the associated results of the original venue graph G^v, and returns the TWPageRank vector of $G^{v,+}$. It is similar to algorithm VTWPR in Fig. 4, except the following. (1) It incrementally computes the topological order O^+ by concatenating the topological orders of ΔG^v and G^v, and (2) it uses the existing TWPageRank vector scaled with constant n/n^+ as the initial vector for the nodes in G^v.

Correctness & time complexity analyses. The correctness of algorithm incVTWPR is essentially based on Proposition 1 and Lemma 6, and that O^+ is indeed a valid topological order.

Algorithm incVTWPR runs in $O(|V^v \cup \Delta V^v| + |E_b^v \cup \Delta E_b^v| + t^+ \cdot |E_w^v \cup \Delta E_w^v|)$, by Lemma 7. Moreover, incVTWPR derives the topological order based on ΔG^v only, instead of $G^{v,+}$, and the number t^+ is very likely smaller than the number t of VTWPR. As shown by the experimental study, incVTWPR is faster than VTWPR, even though they have the same time complexity.

4.3 The Complete Incremental Algorithm

Similar to the popularity computation of the venue ensemble, both the prestige and popularity of the author ensemble are computed exactly the same as batERank, as the prestige and popularity of each article are potentially changed.

We finally present the complete incremental algorithm incERank, which is similar to the batch algorithm batERank, except that (1) it uses algorithms incCTWPR and incVTWPR to incrementally compute the prestige of the citation and venue ensembles, respectively, and (2) it incrementally computes the popularity of the citation ensemble based on Eq. (11).

Time & space complexity analyses. By the analyses above, the time complexity of incERank is the same as batERank, except that incERank takes $O(|V_A^c|)$ time to process G_A^c. Indeed, algorithm incERank is typically faster than batERank in practice as follows. (1) It derives the topological order on updates Δ^c and Δ^v in $O(|\Delta V^c| + |\Delta E^c|)$ and $O(|\Delta V^v| + |\Delta E^v|)$ (comparatively, $O(|V^c \cup \Delta V^c| + |E^c \cup \Delta E^c|)$ and $O(|V^v \cup \Delta V^v| + |E^v \cup \Delta E^v|)$) time; (2) It processes G_A^c of the updated citation graph in $O(|V_A^c|)$ (comparatively, $O(|V_A^c + E_A^c|)$) time and computes popularity in $O(|V^c \cup \Delta V^c| + |\Delta E^c|)$ (comparatively, $O(|V^c \cup \Delta V^c| + |E^c \cup \Delta E^c|)$) time; Finally, (3) It updates TWPageRank scores on the updated venue graph with less iterations.

The efficiency improvement is achieved by using $(|E^c \cup \Delta E^c| + 2 \cdot |V^c \cup \Delta V^c|)$ extra space to store ΔM, ΔPR and the affected and unaffected areas in the citation ensemble.

5. EXPERIMENTAL STUDY

In this section, we present an extensive experimental study of our ensemble enabled approach ERank. Using three real-life scholarly datasets (AAN, DBLP and MAG) and two benchmarks (RECOM and FCITA), we conducted four sets of experiments to evaluate: (1) the effectiveness of ERank, (2) the efficiency of our batch algorithm batERank and incremental algorithm incERank, and (3) the impacts of the time decaying factor σ and paramters α and β.

5.1 Experimental Settings

We first present the settings of our experimental study.

Datasets. We chose three datasets to test our approach.
(1) AAN records the collection of computational linguistics articles published by ACL from 1965 to 2011 [19]. It contains 18,041 articles, 14,386 authors, 273 venues and 69,928 citations.
(2) DBLP records publications in the computer science domain from 1936 to 2016 [28]. It contains around 3.14 million articles, 1.74 million authors, 11,619 venues and 6.38 million citations.
(3) MAG records publications of various disciplines from 1800 to 2016 [26]. It contains around 127 million articles, 115 million authors, 24,024 venues and 526 million citations.

These datasets were further cleaned by detecting citation cycles and removing those edges violating the temporal order if any.

Accuracy metric. We adopted *pairwise accuracy* [23] to evaluate the ranking quality, which is the fraction of times that a ranking agrees with the correct importance orders of scholarly article pairs.

Benchmarks. We constructed two benchmarks (RECOM and FCITA) of scholarly article pairs to test the pairwise accuracy.
(1) RECOM assumes scholarly articles with more recommendations are of higher importance. We used the numbers of recommendations of 93 articles in AAN [19], and then matched articles in DBLP and MAG with titles. Finally, we generated (2133, 966, 1972) pairs on (AAN, DBLP, MAG), respectively.
(2) FCITA assumes scholarly articles with more future citations are of higher importance [18, 31, 32]. We first divided each dataset

Methods	PRank	FRank	HRank	ERank
AAN	0.687	0.737	0.759	**0.802**
DBLP	0.732	0.782	0.786	**0.793**
MAG	0.613	0.655	0.658	**0.676**

Table 1: Accuracy tests on RECOM

into ranking and future data by a splitting year such that ranking (future) data consisted of articles published before (in or after) that year. We then generated article pairs from ranking data, whose importance was evaluated by the numbers of citations from articles in future data. Articles in the same pairs were required to be in similar research fields, by utilizing the Fields-Of-Study information of MAG [26], and published in the same years, similar to [31]. We used all pairs (around 25,000) for AAN, and randomly selected 300,000 pairs for both DBLP and MAG, respectively.

