Query Independent Scholarly Article Ranking

Shuai Ma, Chen Gong, Renjun Hu, Dongsheng Luo, Chunming Hu and Jinpeng Huai
SKLSDE Lab, Beihang University, Beijing, China
Beijing Advanced Innovation Center for Big Data and Brain Computing, Beijing, China
{mashuai, gongchen, hurenjun, lds1995, hucm, huaijp}@buaa.edu.cn

***Abstract*—Ranking query independent scholarly articles is a practical and difficult task, due to the heterogeneous, evolving and dynamic nature of entities involved in scholarly articles. To do this, we first propose a scholarly article ranking model by assembling the importance of involved entities (*i.e.,* articles, venues and authors) such that the importance is a combination of *prestige* and *popularity* to capture the evolving nature of entities. To compute the prestige of articles and venues, we propose a novel Time-Weighted PageRank that extends traditional PageRank with a time decaying factor. We then develop a batch algorithm for scholarly article ranking, in which we propose a block-wise method for Time-Weighted PageRank in terms of an analysis of the citation characteristics of scholarly articles. We further develop an incremental algorithm for dynamic scholarly article ranking, which partitions graphs into *affected and unaffected areas*, and employs different updating strategies for nodes in different areas. Using real-life data, we finally conduct an extensive experimental study, and show that our approach is both effective and efficient for ranking scholarly articles.**

I. Introduction

Query independent ranking of scholarly articles has drawn significant attentions from both academia [1]–[11] and industry [12]–[14]. Generally speaking, a ranking is *a function that assigns each item a numerical score*. Query independent ranking aims to give a static ranking based on the scholarly 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.

In the academia, popular approaches have witnessed a shift from citation-count analysis [1], [2] to graph analysis [3]–[11], as graph-based methods further leverage the global or local structure of bibliographic networks and the interactions among heterogeneous entities, and, hence, are typically more appropriate. Efforts have also been made from the industry. Google Scholar [12] aims to rank articles in the way researchers do, weighing the full text, where they were published, who they were written by, as well as how often and how recently they have been cited; Microsoft Academic [13] considers how often and to which a publication is cited to determine the ranking; And Semantic Scholar [14] proposes to use the citation velocity, which is a weighted average number of article citations in the last three years.

Scholarly articles are involved with multiple entities such as authors, venues, dates and references. That is, 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 difficult task due to the heterogeneous, evolving and dynamic natures of involved entities [15], [16].

First, even if we are only to rank one type of entities (*i.e.,* scholarly articles), the 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, the importance of articles evolves with time in a complex manner [17], [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, which conforms to the universal citation pattern of articles such that the number of citations generally grows in the first two to three years, and then declines in the following years [18]. In addition to the universal one, individual articles indeed follow a diverse set of patterns featured by the peak time of the number of citations [18]. Finally, academic data is dynamic and continuously growing. Indeed, the number of articles in Microsoft Academic Graph has exceeded 126 million, and keeps increasing at around 5.7 million per year [13]. This may cause certain long-term biases into data, *e.g.,* the number of citations increases significantly over time [19], which should be properly considered for scholarly article ranking.

Query independent ranking of scholarly articles remains challenging [20], although there exists quite a bit of work on scholarly article ranking, *e.g.,* [1], [3]–[5]. Most previous work exploits the time-dependent information of scholarly data in the form of exponential decay [8]–[11], which fails to capture the diverse citation patterns of individual articles [18]. Further, to our knowledge, little concern has been paid to dynamic scholarly article ranking except [21] with a strong and impractical assumption that there are no citations between articles published in the same years.

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

(1) We first propose a Scholarly Article Ranking model, referred to as SARank, by assembling the importance of three classes of entities (articles, venues and authors) for scholarly article ranking (Section II). The importance is a combination of *prestige* and *popularity* to capture the evolving nature of entities. To compute the prestige of articles and venues, we propose a novel *Time-Weighted PageRank* with a time decaying factor based on the citation statistics (instead of simple exponential decay), and the prestige of authors is the

average prestige of all their published articles. The popularity of an article is the sum of all its citation freshness (closeness to the current year), while the one of venues and authors is the average popularity of their associated articles. To our knowledge, our Time-Weighted PageRank is among the first to incorporate diverse citation patterns of individual articles and to exploit citation statistics for scholarly article ranking.

(2) We then develop a batch algorithm for scholarly article ranking (Section III), in which we propose a block-wise method for Time-Weighted PageRank in terms of an analysis of the citation characteristics of scholarly articles.

(3) We further develop an incremental algorithm for the block-wise algorithm to deal with dynamic scholarly article ranking (Section IV), which partitions graphs into *affected and unaffected areas*, and employs different updating strategies for nodes in affected and unaffected areas.

(4) Using three real-life scholarly datasets (AAN, DBLP and MAG) and two sets of ground-truth (RECOM and PFCTN), we finally conduct an extensive experimental study (Section V). (a) We find that our model SARank improves the pairwise accuracy [22] over (PRank [23], FRank [10], HRank [3]) by (13.5%, 6.8%, 4.8%) and (12.0%, 3.0%, 3.2%) *w.r.t.* RECOM and PFCTN on AAN, (12.7%, 5.0%, 4.9%) and (14.0%, 6.5%, 4.6%) *w.r.t.* RECOM and PFCTN on DBLP, and (6.5%, 2.5%, 2.2%) and (13.4%, 6.0%, 2.4%) *w.r.t.* RECOM and PFCTN on MAG, on average, respectively. (b) Our batch algorithm batSARank and incremental algorithm incSARank are also efficient. Indeed, incSARank is on average (1.7, 2.8, 116) and (2.0, 4.4, 245) times faster than (batSARank, FRank, HRank) on the large DBLP and MAG, respectively.

II. RANKING MODEL

In this section, we first present Time-Weighted PageRank for evaluating the importance of entities, defined as a combination of the prestige and popularity, and then introduce our ranking model SARank that assembles the importance of articles, venues and authors involved in scholarly articles.

A. Time-Weighted PageRank

We first present Time-Weighted PageRank (TWPageRank) based on citation statistics, as the direct use of PageRank for ranking scholarly articles is problematic as discussed below.
(1) Different articles typically have different impacts in practice, and there is a need to differentiate, while PageRank essentially assumes equal impacts.
(2) The semantics of citation relationships are time-dependent, which means that citations at different periods of time may reveal different information. Note that this has already been exploited for scholarly article ranking [8], [9], [11], while PageRank does not consider this temporal factor at all.

Time-Weighted PageRank (TWPageRank). Most previous work simply exploits temporal information in the form of exponential decay [8]–[11]. We rethink the usage of time information in terms of the impacts of scholarly articles. Recall that articles are categorized into six citation patterns featured

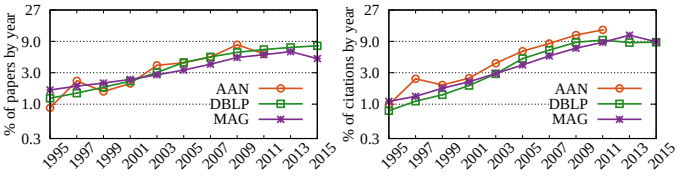


Figure 1. Statistics of scholarly articles: (1) the (logscale) percentage of papers in each year *w.r.t.* all papers (left) and (2) the (logscale) percentage of citations in each year *w.r.t.* all citations (right) on datasets AAN, DBLP and MAG, respectively.

by the time when the articles reach their citation peaks [18]: (a) *PeakInit* with a single citation-count peak in the first five years (but not the first year) after publication, (b) *PeakMul* with distinct multiple peaks, (c) *PeakLate* with a single peak in at least five years (but not the last year) after publication, (d) *MonDec* with monotonically decreasing citations, (e) *MonIncr* with monotonically increasing citations, and, (f) *Other* for articles whose average numbers of citations per year are less than 1. Though the number of citations is an indicator of the impact of an article [1], [17], its impact is time-dependent but not simply in the form of exponential decay that only considers the case of *MonDec*. To unify these distinct citation patterns and to make our ranking model succinct, we adopt that *the impact of an article tends to decay with time after the peak time*, as pointed out by the aging function in [17]. That is, the impacts of articles directly decay with time only for those in *MonDec*, and decay with time after the (highest) peak time for those in *PeakInit*, *PeakLate* and *PeakMul*, and do not decay for ones in *MonIncr* (which is rare). Also observe that *each individual article has its own peak time* as articles may reach their citation peaks in different patterns and time.

