Diversified Ranking on Large Graphs: An Optimization Viewpoint

Hanghang Tong Jingrui He Zhen Wen Ravi Konuru Ching-Yung Lin
IBM T.J. Watson Research Center
Hawthorne, NY, USA
{htong, jingruhe, zhenwen, rkonuru, chingyung}@us.ibm.com

ABSTRACT

Diversified ranking on graphs is a fundamental mining task and has a variety of high-impact applications. There are two important open questions here. The first challenge is the *measure* - how to quantify the goodness of a given top-k ranking list that captures both the relevance and the diversity? The second challenge lies in the *algorithmic* aspect - how to find an optimal, or near-optimal, top-k ranking list that maximizes the measure we defined in a scalable way?

In this paper, we address these challenges from an optimization point of view. Firstly, we propose a goodness measure for a given top-k ranking list. The proposed goodness measure intuitively captures both (a) the relevance between each individual node in the ranking list and the query; and (b) the diversity among different nodes in the ranking list. Moreover, we propose a *scalable* algorithm (*linear* wrt the size of the graph) that generates a *provably near-optimal* solution. The experimental evaluations on real graphs demonstrate its effectiveness and efficiency.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database Applications – Data Mining

General Terms

Algorithm, experimentation

Keywords

Diversity, ranking, scalability, graph mining

1. INTRODUCTION

Given an author-paper network, how to find the top-k most related conferences for a given author? How to diversify the ranking list so that it captures the whole spectrum of the given author's research interest? It is now widely realized that diversity is a key factor to address the uncertainty and ambiguity in an information need; and to cover the different aspects of the information need [32]. Diversity is also positively associated with personnel performance and job retention rate in a large organization [38].

Despite their own success of the previous works (See Section 6 for a review), two important questions remain open in diversified ranking on large graphs. The first challenge is the *measure* - for a given top-k ranking list, how can we quantify its goodness? Intuitively, a good top-k ranking list should capture both the relevance and the diversity. For example, given a task which typically requires a set of different skills, if we want to form a team of experts, not only should the people in the team have relevant skills, but also they should somehow be 'different' from each other so that the whole team can benefit from the diversified, complementary knowledge and social capital. However, there does not exist such a goodness measure for the graph data in the literature. Most of the existing works for diversified ranking on graphs are based on some heuristics. The only exception is [27], where the authors made an important step towards this goal by providing some optimization explanations, which is achieved by defining a *time-varying* objective function at each iteration. But still, it is not clear what *overall* objective function the algorithm tries to optimize.

The second challenge lies in the *algorithmic* aspect - how can we find an optimal, or near-optimal, top-k ranking list that maximizes the goodness measure? Bringing diversity into the design objective implies that we need to optimize on the set level. In other words, the objective function for a subset of nodes is usually not equal to the sum of objective functions of each individual nodes. It is usually very hard to perform such set-level optimization. For instance, a straight-forward method would need exponential enumerations to find the exact optimal solution, which is infeasible even for medium size graphs. This, together with the fact that real graphs are often of large size, reaching billions of nodes and edges, poses the challenge for the optimization algorithm - how can we find a near-optimal solution in a scalable way?

In this paper, we address these challenges from an optimization point of view. We propose a goodness measure which intuitively captures both (a) the relevance between each individual nodes in the ranking list and the query node; and (b) the diversity among different nodes in the ranking list. We further propose a *scalable* algorithm (*linear* wrt the size of the graph) that generates a *provably near-optimal* top-k ranking list. To the best of our knowledge, this is the first work for diversified ranking on large graphs that (1) has a clear optimization formulation; (2) finds a provably near-optimal solution; and (3) enjoys the linearly scalability. The main contributions of the paper are summarized as follows:

- *A measure* to quantify goodness for a top-k ranking list that captures both relevance and diversity;
- *An algorithm* to find a diversified top-k ranking list from large graphs;

KDD'11, August 21–24, 2011, San Diego, CA, USA.

Table 1: Symbols

Symbol	Definition and Description
$\mathbf{A}, \mathbf{B}, \ldots$	matrices (bold upper case)
$\mathbf{A}(i,j)$	the element at the i^{th} row and j^{th} column of $\mathbf{A}$
$\mathbf{A}(i,:)$	the i^{th} row of matrix $\mathbf{A}$
$\mathbf{A}(:,j)$	the j^{th} column of matrix $\mathbf{A}$
$\mathbf{A}'$	transpose of matrix $\mathbf{A}$
$\mathbf{a}, \mathbf{b}, \ldots$	vectors
$\mathcal{I}, \mathcal{J}, \ldots$	sets (calligraphic)
$\otimes$	element-wise Hadamard product
$\mathbf{r}$	an $n \times 1$ ranking vector
$\mathbf{p}$	an $n \times 1$ query vector ($\mathbf{p}(i) \geq 0, \sum_{i=1}^{n} \mathbf{p}(i) = 1$)
$\mathbf{I}$	an identity matrix
$\mathbf{1}$	a vector/matrix with all elements set to 1s
$\mathbf{0}$	a vector/matrix with all elements set to 0s
n, m	the number of the nodes and edges in the graph
k	the budget (i.e., the length of the ranking list)
c	the damping factor $0 < c < 1$

- *Proofs and complexity analysis*, showing that our method is *provably near-optimal* in terms of optimization quality with *linear* scalability;
- *Extensive experimental evaluations*, demonstrating the effectiveness and efficiency of our method.

The rest of the paper is organized as follows. We introduce notation and formally define the problems in Section 2. We present and analyze the proposed measure and algorithm in Section 3 and Section 4, respectively. We provide experimental evaluation in Section 5. We review the related work in Section 6 and conclude in Section 7.

2. PROBLEM DEFINITIONS

Table 1 lists the main symbols we use throughout the paper. In this paper, we consider the most general case of directed, weighted, irreducible unipartite graphs. We represent a general graph by its adjacency matrix[1]. Following the standard notation, we use bold upper-case for matrices (e.g., $\mathbf{A}$), bold lower-case for vectors (e.g., $\mathbf{a}$), and calligraphic fonts for sets (e.g., $\mathcal{I}$). We denote the transpose with a prime (i.e., $\mathbf{A}'$ is the transpose of $\mathbf{A}$). For a bipartite graph with adjacency matrix $\mathbf{W}$, we can convert it to the equivalent unipartite graph: $\mathbf{A} = \begin{pmatrix} \mathbf{0} & \mathbf{W} \\ \mathbf{W} & \mathbf{0} \end{pmatrix}$. We use subscripts to denote the size of matrices/vectors (e.g., $\mathbf{A}_{n \times n}$ means a matrix of size $n \times n$). When the size of matrices/vectors are clear from the context, we omit such subscripts for brevity. Also, we represent the elements in a matrix using a convention similar to Matlab, e.g., $\mathbf{A}(i,j)$ is the element at the i^{th} row and j^{th} column of the matrix $\mathbf{A}$, and $\mathbf{A}(:,j)$ is the j^{th} column of $\mathbf{A}$, etc. With this notation, we can represent a sub-matrix of $\mathbf{A}$ as $\mathbf{A}(\mathcal{I}, \mathcal{I})$, which is a block of matrix $\mathbf{A}$ that corresponds to the rows/columns of $\mathbf{A}$ indexed by the set $\mathcal{I}$.

