Inside the Atoms: Ranking on a Network of Networks

Jingchao Ni [1], Hanghang Tong [2], Wei Fan [3], and Xiang Zhang [1]
[1]Department of Electrical Engineering and Computer Science, Case Western Reserve University
[2]School of Computing, Informatics, Decision Systems Engineering, Arizona State University
[3]Huawei Noahs Ark Lab
[1]{jingchao.ni, xiang.zhang}@case.edu, [2]htong6@asu.edu, [3]david.fanwei@huawei.com

ABSTRACT

Networks are prevalent and have posed many fascinating research questions. How can we spot similar users, e.g., virtual identical twins, in Cleveland for a New Yorker? Given a query disease, how can we prioritize its candidate genes by incorporating the tissue-specific protein interaction networks of those similar diseases? In most, if not all, of the existing network ranking methods, the nodes are the ranking objects with the finest granularity.

In this paper, we propose a new network data model, a Network of Networks (NoN), where each node of the main network itself can be further represented as another (domain-specific) network. This new data model enables to compare the nodes in a broader context and rank them at a finer granularity. Moreover, such an NoN model enables much more efficient search when the ranking targets reside in a certain domain-specific network. We formulate ranking on NoN as a regularized optimization problem; propose efficient algorithms and provide theoretical analysis, such as optimality, convergence, complexity and equivalence. Extensive experimental evaluations demonstrate the effectiveness and the efficiency of our methods.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database Applications—*Data mining*

Keywords

Network of Networks; Ranking; Query

1. INTRODUCTION

How can we spot similar users, e.g., virtual identical twins, in Cleveland for a New Yorker? Given a disease, how can we prioritize its candidate genes by incorporating the tissue-specific protein interaction networks of those similar diseases? Among all the researchers in the area of bioinformatics, who is most likely to collaborate with a given data miner in the near future? In these applications, networks (or graphs[1]) provide a natural data model to capture the relationship among different objects.

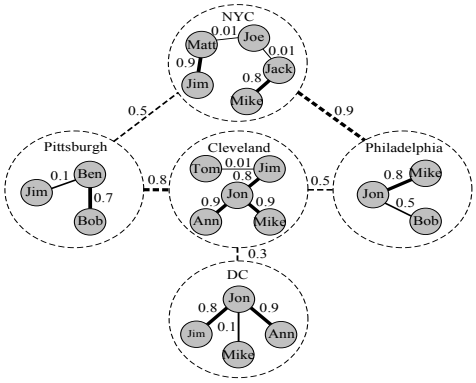


Figure 1: An example of NoN. The main network is represented by dashed nodes and edges. The domain-specific networks are represented by solid nodes and edges.

In the past decade, many graph ranking algorithms have been proposed. Please see Section 6 for a brief review. Despite their own success, a common limitation of these methods is that *ranking stops at the node (atom) level*. That is, the nodes are the ranking objects with the finest granularity in these existing ranking algorithms.

In this paper, we take one step further and propose a new data model - **a Network of Networks (NoN)**. In this NoN model, each node of the main network can be represented as another network. Let us elaborate using the example in Figure 1. Here, the dashed network represents a geo-proximity network (the main network), where the nodes are five different cities and links indicate the geo-proximity among different cities. Each node of this main network is further represented as a social network (the domain-specific network), where nodes are users and links denote their friendship. Collectively, we call this structure as a (geo-proximity) network of (social) networks.

The main advantage of this new NoN model is that it enables to compare the nodes in a broader context and rank them at a finer granularity. Take Figure 1 as the example, using the existing network models, we could either rank the geo-centrality of the cities, or the popularity of the users within each domain-specific network. On the other hand, by the NoN model, we will be able to rank all the users by considering both their popularity within each domain-specific network as well as the geo-centrality of the locations they belong to. We refer to this setting as CROSSRANK. Moreover, we

[1]In this paper, we use network and graph interchangeably.

KDD'14, August 24–27, 2014, New York, NY, USA.

http://dx.doi.org/10.1145/2623330.2623643.

Table 1: Symbols

Symbol	Definition
$\mathbf{G}$	the adjacency matrix of main network
$\mathcal{A}$	domain specific networks $\mathcal{A} = \{\mathbf{A}_1, ..., \mathbf{A}_g\}$
$\mathbf{A}_i$	the i^{th} domain specific network
θ	the one-to-one mapping function
$\mathcal{R}$	a network of networks $\mathcal{R} =< \mathbf{G}, \mathcal{A}, \theta >$
$\mathbf{r}_i$	the ranking vector for $\mathbf{A}_i$
$\mathbf{e}_i$	the i^{th} query vector for $\mathbf{A}_i$
$\mathbf{I}_n$	an $n \times n$ identity matrix
$\mathcal{I}_{ij}$	the set of common nodes between $\mathbf{A}_i$ and $\mathbf{A}_j$
$d_m(i)$	the degree of the i^{th} node in $\mathbf{G}$
$\mathbf{D}_m$	the degree matrix: $\mathbf{D}_m = \text{diag}(d_m(1), ..., d_m(g))$
g	the number of nodes in the main network
n_i	the number of nodes in $\mathbf{A}_i$, $(i = 1, ..., g)$
m_i	the number of edges in $\mathbf{A}_i$, $(i = 1, ..., g)$
c, a	the parameters $0 < c < 1$, and $a > 0$

might be interested in finding who are the most similar from New York City for *Jon* from Cleveland, i.e., who are the virtual identity twins of *Jon* in New York City. In this case, the ranking targets reside in a certain domain-specific network. We refer to this setting as CROSSQUERY. The NoN model will enable much more efficient search for this problem.

The NoN model can be observed in many real-world applications. As another example, consider a NoN, where each node in the main network is a research area, and the domain-specific networks are the co-author networks in different areas. By the NoN model, we can rank authors according to their contributions to multiple areas using CROSSRANK; and find future collaborators for a researcher in a different area using CROSSQUERY.

The main contributions of this paper can be summarized as:

- **New Data Model**. We propose a novel network data model, a network of networks, which enables network ranking to go beyond the atom/node level (Section 2).
- **Algorithms and Analysis**. We propose two new ranking algorithms (CROSSRANK and CROSSQUERY) in NoN and analyze their *optimality*, *convergence*, *complexity* and *relationship* with existing ranking algorithms (Section 3 and 4).
- **Empirical Evaluations**. We perform extensive experiments on real datasets to validate the effectiveness and efficiency of the proposed algorithms (Section 5).

2. PROBLEM DEFINITIONS

Table 1 lists the main symbols used throughout this paper. We use the bold capital letters to denote matrices (e.g., $\mathbf{G}$, $\mathbf{A}$, etc.), the bold lower cases for vectors (e.g., $\mathbf{r}$) and calligraphic letters for sets (e.g., $\mathcal{A}$). In this paper, we consider weighted undirected networks. We denote the networks by their corresponding weighted adjacency matrices. For example, a network with g nodes can be denoted by a $g \times g$ adjacency matrix $\mathbf{G}$. The rows/columns of the matrix $\mathbf{G}$ represents the nodes of the network and its non-zero element $\mathbf{G}(i, j)$ measures the link strength from node i to node j. With the above notations, a network of networks (NoN) can be formally defined as follows.

Definition 1. **Network of Networks (NoN).** Given a $g \times g$ **main** network $\mathbf{G}$, a set of g **domain-specific** networks $\mathcal{A} = \{\mathbf{A}_1, ..., \mathbf{A}_g\}$ and a one-to-one mapping function θ, which maps each node in the main network $\mathbf{G}$ to a domain-specific network, a **Network of Networks (NoN)** is defined as the triplet $\mathcal{R} =< \mathbf{G}, \mathcal{A}, \theta >$. Nodes in the main network are referred to as *main nodes*, nodes in the domain-specific networks are called *domain nodes*. Each main node represents a domain-specific network through the mapping function θ. In addition, we represent the nodes in each domain-specific network as $\mathcal{V}_i (i = 1, ..., g)$. We define $\mathcal{I}_{i,j}$ as the *common nodes* between $\mathbf{A}_i$ and $\mathbf{A}_j$, i.e., $\mathcal{I}_{i,j} = \mathcal{V}_i \cap \mathcal{V}_j$.

Take Figure 1 as an example, the dashed network is the main network $\mathbf{G}$, which has five main nodes, including *NYC, Pittsburgh, Cleveland, Philadelphia, DC*. Each of these main nodes is mapped to a domain-specific network (the solid network). For example, the main node *Pittsburgh* is further represented as a domain-specific network with three nodes, including *Jim, Ben, Bob*.

Given an NoN $\mathcal{R} =< \mathbf{G}, \mathcal{A}, \theta >$, a fundamental task is to compare and rank the nodes in this network. For instance, we might be interested in ranking the importance/popularity of the individuals in each domain-specific network. Further more, given an individual, e.g., *Jim* from *Pittsburgh*, we may also want to find the similar users from *NYC* as well as other cities.

We introduce a query vector $\mathbf{e}_i$ $(i = 1, ..., g)$ for each domain-specific network, which is an $n_i \times 1$ nonnegative vector (n_i represents the number of nodes in the domain-specific network). If we are given a query node of interest from a domain-specific network (e.g., *Jim* from *Pittsburgh*), we set its corresponding entry $\mathbf{e}_i$ to 1, and all other entries to 0. Otherwise, all the entries in $\mathbf{e}_i$ are set to $\frac{1}{n_i}$. Formally, we define the CROSSRANK problem as follows.

PROBLEM 1. CROSSRANK

Given: *(1) an NoN* $\mathcal{R} =< \mathbf{G}, \mathcal{A}, \theta >$*, and (2) the query vectors* $\mathbf{e}_i$ $(i = 1, ..., g)$*;*

Find: *ranking vectors* $\mathbf{r}_i$ *for the nodes in the domain-specific networks* $\mathbf{A}_i$ $(i = 1, ..., g)$*.*

In the above definition, when all the query vectors are uniform, i.e., no query node is given, we essentially want to measure and rank the *global importance* of all the domain nodes within each domain-specific network. On the other hand, if a query node is given, we essentially want to measure the *proximity* (i.e., relevance) from the query node to all the domain nodes within each domain specific network. Note that in the latter case, sometimes, we might be interested in ranking domain nodes from a particular domain-specific network. For instance, given *Jim* from *Pittsburgh*, we might want to find out the top-3 most similar users from *NYC*. We call this special case of CROSSRANK as CROSSQUERY, which is formally defined as follows.