Based on the above discussion, we propose TWPageRank that evaluates the prestige of nodes (*e.g.,* scholarly articles) in a directed graph, such that each node is attached with time information. It differs from PageRank by weighing the influence propagation using the *impact weights on edges*, which represent the relative amounts of time-dependent prestige that should be propagated from the edge sources to targets. Formally, the impact weight on a directed edge (u, v), *i.e.,* an edge from u to v, is defined as:

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

where T_u is the time of node u, $Peak_v$ is the peak time of node v after which the impact weights of edges to v decay with time, and σ is a negative number controlling the decaying speed of the impacts. 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 a half per year when $\sigma = -1$.

For scholarly article ranking, T_u is the publication time of article u and $Peak_v$ should be ideally set to the time when article v has the highest impact. Basically, it could simply be the year when article v obtains the largest number of citations.

However, recent work reveals that the volume of scientific publications and the number of citations grow exponentially with time [19], [24]. We also collect and report the volume and citation statistics on three scholarly datasets in Fig. 1, which verifies the exponential distribution. Hence, we adopt the *scaled* number of citations $\Psi_v^{(t)} = \Phi_v^{(t)} / \log Z^{(t)}$ such that $\Phi_v^{(t)}$ and $Z^{(t)}$ are the number of citations of article v at year t and the total number of citations at year t, respectively. The peak time $Peak_v$ is the time that maximizes $\Psi_v^{(t)}$.

The update rule of TWPageRank is:

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

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 the impact weights on all edges from u, n is the number of nodes and d is a damping parameter in $(0,1)$. By Eq. (2), we can see that prestige is updated based on the impact weights, not equally distributed.

Correspondingly, the matrix form of the update rule is:

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

Here $PR^{(k)}$ is the TWPageRank vector after k iterations, M is the transition matrix such that $M_{u,v} = w(u,v)/W(u)$ and e is an n-dimensional all-one vector $[1]_{n\times 1}$.

The linear system in Eq. (3) is equivalent to *irreducible* and *aperiodic* Markov chains [25], which guarantees the following.

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

B. Ranking with Importance Assembling

In our model, the importance is defined as a combination of the prestige and popularity. Intuitively, prestige favors those with many citations soon after the publication of articles or associated articles of venues and authors, and popularity favors those with recent citations. Both prestige and popularity capture the temporal nature of entities.

Our ranking model SARank, illustrated in Fig. 2, assembles the importance of article, venue and author entities for scholarly article ranking, which is computed by the citation, venue and author components, respectively. We next introduce the details of the three components.

Citation component. The first component computes the importance of articles using the 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 scaled number of citations.

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

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

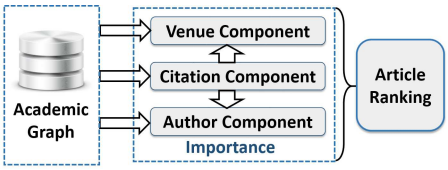


Figure 2. Ranking model SARank

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

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

Intuitively, the more recent citations an article has, the higher its popularity is, no matter how long it has been published. Here popularity is introduced to capture the recent importance of articles, and articles with more recent citations have higher popularity scores. 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 combined to produce the importance of articles. Intuitively, an important article is both prestigious and popular. Hence, the *citation importance score* $Imp_c(v)$ of an article is defined as a weighted combination of its prestige and popularity:

$$Imp_c(v) = Prs_c(v)^{\lambda} Pop_c(v)^{1-\lambda}, \tag{5}$$

where $\lambda \in [0,1]$ is the importance weighting factor. The rationales 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 them; (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 in those of recent attention.

Venue component. The second component computes the importance of venues with their associated articles. As the importance of a venue evolves with time, we treat the venue in each year individually, and its importance is the sum of 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 (in a specific year) s citing articles published in venue (in a specific year) t, and (c) we use the *impact weight* $w_v(s,t)$ to denote the weight from venues s to t, which is the sum of the impact weights from articles published in s to t, *i.e.,*

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

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

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. The prestige and popularity are combined to derive the importance of a venue in a specific year in the same way as the citation component. Finally, the importance of a venue is treated as the *venue importance score* for all articles published in this venue.

Author component. The author component computes the importance of authors with their published articles. Similar to the venue component, we evaluate the importance of each author, and compute the average importance of the authors of an article as its *author importance score*.

One way to do this is to construct an author citation graph such that (a) a node represents an author, and (b) a direct edge (s, t) denotes that there exist articles of author s citing articles of author t. However, it is easy to see that for each citation, the corresponding two sets of authors are fully connected, which makes it computationally expensive to compute the prestige of authors on such an author citation graph with TWPageRank.

Hence, we choose to evaluate the prestige of an author with the average prestige of all articles published by the author. Similar to the venue component, the popularity of an author is defined as the average popularity of her/his published articles. Finally, the prestige and popularity are combined to derive the importance along the same way as the citation component.

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

$$R(v) = \alpha R_c(v) + \beta R_v(v) + (1 - \alpha - \beta) R_a(v). \quad (7)$$

Here aggregating parameters α and β and value $(1 - \alpha - \beta)$ regularize the contributions of the citation, venue and author information, which make our model be able to fit to the various ranking scenarios. As will be seen in the experiments, our model performs well in two reasonable ranking scenarios by using quite different aggregating parameters, and, moreover, these parameters are indeed quite flexible to choose within a certain range. Intuitively, these parameters indicate the intensity of the correlation between the importance of scholarly articles and the specific information.

Remarks. This work follows the graph-based formalization, and further develops efficient batch and incremental algorithms based on graphs for scholarly article ranking (Sections III & IV). However, it is also possible to learn a discriminative model that directly optimizes certain loss functions for ranking, similar to [22] for ranking Web pages.

III. Ranking Computation

In this section, we present our batch algorithm for computing scholarly article ranking based on our model SARank.

A. Algorithm Framework

Our batch algorithm batSARank combines the importance scores computed by the citation, venue and author components. It takes as input academic graph data D and an iteration threshold ϵ for TWPageRank and returns the scholarly article ranking. It first constructs the citation graph $G^c(V^c, E^c)$ and venue graph $G^v(V^v, E^v)$. Then it computes the prestige and popularity of citation, venue and author components. Finally, it combines the prestige and popularity of the three components to produce the final ranking with Eq. (7).

For popularity computation, it is easy to see that (a) the popularity of articles can be computed by scanning through all citations once and adding the freshness of citations to their corresponding articles by Eq. (4), and (b) the popularity of venues in a specific year or authors is computed by averaging the popularity of the articles published in the venues or by the authors. That is, the popularity computation can be done by scanning through all citations once.

For prestige computation, (a) as the one of authors is defined as the average prestige of their published articles, it suffices to scan through all author-article relationships for computing the prestige of authors. (b) The prestige of articles and venues in a specific year is computed by TWPageRank on citation graphs and venue graphs, which is usually computed in an iterative manner [23] and is the most expensive computation. Hence, the key of the computation of our approach is a good solution for computing TWPageRank.

B. TWPageRank Computation

The main result of this section is to speed up computing TWPageRank on scholarly data.

Claim 2: *A block-wise PageRank computation method [26] is a good choice for TWPageRank on scholarly data.* □

The main idea of the block-wise PageRank computation is that each strongly connected component (SCC) of the input graph is treated as a block, and blocks are processed one by one following the *topological order* of the block-wise graph, *i.e.,* each node represents a block of the original graph [26]. We next show Claim 2 by introducing and analyzing such a block-wise computation method.

Block-wise algorithm batTWPR. It takes as input a citation or venue graph $G(V, E)$ and an iteration threshold ϵ, and returns the TWPageRank vector PR of G. To update scores of nodes in an SCC at a time, the edges of G are partitioned into: (a) the set E_i of edges inside SCCs and (b) the set E_a of edges across SCCs such that $E_i \cap E_a = \emptyset$ and $E = E_i \cup E_a$. The update rule in Eq. (3) is revised accordingly to separate E into E_i and E_a as follows.

$$PR(v) = d \sum_{(u,v)\in E_i} M_{u,v} PR(u) + d \sum_{(u,v)\in E_a} M_{u,v} PR(u) + \frac{1-d}{n}. \quad (8)$$

It first computes the block-wise graph G' by treating SCCs in G as single nodes, and derives a topological order O of nodes in G'. It then processes each SCC in the topological order O with Eq. (8). Finally, it returns the TWPageRank vector. When

Table I
STATISTICS OF CITATION/VENUE GRAPHS AND WEB GRAPHS [28]

Graphs	Nodes	Edges	Largest $\|SCC\|$	SCC edge ratio
citation-AAN	18,041	82,944	20	0.9%
citation-DBLP	3,140,081	14,260,658	23	1.6%
citation-MAG	126,909,021	526,498,920	351	0.1%
venue-AAN	565	22,527	18	2.8%
venue-DBLP	56,370	7,094,231	1,467	2.1%
venue-MAG	584,298	162,431,575	10,473	1.8%
web-BS	685,230	7,600,595	334,857	59.51%
web-G	875,713	5,105,039	434,818	66.98%

processing an SCC, it iteratively updates the TWPageRank scores of the nodes in the SCC, and the iteration continues until the sum changes of TWPageRank scores is less than $\epsilon|scc|/|V|$, where $|scc|$ is the number of nodes in the SCC. Note that there must exist a topological order O, as the block-wise graph is a directed acyclic graph [27].