In this paper, we focus on personalized PageRank [30, 11] since it is one of the most fundamental ranking methods on graphs, and has shown its success in many different application domains in the past decade. Formally, it can be defined as follows:

$$\mathbf{r} = c\mathbf{A}'\mathbf{r} + (1-c)\mathbf{p} \tag{1}$$

where $\mathbf{p}$ is an $n \times 1$ personalized vector ($\mathbf{p}(i) \geq 0, \sum_{i=1}^{n} \mathbf{p}(i) = 1$). Sometimes, we also refer to $\mathbf{p}$ as the query vector. c ($0 < c < 1$) is a damping factor; $\mathbf{A}$ is the row-normalized adjacency matrix of the graph (i.e., $\sum_{j=1}^{n} \mathbf{A}(i,j) = 1$ $(i = 1, ..., n)$; and $\mathbf{r}$ is the $n \times 1$ resulting ranking vector. Note that if $\mathbf{p}(i) = 1/n (i = 1, ..., n)$, it is reduced to the standard PageRank [30]; if $\mathbf{p}(i) = 1$ and $\mathbf{p}(j) = 0 (j \neq i)$, the resulting ranking vector $\mathbf{r}$ gives the proximity scores from node i to all the other nodes in the graph [37].

In order to simplify the description of our upcoming method, we also introduce the so-called 'Google matrix' $\mathbf{B}$:

$$\mathbf{B} = c\mathbf{A}' + (1-c)\mathbf{p}\mathbf{1}_{1 \times n} \tag{2}$$

where $\mathbf{1}_{1 \times n}$ is a $1 \times n$ row vector with all elements set to 1s. Intuitively, the 'Google matrix' $\mathbf{B}$ can be viewed as the personalized adjacency matrix that is biased towards the query vector $\mathbf{p}$. It turns out that the ranking vector $\mathbf{r}$ defined in eq. (1) satisfies **r = Br**. In other words, the ranking vector $\mathbf{r}$ is the right eigenvector of the $\mathbf{B}$ matrix with the eigenvalue 1. It can be verified that $\mathbf{B}$ is a column-wise stochastic matrix (i.e., each column of $\mathbf{B}$ sums up to 1). By Perron-Frobenius theorem [10], it can be shown that 1 is the largest (in module) simple eigenvalue of the matrix $\mathbf{B}$; and the ranking vector $\mathbf{r}$ is unique with all non-negative elements since the graph is irreducible.

Our goal is two-fold: (1) we want a goodness measure to quantify the quality of a given top-*k* ranking list that captures both the relevance and the diversity; and (2) given the goodness measure, we want an optimal or near-optimal algorithm to find a top-*k* ranking list that maximizes such goodness measure in a scalable way. With the above notations and assumptions, our problems can be formally defined as follows:

PROBLEM 1. **(Goodness Measure.)**

Given: *A large graph $\mathbf{A}_{n \times n}$, the query vector $\mathbf{p}$, the damping factor c, and a subset of k nodes $\mathcal{S}$;*

Output: *A goodness score $f(\mathcal{S})$ of the subset of nodes $\mathcal{S}$, which measures (a) the relevance of each node in $\mathcal{S}$ wrt the query vector $\mathbf{p}$, and (b) the diversity among all the nodes in the subset $\mathcal{S}$.*

PROBLEM 2. **(Diversified Top-*k* Ranking Algorithm.)**

Given: *A large graph $\mathbf{A}_{n \times n}$, the query vector $\mathbf{p}$, the damping factor c, and the budget k;*

Find: *A subset of k nodes $\mathcal{S}$ that maximizes the goodness measure $f(\mathcal{S})$.*

In the next two sections, we present our solutions for these two problems respectively.

3. THE PROPOSED GOODNESS MEASURE

In this section, we address Problem 1. Our goal is to define a goodness measure to quantify the quality of a given top-*k* ranking list that captures both the relevance and the diversity. We first discuss some design objectives of such a goodness measure; and then present our solution followed by some theoretical analysis.

3.1 Design Objectives

As said before, a good diversified top-*k* ranking list should balance between the relevance and the diversity. The notion of relevance is clear for personalized PageRank, - larger value in the ranking vector $\mathbf{r}$ means more relevant wrt the query vector $\mathbf{p}$. On the other hand, the notion of diversity is more challenging. Intuitively, a diversified subset of nodes should be *dis-similar* with each other. Take the query *'Find the top-k conferences for Philip Yu from the author-conference network'* as an example. Dr. *Philip*

[1] In practice, we store these matrices using an adjacency list representation, since real graphs are often very sparse.

Yu is a professor at University of Illinois at Chicago. His recent major research interest lies in databases and data mining. He also has broad interests in several related domains, including systems, parallel and distributed processing, web applications, and performance modeling, etc. A top-*k* ranking list for this query would have high relevance if it consists of all the conferences from databases and data mining community (e.g., *SIGMOD, VLDB, KDD*, etc) since all these conferences are closely related to his major research interest. However, such a list has low diversity since these conferences are too similar with each other (e.g., having a large overlap of contributing authors, etc). Therefore, if we replace a few databases and data mining conferences by some representative conferences in his other research domains (e.g., *ICDCS* for distributed computing systems, *WWW* for web applications, etc), it would make the whole ranking list more diverse (e.g., the conferences in the list are more dis-similar with each other).

Furthermore, if we go through the ranking list from top down, we would like to see the most relevant conferences to appear first in the ranking list. For example, a ranking list in the order of *'SIGMOD','ICDCS','WWW'* is better than *'ICDCS','WWW','SIGMOD'* since databases (*SIGMOD*) is a more relevant research interest for Dr. *Philip Yu*, compared with distributed computing systems (*ICDCS*), or web applications (*WWW*). In this way, the user can capture Dr. *Philip Yu*'s main research interest by just inspecting a few top-ranked conferences/nodes. This suggests the so-called *diminishing returns property* of the goodness measure - it would help the user to know better about Dr. *Philip Yu*'s whole research interest if we return more conferences/nodes in the ranking list; but the *marginal benefit* becomes smaller and smaller as we go down the ranking list.

Another implicit design objective lies in the algorithmic aspect. The proposed goodness measure should also allow us to develop an effective and scalable algorithm to find an optimal (or at least near-optimal) top-*k* ranking list from large graphs. We will discuss and address this issue in the next section.

To summarize, for a given top-*k* ranking list, we aim to provide a single goodness score that (1) measures the *relevance* between each individual node in the list and the query vector $\mathbf{p}$; (2) measures the *similarity (or dis-similarity)* among all the nodes in the ranking list; (3) exhibits some diminishing returns property wrt the size of the ranking list; and (4) enables some effective and scalable algorithm to find an optimal (or near-optimal) top-*k* ranking list.

3.2 The Proposed Measure

Let $\mathbf{A}$ be the row-normalized adjacency matrix of the graph, $\mathbf{B}$ be the 'Google matrix' defined in eq (2), $\mathbf{p}$ be the personalized vector and $\mathbf{r}$ be the ranking vector. For a given ranking list $\mathcal{S}$ (i.e., $\mathcal{S}$ gives the indices of the nodes in the ranking list; and $|\mathcal{S}| = k$.), the proposed goodness measure is formally defined as follows:

Goodness Measure:

$$\mathrm{f}(\mathcal{S}) = 2\sum_{i\in\mathcal{S}} \mathbf{r}(i) - \sum_{i,j\in\mathcal{S}} \mathbf{B}(i,j)\mathbf{r}(j) \tag{3}$$

We can also represent $\mathrm{f}(\mathcal{S})$ by using the matrix $\mathbf{A}$ instead:

$$\mathrm{f}(\mathcal{S}) = 2\sum_{i\in\mathcal{S}} \mathbf{r}(i) - c\sum_{i,j\in\mathcal{S}} \mathbf{A}(j,i)\mathbf{r}(j) - (1-c)\sum_{j\in\mathcal{S}} \mathbf{r}(j)\sum_{i\in\mathcal{S}} \mathbf{p}(i)$$

where c is the damping factor in personalized PageRank, and $\mathbf{1}_{1\times|\mathcal{S}|}$ is a row vector of length $|\mathcal{S}|$ with all the elements set to 1s. It can be shown that it is equivalent to eq. (3).