PROBLEM 2. CROSSQUERY

Given: *(1) an NoN* $\mathcal{R} =< \mathbf{G}, \mathcal{A}, \theta >$*, (2) a query node of interest from a* source *domain-specific network* $\mathbf{A}_s$*, (3) a* target *domain-specific network* $\mathbf{A}_d$*, and (4) an integer k;*

Find: *the top-k most relevant nodes from the* target *domain-specific network* $\mathbf{A}_d$ *w.r.t. the query node.*

In the next two sections, we give our solutions for CROSSRANK and CROSSQUERY, respectively.

3. CROSSRANK

In this section, we present our solution to CROSSRANK. We start with formulating it as a regularized optimization problem, then present an effective algorithm to solve it, followed by some theoretical analysis.

3.1 Optimization Formulation

The basic idea of our method is to formulate CrossRank as a regularized optimization problem by encoding the following three types of constrains. (1) *Within-network smoothness*: for a given domain-specific network $\mathbf{A}_i$, the ranking scores among the neighboring nodes should be somehow smooth. That is, for any two adjacent nodes x, y on $\mathbf{A}_i$, we want to minimize the difference of the rank scores between them, i.e., $(\mathbf{r}_i(x) - \mathbf{r}_i(y))^2$. (2) *Query preference*: the ranking vector $\mathbf{r}_i$ should also reflect the preference of the specific query node of interest from the corresponding domain-specific network. That is, we want to minimize the difference between the ranking vector $\mathbf{r}_i$ and the corresponding query vector $\mathbf{e}_i$. (3) *Cross-network consistency*: for a pair of domain-specific networks $\mathbf{A}_i$ and $\mathbf{A}_j$ that are connected with each other by the main network (i.e., $\mathbf{G}(i, j) > 0$), they may share some common nodes, i.e., $\mathcal{I}_{ij} = \mathcal{V}_i \cap \mathcal{V}_j$ is non-empty. In this case, those common nodes will receive multiple ranking scores (one from each ranking vector). Intuitively, the ranking scores for the same node should be consistent across highly similar domain-specific networks. In other words, we want to minimize the difference between $\mathbf{r}_i(\mathcal{I}_{ij})$ and $\mathbf{r}_j(\mathcal{I}_{ij})$. Here $\mathbf{r}_i(\mathcal{I}_{ij})$ can be viewed as the projection of the entire ranking vector $\mathbf{r}_i$ on the common node set $\mathcal{I}_{ij}$.

Putting everything together, we have the following regularized objective function $J(\mathbf{r}_1, ..., \mathbf{r}_g)$. The optimal ranking vectors $\mathbf{r}_i$ $(i = 1, ..., g)$ are those minimize the objective function.

$$\begin{aligned} \mathrm{J}(\mathbf{r}_1, ..., \mathbf{r}_g) \;=\; c \underbrace{\sum_{i=1}^{g} \mathbf{r}_i'(\mathbf{I}_{n_i} - \tilde{\mathbf{A}}_i)\mathbf{r}_i}_{\text{within-network smoothness}} + (1-c) \underbrace{\sum_{i=1}^{g} \|\mathbf{r}_i - \mathbf{e}_i\|_F^2}_{\text{query preference}} \\ + \; a \underbrace{\sum_{i=1}^{g}\sum_{j=1}^{g} \| \frac{\mathbf{r}_i(\mathcal{I}_{ij})}{\sqrt{d_m(i)}} - \frac{\mathbf{r}_j(\mathcal{I}_{ij})}{\sqrt{d_m(j)}} \|_F^2 \mathbf{G}(i,j)}_{\text{cross-network consistency}} \end{aligned} \quad (1)$$

where $\|\cdot\|_F$ denotes the Frobenius norm; $\tilde{\mathbf{A}}_i$ is the symmetrically normalized adjacency matrix of $\mathbf{A}_i$; and a and c are two regularization parameters. Note that the length of different ranking vectors could be different and all entries of a given ranking vector $\mathbf{r}_i$ add up to 1. To make the ranking scores comparable with each other for the common nodes, we normalize $\mathbf{r}_i(\mathcal{I}_{ij})$ and $\mathbf{r}_j(\mathcal{I}_{ij})$ by their corresponding degrees in the main networks, respectively.

Discussions. If we ignore the third term (cross-network consistency) in Eq. (1), we can show that the objective function $J(\mathbf{r}_1, ..., \mathbf{r}_g)$ can be decoupled into g independent terms. Each of these terms becomes the same objective function of a popular ranking method on homogeneous network - random walk with restart (also called manifold ranking, the personalized pagerank, etc.) [33, 26]. From this perspective, what we essentially aim to do in Eq. (1) is to solve multiple, inter-correlated such ranking problems simultaneously.

3.2 Optimization Solutions

We first present an equivalent formulation of Eq. (1). Denote the aggregated ranking vector $\mathbf{r} = (\mathbf{r}_1', ..., \mathbf{r}_g')'$, the aggregated query vector $\mathbf{e} = (\mathbf{e}_1', ..., \mathbf{e}_g')'$. Let the aggregated adjacency matrix $\tilde{\mathbf{A}} = \mathrm{diag}(\tilde{\mathbf{A}}_1, ..., \tilde{\mathbf{A}}_g)$ be a $g \times g$ diagonal block matrix, $\mathbf{O}_{ij}$ be an $n_i \times n_j$ binary indicator matrix with $\mathbf{O}_{ij}(x, y) = 1$ if the x^{th} node in $\mathbf{A}_i$ is the y^{th} node in $\mathbf{A}_j$, i.e., common node. Then $\mathbf{O}$ is a block matrix whose $(i, j)^{\text{th}}$ block is $\mathbf{G}(i, j)\mathbf{O}_{ij}$.

Since $\mathbf{O}$ may be singular due to zero-rows/columns, we define $\mathbf{Y} = \mathbf{O} + \mathbf{D}_T$, where $\mathbf{D}_T = \mathrm{diag}(d_m(1)\mathbf{I}_{n_1}, ..., d_m(g)\mathbf{I}_{n_g}) - \mathbf{D}_O$ with $\mathbf{I}_{n_i} (1 \le i \le g)$ being an $n_i \times n_i$ identity matrix and $\mathbf{D}_O$ being the degree matrix of $\mathbf{O}$. $\mathbf{Y}$ is non-singular. The $(i, i)^{th}$ block of $\mathbf{D}_Y$ (i.e., $\mathbf{Y}$'s degree matrix) equals $d_m(i)\mathbf{I}_{n_i}$. Finally, we define $\mathbf{X} = \mathbf{D}_Y^{-\frac{1}{2}}(\mathbf{D}_Y - \mathbf{Y})\mathbf{D}_Y^{-\frac{1}{2}} = \mathbf{I}_n - \tilde{\mathbf{Y}}$, where $\tilde{\mathbf{Y}} = \mathbf{D}_Y^{-\frac{1}{2}}\mathbf{Y}\mathbf{D}_Y^{-\frac{1}{2}}$.

With these definitions, the objective function in Eq. (1) can be re-written as the following matrix form

$$\mathrm{J}(\mathbf{r}) = c\mathbf{r}'(\mathbf{I}_n - \tilde{\mathbf{A}})\mathbf{r} + (1-c)\|\mathbf{r} - \mathbf{e}\|_F^2 + 2a\mathbf{r}'\mathbf{X}\mathbf{r} \quad (2)$$

where $n = \sum_{i=1}^{g} n_i$ is the total number of nodes in all the domain-specific networks.

LEMMA 1. *Equation* (1) *and Equation* (2) *are equivalent.*

PROOF. We only need to prove that the third term in Eq. (1) is equal to that in Eq. (2).

By the definition of $\mathbf{X}$ and $\mathbf{r}$, we have

$$\mathbf{r}'\mathbf{X}\mathbf{r} = \mathbf{r}'\mathbf{I}_n\mathbf{r} - \mathbf{r}'\tilde{\mathbf{Y}}\mathbf{r} = \sum_{i=1}^{g} \mathbf{r}_i'\mathbf{I}_{n_i}\mathbf{r}_i - \sum_{i=1}^{g}\sum_{j=1}^{g} \mathbf{r}_i'\tilde{\mathbf{Y}}_{ij}\mathbf{r}_j$$

where $\tilde{\mathbf{Y}}_{ij}$ is the $(i, j)^{\text{th}}$ block of $\tilde{\mathbf{Y}}$ of size $n_i \times n_j$. Then

$$\begin{aligned} \mathbf{r}'\mathbf{X}\mathbf{r} = \frac{1}{2}\Bigg(& \sum_{i=1}^{g} \frac{\mathbf{r}_i'}{\sqrt{d_m(i)}}(\mathbf{D}_Y)_{ii}\frac{\mathbf{r}_i}{\sqrt{d_m(i)}} \\ & - 2\sum_{i=1}^{g}\sum_{j=1}^{g} \frac{\mathbf{r}_i'}{\sqrt{d_m(i)}}\mathbf{Y}_{ij}\frac{\mathbf{r}_j}{\sqrt{d_m(j)}} + \sum_{j=1}^{g} \frac{\mathbf{r}_j'}{\sqrt{d_m(j)}}(\mathbf{D}_Y)_{jj}\frac{\mathbf{r}_j}{\sqrt{d_m(j)}}\Bigg) \end{aligned}$$

where $(\mathbf{D}_Y)_{ii}$ is the i^{th} diagonal block of $\mathbf{D}_Y$, and $\mathbf{Y}_{ij}$ is the $(i, j)^{th}$ block of $\mathbf{Y}$. Thus