Corollary 3: *The vector PR returned by* batTWPR *converges such that $||PR - PR^*||_1 < \epsilon$ where vector PR^* is the convergent TWPageRank vector [26].* □

Analysis of algorithm batTWPR. While similar block-wise algorithms were originally proposed for Web graphs [26], we next show that they are even better for the TWPageRank computation associated with scholarly data. Given inpute graph $G(V, E)$, the block-wise graph and its topological order can be computed in $O(|V| + |E|)$ time [27], and the edges in E_a are only scanned once when updating TWPageRank scores. From these, we have the following.

Lemma 4: *Given a citation or venue graph $G(V, E)$, algorithm* batTWPR *runs in $O(|V| + |E_a| + t|E_i|)$ time, where t is the maximum number of iterations among all* SCCs. □

Recall that t is very likely to be in the scale of tens to hundreds [23]. Hence, the efficiency of block-wise algorithm batTWPR is mainly affected by $|E_i|$, *i.e.,* the smaller $|E_i|$ is, the faster algorithm batTWPR is.

Observation. It is well-known that citations obey a natural *temporal order*, *i.e.,* an article only cites those published earlier, and it is really rare for the mutual citations between two articles published in the same time. That is, $|E_i|$ *is essentially small for the citation and venue graphs of scholarly data.*

To verify this observation, we also collect the statistics of citation/venue graphs and Web graphs illustrated in Table I, where Web graphs are extracted from berkely.edu and stanford.edu domains in 2002 and from the Google programming contest in 2002, respectively [28]. Due to the existence of the "bow tie" structure and the giant SCC in Web graphs [29], the SCC edge ratios $|E_i|/|E|$ are greater than 59%. In contrast, the SCCs in citation and venue graphs are quite small as a result of the *temporal order of citations*, and $|E_i|/|E|$ is less than 3% for all tested citation and venue graphs. This specific structure in scholarly data has been long ignored in the past literature, which indeed has a positive impact on computations. Taking $t = 100$ for example, algorithm batTWPR only needs to scan $4|E|$ edges on citation and venue graphs, but over $59|E|$ edges on Web graphs.

By Corollary 3, Lemma 4 and the above analysis of our block-wise algorithm, we have informally established Claim 2.

Time complexity analysis of the batch algorithm. By Lemma 4 and the analyses in Section III-A, one can verify that algorithm batSARank takes $O(|V^c| + |E_a^c| + t|E_i^c| + |V^v| + |E_a^v| + t|E_i^v| + |PA|)$ time, where $|PA|$ is the number of author-article relationships. The key of algorithm batSARank is to compute TWPageRank with algorithm batTWPR. Compared with the traditional power method [23], our block-wise algorithm batTWPR speeds up computation by $O((t-1)(|E_a^c| + |E_a^v|))$ at an extra space cost for the block-wise graphs of G^c and G^v and their topological orders.

IV. DYNAMIC RANKING COMPUTATION

Scholarly data is dynamic and continuously growing, and it is impractical to recompute ranking from scratch once it gets updated. In this section, we present an incremental algorithm for our ranking model SARank.

A. Incremental Algorithm Framework

Our incremental algorithm incSARank incrementally computes the popularity and prestige of associate entities. We consider that an update $\Delta = \Delta V \cup \Delta E$ is added to a (citation or venue) graph $G(V, E)$, and the resulting graph is $G^+(V \cup \Delta V, E \cup \Delta E)$, where ΔV is a set of nodes with $\Delta V \cap V = \emptyset$, and ΔE is a set of directed edges on ΔV and from ΔV to V only, as citation relationships obey a natural temporal order, *i.e.,* an article only cites those published earlier, and it is rare for the mutual citations between two articles published in the same time.

Incremental popularity computation. As the popularity of articles is defined as the sum of the freshness of their citations, it is convenient to maintain dynamically. Consider an updated citation graph $G^{c,+}(V^c \cup \Delta V^c, E^c \cup \Delta E^c)$ of $G^c(V^c, E^c)$, and the updated popularity $Pop_c^+(v)$ can be computed as:

$$Pop_c^+(v) = Pop_c(v)e^{\sigma(T_0^+ - T_0)} + \sum_{(u,v)\in\Delta E^c} e^{\sigma(T_0^+ - T_u)}, \quad (9)$$

where $Pop_c^+(v)$ (resp. $Pop_c(v)$) is the popularity of node v on $G^{c,+}$ (resp. G^c), and T_0^+ (resp. T_0) is the current time on $G^{c,+}$ (resp. G^c). By Eq. (9), it is easy to see that updating the popularity of articles takes $O(\ |V^c| + |\Delta V^c| + |\Delta E^c|)$ time.

The popularity of venues and authors is computed along the same lines as their batch counterparts of algorithm batSARank, as almost all venues and authors are affected by the definitions of the popularity of venues and authors.

Incremental prestige computation. The prestige of authors is computed along the same lines as the batch algorithm batSARank, as almost all authors are affected by the definition of the prestige of authors. For articles and venues, we propose an incremental algorithm to maintain their prestige.

B. Incremental TWPageRank Computation

Consider a citation or venue graph $G(V, E)$, its TWPageRank vector PR and the topological order O of its block-wise graph. Given an update $\Delta = \Delta V \cup \Delta E$ to G, the incremental

Input: An update $\Delta = \Delta V \cup \Delta E$, TWPageRank vector PR of G, and the topological order O of the block-wise graph G'.
Output: TWPageRank vector PR^+ of the updated graph G^+.
1. G'_C := the block-wise graph of G_C;
2. ΔO := topological order of G'_C; $O^+ := \Delta O / O$;
3. label SCCs of G_C as C, SCCs of G with outgoing edges having weight changes as B, and the remaining SCCs of G as A;
4. **for** each node v' following O^+ **do**
5. scc := the corresponding SCC of v';
6. **if** scc is labeled as C **then**
7. update $PR^+(v)$ ($v \in scc$) following algorithm batTWPR;
8. label SCC w' as B with $w' \in G'$ and $(v', w') \in E^{+\prime}$;
9. **else if** scc is labeled as B **then**
10. update $PR^+(v)$ where $v \in scc$ with Eq. (10) until the sum of TWPageRank score changes is less than $\epsilon \cdot \frac{|scc|}{|V^+|}$;
11. label SCC w' as B with $(v', w') \in E'$;
12. **else** $PR^+(v) := PR(v) \cdot n/n^+$ where $v \in scc$;
13. **return** PR^+.

Figure 3. Algorithm incTWPR for incremental TWPageRank

prestige computation for articles and venues in a specific year is to compute the TWPageRank vector PR^+ on the updated graph $G^+(V \cup \Delta V, E \cup \Delta E)$.

Auxiliary data structure maintenance. Two auxiliary data structures in the batch algorithm batTWPR need to be maintained: (a) the block-wise graph and a mapping that, given a node of the citation or venue graph, returns the index of the SCC to which it belongs, and (b) the topological order of the nodes in the block-wise graph. Observe that these auxiliary data structures can be easily maintained as follows.

(1) The block-wise graph of G^+ needs to be computed, whose SCCs consist of the SCCs in G and SCCs in the induced subgraph $G^+[\Delta V]$, as the edges of ΔE are on nodes in ΔV and from ΔV to V only. Hence, only those new SCCs in $G^+[\Delta V]$ need to be computed.

(2) The updated topological order $O^+ = \Delta O / O$, where ΔO is the topological order of the block-wise graph of induced subgraph $G^+[\Delta V]$. Hence, only ΔO needs to be computed. One can easily verify the following.

Proposition 5: *$O^+ = \Delta O / O$ is indeed a valid topological order of the block-wise graph of G^+.* □

Analysis of affected and unaffected areas. The TWPageRank vector PR of graph G is mainly affected in two ways.