Notice that the goodness measure in eq (3) is independent on the ordering of the different nodes in the subset $\mathcal{S}$. If we simply change the ordering of the nodes for the same subset $\mathcal{S}$, it does not affect the goodness score. However, as we will show in Section 4, we can still output an *ordered* subset based on the diminishing returns need when the user is seeking for a diverse top-*k* ranking list.

3.3 Proofs and Analysis

Let us analyze how the proposed goodness measure meets our design objectives in subsection 3.1.

There are two terms in eq (3), the first term is twice the sum of the ranking scores in the ranking list. For the second term, recall that $\mathbf{B}$ can be viewed as the personalized adjacency matrix wrt the query vector $\mathbf{p}$, where $\mathbf{B}(i,j)$ indicates the similarity (i.e., the strength of the connection) between nodes i and j. In other words, the second term in eq (3) is the sum of all the similarity scores between any two nodes $i, j (i, j \in \mathcal{S})$ in the ranking list (weighted by $\mathbf{r}(j)$). Therefore, the proposed goodness measure captures both the relevance and the diversity. The more relevant (higher $\mathbf{r}(i)$) each individual node is, the higher the goodness measure $\mathrm{f}(\mathcal{S})$. At the same time, it encourages the diversity within the ranking list by penalizing the (weighted) similarity between any two nodes in $\mathcal{S}$.

The proposed measure $\mathrm{f}(\mathcal{S})$ also exhibits the diminishing returns property, which is summarized in Theorem 1. The intuitions of Theorem 1 are as follows: (1) by P1, it means that the utility of an empty ranking list is always zero; (2) by P2, if we add more nodes into the ranking list, the overall utility of the ranking list does not decrease; and (3) by P3, the marginal utility of adding new nodes is relatively small if we already have a large ranking list.

THEOREM 1. **Diminishing Returns Property of $\mathrm{f}(\mathcal{S})$.** *Let Φ be an empty set; $\mathcal{I}, \mathcal{J}, \mathcal{R}$ be three sets s.t., $\mathcal{I} \subseteq \mathcal{J}$, and $\mathcal{R} \cap \mathcal{J} = \Phi$. The following facts hold for $f(\mathcal{S})$:*

P1: $f(\Phi) = 0$;

P2: $f(\mathcal{S})$ is monotonically non-decreasing, i.e., $f(\mathcal{I}) \leq f(\mathcal{J})$;

P3: $f(\mathcal{S})$ is submodular, i.e., $f(\mathcal{I}\cup\mathcal{R}) - f(\mathcal{I}) \geq f(\mathcal{J}\cup\mathcal{R}) - f(\mathcal{J})$.

PROOF of *P1*. It is obviously held by the definition of $\mathrm{f}(\mathcal{S})$. □

PROOF of *P2*. Let $\mathcal{T} = \mathcal{J} \setminus \mathcal{I}$. Substituting eq (3) into $\mathrm{f}(\mathcal{J}) - \mathrm{f}(\mathcal{I})$ and canceling the common terms, we have

$$\begin{aligned}
& \mathrm{f}(\mathcal{J}) - \mathrm{f}(\mathcal{I}) \\
= \; & 2\sum_{i\in\mathcal{T}} \mathbf{r}(i) - \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{T}} \mathbf{B}(i,j)\mathbf{r}(j) - \sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{J}} \mathbf{B}(i,j)\mathbf{r}(j) \\
= \; & (\sum_{j\in\mathcal{T}} \mathbf{r}(j) - \sum_{j\in\mathcal{T}}\sum_{i\in\mathcal{I}} \mathbf{B}(i,j)\mathbf{r}(j)) \\
& + (\sum_{i\in\mathcal{T}} \mathbf{r}(i) - \sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{J}} \mathbf{B}(i,j)\mathbf{r}(j))
\end{aligned} \tag{4}$$

Recall that the matrix $\mathbf{B}$ is a column-wise stochastic matrix (i.e., each column of $\mathbf{B}$ sums up to 1). The first half of eq (4) satisfies

$$\begin{aligned}
& (\sum_{j\in\mathcal{T}} \mathbf{r}(j) - \sum_{j\in\mathcal{T}}\sum_{i\in\mathcal{I}} \mathbf{B}(i,j)\mathbf{r}(j)) \\
= \; & \sum_{j\in\mathcal{T}} \mathbf{r}(j)(1 - \sum_{i\in\mathcal{I}} \mathbf{B}(i,j)) \\
= \; & \sum_{j\in\mathcal{T}} \mathbf{r}(j) \sum_{i\notin\mathcal{I}} \mathbf{B}(i,j) \geq 0
\end{aligned} \tag{5}$$

For the second half of eq (4), we have that

$$\begin{aligned}
& (\sum_{i\in\mathcal{T}} \mathbf{r}(i) - \sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{J}} \mathbf{B}(i,j)\mathbf{r}(j)) \\
= \; & \sum_{i\in\mathcal{T}} (\mathbf{r}(i) - \sum_{j\in\mathcal{J}} \mathbf{B}(i,j)\mathbf{r}(j)) \\
= \; & \sum_{i\in\mathcal{T}}\sum_{j\notin\mathcal{J}} \mathbf{B}(i,j)\mathbf{r}(j) \geq 0
\end{aligned} \tag{6}$$

The last equality in eq (6) is due to the fact that $\mathbf{r} = \mathbf{B}\mathbf{r}$, and each element in $\mathbf{r}$ is non-negative.

Putting eq (4)-(6) together, we have that $f(\mathcal{J}) \geq f(\mathcal{I})$, which completes the proof of P2. □

PROOF of *P3*. Again, let $\mathcal{T} = \mathcal{J} \setminus \mathcal{I}$. Substituting eq (4) into $(f(\mathcal{I} \cup \mathcal{R}) - f(\mathcal{I})) - (f(\mathcal{J} \cup \mathcal{R}) - f(\mathcal{J}))$ and canceling the common terms, we have

$$\begin{aligned} & (f(\mathcal{I} \cup \mathcal{R}) - f(\mathcal{I})) - (f(\mathcal{J} \cup \mathcal{R}) - f(\mathcal{J})) \\ = & (\sum_{i \in \mathcal{J}} \sum_{j \in \mathcal{R}} \mathbf{B}(i,j)\mathbf{r}(j) - \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{R}} \mathbf{B}(i,j)\mathbf{r}(j)) \\ + & (\sum_{i \in \mathcal{R}} \sum_{j \in \mathcal{J} \cup \mathcal{R}} \mathbf{B}(i,j)\mathbf{r}(j) - \sum_{i \in \mathcal{R}} \sum_{j \in \mathcal{I} \cup \mathcal{R}} \mathbf{B}(i,j)\mathbf{r}(j)) \\ = & \sum_{j \in \mathcal{R}} \sum_{i \in \mathcal{T}} \mathbf{B}(i,j)\mathbf{r}(j) + \sum_{i \in \mathcal{R}} \sum_{j \in \mathcal{T}} \mathbf{B}(i,j)\mathbf{r}(j) \geq 0 \end{aligned}$$

Therefore, we have that $f(\mathcal{I} \cup \mathcal{R}) - f(\mathcal{I}) \geq f(\mathcal{J} \cup \mathcal{R}) - f(\mathcal{J})$, which completes the proof of P3. □

4. THE PROPOSED ALGORITHM

In this section, we address Problem 2. Here, given the initial query vector $\mathbf{p}$ and the budget k, we want to find a subset of k nodes that maximizes the goodness measure defined in eq (3). We first analyze the main challenges in optimizing eq (3); and then present the proposed algorithm DRAGON, followed by some theoretical analysis and discussion.

