Ranking Services by Service Network Structure and Service Attributes

Yang Zhou, Ling Liu
Georgia Institute of Technology
{*yzhou,lingliu*}*@gatech.edu*

Chang-Shing Perng, Anca Sailer, Ignacio Silva-Lepe
IBM T.J.Watson Research Center
{*perng,ancas,isilval*}*@us.ibm.com*

Zhiyuan Su
Dalian University of Technology
suzhiyuan2006@gmail.com

***Abstract*— Service network analysis is an essential aspect of web service discovery, search, mining and recommendation. Many popular web service networks are content-rich in terms of heterogeneous types of entities, attributes and links. A main challenge for ranking services is how to incorporate multiple complex and heterogeneous factors, such as service attributes, relationships between services, relationships between services and service providers or service consumers, into the design of service ranking functions. In this paper, we model services, attributes, and the associated entities, such as providers, consumers, by a heterogeneous service network. We propose a unified neighborhood random walk distance measure, which integrates various types of links and vertex attributes by a local optimal weight assignment. Based on this unified distance measure, a reinforcement algorithm, ServiceRank, is provided to tightly integrate ranking and clustering by mutually and simultaneously enhancing each other such that the performance of both can be improved. An additional clustering matching strategy is proposed to efficiently align clusters from different types of objects. Our extensive evaluation on both synthetic and real service networks demonstrates the effectiveness of ServiceRank in terms of the quality of both clustering and ranking among multiple types of entity, link and attribute similarities in a service network.**

I. Introduction

With the increasing popularity of web services, web service management is becoming an interesting and challenging research problem which has received much attention recently [1], [2], [3], [4], [5], [6], [7], [8]. Service network analysis has emerged as a critical aspect of web service management in both industry and academic research. Many popular web service networks are content-rich in terms of heterogeneous types of entities and links, associated with incomplete attributes. Such web service networks expose two heterogeneity challenges: (1) Multiple types of entities co-exist in the same service network with various attributes, and (2) Links between entities have different types and carry different semantics. Figure 1 presents a real service network from IBM knowledge base. There are two kinds of object vertices: blue service vertices and grey provider nodes. Each service vertex may contain three types of properties: red, purple and green attribute vertices specify service's "Type", "Category" and "Capability", respectively. Each provider vertex may have two kinds of attributes: red "Type" attribute and purple "Category" property. On the other hand, there are three kinds of links: an ochre edge represents the "Provides" relationship between services and providers; a black line specifies the structure relationship between objects with the same type; a dashed edge denotes the attribute edge between object and its attribute.

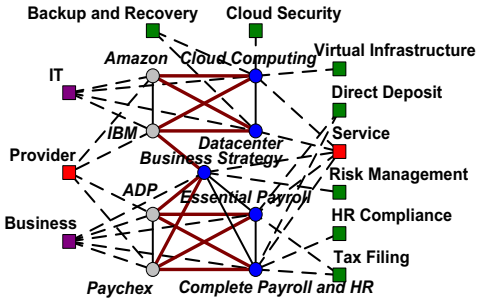


Figure 1. A Heterogeneous Service Network from IBM Knowledge Base

Typical applications of web service management include service retrieval and selection, service ranking and recommendation, service clustering and discovery, service adaptation and composition etc. Among these applications, service clustering and ranking are two significant ones. Ranking entails assigning a score to each service, quantifying its characteristics based on some ranking criteria. With such criteria, "interesting" services to response a specific user request appear high in the returned list. Prominent web ranking algorithms such as HITS [9] and PageRank [10] as well as service ranking methods such as [3] and [4] rank webpages or services on the whole homogeneous network so that the most relevant webpages or services are presented on the top of the returned list. On the other hand, clustering partitions a service network into groups so that services within a cluster are densely connected based on a certain similarity measure. Most existing graph clustering techniques have focused on the topological structures of homogeneous graph based on various criteria.

Although clustering and ranking are two well-known analytical tools for web service management, existing service clustering and ranking approaches are usually regarded as two independent processes. As a result, two main drawbacks characterize such approaches. First, clustering the entire service network without considering service's ranking may result in incorrect clustering result. Suppose we remove providers "Amazon", "Paychex" and their associated links from Figure 1, then service "Business Strategy" has only two providers "IBM" and "ADP". If we want to partition vertices into two clusters, then "IBM" and "ADP" will be in cluster "IT" and cluster "Business", respectively. But there is no statistically convincing evidence on which cluster "Business Strategy" should fall into. Second, ranking services on the

whole service network without considering which groups they belong to often leads to biased ranking result. If we adopt the global ranking algorithm to rank all services in Figure 1, then service "Business Strategy" has a higher score than service "Datacenter" since "Business Strategy" has more peer services and is provided by more providers. This ranking result does not make sense for a customer seeking a "IT" service. However, integrating clustering and ranking together can lead to more comprehensible results. By combining clustering and ranking, "Business Strategy" and "ADP" will belong to cluster "Business" since both rank high in cluster "Business" and rank low in cluster "IT". In contrast, "IBM" ranks high in cluster "IT" and relatively low in "Business". Thus, "Business Strategy" and "ADP" are more similar. Similarly, service "Datacenter" will have a higher score than service "Business Strategy" in cluster "IT" if we rank all services in each cluster.