(1) Let $V_{B,1} \subseteq V$ be the set of nodes reachable from the newly added nodes ΔV, $V_{B,2} \subseteq V$ be the set of nodes with outgoing edges having weight changes, and $V_{B,3} \subseteq V$ be the set of nodes reachable from $V_{B,2}$. Then $V_B = V_{B,1} \cup V_{B,2} \cup V_{B,3}$ is obviously the set of nodes in G affected by the update Δ. TWPageRank scores on V_B are re-iterated as follows, where notations with superscript '+' are defined on G^+.

$$PR^+(v) = d \sum_{(u,v) \in E_i^+} M^+_{u,v} PR^+(u) + d \sum_{(u,v) \in E_d^+} M^+_{u,v} PR^+(u) + \frac{n}{n^+} \Big(PR(v) - d \sum_{(u,v) \in E_i} M_{u,v} PR(u) - d \sum_{(u,v) \in E_a} M_{u,v} PR(u) \Big). \quad (10)$$

(2) Let $V_A = V \backslash V_B$. Since nodes in V_A are not reachable from newly added or affected nodes, V_A is essentially not affected by the update Δ. And TWPageRank scores on V_A only need to scale with constant n/n^+.

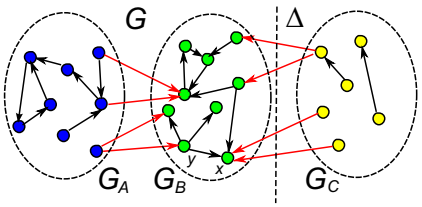


Figure 4. An example of incremental TWPageRank computation

Let $G_A = (V_A, E_A)$, $G_B = (V_B, E_B)$ and $G_C = (V_C, E_C)$, respectively, and let E_{AB} and E_{CB} be the sets of edges from G_A to G_B and from G_C to G_B, respectively. In this way, graph G^+ is divided into subgraphs $\{G_A, G_B, G_C\}$ and edge sets $\{E_{AB}, E_{CB}\}$. We then have $G^+[\Delta V] = G_C$, $\Delta E = E_C \cup E_{CB}$, $V = V_A \cup V_B$ and $E = E_A \cup E_B \cup E_{AB}$.

Incremental algorithm incTWPR. We now present our incremental algorithm for TWPageRank, shown in Fig. 3.

It takes as input an update Δ and the previous results on the original graph $G(V, E)$, and returns the TWPageRank vector of the updated graph G^+. It first incrementally computes the topological order O^+ (lines 1–2). After that, it labels the newly added SCCs with C and existing SCCs with A or B, depending on whether the existing SCCs have weight changes on outgoing edges (line 3). It then processes each SCC in the order O^+ such that the TWPageRank scores of nodes in each SCC are updated according to the labels (lines 4–12), and finally returns the TWPageRank vector (line 13).

When processing V_B with Eq. (10), *edges in E_{AB} can be skipped* since $PR^+(u) = n/n^+ \cdot PR(u)$ for $u \in V_A$ and $M_{u,v} = M^+_{u,v}$ for $(u, v) \in E_{AB}$. Besides, we use $n/n^+ \cdot PR$ as the initial vector. Both of them can speed up the computation.

Example 1: Figure 4 illustrates an example of incremental TWPageRank computation. Consider an update Δ on the original graph G. It is obvious that the update Δ has no impacts on the SCCs of G, and O^+ defined earlier is a valid topological order of $G^{+\prime}$. The original graph G is then partitioned into affected and unaffected areas, and subgraphs G_A, G_B and G_C are associated with node sets V_A, V_B and ΔV, respectively. Here edge weight on (y, x) changes due to the change of the peak time of node x, and, hence, node y as well as all nodes reachable from y are included in G_B. When updating the TWPageRank scores, following O^+, scores of nodes in G_C, G_B and G_A are computed by iterations from scratch, by iterations with Eq. (10) using the existing TWPageRank vector and by scaling, respectively. □

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

Observe that (a) a topological order of the block-wise graph of G_C can be computed in $O(|\Delta V| + |\Delta E|)$ time, (b) updating the TWPageRank scores of nodes in V_B and V_C costs $O(|V_B \cup V_C| + |E_{B,a} \cup E_{C,a} \cup E_{CB}|) + t^+|E_{B,i} \cup E_{C,i}|)$ time, where

Table II
STATISTICS OF AFFECTED AND UNAFFECTED AREAS

Statis.	Citation graphs on			Venue graphs on		
	AAN	DBLP	MAG	AAN	DBLP	MAG
$\|V_A\|$	47.4%	52.3%	69.2%	2.1%	8.7%	12.4%
$\|V_B\|$	46.8%	40.0%	26.3%	92.0%	84.8%	84.6%
$\|V_C\|$	5.8%	7.8%	4.5%	5.8%	6.4%	3.0%
$\|E_A\|$	3.0%	2.4%	0.9%	0.0%	0.0%	0.0%
$\|E_{AB}\|$	26.5%	30.2%	26.6%	1.2%	0.2%	0.1%
$\|E_B\|$	59.8%	59.3%	65.5%	88.6%	92.3%	92.6%
$\|E_{CB}\|$	10.4%	7.2%	7.0%	10.0%	7.3%	7.1%
$\|E_C\|$	0.3%	0.9%	0.1%	0.2%	0.2%	0.1%

t^+ is the maximum number of iterations among all SCCs in G_B and G_C, and, finally, (c) updating the scores of nodes in V_A costs $O(|V_A|)$ time. From these, the following holds.

Proposition 7: *Given an update $\Delta = \Delta V \cup \Delta E$ of citation or venue graph $G(V, E)$, the TWPageRank vector of G and the topological order of G', algorithm* incTWPR *runs in $O(|V \cup \Delta V| + |E_{B,a} \cup \Delta E| + t^+|E_{B,i} \cup E_{C,i}|)$ time.* □

By Propositions 1 & 5 and Theorem 6, one can easily verify the correctness of algorithm incTWPR. Note that (a) algorithm incTWPR computes SCCs and derives the topological order based on Δ only, instead of G^+, (b) it skips edges in $E_A \cup E_{AB}$ when updating the scores of nodes in V_A and V_B, and (c) the number t^+ is very likely smaller than the number t of batTWPR when updating scores of nodes in V_B. All these make incTWPR faster than batTWPR even though they have very similar time complexity.

Time complexity analysis of the incremental algorithm. By the analyses above, the time complexity of incSARank is the same as batSARank, except that incSARank saves $O(|E_A^c \cup E_{AB}^c|)$ and $O(|E_A^v \cup E_{AB}^v|)$ time when computing TWPageRank on the updated citation and venue graphs with an extra space cost for the affected/unaffected division and the copy of original edge weights.

Despite of its similar time complexity to batSARank, algorithm incSARank typically achieves a substantial efficiency improvement over batSARank, according to our statistics of affected/unaffected areas given a yearly update, *i.e.,* articles of 2011 on AAN and 2015 on DBLP and MAG, respectively, shown in Table II. (a) It saves $O(|V^c| + |E^c|)$ and $O(|V^v| + |E^v|)$ time when maintaining SCCs and the topological order based on the update data only, where $(|V|, |E|)$ are more than (92%, 89%) of $(|V^+|, |E^+|)$ on all tested citation and venue graphs; (b) It saves $O(|E_A^c \cup E_{AB}^c|)$ time when updating scores on V^c, where edges in $E_A^c \cup E_{AB}^c$ account for more than 28% of total; (c) It saves $O(|E^c|)$ time when computing popularity of articles, which accounts for more than 89% of total; Finally, (d) it is likely to compute TWPageRank scores on V_B^c and V_B^v with less iterations.

V. EXPERIMENTAL STUDY

In this section, we present an extensive experimental study of our approach SARank, compared with three competitive methods. Using three real-life scholarly datasets (AAN, DBLP and MAG) and two sets of ground-truth (RECOM and PFCTN), we conducted five sets of experiments to evaluate: (1) the effectiveness of SARank, (2) the efficiency of our batch algorithm batSARank and incremental algorithm incSARank, (3) the memory cost, and (4) the impacts of parameters.

A. 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 at ACL conferences from the year of 1965 to 2011 [3]. It contains 18,041 articles, 14,386 authors, 273 venues and 82,944 citations.
(2) DBLP records articles in the computer science domain from 1936 to 2016 [30]. It contains 3.14 million articles, 1.74 million authors, 11,619 venues and 6.38 million citations.
(3) MAG records articles of various disciplines from 1800 to 2016 [13]. It contains around 127 million articles, 115 million authors, 24,024 venues and 529 million citations.

To alleviate the issue of citation missing in DBLP, we added citations by title matching based on MAG, and finally the total number of citations is 14.26 million. These datasets were further cleaned by deleting self-citations and citations from old articles to new ones, which accounted for (0.1%, 0.8%, 0.4%) of the total citations on (AAN, DBLP, MAG), respectively.