4.1 Challenges

Problem 2 is essentially a subset selection problem to find the optimal k nodes that maximize eq (3). Theorem 1 indicates that it is not easy to find the exact optimal solution of Problem 2 - it is *NP-hard* to maximize a monotonic submodular function if the function value is 0 for an empty set [18]. For instance, a straight-forward method would take exponential enumerations $\binom{n}{k}$ to find the exact optimal k nodes, which is not feasible in computation even for a medium size graph (e.g., with a few hundred nodes).

We can also formulate Problem 2 as a binary quadratic programming problem. Let $\mathbf{x}_{n \times 1}$ be a binary indicator vector ($\mathbf{x}(i) = 1$ means node i is selected in the subset $\mathcal{S}$, and 0 means it is not selected). Problem 2 can be expressed as the following binary quadratic programming problem:

$$\begin{aligned} \text{min} \quad & \mathbf{x}'\mathbf{D}\mathbf{x} \\ \text{Subject to:} \quad & \mathbf{x}(i) \in \{0,1\} (i = 1, ..., n) \\ & \sum_{i=1}^{n} \mathbf{x}(i) = k \end{aligned} \tag{7}$$

where $\mathbf{D} = (\mathbf{B} - 2\mathbf{I}_{n \times n})\text{diag}(\mathbf{r})$, $\mathbf{I}_{n \times n}$ is an identity matrix of size $n \times n$, and $\text{diag}(\mathbf{r})$ is a diagonal matrix with $\mathbf{r}(i,i) (i = 1, ..., n)$ being the diagonal elements.

Eq. (7) is still not easy to solve due to (1) the binary constrains on the variable $\mathbf{x}$ and (2) the quadratic term in the objective function. If we relax the binary constrain on $\mathbf{x}$ as $0 \leq \mathbf{x}(i) \leq 1 (i = 1, ..., n)$, we can solve the relaxed problem by standard quadratic programming packages. We refer to this strategy as 'Lin-QP'. However, there are two major limitations of this method. First of all, we do not know what the gap is between eq. (7) and its relaxed version. Therefore, it is not clear how good the final solution is in terms of maximizing the original goodness measure (eq (3)) even if we can solve the relaxed problem optimally[2]. Second, most, if not all, of the existing quadratic programming packages require *polynomial* complexity in computation. This makes this strategy very slow, or even infeasible, for a graph with more than a few thousand nodes.

Another possible solution for eq. (7) is to remove the quadratic term in the objective function as follows. Starting from some initial indicator vector $\hat{\mathbf{x}}$, we iterate between the following two steps: (1) approximate the objective function in eq. (7) by its first order Taylor expansion around $\hat{\mathbf{x}}$; and (2) update $\hat{\mathbf{x}}$ by solving a binary integer programming problem for the approximated objective function, which is linear wrt $\mathbf{x}$. We refer to this strategy as 'Ite-BIP'. However, the two main issues still exist: (1) it is not clear how such approximation will downgrade the overall optimization performance; (2) the binary integer programming itself, again, requires *polynomial* time, which does not scale to large graphs.

4.2 The Proposed DRAGON Algorithm

Our proposed DRAGON algorithm is presented in Alg. 1. In step 1, we compute the ranking vector $\mathbf{r}$ (e.g., by the power method, etc). Then after some initializations (steps 2-5), we select k nodes one-by-one as follows. At each time, we compute the score vector $\mathbf{s}$ in step 7. Then, we select one node with the highest score in the vector $\mathbf{s}$ and add it to the subset $\mathcal{S}$ (steps 8-9). After that, we use the selected node to update the two reference vectors $\mathbf{u}$ and $\mathbf{v}$ (steps 10-11). Note that '$\otimes$' denote the element-wise product between two matrices/vectors. Intuitively, the score vector $\mathbf{s}$ keeps the *marginal* contribution of each node for the goodness measure given the current selected subset $\mathcal{S}$. From step 7, it can be seen that at each iteration, the values of such marginal contribution either keeps unchanged or decreases. This is consistent with P3 of Theorem 1 - as there are more and more nodes in the subset $\mathcal{S}$, the marginal contribution of each node is monotonically non-increasing. It is worth pointing out that we use the original normalized adjacency matrix $\mathbf{A}$, instead of the 'Google matrix' $\mathbf{B}$ in Alg. 1. This is because for many real graphs, the matrix $\mathbf{A}$ is often very sparse, whereas the matrix $\mathbf{B}$ might not be[3]. In the case $\mathbf{B}$ is dense, it is not efficient in either time or space to use $\mathbf{B}$ in Alg. 1.

In Alg. 1, although we try to optimize a goodness measure that is not affected by the ordering of different nodes in the subset, we can still output an ordered list to the user based on in which iteration these nodes are selected - earlier selected nodes in Alg. 1 are placed at the top of the resulting top-*k* ranking list. This ordering naturally meets the diminishing returns need when the user is seeking for a diverse top-*k* ranking list as we analyzed in subsection 3.1.

4.3 Proofs and Analysis

Here, we analyze the *optimality* as well as the *complexity* of the proposed algorithm. We show that our DRAGON leads to a *near-optimal* solution, and at the same time it enjoys *linear scalability* in both time and space.

Optimality. The optimality of the proposed DRAGON is given in Lemma 1. According to Lemma 1, our DRAGON is *near-optimal* - its solution is within a fixed fraction $(1 - 1/e \approx 0.63)$ from the global optimal one. Given the hardness of Problem 2, such near-optimality is acceptable in terms of optimization quality.

LEMMA 1. **Near-Optimality of DRAGON.** *Let $\mathcal{S}$ be the subset found by* DRAGON*; $|\mathcal{S}| = k$; and $\mathcal{S}^* = argmax_{|\mathcal{S}|=k} f(\mathcal{S})$. We have that $f(\mathcal{S}) \geq (1 - 1/e) f(\mathcal{S}^*)$, where e is the base of the natural logarithm.*

PROOF. Omitted for Brevity □

[2]It is worth pointing out that it is not even easy to find an optimal solution for the relaxed problem by quadratic programming because the matrix $\mathbf{D}$ (1) might be asymmetric and (2) is not always semi-positive definite.

[3]To see this, notice that $\mathbf{B}$ is a full matrix if $\mathbf{p}$ is uniform.

Algorithm 1 DRAGON for Problem 2

Input: The row-normalized adjacency matrix $\mathbf{A}$ of the graph, the damping factor c, the query vector $\mathbf{p}$, and the budget k;
Output: A subset of k nodes $\mathcal{S}$.

1: Compute the ranking vector $\mathbf{r}$: $\mathbf{r} = c\mathbf{A}'\mathbf{r} + (1-c)\mathbf{p}$;
2: Initialize $\mathcal{S}$ as the empty set; set $\mathbf{u} = \mathbf{v} = \mathbf{0}_{n\times 1}$;
3: **for** $i = 1 : n$ **do**
4: Initialize $\hat{\mathbf{s}}(i) = (2 - c\mathbf{A}(i,i) - (1-c)\mathbf{p}(i))\mathbf{r}(i)$;
5: **end for**
6: **for** iter $= 1 : k$ **do**
7: Compute the score vector $\mathbf{s} = \hat{\mathbf{s}} - \mathbf{u} \otimes \mathbf{r} - \mathbf{v}$;
8: Find $i = \text{argmax}_j \mathbf{s}(j)(j = 1, ..., n; j \notin \mathcal{S})$;
9: Add node i into $\mathcal{S}$;
10: Update $\mathbf{u} \leftarrow \mathbf{u} + c\mathbf{A}(:,i) + (1-c)\mathbf{p}(i)\mathbf{1}_{n\times 1}$;
11: Update $\mathbf{v} \leftarrow \mathbf{v} + c\mathbf{A}'(:,i)\mathbf{r}(i) + (1-c)\mathbf{r}(i)\mathbf{p}$;
12: **end for**
13: Return the subset $\mathcal{S}$

Time Complexity. The time complexity of the proposed DRAGON is given in Lemma 2. According to Lemma 2, our DRAGON has *linear* time complexity wrt the size of the graph. Therefore it is scalable to large graphs in terms of computational time.