Algorithms. We compared our approach ERank with three baseline methods: PRank [8], FRank [24] and HRank [19].
(1) PRank (PageRank) is a classic method that uses only citation information to generate the scholarly article ranking.
(2) FRank (FutureRank) combines citation, temporal and other heterogeneous information to rank scholarly articles.
(3) HRank (HHGBiRank) is a very recent methods using both citation and other heterogeneous information, such that heterogeneous entities are mutually reinforced based on hypernetworks.

Implementation. All algorithms were implemented with Microsoft Visual C++. For all algorithms, (a) the damping parameter d and the iteration threshold ϵ were fixed to 0.85 and 10^{-8}, (b) the default splitting year was selected such that future data had around 20% of all articles, which was 2008 on AAN and 2013 on both DBLP and MAG, and, (c) aggregating parameters , *e.g.,* α, and β in ERank, were tuned at the granularity of 0.1 and parameters giving the best result were selected. Moreover, ρ was set to -0.2 for FRank following [24], and σ was set to -1 by default for ERank.

All experiments were conduceted on a PC with 2 Intel Xeon E5–2630 2.4GHz CPUs and 64 GB of memory, running 64 bit Windows 7 professional system. When quantity measures are evaluated, the test was repeated over 5 times and the average is reported.

5.2 Experimental Results

We next present our findings.

Exp-1: Effectiveness on RECOM. In the first set of our tests, we used benchmark RECOM to evaluate the effectiveness of our approach. All algorithms only used articles published before 2012, since RECOM is based on this portion of articles. Aggregating parameters were selected as follows: (α, β, γ) = $(0.1, 0.2, 0.2)$ for FRank, (a_{i1}, a_{i2}, a_{i3}) = $(0.6, 0.2, 0.2)$ for HRank ($i \in [1, 3]$), and (α, β) = $(0.1, 0.8)$ for ERank. The results are reported in Table 1.

The PairAcc of PRank is much lower than other algorithms, which indicates that citation information alone is insufficient for scholarly article ranking, and other information helps to refine the results. Moreover, ERank consistently ranks better than all baselines. Indeed, ERank improves the PairAcc over (PRank, FRank, HRank) by (11.5%, 6.5%, 4.3%) on AAN, (6.1%, 1.2%, 0.7%) on DBLP, and (6.3%, 2.1%, 1.8%) on MAG, respectively.

Exp-2: Effectiveness on FCITA. In the second set of tests, we used benchmark FCITA to evaluate the effectiveness. All algorithms produced results based on articles in ranking data. Aggregating parameters were selected as follows: (α, β, γ) = $(0.7, 0.1, 0.2)$ for FRank, (a_{i1}, a_{i2}, a_{i3}) = $(0.3, 0.6, 0.1)$ for HRank ($i \in [1, 3]$), and (α, β) = $(0.8, 0.1)$ for ERank. Here the citation ensemble contributed most in ERank, since FCITA is based on citation information. We tested the impacts of three factors: the splitting year Y_s, the largest number b of existing years of articles in FCITA, and the least difference dif of future citation counts.

Exp-2.1. To evaluate the impacts of the splitting year Y_s, we varied Y_s from 2005 to 2010 on AAN and from 2010 to 2015 on both DBLP and MAG. For each Y_s we generated benchmark pairs as described earlier, and tested PairAcc using all pairs, *i.e.,* $b = +\infty$, $dif = 1$. The results are reported in Figs. 7(a), 7(e) and 7(i), where red markers □ in dashed line mean HRank ran out of memory.

When varying Y_s, the PairAcc of all algorithms increases with the increment of Y_s on both DBLP and MAG, since it is easier to assess short-term importance (larger Y_s) than long-term (lower Y_s). While the results on AAN do not follow this trend, possibly because AAN does not record the complete articles of 2007 and 2009. Moreover, ERank consistently ranks better than all baselines, regardless of assessing long-term or short-term importance. Indeed, ERank improves the PairAcc over (PRank, FRank, HRank) by (19.5%, 4.2%, 5.2%) on AAN, (19.3%, 7.8%, 6.0%) on DBLP, and (11.9%, 4.6%, 1.8%) on MAG, on average, respectively.

Exp-2.2. To evaluate the impacts of number b, we varied b from 1 to $+\infty$, while fixed Y_s to default values and $dif = 1$. The results are reported in Figs. 7(b), 7(f) and 7(j).