$$\begin{aligned} \mathbf{r}'\mathbf{X}\mathbf{r} = \frac{1}{2}\Bigg(& \sum_{i=1}^{g} \Big(\frac{\mathbf{r}_i'}{\sqrt{d_m(i)}}(\mathbf{D}_O)_{ii}\frac{\mathbf{r}_i}{\sqrt{d_m(i)}} + \frac{\mathbf{r}_i'}{\sqrt{d_m(i)}}(\mathbf{D}_T)_{ii}\frac{\mathbf{r}_i}{\sqrt{d_m(i)}}\Big) \\ & - 2\sum_{i=1}^{g}\sum_{j=1}^{g} \Big(\frac{\mathbf{r}_i'}{\sqrt{d_m(i)}}\mathbf{G}(i,j)\mathbf{O}_{ij}\frac{\mathbf{r}_j}{\sqrt{d_m(j)}} + \frac{\mathbf{r}_i'}{\sqrt{d_m(i)}}(\mathbf{D}_T)_{ij}\frac{\mathbf{r}_j}{\sqrt{d_m(j)}}\Big) \\ & + \sum_{j=1}^{g} \Big(\frac{\mathbf{r}_j'}{\sqrt{d_m(j)}}(\mathbf{D}_\mathbf{O})_{jj}\frac{\mathbf{r}_j}{\sqrt{d_m(j)}} + \frac{\mathbf{r}_j'}{\sqrt{d_m(j)}}(\mathbf{D}_T)_{jj}\frac{\mathbf{r}_j}{\sqrt{d_m(j)}}\Big)\Bigg) \end{aligned}$$

where $(\mathbf{D}_T)_{ij}$, $(\mathbf{D}_O)_{ij}$ and $\mathbf{O}_{ij}$ are block matrices representing $(i, j)^{th}$ block of $\mathbf{D}_T$, $\mathbf{D}_O$ and $\mathbf{O}$, respectively. Since $\mathbf{D}_T$ is a diagonal block matrix, we have

$$2\sum_{i=1}^{g}\sum_{j=1}^{g} \frac{\mathbf{r}_i'}{\sqrt{d_m(i)}}(\mathbf{D}_T)_{ij}\frac{\mathbf{r}_j}{\sqrt{d_m(j)}} = 2\sum_{i=1}^{g} \frac{\mathbf{r}_i'}{\sqrt{d_m(i)}}(\mathbf{D}_T)_{ii}\frac{\mathbf{r}_i}{\sqrt{d_m(i)}}$$

and

$$\begin{aligned} \sum_{i=1}^{g} \frac{\mathbf{r}_i'}{\sqrt{d_m(i)}}(\mathbf{D}_T)_{ii}\frac{\mathbf{r}_i}{\sqrt{d_m(i)}} - 2\sum_{i=1}^{g}\sum_{j=1}^{g} \frac{\mathbf{r}_i'}{\sqrt{d_m(i)}}(\mathbf{D}_T)_{ij}\frac{\mathbf{r}_j}{\sqrt{d_m(j)}} \\ + \sum_{j=1}^{g} \frac{\mathbf{r}_j'}{\sqrt{d_m(j)}}(\mathbf{D}_T)_{jj}\frac{\mathbf{r}_j}{\sqrt{d_m(j)}} = 0 \end{aligned}$$

This means $\mathbf{D}_T$ does not affect the ranking results. Therefore, we have that

$$\begin{aligned} \mathbf{r}'\mathbf{X}\mathbf{r} &= \frac{1}{2}\sum_{i=1}^{g}\sum_{j=1}^{g} \mathbf{G}(i,j)\Bigg(\frac{\mathbf{r}_i'(\mathcal{I}_{ij})}{\sqrt{d_m(i)}}\frac{\mathbf{r}_i(\mathcal{I}_{ij})}{\sqrt{d_m(i)}} - 2\frac{\mathbf{r}_i'(\mathcal{I}_{ij})}{\sqrt{d_m(i)}}\frac{\mathbf{r}_j(\mathcal{I}_{ij})}{\sqrt{d_m(j)}} \\ &\quad + \frac{\mathbf{r}_j'(\mathcal{I}_{ij})}{\sqrt{d_m(j)}}\frac{\mathbf{r}_j(\mathcal{I}_{ij})}{\sqrt{d_m(j)}}\Bigg) \\ &= \frac{1}{2}\sum_{i=1}^{g}\sum_{j=1}^{g} \mathbf{G}(i,j)\| \frac{\mathbf{r}_i(\mathcal{I}_{ij})}{\sqrt{d_m(i)}} - \frac{\mathbf{r}_j(\mathcal{I}_{ij})}{\sqrt{d_m(j)}} \|_F^2 \end{aligned}$$

This completes the proof. □

Algorithm 1: CROSSRANK

Input: (1) a network of networks $\mathcal{R} =< \mathbf{G}, \mathcal{A}, \theta >$; (2) the query vectors $\mathbf{e}_i$, $(i = 1, ..., g)$; and (3) the parameters a and c

Output: the ranking vectors $\mathbf{r}_i$, $(i = 1, ..., g)$

Offline-computation: Construct $\tilde{\mathbf{A}}$ and $\tilde{\mathbf{Y}}$ from $\mathcal{R}$;
Online-ranking:
Construct the aggregated query vector $\mathbf{e} = (\mathbf{e}_1', ..., \mathbf{e}_g')'$;
Initialize the aggregated ranking vector $\mathbf{r} = \mathbf{e}$;
while *not convergence* **do**
  Update: $\mathbf{r} \leftarrow (\frac{c}{1+2a}\tilde{\mathbf{A}} + \frac{2a}{1+2a}\tilde{\mathbf{Y}})\mathbf{r} + \frac{1-c}{1+2a}\mathbf{e}$;
end
return the ranking vectors $\mathbf{r}_1, ..., \mathbf{r}_g$ based on $\mathbf{r}$

From Eq. (2), it can be seen that the objective function J is a quadratic function of the ranking vector $\mathbf{r}$. We have

$$\frac{\partial \mathrm{J}}{\partial \mathbf{r}} = 2((1+2a)\mathbf{I}_n - c\tilde{\mathbf{A}} - 2a\tilde{\mathbf{Y}})\mathbf{r} - 2(1-c)\mathbf{e}$$

If we set $\mathbf{r} = \mathbf{r} - \eta\frac{\partial \mathrm{J}}{\partial \mathbf{r}}$, where $\eta = \frac{1}{2+4a}$, we have

$$\mathbf{r} = (\frac{c}{1+2a}\tilde{\mathbf{A}} + \frac{2a}{1+2a}\tilde{\mathbf{Y}})\mathbf{r} + \frac{1-c}{1+2a}\mathbf{e} \tag{3}$$

Based on Eq. (3), we have a fixed-point approach to compute the optimal ranking vector $\mathbf{r}$ that minimizes the objective function $\mathrm{J}(\mathbf{r})$ as illustrated in Algorithm 1.

3.3 Proofs and Analysis

In this subsection, we analyze Algorithm 1 in terms of its *convergence*, *optimality*, *complexity* and discuss its *relationship* with existing ranking algorithms.

We start with the the convergence of Algorithm 1, which is summarized in Theorem 1. It says that the proposed CROSSRANK algorithm converges to its closed-form solution.

THEOREM 1. ***Convergence of* CROSSRANK.** *Algorithm 1 converges to the closed-form solution:* $\mathbf{r} = (\mathbf{I}_n - \frac{c}{1+2a}\tilde{\mathbf{A}} - \frac{2a}{1+2a}\tilde{\mathbf{Y}})^{-1}\frac{1-c}{1+2a}\mathbf{e}$.

PROOF. First, the closed-form solution can be obtained by solving $\frac{\partial J}{\partial \mathbf{r}} = 0$. Then let $\mathbf{M} = \frac{c}{1+2a}\tilde{\mathbf{A}} + \frac{2a}{1+2a}\tilde{\mathbf{Y}}$. Eq. (3) becomes $\mathbf{r} = \mathbf{M}\mathbf{r} + \frac{1-c}{1+2a}\mathbf{e}$. Next, we show that the eigenvalues of $\mathbf{M}$ are in $(-1, 1)$.

Since $\tilde{\mathbf{A}}$ and $\tilde{\mathbf{Y}}$ are similar to the stochastic matrices $\mathbf{A}\mathbf{D}_A^{-1} = \mathbf{D}_A^{\frac{1}{2}}\tilde{\mathbf{A}}\mathbf{D}_A^{-\frac{1}{2}}$ and $\mathbf{Y}\mathbf{D}_Y^{-1} = \mathbf{D}_Y^{\frac{1}{2}}\tilde{\mathbf{Y}}\mathbf{D}_Y^{-\frac{1}{2}}$, respectively, both $\tilde{\mathbf{A}}$ and $\tilde{\mathbf{Y}}$ have eigenvalues within $[-1, 1]$.

One result of the Weyl's inequality theorem [1] states that for matrices $\hat{\mathbf{H}}, \mathbf{H}, \mathbf{P} \in \mathcal{H}_n$, where $\mathcal{H}_n$ is the set of $n \times n$ Hermitian matrices, if $\hat{\mathbf{H}} = \mathbf{H} + \mathbf{P}$ and their eigenvalues are arranged in non-increasing orders, i.e., $\lambda_1(\hat{\mathbf{H}}) \geq ... \geq \lambda_n(\hat{\mathbf{H}})$, $\lambda_1(\mathbf{H}) \geq ... \geq \lambda_n(\mathbf{H})$, $\lambda_1(\mathbf{P}) \geq ... \geq \lambda_n(\mathbf{P})$, then the following inequalities hold:

$$\lambda_n(\mathbf{P}) \leq \lambda_i(\hat{\mathbf{H}}) - \lambda_i(\mathbf{H}) \leq \lambda_1(\mathbf{P}), \forall i = 1, ..., n$$

Since $\tilde{\mathbf{A}}, \tilde{\mathbf{Y}}, \mathbf{M} \in \mathcal{H}_n$ and $\mathbf{M} = \frac{c}{1+2a}\tilde{\mathbf{A}} + \frac{2a}{1+2a}\tilde{\mathbf{Y}}$, we have

$$\lambda_1(\mathbf{M}) \leq \frac{c}{1+2a}\lambda_1(\tilde{\mathbf{A}}) + \frac{2a}{1+2a}\lambda_1(\tilde{\mathbf{Y}})$$
$$\lambda_n(\mathbf{M}) \geq \frac{c}{1+2a}\lambda_n(\tilde{\mathbf{A}}) + \frac{2a}{1+2a}\lambda_n(\tilde{\mathbf{Y}})$$

which means the eigenvalues of $\mathbf{M}$ are in the range of $[-\frac{c+2a}{1+2a}, \frac{c+2a}{1+2a}]$. Since $0 < c < 1$, the eigenvalues of $\mathbf{M}$ are in $(-1, 1)$.