LEMMA 2. **Time Complexity of DRAGON.** *The time complexity of Alg. 1 is* $O(m + nk)$.

PROOF. Omitted for brevity. □

We would like to point out that the proposed DRAGON can be further sped up. Firstly, notice that the $O(m)$ term in Lemma 2 comes from computing the ranking vector $\mathbf{r}$ (step 1) by the most commonly used power method. There are a lot of fast methods for computing $\mathbf{r}$, either by effective approximation (e.g., [37]), or by parallelism (e.g. [13]). These methods can be naturally plugged in our DRAGON, which might lead to further computational savings. Secondly, the $O(nk)$ term in Lemma 2 comes from the greedy selection step in steps 6-12. Thanks to the monotonicity of $f(\mathcal{S})$ as we show in Theorem 1, we can use the similar lazy evaluation strategy as [20] to speed up this process, without sacrificing the optimization quality.

Space Complexity. The space complexity of the proposed DRAGON is given in Lemma 3. According to Lemma 3, our DRAGON has *linear* space complexity wrt the size of the graph. Therefore it is also scalable to large graphs in terms of space cost.

LEMMA 3. **Space Complexity of DRAGON.** *The space complexity of Alg. 1 is* $O(m + n + k)$.

PROOF. Omitted for brevity. □

4.4 Discussion - Comparisons

In literature, there exist two other methods to encourage diversity in the top-*k* ranking list for personalized PageRank. Here, we make a comparison in terms of *optimality, convergence,* and *scalability* of different methods. ARW [42] is based on an intuitive heuristic by greedily selecting the highest ranked node and setting it as the absorbing state. From theoretical point of view, it is not clear what ARW [42] tries to optimize. And also, it requires a matrix inverse of the same size of the graph, which is not scalable to large graphs. RRW [27] is based on vertex reinforced random walk [31]. Compared with ARW [42], it makes an important step forward by providing some optimization explanations via defining a *time-varying* objective function that changes at each iteration step. However, it is still not clear what overall metric it tries to measure; and how good its optimization solution is. Moreover, RRW [27] introduced some modifications and approximation techniques to the original vertex reinforced random walk, and it is not clear how the modified vertex reinforcement random walk converges[4].

Table 2: Comparison of different methods. Our proposed DRAGON is the only method that leads to a near-optimal solution with linear scalability.

Method	Measure	Optimality	Scalability	Convergence
ARW [42]	NA	NA	No	Yes
RRW [27]	Partial	NA	Yes	NA
DRAGON	Yes	Near-optimal	Yes	Yes

5. EXPERIMENTAL EVALUATIONS

In this section, we provide empirical evaluations for the proposed DRAGON. Our evaluations mainly focus on (1) the effectiveness and (2) efficiency of the proposed DRAGON.

5.1 Experimental Setup

Data sets. We use the DBLP publication data[5] to construct a co-authorship network, where each node is an author and the edge weight is the number of the co-authored papers between the two corresponding persons. Overall, we have $n = 418,236$ nodes and $m = 2,753,798$ edges. We also construct much smaller co-authorship networks, using the authors from only one conference (e.g., *KDD, SIGIR, SIGMOD*, etc.). For example, *KDD* is the co-authorship network for the authors in the 'KDD' conference. These smaller co-authorship networks typically have a few thousand nodes and up to a few tens of thousands edges. We also construct the co-authorship networks, using the authors from multiple conferences (e.g., *KDD+SIGIR*). For these graphs, we denote them as *Sub(n,m)*, where n and m are the numbers of nodes and edges in the graph, respectively.

Machine configurations. For the computational cost and scalability, we report the wall-clock time. All the experiments ran on the same machine with four 2.4GHz AMD CPUs and 48GB memory, running Linux (2.6 kernel). For all the quantitative results, we randomly generate a query vector $\mathbf{p}$ and feed it into different methods for a top-*k* ranking list with the same length. We repeat it 100 times and report the average.

Evaluation criteria. To the best of our knowledge, there is no universally accepted measure for diversity. In [27], the authors suggested an intuitive notion based on the density of the induced subgraph from the original graph $\mathbf{A}$ by the subset $\mathcal{S}$. The intuition is as follows: the lower the density (i.e., the less 1-step neighbors) of the induced subgraph, the more diverse the subset $\mathcal{S}$. Here, we generalize this notion to the t-step graph in order to also take into account the effect of those in-direct neighbors. Let Sign(.) be a binary function operated element-wise on a matrix, i.e., $\mathbf{Y} = \text{Sign}(\mathbf{X})$, where $\mathbf{Y}$ is a matrix of the same size as $\mathbf{X}$, $\mathbf{Y}(i,j) = 1$ if $\mathbf{X}(i,j) > 0$, $\mathbf{Y}(i,j) = 0$ otherwise. We define the t-step connectivity matrix $\mathbf{C}^t$ as $\mathbf{C}^t = \text{Sign}(\sum_{i=1}^{t} \mathbf{A}^i)$. That is, $\mathbf{C}^t(i,j) = 1$ (0) means that node i can (cannot) reach node j on the graph $\mathbf{A}$ within t-steps/hops. With this $\mathbf{C}^t$ matrix, we define the diversity of a given subset $\mathcal{S}$ as eq (8). Here, the value of $\text{Div}(t)$ is always between 0.5 and 1 - higher means more diverse. If all the nodes in $\mathcal{S}$ are reachable from each other within t-steps, we say that the subset $\mathcal{S}$

[4]Even if it converges, its stationary state might not be unique according to [31].

[5]http://www.informatik.uni-trier.de/~ley/db/

is the least diverse ($\text{Div}(t) = 0.5$). On the other extreme, If all the nodes in $\mathcal{S}$ cannot reach each other within t-steps, the subset $\mathcal{S}$ is the most diverse ($\text{Div}(t) = 1$).

$$\text{Div}(t) = \frac{1}{1 + \sum_{i,j \in \mathcal{S}, i \neq j} \mathbf{C}^t(i,j) / (|\mathcal{S}| \cdot (|\mathcal{S}| - 1))} \quad (8)$$

For the task of top-*k* ranking, the notion of diversity alone, though important, might not be enough for the information need. For example, if we simply randomly select k nodes as the top-*k* ranking list, these k nodes might not connected with each other at all given that the length of the ranking list k is usually much smaller than the number of nodes n in the graph. Therefore, it has a high diversity. However, it is unlikely that such a ranking list can well fit the user's information need since each of them might have very low relevance score. In other words, a diversified top-*k* ranking list should also have high relevance. That said, we will mainly focus on evaluating how different methods balance between the diversity and the relevance.

Notice that the relevance score for each individual node is often very small on large graphs (since the L_1 norm of the ranking vector is 1). To make the two quantities (diversity vs. relevance) comparable with each other, we need to normalize the relevance scores. Let $\hat{\mathcal{S}}$ be the top-*k* ranking list by the original personalized PageRank, we define the normalized relevance score for a given subset $\mathcal{S}(|\mathcal{S}| = k)$ as eq (9). Since the personalized PageRank always gives the k most relevant nodes, the Rel defined in eq (9) is always between 0 and 1 - higher means more relevant.

$$\text{Rel} = \frac{\sum_{i \in \mathcal{S}} \mathbf{r}(i)}{\sum_{i \in \hat{\mathcal{S}}} \mathbf{r}(i)} \quad (9)$$

5.2 Effectiveness of DRAGON: Case Studies

Let us start with an illustrative example to gain some visual intuitions. In Fig. 1, we show a fictitious co-authorship network, where each node corresponds to an author (e.g., *John, Smith, etc*), and the edge weight is the number of the co-authored papers. There are three communities in this network (e.g., *DM, DB* and *IR*). From Fig. 1, we can see that node 1 has very strong connections to the *DM* community. In other words, *DM* might be his/her major research interest. In addition, s/he also has some connections to the *IR* and *DB* communities. Given the budge $k = 3$, personalized PageRank returns all the three nodes (nodes 2, 3 and 5) from *DM* community which is consistent with the intuition since personalized PageRank solely focuses on the relevance. In contrast, the proposed DRAGON returns nodes 2, 6, and 10, each of which is still relevant enough to the query node 1. At the same time, they are diversified from each other, covering the whole spectrum of his/her research interest (*DM, DB,* and *IR*).