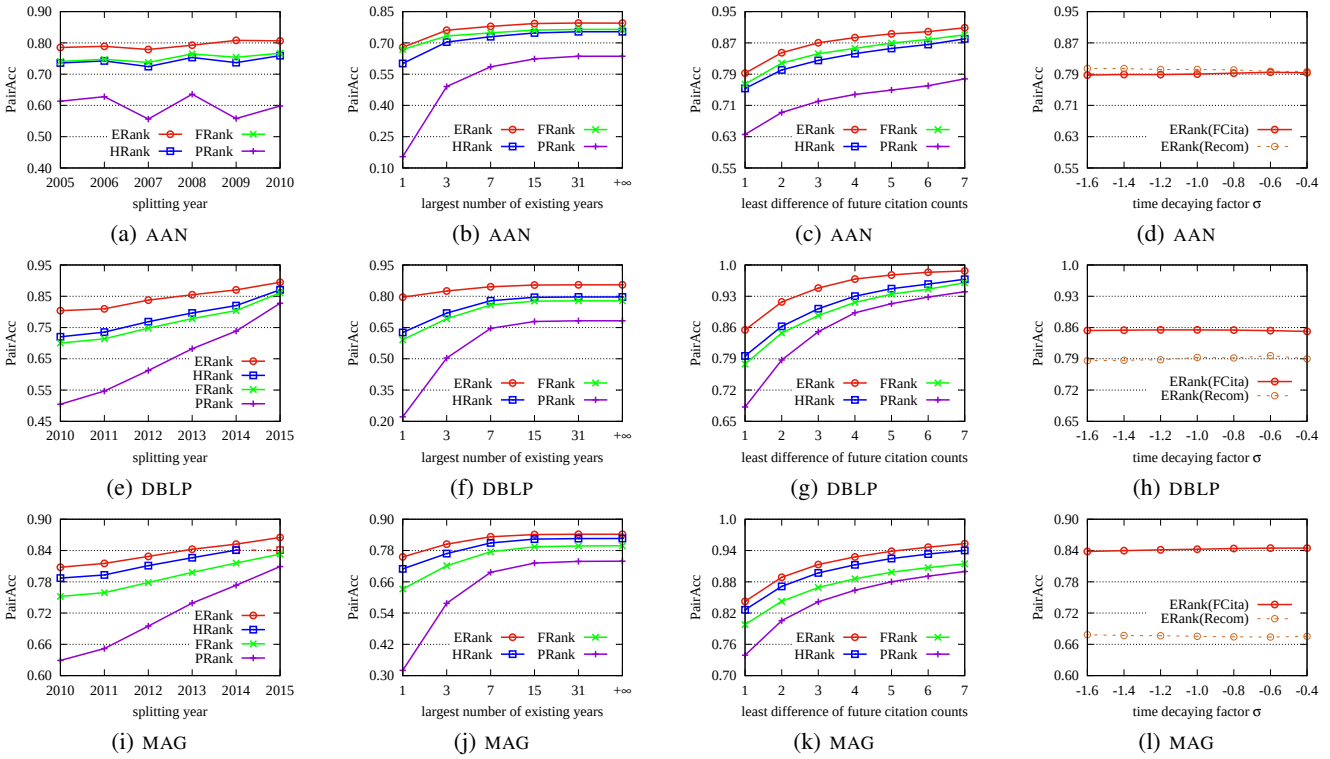
When varying b, the PairAcc of all algorithms increases with the increment of b, since old articles (larger b) are easy to rank based on adequate information, while new articles (lower b) are hard to rank with only little information available. Moreover, ERank consistently ranks better than all baselines, especially when $b \leq 3$, *i.e.,* ranking newly published articles. Indeed, ERank improves the PairAcc over (PRank, FRank, HRank) by (24.6%, 2.7%, 5.2%) on AAN, (26.9%, 10.9%, 8.6%) on DBLP, and (18.6%, 6.7%, 2.6%) on MAG, on average, respectively.

Exp-2.3. To evaluate the impacts of difference dif, we varied dif from 1 to 7, while fixed Y_s to default values and $b = +\infty$. The results are reported in Figs. 7(c), 7(g) and 7(k).

When varying dif, the PairAcc of all algorithms increases with the increment of dif, since pairs with larger difference of future citation counts are easier to rank. Moreover, ERank consistently ranks better than all baselines, regardless of larger or lower difference. Indeed, ERank improves the PairAcc over (PRank, FRank, HRank) by (14.5%, 2.5%, 3.8%) on AAN, (9.1%, 5.2%, 3.9%) on DBLP, and (7.0%, 4.2%, 1.5%) on MAG, on average, respectively.

Exp-3: Efficiency. In the third set of tests, we evaluated the efficiency of our batch and incremental algorithms. We compared our algorithms with powERank (powCTWPR, powVTWPR), which was exactly the same to batERank (CTWPR, VTWPR) except using power method for TWPageRank computation, and baselines FRank and HRank. Here PRank was omitted due to its effectiveness. We varied the splitting year Y_s from 2009 to 2016 and tested the running time based on articles published before Y_s on both MAG and DBLP. For incremental algorithms, base and update data consisted of articles published before 2008 and within $[2008, Y_s)$, respectively. The results on the largest MAG are reported in Fig. 8, where red markers □ in dashed line mean HRank ran out of memory, and the results on DBLP are left in [1].

When varying Y_s, the running time of all algorithms increases with the increment of Y_s, and the incremental algorithms consistently run faster than other counterparts. For TWPageRank, algorithms CTWPR and VTWPR are 2.1 and 1.7 times faster than powCTWPR and powVTWPR on average, respectively. And algorithms incCTWPR and incVTWPR further improve the efficiency by 23% and 38% on average, compared with CTWPR and VTWPR, respectively. For scholarly article ranking, algorithm batERank is on average (1.3, 2.5, 348) times faster than (powERank, FRank, HRank), respectively. And algorithm incERank further improves the efficiency by 22% on average, compared with batERank.