Based on this property, we can show the convergence of the fixed-point approach. Without loss of generality, let $\mathbf{r}^{(0)} = \mathbf{e}$, and t be the iteration index ($t \geq 1$). According to Eq. (3), we have

$$\mathbf{r}^{(t)} = \mathbf{M}^t\mathbf{e} + \sum_{i=0}^{t-1}\mathbf{M}^i\frac{1-c}{1+2a}\mathbf{e}$$

Since the eigenvalues of $\mathbf{M}$ are all in $(-1, 1)$, we have

$$\lim_{t\to\infty}\mathbf{M}^t = 0, \text{and } \lim_{t\to\infty}\sum_{i=0}^{t-1}\mathbf{M}^i = (\mathbf{I}_n - \mathbf{M})^{-1}$$

Therefore

$$\mathbf{r} = \lim_{t\to\infty}\mathbf{r}^{(t)} = (\mathbf{I}_n - \mathbf{M})^{-1}\frac{1-c}{1+2a}\mathbf{e}$$
$$= (\mathbf{I}_n - \frac{c}{1+2a}\tilde{\mathbf{A}} - \frac{2a}{1+2a}\tilde{\mathbf{Y}})^{-1}\frac{1-c}{1+2a}\mathbf{e}$$

which is the closed-form solution. □

Next, we show in Lemma 2 that Algorithm 1 finds the *global optimal* solution of the objective function $\mathrm{J}(\mathbf{r})$ defined in Eq. (2).

LEMMA 2. ***Optimality of* CROSSRANK.** *At convergence, Algorithm 1 finds the global minimal solution of* $J(\mathbf{r})$ *defined in Eq.* (2)

PROOF. This can be proved by showing that the function in Eq. (2) is convex. The Hessian matrix of Eq. (2) is $\nabla^2\mathrm{J} = 2((1+2a)\mathbf{I}_n - c\tilde{\mathbf{A}} - 2a\tilde{\mathbf{Y}})$. Following the similar idea as in the proof of Theorem 1, we have that the eigenvalues of $\nabla^2\mathrm{J}$ are no less than $2(1-c)$. Since $0 < c < 1$, $\nabla^2\mathrm{J}$ is positive-definite. Therefore, Eq. (2) is convex. □

The complexity of Algorithm 1 is summarized in Lemma 3, which says that it is *linear* w.r.t. the overall number of nodes and edges in NoN in terms of both space and time cost.

LEMMA 3. ***Complexity of* CROSSRANK.** *The time complexity of Algorithm 1 is* $O(T^*(m + ng))$ *and its space complexity is* $O(m + n)$*, where* $m = \sum_{i=1}^{g} m_i$*,* $n = \sum_{i=1}^{g} n_i$*, and* T^* *is the total iteration number.*

PROOF. In the preprocessing stage, we need to construct $\tilde{\mathbf{A}}$ and $\tilde{\mathbf{Y}}$. The construction time of $\tilde{\mathbf{A}}$ is $O(m)$, since there are $O(m)$ nonzero entries in $\tilde{\mathbf{A}}$. The construction time of $\tilde{\mathbf{Y}}$ is $O(gn)$, since the nonzero entries of $\tilde{\mathbf{Y}}$ are pairwise intersection of node lists of all the domain-specific networks, i.e., $\sum_{i=1}^{g} gn_i = gn$. Therefore, preprocessing cost is $O(m + gn)$. The online-computation cost depends on the fix-point updating step in Eq. (3). Since there are $O(m+gn)$ nonzero entries in $\frac{c}{1+2a}\tilde{\mathbf{A}} + \frac{2a}{1+2a}\tilde{\mathbf{Y}}$, the online-computation cost is $O(T^*(m+gn))$. In Algorithm 1, we need to store $\tilde{\mathbf{A}}$ and $\tilde{\mathbf{Y}}$, with space complexity $O(m+gn)$. Usually g is much smaller than n, we can regard time and space complexities as $O(T^*(m+n))$ and $O(m+n)$ respectively. □

Finally, we show the relationship between the proposed CROSSRANK algorithm and random walk with restart (RWR).

LEMMA 4. ***Relationship between* CROSSRANK *and RWR.*** CROSSRANK *fixed-point approach is equivalent to random walk with restart with the transition matrix* $\tilde{\mathbf{W}} = \frac{c}{c+2a}\tilde{\mathbf{A}} + \frac{2a}{c+2a}\mathbf{Y}\mathbf{D}_Y^{-1}$ *if* $\tilde{\mathbf{A}}$ *is stochastic.*

PROOF. Let $\tilde{c} = \frac{c+2a}{1+2a}$ and $\mathbf{W} = \frac{c}{c+2a}\tilde{\mathbf{A}} + \frac{2a}{c+2a}\tilde{\mathbf{Y}}$, we can transform Eq. (3) to

$$\mathbf{r} = \tilde{c}\mathbf{W}\mathbf{r} + (1-\tilde{c})\mathbf{e} \tag{4}$$

and the closed-form solution of Eq. (2) to

$$\mathbf{r} = (\mathbf{I}_n - \tilde{c}\mathbf{W})^{-1}(1-\tilde{c})\mathbf{e} \tag{5}$$

These formulations have a RWR form except that $\mathbf{W}$ is not stochastic, since $\tilde{\mathbf{A}}$ and $\tilde{\mathbf{Y}}$ are symmetrically normalized. If we denote $\tilde{\mathbf{W}} = \frac{c}{c+2a}\tilde{\mathbf{A}} + \frac{2a}{c+2a}\mathbf{Y}\mathbf{D}_Y^{-1}$, we can further transform the closed-form solution to

$$\begin{aligned}\mathbf{r} &= (\mathbf{I}_n - \frac{c}{1+2a}\tilde{\mathbf{A}} - \frac{2a}{1+2a}\tilde{\mathbf{Y}})^{-1}\frac{1-c}{1+2a}\mathbf{e} \\ &= \mathbf{D}_Y^{-\frac{1}{2}}(\mathbf{I}_n - \frac{c}{1+2a}\tilde{\mathbf{A}} - \frac{2a}{1+2a}\mathbf{Y}\mathbf{D}_Y^{-1})^{-1}\mathbf{D}_Y^{\frac{1}{2}}\frac{1-c}{1+2a}\mathbf{e} \\ &= \mathbf{D}_Y^{-\frac{1}{2}}(\mathbf{I}_n - \tilde{c}\tilde{\mathbf{W}})^{-1}\mathbf{D}_Y^{\frac{1}{2}}(1-\tilde{c})\mathbf{e}\end{aligned} \quad (6)$$

In the above equation, $\mathbf{D}_Y^{\frac{1}{2}}\tilde{\mathbf{A}}\mathbf{D}_Y^{-\frac{1}{2}} = \tilde{\mathbf{A}}$ since $\mathbf{D}_Y^{\frac{1}{2}}\tilde{\mathbf{A}}\mathbf{D}_Y^{-\frac{1}{2}}$ is a diagonal block matrix, of which each diagonal block can be represented as $(\mathbf{D}_Y)_{ii}^{\frac{1}{2}}(\tilde{\mathbf{A}})_{ii}(\mathbf{D}_Y)_{ii}^{-\frac{1}{2}}$ where $(\mathbf{D}_Y)_{ii}$ and $\tilde{\mathbf{A}}_{ii}$ are the i^{th} diagonal block of $\mathbf{D}_Y$ and $\tilde{\mathbf{A}}$, respectively. Since the diagonal values of $(\mathbf{D}_Y)_{ii}$ equal to $d_m(i)$, $(\mathbf{D}_Y)_{ii}^{\frac{1}{2}}(\tilde{\mathbf{A}})_{ii}(\mathbf{D}_Y)_{ii}^{-\frac{1}{2}} = (\tilde{\mathbf{A}})_{ii}$ and $\mathbf{D}_Y^{\frac{1}{2}}\tilde{\mathbf{A}}\mathbf{D}_Y^{-\frac{1}{2}} = \tilde{\mathbf{A}}$.

Note that we can view $\tilde{\mathbf{W}}$ as the normalized matrix of a network generated by linking the common nodes between any two domain-specific networks if they are connected in the main network. Eq. (6) shows that if we normalize the initial vector $\mathbf{e}$ of CrossRank as $\tilde{\mathbf{e}} = \mathbf{D}_Y^{\frac{1}{2}}\mathbf{e}$, the process $\tilde{\mathbf{r}} = \tilde{c}\tilde{\mathbf{W}}\tilde{\mathbf{r}} + (1-\tilde{c})\tilde{\mathbf{e}}$ spreads information symmetrically within domain-specific networks and stochastically across domain-specific networks (note this process converges since $\tilde{\mathbf{W}}$ and $\mathbf{W}$ are similar). It gives the same ranking results as Eq. (4), since multiplying $\mathbf{D}_Y^{-\frac{1}{2}}$ on the ranking scores $\tilde{\mathbf{r}}$ does not affect the relative ranking order within the domain-specific networks. If $\tilde{\mathbf{A}}$ is stochastic, $\tilde{\mathbf{W}}$ is stochastic and $\tilde{\mathbf{r}} = \tilde{c}\tilde{\mathbf{W}}\tilde{\mathbf{r}} + (1-\tilde{c})\tilde{\mathbf{e}}$ becomes a stochastic process. This completes the proof. □

4. CrossQuery

In this section, we address CrossQuery. Note that CrossQuery is a special case of CrossRank, by (1) requiring the query node being from a source domain-specific network $\mathbf{A}_s$; (2) restricting the query results within a target domain-specific network $\mathbf{A}_d$; and (3) searching for a set of k most relevant nodes from $\mathbf{A}_d$ (as opposed to demanding the exacting ranking vectors). Thus, a default solution for CrossQuery is to run the CrossRank algorithm to get the ranking vector $\mathbf{r}_d$ for the target domain-specific network and return the top-k nodes with the highest ranking scores. In this section, we aim to further speed-up its computation.