We also conduct cast studies on real graphs. We construct a co-authorship networks from *SIGIR* (the major conference on information retrieval) and *ICML* (the major conference on machine learning). We issue a query to find the top-*10* co-authors for Prof. *Yiming Yang*. The results are shown in Fig. 2. We compare it with the original personalized PageRank. *Yiming Yang* is a professor from Carnegie Mellon University; and she has broad interest in information retrieval and machine learning. From Fig. 2, we have the following observations. Firstly, both DRAGON and personalized PageRank share the same authors for the top-*3* returned authors, indicating that DRAGON also captures those highly relevant authors wrt the querying author. Secondly, our DRAGON returns a more diverse list of authors. For example, although *ChengXiang Zhai* is not a co-author of *Yiming Yang*, they shares a lot of research interest in information retrieval, and has a lot of indirect connections through other *IR* people. In contrast, the existence of some

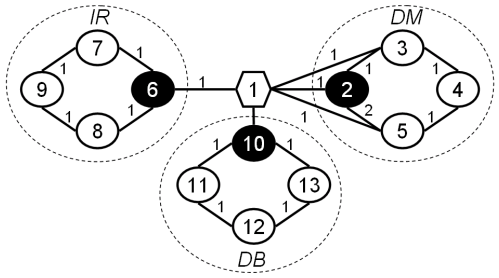


Figure 1: An illustrative example of a co-authorship network with three communities. Given the query node 1 and the budget $k = 3$, the proposed DRAGON returns three relevant and diversified nodes (2, 6, and 10, in black). In contrast, personalized PageRank returns nodes 2, 3 and 5 (all from the *DM* community).

authors in the ranking list by personalized PageRank is somehow redundant, in terms of helping the user to understand Prof. *Yiming Yang*'s whole collaboration network. For example, Prof. *Alex G. Hauptmann* is also from Carnegie Mellon University. Although, he has a lot of co-authored papers with *Yiming Yang*, they are also co-authored with *Jian Zhang* and *Rong Jin*. Therefore, given that *Jian Zhang* and *Rong Jin* are already in the ranking list, his existence does not provide much marginal information about *Yiming Yang*'s collaboration network. As a quantitative indicator, the average degree of the induced subgraph by DRAGON is only *2.8*, which is much lower (i.e., more diverse) than that by personalized PageRank (*4.3*). Finally, notice that for some authors, although they show up in both lists, their positions in the ranking list are different. For example, *Jian-Yun Nie* shows at the 4^{th} and the 8^{th} positions in the two ranking lists, respectively. This is because *Jian-Yun Nie* makes the top-*4* authors more diverse compared with *Thomas Pierce*, although its individual relevance score is lower than the latter.

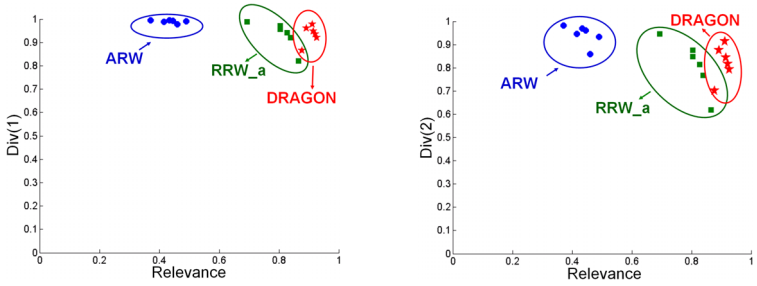
5.3 Comparison with Alternative Methods for Diversified Ranking on Graphs

We compare the proposed DRAGON with ARW [42] and RRW [27], both of which also aim to improve the diversity of personalized PageRank. We skip the comparison with MMR [6] for brevity since [27] shows that its performance is not as good as RRW for the graph-type data. For RRW [27], it has two variants based on different approximation methods it actually uses: the one based on the cumulative estimation (referred to as 'RRW-a') and the other one based on the pointwise estimation (referred to as 'RRW-b').

First, let us compare how different methods balance between the relevance and the diversity. Fig. 3 shows the results on the *NIPS* co-authorship network. We test with different budgets ($k = 10, 20, 30, 40, 50, 100$). In Fig. 3, $\text{Div}(1)$ means that we only consider 1-step neighbors to measure the diversity (i.e., setting $t = 1$ in eq (8)). $\text{Div}(2)$ means that we consider both 1-step and 2-step neighbors (i.e., setting $t = 2$ in eq (8)). We only present the results by RRW-a since RRW-b gives similar results. From Fig. 3, we can see that all the three methods are effective to improve the diversity. The proposed DRAGON achieves a better balance between the relevance and the diversity. For ARW, although it gives the highest diversity score, its (normalized) relevance score is too low - only about half of the other two methods. This is because in ARW, *only* the first node is selected according to the relevance;

Jian Zhang	Jian Zhang
Rong Jin	Bryan Kisiel
Bryan Kisiel	Rong Jin
Jian-Yun Nie	Thomas Pierce
Wei-Ying Ma	Monica Rogati
Thomas Pierce	Tom Ault
Jan O. Pedersen	Alex G. Hauptmann
Ni Lao	Jian-Yun Nie
Jaime G. Carbonell	Ni Lao
ChengXiang Zhai	Abhimanyu Lad
(a) Dragon	(b) Personalized PageRank

Figure 2: Top-10 authors for Prof. Yiming Yang. Our DRAGON return a relevant, but more diverse list of authors. The difference between the two lists is highlighted in black.



(a) Diversity by 1-step neighbors (b) Diversity by 1- and 2-step neighbors

Figure 3: Diversity and Relevance trade-off of different methods.

and all the remaining *(k-1)* are selected by diversity. As for RRW-a, both its relevance and diversity scores are lower than the proposed DRAGON. It is interesting to notice from Fig. 3(b) that the diversity of RRW-a drops a lot when it is measured by within 2-step neighbors (i.e., Div(2)). This is consistent with the intuition of RRW. In RRW (both RRW-a and RRW-b), it achieves the diversity by encouraging 1-step neighboring nodes to compete with each other. Consequently, the density of its within 1-step induced subgraph might low (i.e., high diversity), it is not necessary the case for the within t-step ($t \geq 2$) induced subgraph.

In order to test how the overall performance of different methods vary across different data sets, we take the average between relevance and diversity scores. The results are presented in Fig. 4, using four different co-authorship networks (*SIGMOD, NIPS, SIGIR, SIGGRAPH*). For the space limitation, we omit the results when the diversity is measured by within 1-steps neighbors, which is similar as the results by within 2-steps neighbors. It can be seen that the proposed DRAGON consistently performs the best.