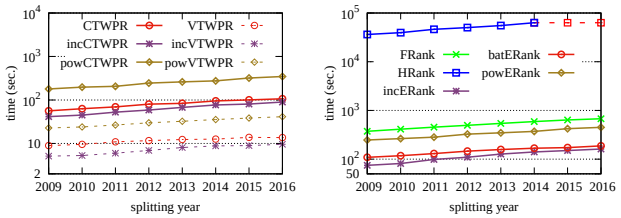


(a) AAN (b) AAN (c) AAN (d) AAN

(e) DBLP (f) DBLP (g) DBLP (h) DBLP

(i) MAG (j) MAG (k) MAG (l) MAG

Figure 7: Accuracy tests on FCITA (all) and RECOM ((d), (h) and (l) only)



(a) TWPageRank (batch vs. inc.) (b) Comparison of ranking algorithms

Figure 8: Efficiency tests on MAG

Exp-4: Impacts of parameters. In the last set of tests, we evaluated the impacts of time decaying factor σ and parameters α, β.

We present main results only and more details are available in [1].

Exp-4.1. To evaluate the impacts of the time decaying factor σ, we varied σ from -1.6 to -0.4, while fixed Y_s to default values, $b = +\infty$ and $dif = 1$. We tested the PairAcc on both benchmarks. The results of PairAcc are reported in Figs. 7(d), 7(h) and 7(l).

When varying σ, the PairAcc of ERank is almost stable on all datasets using both FCITA and RECOM. Indeed, the PairAcc using (FCITA, RECOM) only varies (0.73%, 0.80%) on AAN, (0.34%, 1.1%) on DBLP, and (0.65%, 0.46%) on MAG, respectively.

The running time indeed varies (2.1%,1.0%) on average only on (MAG, DBLP), respectively.

Exp-4.2. To evaluate the impacts of parameters α and β on effectiveness, we varied α and β at the granularity of 0.01, while fixed Y_s to default values, $b = +\infty$, $dif = 1$ and $\sigma = -1.0$. Note that parameters α and β have no impacts on efficiency.

Indeed ERank is very robust to parameters α and β. (a) When varying α and β, the PairAcc of ERank changes gently. (b) PairAcc also keeps at a high level within a certain (α, β) combination space. Finally, (c) the optimal parameters on the same benchmarks are very similar for (AAN, DBLP and MAG). That is, it is quite flexible for choosing proper values of parameters α and β.

Summary. From these tests we find the followings.

(1) Our approach ERank is effective for ranking scholarly articles. The PairAcc of ERank is consistently better than the compared methods in all tests. Indeed, using RECOM and FCITA, ERank improves PairAcc over (PRank, FRank, HRank) by (11.5%, 6.5%, 4.3%) and (14.5%, 2.5%, 3.8%) on AAN, (6.1%, 1.2%, 0.7%) and (9.1%, 5.2%, 3.6%) on DBLP, and (6.3%, 2.1%, 1.8%) and (7.0%, 4.2%, 1.5%) on MAG, on average, respectively.

(2) Our approach ERank is also very efficient. The batch algorithm batERank is on average (1.3, 2.5, 348) times faster than (powERank, FRank, HRank) on the largest MAG, respectively.

(3) Our incremental algorithms are much faster than their batch counterparts in practice, even their time complexity is very close. Indeed, algorithms incCTWPR, incVTWPR and incERank further improve the efficiency of (CTWPR, VTWPR, batERank) by (23%, 38%, 22%) on average, respectively.

(4) Our ensemble ranking model ERank introduces the time decaying factor σ and parameters α and β for the sake of practicability and flexibility of real-life applications. We have experimentally shown that ERank is very robust to these parameters.

6. CONCLUSIONS

We have proposed an ensemble enabled model ERank for scholarly article ranking, which combines the importance of article, venue and author entities. We have also proposed efficient batch and incremental algorithms for the computation of their importance, defined as a combination of prestige and popularity. As shown by the experimental study, ERank is both effective and efficient for scholarly article ranking.

A couple of topics need further investigation. First, we are to clean scholarly article data with external data sources for further improving the quality of ranking. Second, we are to study distributed algorithms for ensemble computations, similar to [37] that computes PageRank in distributed scenarios.

7. REFERENCES

[1] Full version. *http://mashuai.buaa.edu.cn/ERank-full.pdf*.

[2] Google Scholar. *https://scholar.google.com/*.

[3] Semantic Scholar. *https://www.semanticscholar.org/*.

[4] Spectral Radius. *https://en.wikipedia.org/wiki/Spectral_radius*.

[5] WSDM CUP'2016. *https://wsdmcupchallenge.azurewebsites.net/*.

[6] S. Abiteboul, M. Preda, and G. Cobena. Adaptive on-line page importance computation. In *WWW*, 2003.

[7] C. C. Aggarwal and K. Subbian. Evolutionary network analysis: A survey. *ACM Comput. Surv.*, 47(1):10:1–10:36, 2014.