4.1 CrossQuery-Basic

According to Lemma 4, CrossRank has a RWR form on the integrated normalized matrix $\mathbf{W}$. This not only reveals an interesting relationship between CrossRank and RWR, but also opens the door to take the advantage of the vast machinery of the existing scalable algorithms for RWR [25, 5, 6]. For example, we could modify the algorithm in [6] to solve CrossQuery since $\lim_{t\to\infty}(\tilde{c}\mathbf{W})^t = 0$ (a prerequisite of this algorithm), which is summarized in Algorithm 2. This is an exact method that improves the basic power iteration method by estimating the node ranking scores using only a small number of iterations.

Time Complexity. Following the notations in Lemma 3, the overall time complexity of CrossQuery-Basic is $O(T(bf + h) + m + gn)$, where b is the average size of the neighborhood for each node, f and h are the average size of the subtree $\mathcal{T}^{(t)}$ and the selected nodes set $\mathcal{S}^{(t)}$ over all iterations respectively, T is the total number of iterations. Compared with the CrossRank method, whose time complexity is $O(T^*(m + gn))$, where T^* is the total number of iterations, Algorithm 2 is more efficient since in practice $bf << m + gn$, $h << n$ and $T << T^*$.

Algorithm 2: CrossQuery-Basic (adopted from [6])

Input: (1) matrices $\mathbf{W}, \mathbf{W}_{max}$ ($\mathbf{W}_{max}(u) = \max_{v\in\mathcal{N}(u)}\mathbf{W}(u,v)$ where $\mathcal{N}(u)$ is the neighbourhood of u); (2) node sets $\mathcal{V}_1, ..., \mathcal{V}_g$; (3) nodes q, s, d and (4) parameters $\tilde{c}, k$
Output: Top k relevant nodes set $\mathcal{K}$

Set $\mathbf{e}_s(q) = 1, \mathbf{e}_i = \mathbf{0}, \forall i \in (1, ..., g)\backslash s$, and the query vector $\mathbf{e} = (\mathbf{e}_1', ..., \mathbf{e}_g')'$;
Initialize the random walk vector $\mathbf{p} = \mathbf{e}$, the lower bound vector $\underline{\mathbf{r}}_d^{(0)} = \mathbf{0}$, and the upper bound vector $\bar{\mathbf{r}}_d^{(0)} = \mathbf{0}$;
Initialize the iteration number $t = 0$, Subtree $\mathcal{T}^{(0)} = \{q\}$, the selected nodes set $\mathcal{S}^{(0)} = \mathcal{V}_d$, the threshold $\sigma = -1$;
while $|\mathcal{S}^{(t)}| > k$ **do**
  Increase the iteration number $t = t + 1$;
  $layer^{(t)} = \{u | u.layer = t$ by BFS rooted at $q, \forall u \in \bigcup_{i=1}^{g}\mathcal{V}_i\}$;
  expand $\mathcal{T}^{(t)} = \mathcal{T}^{(t-1)} \cup layer^{(t)}$;
  update $\mathbf{p}(\mathcal{T}^{(t)}) \leftarrow \mathbf{W}\mathbf{p}(\mathcal{T}^{(t)})$;
  update lower bound: $\underline{\mathbf{r}}_d^{(t)} \leftarrow \underline{\mathbf{r}}_d^{(t-1)} + (1-\tilde{c})\tilde{c}^t\mathbf{p}(\mathcal{S}^{(t-1)})$;
  update upper bound: $\bar{\mathbf{r}}_d^{(t)} \leftarrow \underline{\mathbf{r}}_d^{(t)} + \tilde{c}^{t+1}\mathbf{W}_{max}(\mathcal{S}^{(t-1)})$;
  update threshold: $\sigma \leftarrow$ the k^{th} largest value in $\underline{\mathbf{r}}_d^{(t)}$;
  shrink selected nodes: $\mathcal{S}^{(t)} = \{u | \bar{\mathbf{r}}_d^{(t)}(u) \geq \sigma, \forall u \in \mathcal{S}^{(t-1)}\}$;
  shrink bounds lists: $\underline{\mathbf{r}}_d^{(t)} \leftarrow \underline{\mathbf{r}}_d^{(t)}(\mathcal{S}^{(t)})$, $\bar{\mathbf{r}}_d^{(t)} \leftarrow \bar{\mathbf{r}}_d^{(t)}(\mathcal{S}^{(t)})$;
end
return $\mathcal{K} = \mathcal{S}^{(t)}$

4.2 CrossQuery-Fast

Next, we propose a more efficient algorithm for CrossQuery. The basic idea is as follows. Let the corresponding main nodes for $\mathbf{A}_s$ and $\mathbf{A}_d$ be node s and node d, respectively. Given s and d, parts of the main network $\mathbf{G}$ might have little contributions to measure the relevance for the nodes from $\mathbf{A}_d$ w.r.t. the query node, and thus can be pruned with little impact on the final query accuracy. In the context of NoN, even if we can only prune a small portion of the main network, the computational efficiency can be significantly improved since all the corresponding domain-specific networks can be filtered out during the ranking process.

Let $\mathbf{r}(v)$ be the ranking score of any node v w.r.t. a query node q by RWR. It is well known that $\mathbf{r}(v)$ can be represented by the so-called *inverse P-distance* [9].

$$\mathbf{r}(v) = \sum_{p\in\{q\rightsquigarrow v\}} Prob(p)(1-c)c^{l(p)}$$

where $p \in \{q \rightsquigarrow v\}$ is a path from q to v, and $l(p)$ is the unweighed length of p. For any path $p : v_1 \to v_2 \to ... \to v_{l(p)}$, $Prob(p) = \prod_{i=1}^{l(p)-1} Prob(v_{i+1}|v_i)$, where $Prob(v_{i+1}|v_i)$ is the transition probability from v_i to v_{i+1}. We call $Prob(p)$ the transition probability of path p.

Inspired by the relationship between CrossRank and RWR (as indicated by Eq. (6)), here we consider $\tilde{\mathbf{A}}$ to be stochastic (i.e., column normalized) to guide the pruning process. Then we can express the CrossQuery score of a node v w.r.t. the query node q as

$$\mathbf{r}(v) = \sum_{p\in\{q\rightsquigarrow v\}} \frac{\sqrt{d_m(s)}}{\sqrt{d_m(d)}} Prob(p)(1-\tilde{c})\tilde{c}^{l(p)} \quad (7)$$

The following lemma says that $Prob(p), \forall p \in \{q \rightsquigarrow v\}$, is upper bounded by the transition probability of some path in the main network.

Lemma 5. **Upper Bound of** *Prob(p). For any path* $p \in \{q \rightsquigarrow v\}$, $\exists p' \in \{s \rightsquigarrow d\}$ *s.t.* $Prob(p) \leq Prob(p')$, *where* p' *is a path con-*

necting s and d in the main network. Here $Prob(p') = \prod_{e_{ij} \in p'} \frac{\mathbf{G}(i,j)}{d_m(i)}$ where e_{ij} is the edge connecting main nodes i and j in $\mathbf{G}$.

PROOF. For any edge $e_{xy} \in p$, let i and j be the main nodes that contain domain nodes x and y respectively. We have

$$\begin{aligned} Prob(p) &= \prod_{\substack{e_{xy} \in p \\ i=j}} \frac{c}{c+2a} \tilde{\mathbf{A}}_i(y,x) \prod_{\substack{e_{xy} \in p \\ i \neq j}} \frac{2a}{c+2a} (\mathbf{Y}\mathbf{D}_Y^{-1})_{ji}(y,x) \\ &\leq \prod_{\substack{e_{xy} \in p \\ i \neq j}} (\mathbf{Y}\mathbf{D}_Y^{-1})_{ji}(y,x) \\ &= \prod_{\substack{e_{ij} \in p' \\ i \neq j}} \frac{\mathbf{G}(i,j)}{d_m(i)} = Prob(p') \end{aligned}$$

where $(\mathbf{Y}\mathbf{D}_Y^{-1})_{ji}$ is the $(j,i)^{th}$ block of $\mathbf{Y}\mathbf{D}_Y^{-1}$, which is the transition matrix from domain-specific network i to j. □

In the above lemma, p' consists of edges in p that go across domain-specific networks. Let φ be the function that maps each p to its corresponding p', i.e., $\varphi(p) = p'$, then we have the following upper bound of the ranking score $\mathbf{r}(v)$.

COROLLARY 1. **Upper Bound of $\mathbf{r}(v)$.** *The* CROSSQUERY *score of a node $v \in \mathcal{V}_d$ w.r.t. the query node q, i.e., $\mathbf{r}(v)$, is upper bounded by*

$$\mathbf{r}(v) \leq \sum_{p \in \{q \rightsquigarrow v\}} \frac{\sqrt{d_m(s)}}{\sqrt{d_m(d)}} Prob(\varphi(p))(1-\tilde{c})\tilde{c}^{l(\varphi(p))}$$

PROOF. This can be derived directly from Eq. (7) and Lemma 5. The detailed proof is omitted. □

From Corollary 1, we have that if $Prob(\varphi(p))$ is small, the contribution from the path p to $\mathbf{r}(v)$ will be small. Based on this observation, we can extract a subnetwork of the main network consisting of paths with large $Prob(\varphi(p))$.