Accuracy metric and ground-truth. We adopted the *pairwise accuracy* introduced by Microsoft [20], [22] to evaluate the ranking quality, *i.e.,* the fraction of times that a ranking agrees with the correct ranking orders of scholarly article pairs:

$$\mathsf{PairAcc} = \frac{\#\text{ of agreed pairs}}{\#\text{ of all pairs}}. \tag{11}$$

We constructed two sets of ground-truth importance ranking orders of article pairs, referred to as RECOM and PFCTN.
(1) RECOM assumes that scholarly articles with more recommendations are of higher importance. We used the number of recommendations of 93 articles on AAN [3], and, by exact title matching, generated (2133, 966, 1972) scholarly article pairs on (AAN, DBLP, MAG), respectively.
(2) PFCTN assumes that scholarly articles with more citations are of higher importance. However, the number of entire citations is obviously biased to old articles. Some work adopts the number of future citations [6], [8], [9], which is also not appropriate since this only estimates future impacts of articles, not at the concerned time. For a fair ranking benchmark, we propose to use both past and future citations with the same period of time *w.r.t.* the concerned time, such that the number of citations within these two periods reveals the importance of articles at the concerned time. We hence divide each dataset into two parts with a splitting (concerned) time such that (a) the data before the splitting time is used for ranking model, (b) the remaining part of data is used to collect future citations, and (c) the most recent part of the data for ranking model with the same time span as the future citations is used to collect past citations. Moreover, articles in the same pairs were required to be in similar research fields, by utilizing the Fields-Of-Study information on MAG [13], and published in the same years, similar to [6]. We used all pairs (around 50,000) for AAN, and randomly chose 300,000 pairs for both DBLP and MAG.

Table III
ACCURACY EVALUATION WITH RECOM

Datasets	PRank	FRank	HRank	SARank
AAN	0.671	0.738	0.758	**0.805**
DBLP	0.651	0.729	0.730	**0.778**
MAG	0.615	0.655	0.658	**0.680**

Algorithms. We compared our approach with three competitive methods: PRank [23], FRank [10] and HRank [3].
(1) PRank (PageRank) is a classic method that uses only citation information to rank scholarly articles.
(2) FRank (FutureRank) combines citation, temporal and other heterogeneous information to rank scholarly articles.
(3) HRank (HHGBiRank) is a very recent method using both citation and 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}, respectively, (b) the default splitting years (time) were selected such that the part of data for ranking model accounted for around 75% of the entire data, which were 2008 on AAN and 2012 on both DBLP and MAG, and, (c) for the sake of fairness, aggregating parameters of FRank, HRank and SARank were tuned at the granularity of 0.1 and the best results were reported. Moreover, ρ was set to -0.2 for FRank following [10], and the time decaying factor σ and the importance weighting factor λ were set to -1 and 0.5 by default for SARank.

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

B. Experimental Results

We next present our findings.

Exp-1: Effectiveness with RECOM. In the first set of our tests, we used ground-truth RECOM to evaluate the effectiveness of our approach. All algorithms used articles published before 2012, since article pairs of RECOM were from this portion of articles. Aggregating parameters were selected as follows: $(\alpha, \beta, \gamma) = (0.1, 0.2, 0.2)$ for FRank, $(a_{i1}, a_{i2}, a_{i3}) = (0.6, 0.2, 0.2)$ for HRank ($i \in [1, 3]$), and $(\alpha, \beta) = (0.1, 0.8)$ for SARank. The results of PairAcc are reported in Table III.

The PairAcc of PRank is much lower than the one of other algorithms, indicating that citation information alone is insufficient for scholarly article ranking, and other information helps to refine the results. Moreover, SARank consistently ranks better than all competitors. Indeed, SARank improves the PairAcc over (PRank, FRank, HRank) by (13.5%, 6.8%, 4.8%) on AAN, (12.7%, 5.0%, 4.9%) on DBLP, and (6.5%, 2.5%, 2.2%) on MAG, respectively.

Exp-2: Effectiveness with PFCTN. In the second set of tests, we used ground-truth PFCTN to evaluate the effectiveness. Aggregating parameters were selected as follows: $(\alpha, \beta, \gamma) = (0.7, 0.1, 0.2)$ for FRank, $(a_{i1}, a_{i2}, a_{i3}) = (0.3, 0.6, 0.1)$ for HRank ($i \in [1, 3]$), and $(\alpha, \beta) = (0.8, 0.1)$ for SARank. Note that with PFCTN the values of paraments α and β for SARank are quite different from the ones with RECOM. To evaluate the effectiveness of ranking in different scenarios, we varied three factors in our tests: the splitting year Y_s, the number T_p of published years of articles, and the difference dif of past and future citation counts. Given Y_s, T_p and dif, we only used article pairs whose articles were published within $[Y_s - T_p, Y_s)$ and the difference of past and future citation counts was equal to or larger than dif to test the PairAcc.

Exp-2.1. To evaluate the effectiveness of ranking *w.r.t. short-term and long-term importance*, we varied the splitting year Y_s from 2006 to 2011 on AAN and from 2010 to 2015 on both DBLP and MAG, while fixed $T_p = +\infty$ and $dif = 1$, *i.e.,* using all scholarly article pairs. Intuitively, large and small Y_s correspond to short-term and long-term importance, respectively. The results of PairAcc are reported in Figs. 5(a), 5(f) and 5(k), in which the red markers □ in dashed lines mean that 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, indicating that it is easier to assess short-term (large Y_s) than long-term (small Y_s) importance. While the results on AAN do not follow this trend, possibly because AAN does not record the complete articles of 2007 and 2009. Moreover, SARank consistently ranks better than all competitors, regardless of assessing short-term or long-term importance. Indeed, SARank improves the PairAcc over (PRank, FRank, HRank) by (17.9%, 5.4%, 5.5%) on AAN, (18.6%, 7.7%, 5.8%) on DBLP, and (16.7%, 7.2%, 2.9%) on MAG, respectively.

Exp-2.2. To evaluate the effectiveness of ranking *w.r.t. the published time of articles*, we varied the number T_p of published years from 1 to $+\infty$, while fixed Y_s to default values of three datasets and $dif = 1$, respectively. The results of PairAcc are reported in Figs. 5(b), 5(g) and 5(l).

When varying T_p, the PairAcc of all algorithms increases with the increment of T_p, since old articles (large T_p) are easier to rank based on adequate information, while new articles (small T_p) are hard to rank with little information available. Moreover, SARank consistently ranks better than all competitors, especially when $T_p \leq 3$, *i.e.,* ranking recently published articles. Indeed, SARank improves the PairAcc over (PRank, FRank, HRank) by (19.0%, 3.1%, 3.9%) on AAN, (25.0%, 8.2%, 6.3%) on DBLP, and (23.6%, 8.3%, 3.2%) on MAG, on average, respectively.

Exp-2.3. To evaluate the effectiveness of ranking *w.r.t. the difference of past and future citations*, we varied the difference dif of past and future citation counts from 1 to 7, while fixed Y_s to default values of three datasets and $T_p = +\infty$. The results of PairAcc are reported in Figs. 5(c), 5(h) and 5(m).

When varying dif, the PairAcc of all algorithms increases with the increment of dif, since pairs with larger dif are easier to rank. Moreover, SARank consistently ranks better, regardless of easy or difficult article pairs. Indeed, SARank improves the PairAcc over (PRank, FRank, HRank) by (12.0%, 3.0%,