[8] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks*, 30(1-7):107–117, 1998.

[9] P. Chen, H. Xie, S. Maslov, and S. Redner. Finding scientific gems with google's pagerank algorithm. *J. Informetrics*, 1(1):8–15, 2007.

[10] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. The MIT Press, 2001.

[11] P. K. Desikan, N. Pathak, J. Srivastava, and V. Kumar. Incremental page rank computation on evolving graphs. In *WWW, Special interest tracks and posters*, 2005.

[12] L. Duan, C. Aggarwal, S. Ma, R. Hu, and J. Huai. Scaling up link prediction with ensembles. In *WSDM*, 2016.

[13] E. Garfield. Citation analysis as a tool in journal evaluation. *Science*, 178(4060):471–479, 1972.

[14] R. Ghosh, T. Kuo, C. Hsu, S. Lin, and K. Lerman. Time-aware ranking in dynamic citation networks. In *ICDM Workshops*, 2011.

[15] J. E. Hirsch. An index to quantify an individual's scientific research output. *Proceedings of the National Academy of Sciences of the United States of America*, 102(46):16569–16572, 2005.

[16] X. Jiang, X. Sun, and H. Zhuge. Towards an effective and unbiased ranking of scientific literature through mutual reinforcement. In *CIKM*, 2012.

[17] C. T. Kelley. *Iterative methods for optimization*. SIAM, 1999.

[18] X. Li, B. Liu, and P. Yu. Time sensitive ranking with application to publication search. In *ICDM*, 2008.

[19] R. Liang and X. Jiang. Scientific ranking over heterogeneous academic hypernetwork. In *AAAI*, 2016.

[20] S. Ma, J. Li, C. Hu, X. Lin, and J. Huai. Big graph search: challenges and techniques. *FCS*, 10(3):387–398, 2016.

[21] M. K.-P. Ng, X. Li, and Y. Ye. Multirank: Co-ranking for objects and relations in multi-relational data. In *KDD*, 2011.

[22] G. Ramalingam and T. W. Reps. A categorized bibliography on incremental computation. In *POPL*, 1993.

[23] M. Richardson, A. Prakash, and E. Brill. Beyond pagerank: Machine learning for static ranking. In *WWW*, 2006.

[24] H. Sayyadi and L. Getoor. Future rank: Ranking scientific articles by predicting their future pagerank. In *SDM*, 2009.

[25] J. Shen, Z. Song, S. Li, Z. Tan, Y. Mao, L. Fu, L. Song, and X. Wang. Modeling topic-level academic influence in scientific literatures. In *AAAI Workshop on Scholarly Big Data*, 2016.

[26] A. Sinha, Z. Shen, Y. Song, H. Ma, D. Eide, B.-J. Hsu, and K. Wang. An overview of microsoft academic service (MAS) and applications. In *WWW*, 2015.

[27] Z. Tan, C. Liu, Y. Mao, Y. Guo, J. Shen, and X. Wang. Acemap: A novel approach towards displaying relationship among academic literatures. In *WWW,Companion Volume*, 2016.

[28] J. Tang, J. Zhang, L. Yao, J. Li, L. Zhang, and Z. Su. Arnetminer: Extraction and mining of academic social networks. In *KDD*, pages 990–998, 2008.

[29] D. Walker, H. Xie, K.-K. Yan, and S. Maslov. Ranking scientific publications using a model of network traffic. *Journal of Statistical Mechanics: Theory and Experiment*, 2007(06):P06010, 2007.

[30] L. Waltman and E. Yan. *PageRank-Related Methods for Analyzing Citation Networks*, pages 83–100. Springer, 2014.

[31] S. Wang, S. Xie, X. Zhang, Z. Li, P. S. Yu, and Y. He. Coranking the future influence of multiobjects in bibliographic network through mutual reinforcement. *ACM TIST*, 7(4):64:1–64:28, 2016.

[32] Y. Wang, Y. Tong, and M. Zeng. Ranking scientific articles by exploiting citations, authors, journals and time information. In *AAAI*, 2013.

[33] Y. Wu and L. Raschid. Approxrank: Estimating rank for a subgraph. In *ICDE*, 2009.

[34] W. Xing and A. Ghorbani. Weighted pagerank algorithm. In *CNSR*, 2004.

[35] D. Zhou, S. A. Orshanskiy, H. Zha, and C. L. Giles. Co-ranking authors and documents in a heterogeneous network. In *ICDM*, 2007.

[36] Z.-H. Zhou. *Ensemble Methods: Foundations and Algorithms*. Chapman & Hall/CRC, 2012.

[37] Y. Zhu, S. Ye, and X. Li. Distributed pagerank computation based on iterative aggregation-disaggregation methods. In *CIKM*, 2005.

APPENDIX A: Further Experiments

1. Exp-3: Efficiency

In the third of tests, we evaluated the efficiency of our batch and incremental algorithms. We compared our algorithms with powERank, powCTWPR, and powVTWPR, as well as baselines FRank and HRank. The results on DBLP are reported in Fig. 9.