For any p, let $\tilde{Prob}(\varphi(p)) = \frac{\sqrt{d_m(s)}}{\sqrt{d_m(d)}} Prob(\varphi(p)) = \prod_{e_{ij} \in \varphi(p)} \frac{\mathbf{G}(i,j)}{\sqrt{d_m(i)}\sqrt{d_m(j)}}$. We define the distance between node i and node j as

$$L_{ij} = -log(\frac{\mathbf{G}(i,j)}{\sqrt{d_m(i)}\sqrt{d_m(j)}}) \tag{8}$$

Then the upper bound of the ranking score $\mathbf{r}(v)$ in Corollary 1 becomes

$$\mathbf{r}(v) \leq \sum_{p \in \{q \rightsquigarrow v\}} 10^{-\sum_{e_{ij} \in \varphi(p)} L_{ij}} (1-\tilde{c})\tilde{c}^{l(\varphi(p))} \tag{9}$$

This transformation allows us to search for short paths connecting s and d in the main network that capture high transition probabilities. We adopt a fast heuristics proposed in [12] to extract these paths. With the subgraph consisting of the extracted paths in the main network, we then apply CROSSQUERY-BASIC on the corresponding NoN to retrieve the top-k most relevant nodes from the target domain-specific network.

Algorithm 3 summarizes this approach. The algorithm starts the shortest path search rooted from s and d simultaneously in $\mathbf{G}$ (e.g., using Dijkstra's algorithm). Let $\mathcal{N}(s)$ and $\mathcal{N}(t)$ be the neighborhoods of s and d expanded by the algorithm so far. The first overlapping node in $\mathcal{N}(s)$ and $\mathcal{N}(t)$ identifies the shortest path between s and d. Denote this path as p'_{max} as it has the maximal transition probability from s to d (Steps 4 to 6). The algorithm continues expanding and adding new nodes in $\mathcal{N}(s)$ and $\mathcal{N}(t)$. The overlapping nodes in $\mathcal{N}(s)$ and $\mathcal{N}(t)$ will generate new paths. The expansion stops when the transition probability of the newly discovered path

Algorithm 3: CROSSQUERY-FAST

Input: (1) matrices $\mathbf{G}, \tilde{\mathbf{A}}, \mathbf{Y}$; (2) node sets $\mathcal{V}_1, ..., \mathcal{V}_g$; (3) nodes s, d, q and (4) parameters: ϵ, a, c, k
Output: Top k relevant nodes set $\mathcal{K}$

1 Initialize $L(p'_{max}) = \infty$, neighbourhoods $\mathcal{N}(s) = \phi, \mathcal{N}(d) = \phi$, radiuses $r_s = 0, r_d = 0$, selected nodes set $\mathcal{G} = \phi$;
2 Transform similarity in $\mathbf{G}$ into distances by Eq. (8);
3 **while** $r_s \leq \frac{L(p'_{max})-log(\epsilon)}{2} \vee r_d \leq \frac{L(p'_{max})-log(\epsilon)}{2}$ **do**
4 **if** $\mathcal{N}(s)$ *and* $\mathcal{N}(d)$ *first overlap at node* u **then**
5 $L(p'_{max}) = L(s \rightsquigarrow u \rightsquigarrow d)$;
6 **end**
7 **if** $r_s \leq \frac{L(p'_{max})-log(\epsilon)}{2}$ **then**
8 Expand $\mathcal{N}(s)$ and update r_s;
9 **end**
10 **if** $r_d \leq \frac{L(p'_{max})-log(\epsilon)}{2}$ **then**
11 Expand $\mathcal{N}(d)$ and update r_d;
12 **end**
13 **end**
14 Further prune $\mathcal{N}(s)$ and $\mathcal{N}(d)$: $\mathcal{G} = \{u | L(s,u) + L(u,d) \leq L(p'_{max}) - log(\epsilon), u \in \mathcal{N}(s) \cup \mathcal{N}(d)\}$;
15 Calculate $\mathbf{W}$, $\mathbf{W}_{max}$, $\tilde{c}$ based on $\mathcal{G}$, $\tilde{\mathbf{A}}$, $\mathbf{Y}$, a and c;
16 Apply CROSSQUERY-BASIC on the extracted NoN;
17 **return** $\mathcal{K}$

p'_{new} drops below $\epsilon \times \tilde{Prob}(p'_{max})$, where $0 < \epsilon < 1$ is an error factor. Thus we have $L(p'_{new}) > L(p'_{max}) - log(\epsilon)$, where $L(\cdot)$ is the weighed length of a path. Therefore, the neighborhoods with radius $\frac{L(p'_{max})-log(\epsilon)}{2}$ from s and d (i.e., r_s and r_d) will contain all significant paths based on ϵ (Steps 7 to 12). A node u in the neighborhoods can be pruned if $L(s,u) + L(u,d) > L(p'_{max}) - log(\epsilon)$, where $L(s,u)$ and $L(u,d)$ are the shortest distances between s and u, u and d, respectively (Step 14). Finally we apply CROSSQUERY-BASIC on the pruned NoN (Step 15 to 16).

Time Complexity. Let the number of nodes and edges in the main network be g and z respectively, and T be the number of iterations. Subgraph extraction takes $O(T(glog(g) + z))$ using Dijkstra's algorithm. The pruning step takes $O(g)$. Thus the overall complexity of subgraph extraction is $O(T(glog(g) + z))$. Note that since g and z are typically very small compared to the overall size of the domain-specific networks, this step is usually very efficient. Experimental results show that CROSSQUERY-FAST brings 4.5× to 7.5× additional speedup while preserving a high accuracy.

5. EXPERIMENTS

In this section, we perform comprehensive experiments to evaluate the performance of the proposed methods. We study NoN in two real-world applications: ranking authors in the network of co-author networks constructed from the DBLP bibliographic data, and prioritizing candidate gene in the network of tissue-specific protein interaction networks. Synthetic datasets are also generated to evaluate the efficiency of the developed optimization strategies. All experiments are performed on a 3.00 GHz desktop PC with 8G memory.

5.1 Co-Author NoN

We construct the co-author NoN from the DBLP data [24] as follows. The main network consists of 5 research areas, i.e., data mining (DM), machine learning (ML), database (DB), information retrieval (IR), and bioinformatics (BIO). Each area corresponds to a node in the main network. The conferences included in each area are summarized in Table 2. The similarity between two areas is measured by the ratio between the number of citations across these

Table 2: Areas in the main network

Area	Conference included
DM	KDD, ICDM, SDM, PKDD, PAKDD
ML	ICML, NIPS, AAAI, IJCAI, UAI, ECML
DB	VLDB, SIGMOD, ICDE, ICDT, EDBT, PODS
IR	SIGIR, WWW, ACL, ECIR, CIKM
BIO	ISMB, RECOMB, ECCB, BIBE, BIBM, WABI

two areas and the total number of cited papers in them. This is intuitive since two similar areas are more likely to cite each other's papers. The co-author network in each area is used as the domain-specific network.

5.1.1 *Effectiveness Evaluation*

To evaluate the effectiveness of the NoN model, we first apply CROSSRANK and study how the top-10 authors in DB change when varying a. Table 3 shows the ranking results for different a values. Note that when $a = 0$, the ranking result is the same as that of applying random walk with restart in each individual area independently.

From the table, we can see that the rank of three authors, Jiawei Han, Christos Faloutsos and Philip S. Yu, increases as a increases. When $a = 0$, their rank indicates their contributions to DB only. When a increases, their contributions to areas related to DB are also taken into consideration. In particular, they are the top-3 researchers in DM, which is the most similar area to DB in the main network. This indicates that CROSSRANK can effectively recognize a researcher's broader impact by incorporating his/her contribution across multiple areas.

Next, we evaluate the effectiveness of CROSSQUERY by studying an interesting cross-area link prediction task, i.e., for a given query author, we would like to identify future collaborators for this author in a relevant area. In particular, we partition the DBLP dataset into two parts, one from 2001 to 2005 (time interval T_1) and another from 2006 to 2010 (time interval T_2). We are interested in the DB researchers who have no DM publications during $T1$ but collaborate with DM researchers and publish in DM during T_2. We select such DM-DB author-pairs and use network in T_1 to predict their co-authorship in T_2.

For comparison, we select methods developed for (1) a single merged co-author network, and (2) heterogeneous information network. For a single merged network, we select the following methods. The path counting (PC) method uses the number of paths between each pair. Here we consider paths with length no longer than 5. The Katz method [11] is also a path counting method that penalizes longer paths. Personalized PageRank [9] and PropFlow [16] are random walk based methods. PathSim is designed for heterogeneous information network [22]. We use the 9 meta paths proposed in [21] for co-authorship link prediction. The parameters of the selected methods are tuned for their optimal performance.

Note that for all the selected methods, we focus on authors with at least certain number of publications, and author-pairs who are within a few hops in the merged overall co-author networks. This is reasonable since highly productive authors are more likely to cooperate with authors within a small distance [21].

We perform leave-one-out cross validation and use *accuracy* and *AUC* value as the evaluation criteria. Specifically, we test one selected DM-DB author-pair at each time. The DM researcher is used as the query node and DB is the target query area. A prediction is considered accurate if the test DB researcher is among the top-20 ranked authors in DB.

Table 4: Co-authorship prediction results

#Papers	Hops	#Pairs	Methods	AUC	Accuracy
≥ 3	[3, 4]	45	PC	0.7196	0.4444
			Katz	0.7439	0.5556
			PropFlow	0.7558	0.6222
			PathSim	0.5636	0.2444
			PageRank	0.7417	0.5333
			CrossQuery	**0.7685**	**0.6444**
≥ 3	[3, 6]	70	PC	0.6009	0.3000
			Katz	0.6243	0.3714
			PropFlow	0.6268	0.4429
			PathSim	0.5278	0.2143
			PageRank	0.6378	0.3714
			CrossQuery	**0.6632**	**0.4571**
≥ 5	[3, 4]	23	PC	0.6521	0.2609
			Katz	0.6717	**0.3478**
			PropFlow	0.6850	**0.3478**
			PathSim	0.4279	0.1304
			PageRank	0.6743	**0.3478**
			CrossQuery	**0.7099**	**0.3478**
≥ 5	[3, 6]	38	PC	0.5692	0.2105
			Katz	0.5786	0.2368
			PropFlow	0.5950	**0.2895**
			PathSim	0.4362	0.1053
			PageRank	0.5880	0.2368
			CrossQuery	**0.6308**	**0.2895**

Table 4 shows the results. From the table, we can see that CROSSQUERY outperforms all alternative methods. The performance gain becomes larger when more test author-pairs are available. The alternative methods are not suitable for this cross-area query task, since they only consider the similarity between the authors but not areas. Note that the meta paths used in PathSim do not consider the similarity between areas and conferences either.