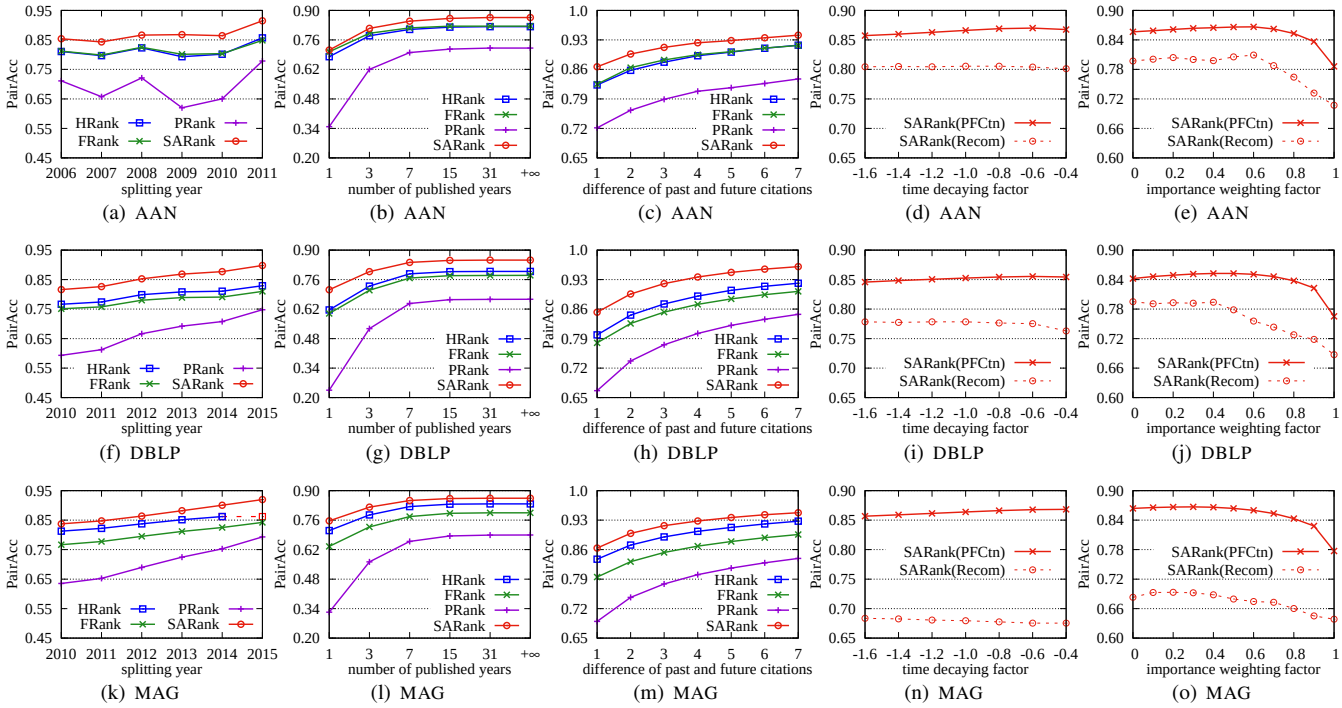


Figure 5. Accuracy evaluation with PFCTN (all) and RECOM ((d)–(e), (i)–(j) and (n)–(o))

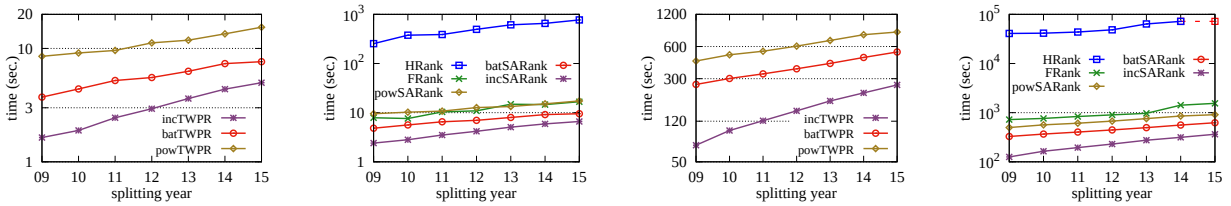


Figure 6. Efficiency evaluation on DBLP ((a)–(b)) and MAG ((c)–(d))

3.2%) on AAN, (14.0%, 6.5%, 4.6%) on DBLP, and (13.4%, 6.0%, 2.4%) on MAG, on average, respectively.

Exp-3: Efficiency. In the third set of tests, we evaluated the efficiency of our algorithms. We compared our algorithms with powTWPR and powSARank, which were the same to batTWPR and batSARank except using power method for TWPageRank computation, and with algorithms FRank and HRank. Here PRank was omitted due to its effectiveness. We varied the splitting year Y_s from 2009 to 2015 and tested the running time on both DBLP and MAG. For incremental algorithms, base and update parts consisted of data before 2008 and within $[2008, Y_s)$, respectively. The results of running time are reported in Fig. 6, where the red markers □ in dashed lines mean that HRank ran out of memory.

When varying Y_s, the running time of all algorithms increases with the increment of Y_s, and our incremental algorithms consistently run faster than all competitors, especially with less update data. For TWPageRank computation, algorithm incTWPR is on average (1.9, 3.8) and (2.5, 4.1) times faster than (batTWPR, powTWPR) on DBLP and MAG, respectively. For scholarly article ranking, algorithm incSARank is on average (1.7, 3.1, 2.8, 117) and (2.0, 3.0, 4.4, 245) times faster than (batSARank, powSARank, FRank, HRank) on DBLP and MAG, respectively.

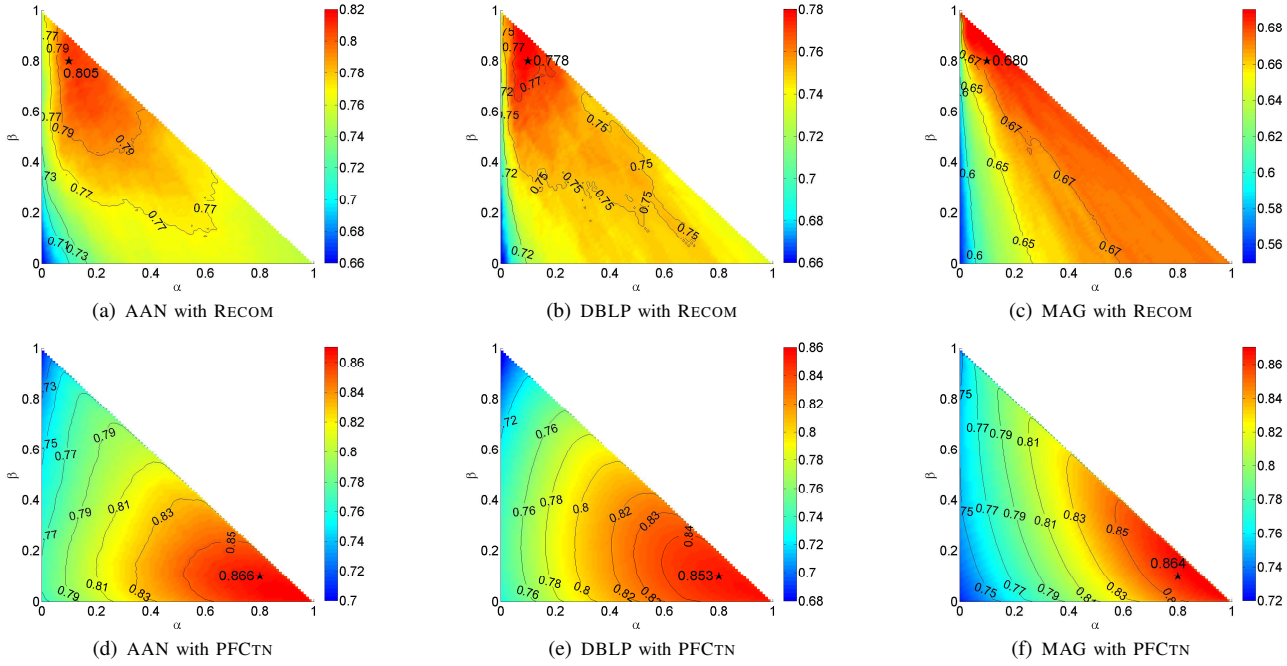
Table IV
MEMORY COSTS ON DBLP AND MAG

Datasets	Data	PRank	FRank	HRank	SARank
DBLP	289MB	264MB	404MB	1.34GB	1.28GB
MAG	10.5GB	8.7GB	14.3GB	61.4GB	48.1GB

In our tests we adopted a yearly update policy due the limitation of available time information. In practice our algorithms may bring more efficiency benefits since the update is usually more frequent, such that the data updates are smaller and the unaffected area is very likely much larger.

Exp-4: Memory cost. In the fourth set of tests, we evaluated the memory cost of our algorithm on the large DBLP and MAG. For a fair comparison, we used all data on DBLP and data before 2014 on MAG such that HRank could finish the test. The results are reported in Table IV, where column 'Data' records the size of the academic graph data, *i.e.,* the citation graph, author-article relationships, article-venue relationships and time information of articles.

Algorithm PRank uses the least memory, followed by algorithms FRank, SARank and HRank, respectively. The



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

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

Figure 7. Accuracy evaluation: varying aggregating parameters α and β

Table V
ACCURACY EVALUATION USING DIFFERENT COMPONENTS WITH RECOM (ROWS 2–4) AND PFCTN (ROWS 5–7).