Similar to the results on MAG, when varying Y_s, the running time of all algorithms increases with the increment of Y_s, and the incremental algorithms consistently run faster than other counterparts on DBLP. For TWPageRank, algorithms CTWPR and VTWPR are 1.8 and 2.1 times faster than powCTWPR and powVTWPR on average, respectively. And algorithms incCTWPR and incVTWPR further improve the efficiency by 31% and 35% on average, compared with CTWPR and VTWPR, respectively. For scholarly article ranking, algorithm batERank is on average (1.1, 2.0, 181) times faster than (powERank, FRank, HRank), respectively. And algorithm incERank further improves the efficiency by 27% on average, compared with batERank.

2. Detailed results of Exp-4

We present detailed results of Exp-4, which are omitted earlier due to the space constraints, *i.e.,* the impacts of the time decaying factor σ on efficiency (*Exp-4.1*) and the impacts of parameters α and β on effectiveness (*Exp-4.2*).

Exp-4.1. To evaluate the impacts of the time decaying factor σ, we varied σ from -1.6 to -0.4, while fixed Y_s to default values, $b = +\infty$ and $dif = 1$. The efficiency results are reported in Fig. 10. Note that the running time of algorithm batERank is different on RECOM and FCITA because of the different splitting year Y_s.

When varying σ, the running time of batERank is almost stable on both DBLP and MAG using FCITA and RECOM. Indeed, the running time using (FCITA, RECOM) only varies (1.4%, 0.5%) on DBLP, and (2.1%, 2.2%) on MAG, on average, respectively.

Exp-4.2. To evaluate the impacts of parameters α and β on effectiveness, we varied α and β at the granularity of 0.01, while fixed Y_s to default values, $b = +\infty$, $Dif = 1$ and $\sigma = -1.0$. The results are reported in Fig. 11, where the parameters selected earlier and their corresponding PairAcc are marked with $*$.

When varying α and β, the PairAcc of ERank changes gently, as shown in Fig. 11. The optimal PairAcc is obtained within a single region, rather than a complex collection of optimal regions. Moreover, the PairAcc keeps at a high level within a certain (α, β) combination space around the optimal region. For instance, consider a square of length 0.3, which covers 8.5% of the parameter combination space. The fraction of parameters such that the PairAcc is no worse than 1% of the corresponding PairAcc with marker $*$ is (73%, 94%) on AAN, (96%, 87%) on DBLP and (83%, 95%) on MAG, using (RECOM, FCITA), respectively. Further, the optimal parameters on the same benchmarks are very similar for (AAN, DBLP and MAG), indicating that the setting of α and β can be easily transferred across different datasets. To conclude, ERank is very robust to parameters α and β, and it is quite flexible for choosing proper values of parameters α and β.

APPENDIX B: Detailed Proofs

1. Proof of Lemma 3

The TWPageRank vector PR returned by CTWPR equals to the convergent TWPageRank vector PR^*.

Proof: We have $PR^* = d \cdot M^T \cdot PR^* + \frac{1-d}{n} \cdot e$, as PR^* is the convergent TWPageRank vector. Hence, we also have

$$PR^*(v) = d \cdot \sum_{(u,v)\in E^c} M_{u,v} \cdot PR^*(u) + \frac{1-d}{n}. \quad (12)$$

Consider a topological order $v_1/\ldots/v_n$ on citation graph G^c. We then prove $PR(v_k) = PR^*(v_k)$ ($k \in [1,n]$) by induction.
(1) When $k = 1$, it is obvious that $PR(v_k) = PR^*(v_k) = \frac{1-d}{n}$.
(2) Assume that it holds for $1 \le k \le q$, we then show $PR^*(v_k) = PR(v_k)$ when $k = q+1$ as follows.

$$\begin{aligned} PR^*(v_{q+1}) &= d \cdot \sum_u M_{u,v_{q+1}} \cdot PR^*(u) + \frac{1-d}{n} \\ &= d \cdot \sum_u M_{u,v_{q+1}} \cdot PR(u) + \frac{1-d}{n} = PR(v_{q+1}). \end{aligned}$$

Here $\{u|(u,v_{q+1}) \in E^c\} \subseteq \{v_1,\ldots,v_q\}$. □

2. Proof of Proposition 6

The TWPageRank vector PR returned by VTWPR converges such that $||PR - PR^*||_1 < \epsilon$, where PR^* is the convergent TWPageRank vector.

Proof: We first prove that the sum of changes after another iteration from PR is smaller than ϵ, *i.e.,* $||PR^+ - PR||_1 < \epsilon$ where $PR^+ = d \cdot M^T \cdot PR + \frac{1-d}{n} \cdot e$, and then prove that $||PR^* - PR||_1$ is smaller than the sum of changes.

Consider $scc_1,\ldots,scc_m$ of the venue graph G^v such that $v'_1/\ldots/v'_m$ is indeed a valid topological order of the converted G' of G^v, where m is the number of SCCs in G^v and v'_k ($k \in [1,m]$) is the corresponding nodes of scc_k in G'.

Let PR_k and PR_k^- be the current and the previous TWPageRank vectors of nodes in scc_k produced by VTWPR, and PR_k^+ be the TWPageRank vector of nodes in scc_k extracted from PR^+. Further let $\Delta_k^- = PR_k - PR_k^-$ and we have: $\sum_{k=1}^m ||\Delta_k^-||_1 < \epsilon$.