5.1.2 *Efficiency Evaluation*

We evaluate the efficiency of the proposed methods using both the DBLP and synthetic datasets. The main network in the DBLP NoN consists of 121 conferences. The number of nodes in the conference-specific coauthor networks ranges from 88 to 14,636 depending on the size of the conferences, with a total of 259,822 nodes. The synthetic NoN are generated using the R-MAT model [2] so that the resulting networks resemble real-word networks. The main network consists of 1,023 nodes. The domain-specific networks contain 935 to 8,100 nodes with a total of 3,773,519 nodes.

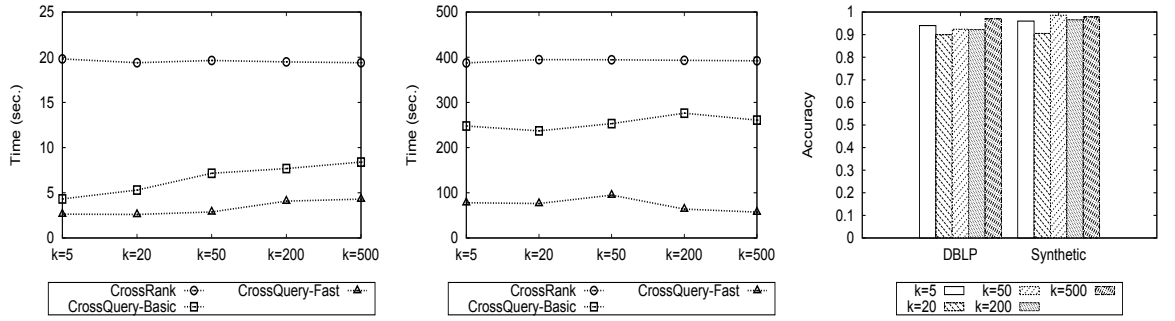
Figure 2 shows the running time of CROSSQUERY-BASIC and CROSSQUERY-FAST when varying the number of returned nodes k. The running time of CROSSRANK is also reported in the figure as a reference. The running time is averaged over 10 randomly selected query nodes and target domain-specific networks. The parameters are $a = 0.2, c = 0.85$, and $\epsilon = 10^{-3}$. It can be seen that both CROSSQUERY-BASIC and CROSSQUERY-FAST are much more efficient than CROSSRANK. The running time of CROSSQUERY-BASIC and CROSSQUERY-FAST increases as k increases. The randomly selected query nodes may affect the trend, e.g., in Figure 2(b) when $k = 500$. The corresponding averaged accuracy of CROSSQUERY-FAST is shown in 2(c). From the results, we can see that CROSSQUERY-FAST achieves more than 90% accuracy for all settings and dramatically improves the efficiency.

5.2 Protein Interaction NoN

In this section, we apply the proposed NoN model to the candidate gene prioritization problem, which has recently attracted in-

Table 3: Top ranked authors in the database area when varying a

Rank	$a = 0$	$a = 0.05$	$a = 0.1$	$a = 0.3$	$a = 0.5$
1	Divesh Srivastava	**Jiawei Han**	**Jiawei Han**	**Jiawei Han**	**Jiawei Han**
2	**Jiawei Han**	Divesh Srivastava	Divesh Srivastava	**Philip S. Yu**	**Philip S. Yu**
3	**Philip S. Yu**	**Philip S. Yu**	**Philip S. Yu**	Divesh Srivastava	**Christos Faloutsos**
4	Hector Garcia-Molina	Hector Garcia-Molina	Hector Garcia-Molina	**Christos Faloutsos**	Michael Stonebraker
5	Raghu Ramakrishnan	Raghu Ramakrishnan	**Christos Faloutsos**	Michael Stonebraker	Divesh Srivastava
6	Gerhard Weikum	Gerhard Weikum	Michael Stonebraker	Hector Garcia-Molina	Hector Garcia-Molina
7	Beng Chin Ooi	**Christos Faloutsos**	Raghu Ramakrishnan	Michael J. Carey	Michael J. Carey
8	H. V. Jagadish	Michael Stonebraker	Gerhard Weikum	Raghu Ramakrishnan	Gerhard Weikum
9	Michael J. Carey	Michael J. Carey	Michael J. Carey	Gerhard Weikum	Raghu Ramakrishnan
10	Michael Stonebraker	Beng Chin Ooi	Beng Chin Ooi	Elke A. Rundensteiner	Elke A. Rundensteiner



(a) Query time on DBLP NoN (b) Query time on synthetic NoN (c) Accuracy of CROSSQUERY-FAST

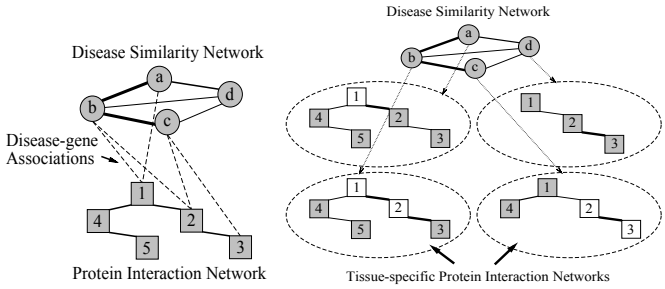
Figure 2: Efficiency and accuracy of the proposed techniques

tensive research interests. Various methods have been proposed including regression based methods [29], alignment based methods [30], random walk based methods [31, 14, 28], and maximum flow based methods [3].

Most of the state-of-the-art methods use a heterogeneous network as shown in Figure 3(a). The heterogeneous network consists of three components: a disease similarity network, a generic protein interaction network, and known disease-gene associations connecting the two networks. Based on the "guilt-by-association" principle, these methods utilizes similarities between diseases to infer genes that are associated with diseases [4].

Since the majority of genetic disorders tend to manifest only in a few tissues, recent studies have shown that it is more reasonable to utilize tissue-specific protein interaction networks in candidate gene prioritization [17].

We construct the tissue-specific protein interaction NoN as shown in Figure 3(b). The similarity network between 5,080 diseases is retrieved from the OMIM database [7] and used as the main network in NoN. We use the tissue-specific protein interaction network contributed in [17]. The authors generated tissue-specific protein interaction networks of 9,998 proteins for 60 human tissues using gene expression profiles in these tissues. For each disease in the main network, we use the protein interaction network in the tissue that is most relevant to the disease as its domain-specific network [13]. We collect 2,187 known associations between 1,524 diseases and 1,326 genes from the OMIM database [7] and select a subset of known disease-gene associations by filtering out diseases with the MAS (Maximum Association Score) less than 8%, which is used for selecting diseases having high associations with relevant tissues [17, 13], and diseases whose maximum similarity to all other diseases is less than 0.5 to preserve significant similarities among diseases [27]. The resulting disease-gene associations involve 147 associations between 106 diseases and 102 genes. These genes are

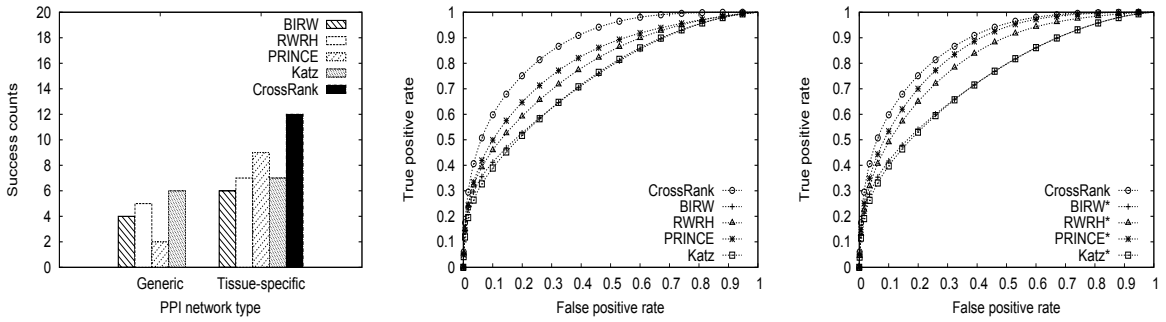


(a) Heterogeneous network structure (b) NoN structure

Figure 3: Heterogeneous network and NoN for the candidate gene prioritization problem. The protein interaction network in (a) is unweighed while those in (b) are weighed, denoting by the edge thickness

used as seed nodes in CROSSRANK which are represented by the white rectangles in Figure 3(b)

We select four state-of-the-art methods for comparison, including RWRH [14], BIRW [31], PRINCE [28] and Katz [20]. We use the standard leave-one-out cross validation to compare the prioritization accuracy of the selected methods. Specifically, at each time, we remove an association between the query disease and one causal gene, as well as all the associations involving the causal gene to avoid the trivial cases where mutations in the same gene cause two very similar diseases [28, 17]. This causal gene is used as the test gene. If the test gene is ranked within the top-k genes, where k is the total number of known genes for the query disease, this is considered as a successful prediction. This evaluation method is



(a) # successful preditions (b) ROC using generic protein network (c) ROC using tissue-specific protein network

Figure 4: Comparison between CROSSRANK and baseline methods

Table 5: AUC value comparison

Method	AUC50	AUC100	AUC300	AUC500	AUC
RWRH	0.2222	0.2846	0.3335	0.3601	0.8049
RWRH*	0.2143	0.2616	0.3364	0.3738	0.8475
BIRW	0.2261	0.2786	0.3198	0.3381	0.7586
BIRW*	0.2233	0.2653	0.3203	0.3393	0.7672
PRINCE	0.2373	0.2799	0.3454	0.3827	0.8339
PRINCE*	0.2446	0.2899	0.3656	0.4133	0.8832
Katz	0.1869	0.2343	0.2896	0.3073	0.7594
Katz*	0.1948	0.2299	0.2804	0.3077	0.7636
CrossRank	**0.2935**	**0.3477**	**0.4280**	**0.4800**	**0.9048**

Table 6: Ranking results comparison

Method	better	tie	worse	p-value
CrossRank vs. RWRH	106	2	39	2.04×10^{-11}
CrossRank vs. RWRH*	94	6	47	2.38×10^{-6}
CrossRank vs. BIRW	106	4	37	1.82×10^{-11}
CrossRank vs. BIRW*	100	9	38	1.44×10^{-9}
CrossRank vs. PRINCE	108	4	35	1.08×10^{-10}
CrossRank vs. PRINCE*	79	6	62	1.17×10^{-2}
CrossRank vs. Katz	108	5	34	2.32×10^{-12}
CrossRank vs. Katz*	103	9	35	1.23×10^{-10}

commonly used in the candidate gene prioritization studies [28]. The parameters are tuned for their optimal performance for the selected methods.