Datasets	C	V	A	CV	CA	VA	CVA
AAN	0.752	0.616	0.649	0.809	0.764	0.747	**0.810**
DBLP	0.735	0.581	0.640	0.784	0.749	0.729	**0.785**
MAG	0.635	0.534	0.553	0.697	0.673	0.648	**0.698**
AAN	0.785	0.557	0.761	0.849	0.866	0.771	**0.870**
DBLP	0.713	0.603	0.725	0.843	0.847	0.740	**0.856**
MAG	0.736	0.628	0.718	0.848	0.857	0.751	**0.874**

memory used by PRank is even less than the data size as it uses the citation graph only. Our SARank model costs more memory than FRank, but less than HRank. This is because SARank further combines article-venue relationships ignored by FRank, and assembles multiple rankings based on prestige and popularity, a price paid to achieve better effectiveness.

Exp-5: Impacts of parameters. In the last set of tests, we evaluated the impacts of time decaying factor σ, importance weighting factor λ, aggregating parameters α and β, and the TWPageRank. We fixed these parameters as well as Y_s to their default values, used the TWPageRank proposed in this work by default, and tested the PairAcc with the entire RECOM and PFCTN ($T_p = +\infty$, $dif = 1$).

Exp-5.1. To evaluate the impacts of the time decaying factor σ, we varied σ from -1.6 to -0.4. The results of PairAcc are reported in Figs. 5(d), 5(i) and 5(n).

When varying σ, the PairAcc of SARank is very stable on all datasets using both RECOM and PFCTN. Indeed, with RECOM and PFCTN, the PairAcc only varies (0.42%, 1.55%, 0.81%) and (1.26%, 0.96%, 1.16%) on (AAN, DBLP, MAG), respectively. The running time varies (11.3%, 8.6%) on average only on (DBLP, MAG), respectively.

Exp-5.2. To evaluate the impacts of importance weighting factor λ, we varied λ from 0 to 1. The results of PairAcc are reported in Figs. 5(e), 5(j) and 5(o). Note that parameter λ has no impacts on efficiency.

When varying λ, the PairAcc of SARank first increases and then decreases on all datasets with both PFCTN and RECOM, except on DBLP with RECOM. This result indicates that combining prestige and popularity generally produces more robust results than using either of prestige and popularity. Indeed, with RECOM and PFCTN, the PairAcc of combining prestige and popularity is (10.2%, 10.7%, 5.5%) and (8.0%, 8.7%, 9.0%) higher than using prestige alone, and is (1.2%, -0.1%, 1.0%) and (1.0%, 1.0%, 0.3%) higher than using popularity alone on (AAN, DBLP, MAG), respectively.

Exp-5.3. To evaluate the impacts of aggregating parameters α and β, we varied α and β at the granularity of 0.01. Again, parameters α and β have few impacts on efficiency. The results are reported in Fig. 7, where the parameters α and β are set by default (used earlier) to the corresponding values of α-axis and β-axis of the PairAcc marked with $\star$, respectively.

When varying α and β, the PairAcc of SARank changes gently, as shown in Fig. 7. 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, as shown in Fig. 7. Further, the optimal parameters on the same set of ground-truth 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, SARank is very robust to parameters α and β, and it is quite flexible for choosing proper values of parameters α and β.

Moreover, this enables to verify the effectiveness of importance assembling from different components, whose results are reported in Table V, in which letters C, V and A stand

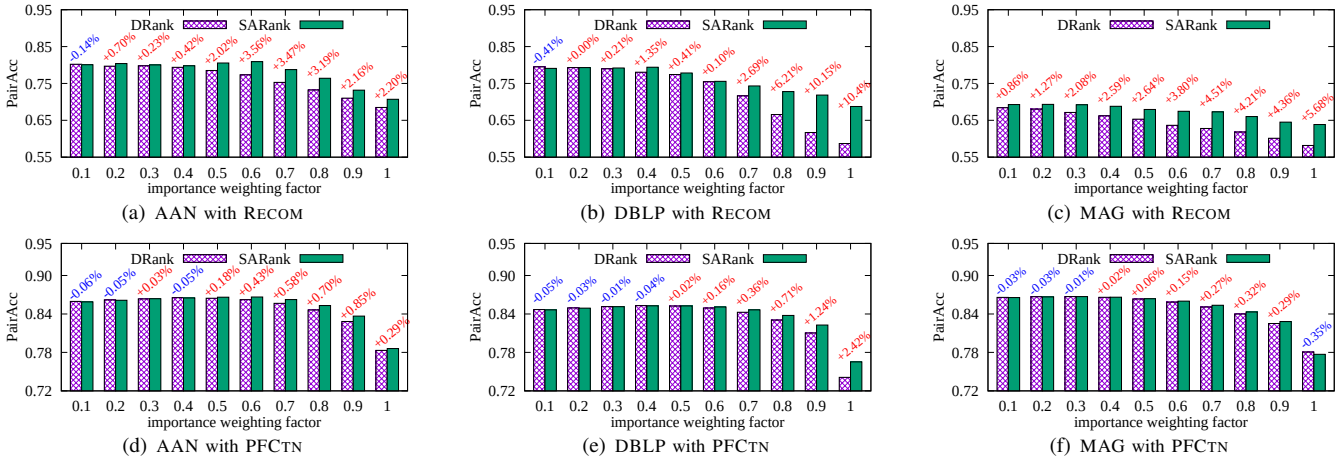


Figure 8. Impacts of TWPageRank on accuracy: varying importance weighting factor λ

for citation, venue and author components, respectively. The ranking based on all components consistently performs the best, using both RECOM and PFCTN, which justifies the use of importance assembling for ranking scholarly articles.

Exp-5.4. To evaluate the impacts of our TWPageRank, we compared our SARank with DRank, an alternative of SARank by removing the peak time in Eq. (1) of TWPageRank, *i.e.,* $w(u,v) = e^{\sigma(T_u - T_v)}$. To better understand the impacts, we varied the importance weighting factor λ from 0.1 to 1. Note that the ranking results are the same when $\lambda = 0$ due to the same popularity computation. The results are reported in Fig. 8, where the numbers represent the improvement of PairAcc by SARank over DRank.

When varying λ, the PairAcc of SARank is better than the one of DRank in most cases, which shows the superiority of the TWPageRank than directly decaying without introducing the peak time. The PairAcc difference of these two algorithms is higher with RECOM than with PFCTN, since the two algorithms are just using citation information to predict past and future citations with PFCTN. Moreover, algorithm SARank is consistently better than DRank when $0.5 \leq \lambda \leq 0.9$. The improvement decreases with the decrease of λ as the popularity dominates the ranking with small λ, and in some cases, DRank outperforms SARank. Overall, with RECOM and PFCTN, SARank improves the PairAcc over DRank by (1.78%, 3.07%, 3.20%) and (0.29%, 0.48%, 0.11%) on (AAN, DBLP, MAG) on average, respectively.

The TWPageRank has a minor impact on efficiency, and the running time of the two algorithms only varies (6.34%, 4.83%) on (DBLP, MAG) on average, respectively.

Summary. From these tests, we find the followings.

(1) Our model SARank is effective for ranking scholarly articles, which is consistently better than competitive methods in all tests. With RECOM and PFCTN, SARank improves PairAcc over (PRank, FRank, HRank) by (13.5%, 6.8%, 4.8%) and (12.0%, 3.0%, 3.2%) on AAN, (12.7%, 5.0%, 4.9%) and (14.0%, 6.5%, 4.6%) on DBLP, and (6.5%, 2.5%, 2.2%) and (13.4%, 6.0%, 2.4%) on MAG, on average, respectively.

(2) Our batch algorithm batSARank and incremental algorithm incSARank are also efficient. Our incremental algorithm incSARank is on average (1.7, 3.1, 2.8, 117) and (2.0, 3.0, 4.4, 245) times faster than (batSARank, powSARank, FRank, HRank) on the large DBLP and MAG, respectively.

(3) Our ranking model SARank introduces the time decaying factor σ, importance weighting factor λ and aggregating parameters α and β for the sake of practicability and flexibility in real-life applications, and, from our tests, SARank is very robust to these parameters. Moreover, the proposed TWPageRank is generally more effective than directly using exponentially decayed impact weights.

VI. RELATED WORK

Scholarly article ranking has shifted from citation-count analysis [1], [2] to graph analysis [3]–[11]. Based on the information used, these methods are divided into four categories: (a) using the citation information only [1], [2], [7], (b) using the citation and temporal information [8], [11], (c) using the citation information and other heterogeneous information, *e.g.,* authors and venues of articles [3], [4], and (d) combining the citation, temporal and other heterogeneous information [6], [9], [10]. Our work belongs to the last category aiming at fully employing information available for scholarly article ranking.