Consider M_{ij} ($i,j \in [1,m]$), the transition submatrix from scc_i to scc_j. We have $M_{ij} = \mathbf{0}$ when $i > j$, since there exists no edges from nodes in later scc_i back to nodes in earlier scc_j. And, hence, PR_k and PR_k^+ are updated as:

$$\begin{aligned} PR_k &= \frac{1-d}{n} \cdot e_k + d \cdot \sum_{j=1}^{k-1} \cdot M_{jk}^T \cdot PR_j + d \cdot M_{kk}^T \cdot PR_k^-, \\ PR_k^+ &= \frac{1-d}{n} \cdot e_k + d \cdot \sum_{j=1}^{k-1} \cdot M_{jk}^T \cdot PR_j + d \cdot M_{kk}^T \cdot PR_k, \end{aligned}$$

respectively, where $e_k = [1]_{|scc_k|\times 1}$.

Hence, the sum of changes between PR^+ and PR is:

$$\begin{aligned} ||PR^+ - PR||_1 &= \sum_{k=1}^m ||PR_k^+ - PR_k||_1 = \sum_{k=1}^m ||d \cdot M_{kk}^T \cdot \Delta_k^-||_1 \\ &\le d \cdot \sum_{k=1}^m ||\Delta_k^-||_1 < \epsilon. \end{aligned}$$

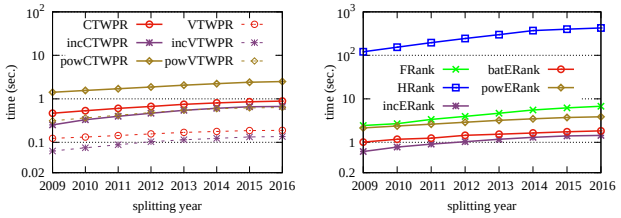
based on the fact that the row sums of M_{kk} are always ≤ 1;

Moreover, $||PR^+ - PR||_1 = ||PR^+ - PR^* + PR^* - PR||_1 = ||d \cdot M^T \cdot (PR - PR^*)||_1 + ||PR - PR^*||_1$, which gives $||PR - PR^*||_1 < \epsilon$ and proves the conclusion. □

3. Proof of Proposition 9

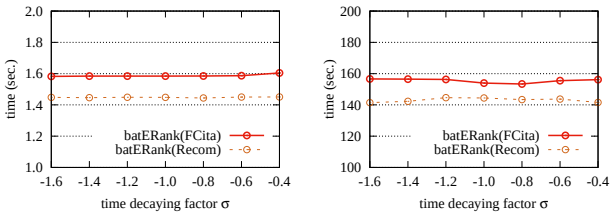
For nodes v within G_A^c, $PR^+(v) = PR(v) \cdot n/n^+$.

Proof: Assume a topological order $v_1/\ldots/v_{n_A}$ of graph G_A^c with $n_A = |V_A^c|$. We prove $PR^+(v_k) = n/n^+ \cdot PR(v_k)$ ($k \in [1,n_A]$) by induction. Note that given $v \in V_A^c$, $\{(u,v)|(u,v) \in E^{c,+}\} = \{(u,v)|(u,v) \in E_A^c\}$, and, hence, $\Delta M_{u,v} = 0$ for $(u,v) \in E^{c,+}$.



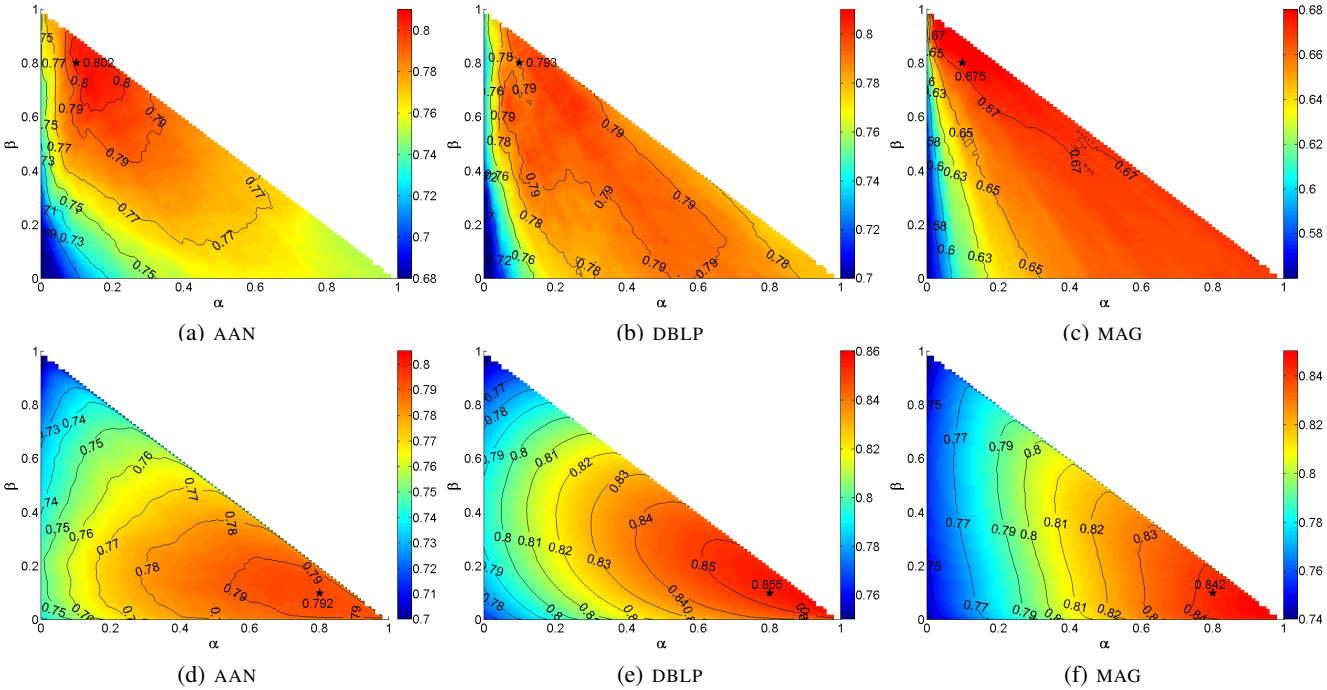
(a) TWPageRank (batch vs. inc.) (b) Comparison of ranking algorithms