5.4 Comparison with Alternative Optimization Methods

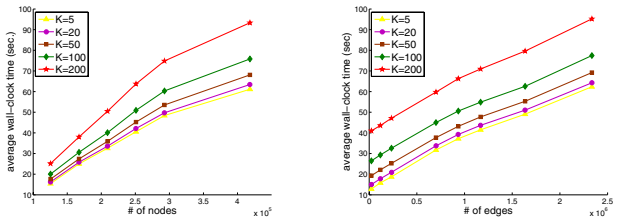
Here, we evaluate the effectiveness and the efficiency of the proposed DRAGON in terms of maximizing the goodness measure $f(\mathcal{S})$. We compared it with the two methods we introduced in subsection 4.1. We also compare it with two other heuristics. The first method (referred to as 'Heuristic1') starts with generating a candidate pool (e.g., the top $10 \times k$ most relevant nodes), picks one seed node, and then repeatedly adds the most dis-similar (measured by **A**) node into the ranking list from the candidate pool. The second method (referred to as 'Heuristic2') also starts with generating a candidate pool, puts all the nodes from candidate pool in the list, and then repeatedly drops a most similar (measured by **A**) node from the list.

First, let us evaluate how the different methods balance between the optimization quality (measured by $f(\mathcal{S})$) and the speed (measured by wall-clock time). Fig. 5 shows the results from the co-authorship network of *NIPS* and *KDD* conferences with the budget $k = 20$, where $f(\mathcal{S})$ is normalized by the highest one among different methods. It can be seen that the proposed DRAGON is the best - it leads to the highest optimization quality (i.e., highest $f(\mathcal{S})$) with the least amount of wall-clock time. Notice that the y-axis is in logarithm scale.

We also conduct experiments on the co-authorship network constructed from multiple conferences. Fig. 6 shows the results on these data sets with the budget $k = 20$. Here *Sub(n,m)* means a co-authorship network with n nodes and m edges. We stop the program if it takes more than 100,000 seconds (i.e., more than 10 days). It can be seen from Fig. 6 that the proposed DRAGON is consistently best across all the different data sets - it leads to highest optimization quality (i.e., highest $f(\mathcal{S})$) with least amount of wall-clock time. The normalized $f(\mathcal{S})$ for 'Lin-QP' is missing for *Sub(24K,114K)* because it fails to finish within 100,000 seconds. This indicates that it is not feasible for large graphs. For the smaller graphs, 'Lin-QP' leads to slightly lower $f(\mathcal{S})$ than the proposed DRAGON; but it requires 3-5 orders of magnitude wall-clock time. For all the other comparative methods, they lead to worse optimization quality with longer wall-clock time.

5.5 Scalability

We also evaluate the scalability of DRAGON. When we evaluate the scalability wrt the number of the nodes in the graph, we fix the number of edges and vice versa. The results in Fig. 7 are consistent with the complexity analysis in subsection 4.3 - the proposed DRAGON scales linearly wrt both n and m, which means that it is suitable for large graphs.



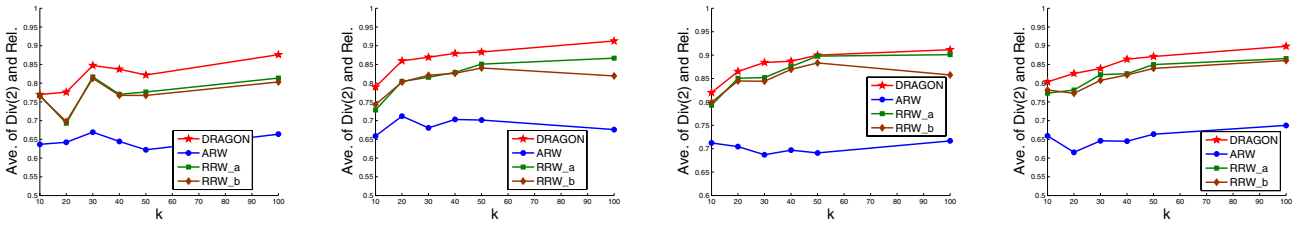
(a) Time vs. # of nodes (b) Time vs. # of edges

Figure 7: Scalability of DRAGON. The proposed DRAGON scales linearly wrt the size of the graph.

6. RELATED WORK

In this section, we review the related work, which can be categorized into four parts: ranking on graphs, diversity, set-level optimization for data mining and general graph mining.

Ranking on Graphs. Personalized PageRank is one of the most fundamental and most widely used ranking methods on graphs. It has been successfully applied to many high-impact applications [30, 11].Many other ranking methods on graphs are built upon, and/or share the similar ideas as personalized PageRank, such as RalationalRank [9, 3], random walk with restart [37], SimRank [22], etc. Because of its generality and wide applicability, we choose Personalized PageRank as the starting point of our method. Other ranking methods on graphs include HITS [15], electricity-based methods [17], etc. There also exist a lot of work to speed-up the computation of personalized PageRank, such as [37, 33, 13]. It is worth pointing out that all these fast algorithms can be naturally plugged into the proposed DRAGON to gain further savings in computational time.



(a) *SIGMOD* (b) *NIPS* (c) *SIGIR* (d) *SIGGRAPH*

Figure 4: Comparison with alternative methods for diversified ranking on graphs. The average of the relevance and the diversity vs. the budget k (larger is better). The proposed DRAGON(red star) consistently performs the best.

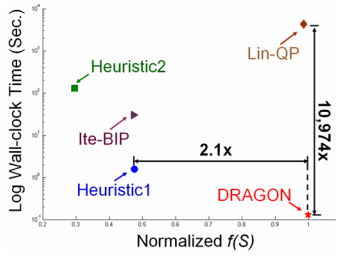
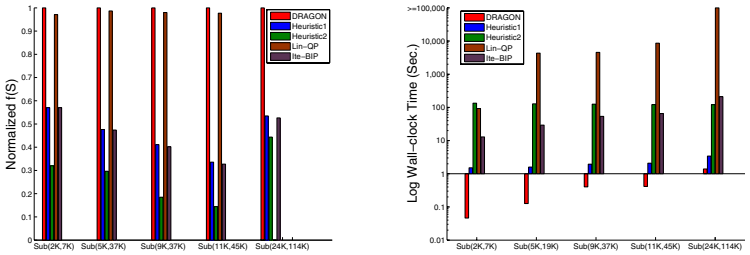


Figure 5: Wall-clock time vs. quality on the *NIPS+KDD* co-authorship Network. The y-axis is in logarithm scale. The proposed DRAGON is the best. It has the highest f($\mathcal{S}$) with the least amount of time.



(a) Comparison on the normalized f($\mathcal{S}$) (higher is better)

Comparison on wall-clock time (lower is better)

Figure 6: Comparison of different optimization methods. Our DRAGON(the left most one) always leads to the highest f($\mathcal{S}$), with the least amount of time. Best viewed in color.

Diversity. It is now widely recognized that diversity is a highly desired property in many data mining tasks, such as expertise and legal search [32], recommendation system [43], blog filtering [7], document summarization [6], etc. It is a powerful tool to address the uncertainty and ambiguity; and/or to cover the different aspects of an information need [32]. For the graph-type data, [42] was among the first to address the diversified ranking on graphs. [27] proposed to balance between the relevance and diversity based on the vertex reinforcement random walk on graphs . However, they suffer from some important subtle issues as we show in subsection 4.4. There are also a lot of algorithms to improve the diversity for other types of data (e.g., document, etc), including [6, 41, 21], etc.

Set-level Optimization for Data Mining. In the recent years, set-level optimizationhas been playing a very important role in many data mining tasks. Many set-level optimization problems are NP-hard. Therefore, it is difficult, if not impossible, to find the global optimal solutions. However, if the function is monotonic sub-modular with 0 function value for the empty set, a greedy strategy can lead to a provably near-optimal solution [18]. This powerful strategy has been recurring in many different settings, e.g., immunization, outbreak detection, blog filtering, sensor placement, influence maximization, structure learning, etc. (See [18] for a comprehensive review). In this paper, we introduce a new type of submodular function tailored for diversified ranking on large graphs.