Note that all these baseline methods use a generic (non tissue-specific) protein interaction network. For a fair comparison, we also apply these methods to tissue-specific protein interaction networks. Specifically, for each query disease, we replace the generic protein interaction network by its most relevant tissue-specific protein interaction network.

Figure 4(a) shows the number of successful predictions made by the selected methods. As we can see, using tissue-specific protein interaction networks can improve the accuracy for the baseline methods. CROSSRANK has more successful predictions than all other methods. Figures 4(b) and 4(c) show the ROC curve. The results are consistent with the previous ones. The corresponding AUC values are reported in Table 5, where AUC values are reported by considering up to 50, 100, 300, 500 and all false positives. Note that a "*" indicates the use of tissue-specific protein interaction networks.

We further study how often CROSSRANK gives higher ranks to test genes than other methods. The results are shown in Table 6. The Wilcoxon signed rank test is used to assess the statistical significance of the difference between the ranking lists given by the two compared methods. It is clear from the table that the results of CROSSRANK are significantly better than that of the alternatives.

6. RELATED WORK

Networks are ubiquitous in real-life applications. The simplest model uses a single graph to represent a network. A variety of ranking algorithms have been developed for a single network [9, 8, 19, 6, 15]. Recently, various advanced network models have been proposed, such as multi-relational network [18], heterogenous information network [23], and hypergraph [34]. The multi-relational network and heterogeneous information network can incorporate node or edge type information: in a multi-relational network, edges connecting two nodes may be of different types; while different types of objects can co-exist in a heterogenous information network. Some work uses them interchangeably [32]. A tensor based co-ranking framework is proposed in [18] for multi-relational network. In heterogeneous information network, ranking-based clustering [23] and ranking-based classification [10] frameworks are developed, where ranking and clustering/classification can be mutually enhanced. In a hypergraph, an edge can connect any subset of nodes and is similar to a main node in our NoN model. But the set of nodes connected by an edge in a hypergraph do not form any network topology. If we treat each hyperedge as a main node, hypergraph can be viewed as special case of NoN where we do not have any links among domain nodes at all.

7. CONCLUSION

Ranking is a primitive operation in network analysis. In this paper, we propose a new network data model, Network of Networks (NoN), which enables novel ranking tasks such as CROSSRANK and CROSSQUERY. We formulate ranking on NoN as a regularized optimization problem, develop efficient algorithms, and provide rigorous theoretical analysis. Experimental results on real-world datasets demonstrate the effectiveness and efficiency of the proposed methods.

8. ACKNOWLEDGEMENT

This work was partially supported by the National Science Foundation grants IIS-1162374, IIS-1218036 and IIS1017415, by the Army Research Laboratory under Cooperative Agreement Number W911NF-09-2-0053, by Defense Advanced Research Projects Agency (DARPA) under Contract Number W911NF-11-C-0200 and

W911NF-12-C-0028, by Region II University Transportation Center under the project number 49997-33 25, and by China 973 Fundamental R&D Program (No.2014CB340304).

9. REFERENCES

[1] R. Bhatia. Linear algebra to quantum cohomology: the story of alfred horn's inequalities. *American Mathematical Monthly*, pages 289–318, 2001.

[2] D. Chakrabarti, Y. Zhan, and C. Faloutsos. R-mat: A recursive model for graph mining. In *SDM*, 2004.

[3] Y. Chen, T. Jiang, and R. Jiang. Uncover disease genes by maximizing information flow in the phenome–interactome network. *Bioinformatics*, 27(13):i167–i176, 2011.

[4] Y. Chen, W. Zhang, M. Gan, and R. Jiang. Constructing human phenome-interactome networks for the prioritization of candidate genes. *STATISTICS AND ITS INTERFACE*, 5(1):137–148, 2012.

[5] Y. Fujiwara, M. Nakatsuji, M. Onizuka, and M. Kitsuregawa. Fast and exact top-k search for random walk with restart. *VLDB*, 5(5):442–453, 2012.

[6] Y. Fujiwara, M. Nakatsuji, H. Shiokawa, T. Mishima, and M. Onizuka. Efficient ad-hoc search for personalized pagerank. In *SIGMOD*, pages 445–456, 2013.

[7] A. Hamosh, A. F. Scott, J. S. Amberger, C. A. Bocchini, and V. A. McKusick. Online mendelian inheritance in man (omim), a knowledgebase of human genes and genetic disorders. *Nucleic acids research*, 33(suppl 1):D514–D517, 2005.

[8] G. Jeh and J. Widom. Simrank: a measure of structural-context similarity. In *KDD*, pages 538–543, 2002.

[9] G. Jeh and J. Widom. Scaling personalized web search. In *WWW*, pages 271–279, 2003.

[10] M. Ji, J. Han, and M. Danilevsky. Ranking-based classification of heterogeneous information networks. In *KDD*, pages 1298–1306, 2011.

[11] L. Katz. A new status index derived from sociometric analysis. *Psychometrika*, 18(1):39–43, 1953.

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

[13] K. Lage, N. T. Hansen, E. O. Karlberg, A. C. Eklund, F. S. Roque, P. K. Donahoe, Z. Szallasi, T. S. Jensen, and S. Brunak. A large-scale analysis of tissue-specific pathology and gene expression of human disease genes and complexes. *Proceedings of the National Academy of Sciences*, 105(52):20870–20875, 2008.

[14] Y. Li and J. C. Patra. Genome-wide inferring gene–phenotype relationship by walking on the heterogeneous network. *Bioinformatics*, 26(9):1219–1224, 2010.

[15] D. Liben-Nowell and J. Kleinberg. The link-prediction problem for social networks. *Journal of the American society for information science and technology*, 58(7):1019–1031, 2007.

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

[17] O. Magger, Y. Y. Waldman, E. Ruppin, and R. Sharan. Enhancing the prioritization of disease-causing genes through tissue specific protein interaction networks. *PLoS Computational Biology*, 8(9):e1002690, 2012.

[18] M. K.-P. Ng, X. Li, and Y. Ye. Multirank: co-ranking for objects and relations in multi-relational data. In *KDD*, pages 1217–1225, 2011.

[19] P. Sarkar, A. W. Moore, and A. Prakash. Fast incremental proximity search in large graphs. In *ICML*, pages 896–903, 2008.

[20] U. M. Singh-Blom, N. Natarajan, A. Tewari, J. O. Woods, I. S. Dhillon, and E. M. Marcotte. Prediction and validation of gene-disease associations using methods inspired by social network analyses. *PloS one*, 8(5):e58977, 2013.

[21] Y. Sun, R. Barber, M. Gupta, C. C. Aggarwal, and J. Han. Co-author relationship prediction in heterogeneous bibliographic networks. In *ASONAM*, pages 121–128, 2011.

[22] Y. Sun, J. Han, X. Yan, P. S. Yu, and T. Wu. Pathsim: Meta path-based top-k similarity search in heterogeneous information networks. *VLDB*, 2011.

[23] Y. Sun, J. Han, P. Zhao, Z. Yin, H. Cheng, and T. Wu. Rankclus: integrating clustering with ranking for heterogeneous information network analysis. In *EDBT*, pages 565–576, 2009.

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

[25] H. Tong, C. Faloutsos, and J.-Y. Pan. Random walk with restart: fast solutions and applications. In *ICDM*, pages 327–346, 2006.

[26] H. Tong, J. He, M. Li, W.-Y. Ma, H.-J. Zhang, and C. Zhang. Manifold-ranking-based keyword propagation for image retrieval. *EURASIP J. Adv. Sig. Proc.*, 2006.

[27] M. A. van Driel, J. Bruggeman, G. Vriend, H. G. Brunner, and J. A. Leunissen. A text-mining analysis of the human phenome. *European journal of human genetics*, 14(5):535–542, 2006.

[28] O. Vanunu, O. Magger, E. Ruppin, T. Shlomi, and R. Sharan. Associating genes and protein complexes with disease via network propagation. *PLoS computational biology*, 6(1):e1000641, 2010.

[29] X. Wu, R. Jiang, M. Q. Zhang, and S. Li. Network-based global inference of human disease genes. *Molecular Systems Biology*, 4(1), 2008.

[30] X. Wu, Q. Liu, and R. Jiang. Align human interactome with phenome to identify causative genes and networks underlying disease families. *Bioinformatics*, 25(1):98–104, 2009.

[31] M. Xie, T. Hwang, and R. Kuang. Reconstructing disease phenome-genome association by bi-random walk. *Bioinformatics*, 2012.

[32] Y. Yang, N. Chawla, Y. Sun, and J. Hani. Predicting links in multi-relational and heterogeneous networks. In *ICDM*, pages 755–764, 2012.

[33] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf. Learning with local and global consistency. *Advances in neural information processing systems*, 16(16):321–328, 2004.

[34] D. Zhou, J. Huang, and B. Schölkopf. Learning with hypergraphs: Clustering, classification, and embedding. *Advances in neural information processing systems*, 19:1601, 2007.