PageRank [23] and its extensions have been extensively used for citation analyses [5]. While PageRank equally propagates scores along outlinks, Weighted PageRank extends PageRank by distributing scores based on certain criteria such as popularity of pages [31] or authority of authors [32]. Scholarly graphs belong to temporal graphs [33], and temporal information is a key factor for scholarly article ranking. There has been work extending temporal information into PageRank, *e.g.,* exponentially decayed weights [8], exponentially decayed initial vectors [11] and time-dependent weights based on co-authorship [34]. Differently, our Time-Weighted PageRank is designed based on a deep analysis of scholarly articles, and discriminately propagates scores in terms of citation statistics.

Dynamic algorithms have proven useful for various tasks by avoiding computing from scratch [35]. To our knowledge,

little concern has been paid to dynamic scholarly article ranking except that [21] uses PageRank in dynamic citation networks. However, its solution is based on a strong and impractical assumption that there are no citations between articles in the same years. Further, although there exist several studies on incremental PageRank computation [36]–[38] and on incremental PageRank approximation [39], [40], they are not designed for scholarly article ranking. In this work, we study dynamic scholarly article ranking in the general setting by eliminating the strong and impractical assumption. Our incremental algorithm is designed for the block-wise algorithm of Time-Weighted PageRank, and is based on the citation characteristics, both of which have never been exploited before.

Ensemble methods use multiple learners to obtain better performance than could be obtained from a constituent learner alone [41]. In this work, we leverage importance assembling to produce better and more robust ranking for scholarly articles [20], [41], [42].

VII. Conclusions

We have proposed a new model SARank for scholarly article ranking, which assembles the importance of article, venue and author entities. We have also proposed efficient batch and incremental algorithms for the computation of their importance, a combination of prestige and popularity. As shown by the experimental study, our approach is both effective and efficient for scholarly article ranking.

A couple of topics need further investigation. First, we are to clean scholarly data with external data sources and to extend our model with affiliation and discipline information for further improving the quality of ranking. Second, we are to study distributed algorithms, similar to [43] that computes PageRank in a distributed environment.

Acknowledgments

This work is supported in part by NSFC U1636210, 973 Program 2014CB340300, NSFC 61421003, and MSRA Collaborative Research Program. For any correspondence, please refer to Renjun Hu and Chunming Hu.

References

[1] E. Garfield, "Citation analysis as a tool in journal evaluation," *Science*, vol. 178, no. 4060, pp. 471–479, 1972.
[2] J. E. Hirsch, "An index to quantify an individual's scientific research output," *PNAS*, vol. 102, no. 46, pp. 16 569–16 572, 2005.
[3] R. Liang and X. Jiang, "Scientific ranking over heterogeneous academic hypernetwork," in *AAAI*, 2016.
[4] X. Jiang, X. Sun, and H. Zhuge, "Towards an effective and unbiased ranking of scientific literature through mutual reinforcement," in *CIKM*, 2012.
[5] L. Waltman and E. Yan, *PageRank-Related Methods for Analyzing Citation Networks*. Springer, 2014, pp. 83–100.
[6] 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*, vol. 7, no. 4, pp. 64:1–64:28, 2016.
[7] M. K.-P. Ng, X. Li, and Y. Ye, "Multirank: Co-ranking for objects and relations in multi-relational data," in *KDD*, 2011.
[8] X. Li, B. Liu, and P. Yu, "Time sensitive ranking with application to publication search," in *ICDM*, 2008.
[9] Y. Wang, Y. Tong, and M. Zeng, "Ranking scientific articles by exploiting citations, authors, journals and time information," in *AAAI*, 2013.
[10] H. Sayyadi and L. Getoor, "Future rank: Ranking scientific articles by predicting their future pagerank," in *SDM*, 2009.
[11] 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*, vol. 2007, no. 06, p. P06010, 2007.
[12] Google Scholar, *https://scholar.google.com/*.
[13] 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.
[14] Semantic Scholar, *https://www.semanticscholar.org/*.
[15] C. C. Aggarwal and K. Subbian, "Evolutionary network analysis: A survey," *ACM Comput. Surv.*, vol. 47, no. 1, pp. 10:1–10:36, 2014.
[16] S. Ma, J. Li, C. Hu, X. Lin, and J. Huai, "Big graph search: challenges and techniques," *FCS*, vol. 10, no. 3, pp. 387–398, 2016.
[17] D. Wang, C. Song, and A. Barabási, "Quantifying long-term scientific impact," *Science*, vol. 342, no. 6154, pp. 127–132, 2013.
[18] T. Chakraborty, S. Kumar, P. Goyal, N. Ganguly, and A. Mukherjee, "On the categorization of scientific citation profiles in computer science," *Commun. ACM*, vol. 58, no. 9, 2015.
[19] L. Bornmann and R. Mutz, "Growth rates of modern science: A bibliometric analysis based on the number of publications and cited references," *JASIST*, vol. 66, no. 11, pp. 2215–2222, 2015.
[20] A. D. Wade, K. Wang, Y. Sun, and A. Gulli, "Wsdm cup 2016: Entity ranking challenge," in *WSDM*, 2016.
[21] R. Ghosh, T. Kuo, C. Hsu, S. Lin, and K. Lerman, "Time-aware ranking in dynamic citation networks," in *ICDM Workshops*, 2011.
[22] M. Richardson, A. Prakash, and E. Brill, "Beyond pagerank: Machine learning for static ranking," in *WWW*, 2006.
[23] S. Brin and L. Page, "The anatomy of a large-scale hypertextual web search engine," *Computer Networks*, vol. 30, no. 1-7, pp. 107–117, 1998.
[24] Y. Dong, H. Ma, Z. Shen, and K. Wang, "A century of science: Globalization of scientific collaborations, citations, and innovations," in *KDD*, 2017.
[25] G. M. D. Corso, A. Gulli, and F. Romani, "Fast pagerank computation via a sparse linear system," *Internet Mathematics*, vol. 2, no. 3, pp. 251–273, 2005.
[26] P. Berkhin, "Survey: A survey on pagerank computing," *Internet Mathematics*, vol. 2, no. 1, pp. 73–120, 2005.
[27] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*. The MIT Press, 2001.
[28] J. Leskovec, K. J. Lang, A. Dasgupta, and M. W. Mahoney, "Community structure in large networks: Natural cluster sizes and the absence of large well-defined clusters," *Internet Mathematics*, vol. 6, no. 1, pp. 29–123, 2009.
[29] A. Z. Broder, R. Kumar, F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A. Tomkins, and J. L. Wiener, "Graph structure in the web," *Computer Networks*, vol. 33, no. 1-6, pp. 309–320, 2000.
[30] J. Tang, J. Zhang, L. Yao, J. Li, L. Zhang, and Z. Su, "Arnetminer: Extraction and mining of academic social networks," in *KDD*, 2008.
[31] W. Xing and A. Ghorbani, "Weighted pagerank algorithm," in *CNSR*, 2004.
[32] Y. Ding, "Applying weighted pagerank to author citation networks," *JASIST*, vol. 62, no. 2, pp. 236–245, 2011.
[33] P. Holme and J. Saramaki, "Temporal networks," *Physics Reports*, vol. 519, no. 3, pp. 97–125, 2012.
[34] D. Fiala, "Time-aware pagerank for bibliographic networks," *Journal of Informetrics*, vol. 6, no. 3, pp. 370–388, 2012.
[35] G. Ramalingam and T. W. Reps, "A categorized bibliography on incremental computation," in *POPL*, 1993.
[36] P. K. Desikan, N. Pathak, J. Srivastava, and V. Kumar, "Incremental page rank computation on evolving graphs," in *WWW*, 2005.
[37] S. Abiteboul, M. Preda, and G. Cobena, "Adaptive on-line page importance computation," in *WWW*, 2003.
[38] Y. Wu and L. Raschid, "Approxrank: Estimating rank for a subgraph," in *ICDE*, 2009.
[39] B. Bahmani, A. Chowdhury, and A. Goel, "Fast incremental and personalized pagerank," *PVLDB*, vol. 4, no. 3, pp. 173–184, 2010.
[40] B. Bahmani, R. Kumar, M. Mahdian, and E. Upfal, "Pagerank on an evolving graph," in *KDD*, 2012.
[41] Z.-H. Zhou, *Ensemble Methods: Foundations and Algorithms*. Chapman & Hall/CRC, 2012.
[42] L. Duan, S. Ma, C. Aggarwal, T. Ma, and J. Huai, "An ensemble approach to link prediction," *TKDE*, vol. 29, no. 11, pp. 2402–2416, 2017.
[43] Y. Zhu, S. Ye, and X. Li, "Distributed pagerank computation based on iterative aggregation-disaggregation methods," in *CIKM*, 2005.