General Graph Mining. There is a lot of work on graph mining. Representative works include pattern and law mining [5], frequent substructure discovery [39], compression [26], fraud and anomaly detection [29], community mining and graph partition [14, 34, 25, 40], social action tracking [36], user click-through modeling [1, 2], collaborative filtering [16, 8, 35, 12], term formation [19], network classification [28], link prediction [23, 24, 4], etc.

7. CONCLUSION

In this paper, we address the diversified ranking on large graphs from an optimization point of view. To the best of our knowledge, this is the first work for diversified ranking on large graphs that (1) has a clear optimization formulation (see eq. (3)); (2) finds provably near-optimal solutions (see Theorem 1 and Lemma 1); and (3) enjoys the linear scalability (see Lemma 2 and Lemma 3). Our experimental evaluations on real graphs validate that our method is (1) indeed effective to balance the relevance and the diversity in top-k ranking; and (2) scalable to large graphs.

8. ACKNOWLEDGEMENT

Research was sponsored by the Army Research Laboratory and was accomplished under Cooperative Agreement Number W911NF-09-2-0053. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

9. REFERENCES

[1] C. L. 0001, F. Guo, and C. Faloutsos. Bbm: bayesian browsing model from petabyte-scale data. In *KDD*, pages 537–546, 2009.

[2] D. Agarwal, A. Z. Broder, D. Chakrabarti, D. Diklic, V. Josifovski, and M. Sayyadian. Estimating rates of rare events at multiple resolutions. In *KDD*, pages 16–25, 2007.

[3] A. Angel, S. Chaudhuri, G. Das, and N. Koudas. Ranking objects based on relationships and fixed associations. In *EDBT'09*, pages 910–921, 2009.

[4] L. Backstrom and J. Leskovec. Supervised random walks: predicting and recommending links in social networks. In *WSDM*, pages 635–644, 2011.

[5] A. Broder, R. Kumar, F. Maghoul1, P. Raghavan, S. Rajagopalan, R. Stata, A. Tomkins, and J. Wiener. Graph structure in the web: experiments and models. In *WWW Conf.*, 2000.

[6] J. G. Carbonell and J. Goldstein. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *SIGIR*, pages 335–336, 1998.

[7] K. El-Arini, G. Veda, D. Shahaf, and C. Guestrin. Turning down the noise in the blogosphere. In *KDD*, pages 289–298, 2009.

[8] Y. Ge, H. Xiong, A. Tuzhilin, K. Xiao, M. Gruteser, and M. J. Pazzani. An energy-efficient mobile recommender system. In *KDD*, pages 899–908, 2010.

[9] F. Geerts, H. Mannila, and E. Terzi. Relational link-based ranking. In *VLDB*, pages 552–563, 2004.

[10] G. H. Golub and C. F. V. Loan. *Matrix Perturbation Theory*. The Johns Hopkins University Press, 1996.

[11] T. H. Haveliwala. Topic-sensitive pagerank. *WWW*, pages 517–526, 2002.

[12] D. Heckerman, D. M. Chickering, C. Meek, R. Rounthwaite, and C. M. Kadie. Dependency networks for collaborative filtering and data visualization. In *UAI*, pages 264–273, 2000.

[13] U. Kang, C. E. Tsourakakis, and C. Faloutsos. Pegasus: A peta-scale graph mining system. In *ICDM*, pages 229–238, 2009.

[14] G. Karypis and V. Kumar. Multilevel -way hypergraph partitioning. In *DAC*, pages 343–348, 1999.

[15] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *J. ACM*, 46(5):604–632, 1999.

[16] Y. Koren. Collaborative filtering with temporal dynamics. In *KDD*, pages 447–456, 2009.

[17] Y. Koren, S. C. North, and C. Volinsky. Measuring and extracting proximity in networks. In *KDD*, pages 245–255, 2006.

[18] A. Krause and C. Guestrin. Beyond convexity - submodularity in machine learning. In *ICML*, 2008.

[19] T. Lappas, K. Liu, and E. Terzi. Finding a team of experts in social networks. In *KDD*, pages 467–476, 2009.

[20] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. M. VanBriesen, and N. S. Glance. Cost-effective outbreak detection in networks. In *KDD*, pages 420–429, 2007.

[21] L. Li, K. Zhou, G.-R. Xue, H. Zha, and Y. Yu. Enhancing diversity, coverage and balance for summarization through structure learning. In *WWW*, pages 71–80, 2009.

[22] P. Li, H. Liu, J. X. Yu, J. He, and X. Du. Fast single-pair simrank computation. In *SDM*, pages 571–582, 2010.

[23] D. Liben-Nowell and J. Kleinberg. The link prediction problem for social networks. In *Proc. CIKM*, 2003.

[24] R. Lichtenwalter, J. T. Lussier, and N. V. Chawla. New perspectives and methods in link prediction. In *KDD*, pages 243–252, 2010.

[25] A. S. Maiya and T. Y. Berger-Wolf. Sampling community structure. In *WWW*, pages 701–710, 2010.

[26] H. Maserrat and J. Pei. Neighbor query friendly compression of social networks. In *KDD*, pages 533–542, 2010.

[27] Q. Mei, J. Guo, and D. R. Radev. Divrank: the interplay of prestige and diversity in information networks. In *KDD*, pages 1009–1018, 2010.

[28] J. Neville, B. Gallagher, and T. Eliassi-Rad. Evaluating statistical tests for within-network classifiers of relational data. In *ICDM*, pages 397–406, 2009.

[29] C. C. Noble and D. J. Cook. Graph-based anomaly detection. In *KDD*, pages 631–636, 2003.

[30] L. Page, S. Brin, R. Motwani, and T. Winograd. The PageRank citation ranking: Bringing order to the web. Technical report, Stanford Digital Library Technologies Project, 1998. Paper SIDL-WP-1999-0120 (version of 11/11/1999).

[31] R. Pemantle. Vertex reinforced random walk. *Prob. Th. and Rel. Fields*, pages 117–136, 1992.

[32] F. Radlinski, P. N. Bennett, B. Carterette, and T. Joachims. Redundancy, diversity and interdependent document relevance. *SIGIR Forum*, 43(2):46–52, 2009.

[33] P. Sarkar and A. W. Moore. Fast nearest-neighbor search in disk-resident graphs. In *KDD*, pages 513–522, 2010.

[34] V. Satuluri and S. Parthasarathy. Scalable graph clustering using stochastic flows: applications to community discovery. In *KDD*, pages 737–746, 2009.

[35] H. Shan and A. Banerjee. Generalized probabilistic matrix factorizations for collaborative filtering. In *ICDM*, pages 1025–1030, 2010.

[36] C. Tan, J. Tang, J. Sun, Q. Lin, and F. Wang. Social action tracking via noise tolerant time-varying factor graphs. In *KDD*, pages 1049–1058, 2010.

[37] H. Tong, C. Faloutsos, and J.-Y. Pan. Fast random walk with restart and its applications. In *ICDM*, pages 613–622, 2006.

[38] L. Wu. Social network effects on performance and layoffs: Evidence from the adoption of a social networking tool. *Job Market Paper*, 2011.

[39] D. Xin, J. Han, X. Yan, and H. Cheng. Mining compressed frequent-pattern sets. In *VLDB*, pages 709–720, 2005.

[40] X. Yin, J. Han, and P. S. Yu. Cross-relational clustering with user's guidance. In *KDD*, pages 344–353, 2005.

[41] Y. Yue and T. Joachims. Predicting diverse subsets using structural svms. In *ICML*, pages 1224–1231, 2008.

[42] X. Zhu, A. B. Goldberg, J. V. Gael, and D. Andrzejewski. Improving diversity in ranking using absorbing random walks. In *HLT-NAACL*, pages 97–104, 2007.

[43] C.-N. Ziegler, S. M. McNee, J. A. Konstan, and G. Lausen. Improving recommendation lists through topic diversification. In *WWW*, pages 22–32, 2005.