Figure 9: Efficiency tests on DBLP



(a) DBLP (b) MAG

Figure 10: Efficiency tests: varying σ



(a) AAN (b) DBLP (c) MAG

(d) AAN (e) DBLP (f) MAG

Figure 11: Accuracy tests on RECOM ((a)–(c)) and FCITA ((d)–(f)): varying parameters α and β

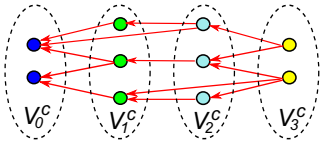


Figure 12: An example of a four-layer citation graph

(1) When $k = 1$, $PR^+(v_k) = n/n^+ \cdot PR(v_k)$ obviously holds since $\{(u, v_1)|(u, v_1) \in E^{c,+}\} = \emptyset$;
(2) Assume that it holds for $1 \leq k \leq q$. We then show $PR^+(v_k) = n/n^+ \cdot PR(v_k)$ for $k = q+1$, since both $(n^+ - n)/n^+ \cdot PR(u) + \Delta PR(u) = 0$ and $\Delta M_{u,v_{q+1}} = 0$ when $(u, v_{q+1}) \in E^{c,+}$. Here $\{u|(u, v_{q+1}) \in E^{c,+}\} \subseteq \{v_1, \ldots, v_q\}$. □

4. A Stronger Convergence Result

Proposotion 1 has shown the convergence of TWPageRank. We further present a stronger convergence result giving the number of iterations needed for power method to achieve convergence, which is based on dividing citation graphs into ordered layers.

Since the citation graph $G^c(V^c, E^c)$ can be treated as a directed acyclic graph, V^c can be organized into ordered layers such that all edges are from later layers to earlier layers. To do this, let l_v be the length of the longest path starting from node v, and L be the length of the longest path starting from any node in G^c, *i.e.,* $L = \max_{v \in V^c} l_v$. Based on l_v, V^c is then divided into $L + 1$ disjoint layers $V_0^c, V_1^c, \ldots, V_L^c$ such that $V_0^c \bigcup V_1^c \bigcup \cdots \bigcup V_L^c = V^c$ and node $v \in V_k^c$ iff $l_v = k$.

Example 2: Fig. 12 illustrates a four-layer citation graph, where L equals to the length of the longest path, *i.e.,* 3, and the nodes are divided into 4 layers $[V_0^c, \ldots, V_3^c]$, such that V_k^c ($k \in [0, 3]$) contains all nodes starting from whom the length of the longest path is exactly k, and all edges are from V_i^c to V_j^c where $i > j$. □

Proposition 11: *TWPageRank converges to a unique vector on an $(L + 1)$-layer citation graph after $L + 1$ iterations, regardless of the initial vector.* □

Proof: Given the initial TWPageRank vector $PR^{(0)}$, the PageRank vector after t iterations is:

$$PR^{(t)} = d^t \cdot (M^T)^t \cdot PR^{(0)} + \frac{1-d}{n} \cdot \sum_{k=1}^{t-1} (d \cdot M^T)^k \cdot e + \frac{1-d}{n} \cdot e, \quad (13)$$

which is derived by iteratively computing $PR^{(1)}$ up to $PR^{(t)}$.

Without loss of generality, we consider G^c whose nodes are properly arranged such that nodes in V_0^c come first, followed by ones in V_1^c till V_L^c. In this case the transition matrix M of G^c is a strictly lower triangular matrix since all edges are from V_i^c to V_j^c where $i > j$. Moreover, $M^{L+1} = \mathbf{0}$.

When $t \geq L + 1$, the first term in the right hand of Eq. (13) becomes $\mathbf{0}$, and $PR^{(t)}$ equals to $\frac{1-d}{n} \cdot \sum_{k=1}^{L} (d \cdot M^T)^k \cdot e + \frac{1-d}{n} \cdot e$, which is the unique convergent TWPageRank vector of G^c, regardless of the initial vector $PR^{(0)}$. □