In this paper, we identify four requirements for ranking and clustering a heterogeneous network of services. First, we need to integrate various types of links and attributes into a unified distance model to estimate the pairwise vertex closeness. Second, we need to design a robust probabilistic clustering method for heterogeneous service network to make our approach applicable to a wide range of applications. Third, we need to design a more useful service ranking approach with combining ranking information from peer services, providers and associated attributes. Finally, we need to develop a framework that can smoothly integrate ranking and clustering techniques.

The main contributions of this paper are outlined below. First, we propose a unified neighborhood random walk distance measure integrating various types of link and attribute information with the local optimal weight assignment on a heterogeneous service network. Second, we propose a greedy strategy to efficiently execute clustering matching process to align clusters for each type of objects on the heterogeneous service network. Third, we present a general algorithm that smoothly integrates ranking and clustering techniques, while mutually enhances the individual performance of each of those techniques. Finally, we perform extensive evaluation of our proposed ranking approach on both synthetic and real service datasets to demonstrate the effectiveness and efficiency of our method.

II. Problem Statement

In this section, we first introduce the problem formulation of service clustering and ranking considering both service network structure and service attribute information.

A **heterogeneous service network** is denoted as $G = (V, A, E)$, where V represents the set of object vertices, A denotes the set of property vertices, E contains multiple types of edges between object or property vertices. V contains two types of object vertices: the service set S and the provider set P, i.e., $V = S \cup P$. $A = \{a_{11}, \ldots, a_{1n_1}, \ldots, a_{m1}, \ldots, a_{mn_m}\}$ represents m associated attributes and their values for describing object properties. $Dom(a_i) = \{a_{i1}, \ldots, a_{in_i}\}$ represents the domain of attribute a_i with a size of $|Dom(a_i)| = n_i$. An attribute vertex $v_{ij} \in A$ represents that attribute a_i takes the j^{th} value. An attribute edge $(v_i, v_{jk}) \in E$ iff vertex v_i takes the value of a_{jk} on attribute a_j. Thus, there are five types of edges between different kinds of vertices: $E = E_{SS} \cup E_{PP} \cup E_{SP} \cup E_{SA} \cup E_{PA}$ where subscripts S, P and A represent the service set, the provider set and the attribute set, respectively. For ease of presentation, we call an edge between object vertices, i.e., $e \in E_{SS} \cup E_{PP} \cup E_{SP}$, as a structure edge and an edge between object and attribute, i.e., $e \in E_{SA} \cup E_{PA}$, as an attribute edge.

In such a heterogeneous service network with various types of objects, links and attributes, a good measure of "similarity" between objects is crucial to efficient social network analysis on real applications. A **unified neighborhood random walk distance measure** is to measure the closeness between vertices based on connectivity, vicinity and transition probabilities at different types of vertices.

Given k initial disjoint clusters of $G = (V, A, E)$ for each type of objects, $S = \cup_{i=1}^{k} S_i$ $(S_i \cap S_j = \emptyset, \forall i \neq j)$ and $P = \cup_{i=1}^{k} P_i$ $(P_i \cap P_j = \emptyset, \forall i \neq j)$ as the initial service influence, **service influence based probabilistic clustering** is to execute service influence propagation to partition each service $s \in S$ into each of k clusters based on the dynamic social activities with the initial disjoint service clusters and provider clusters. In the final clustering result, $\sum_{i=1}^{k} p_{si} = 1, \forall s \in S$ where p_{si} represents the normalized probability of service s will be partitioned into the i^{th} cluster. A desired clustering of a heterogeneous service network should achieve a good balance between the following two properties: (1) services within one cluster are close to each other in terms of structure links, while services between clusters are distant from each other; and (2) services within one cluster have similar properties, while services between clusters could have quite different attribute values.

Given a service influence based probabilistic clustering, **service influence based ranking** is to rank each service $s \in S$ and each provider $p \in P$ in each of k clusters based on some heuristics rules so that good service ranking generates good provider ranking, good provider ranking promotes good service ranking.

III. A Unified Weighted Distance Measure

In a heterogeneous service network, each service is associated with a service set and a provider set through structure links and an attribute set through attribute links, we propose to use a unified distance measure based on the neighborhood random walk model to integrate various types of structural and attribute similarities. In the heterogeneous service network, there exists a random walk path between two services $s_1, s_2 \in S$ if (1) s_1 and s_2 have the same peer service $s_3 \in S$; (2) both s_1 and s_2 are provided by the same

provider $p \in P$; or (3) s_1 and s_2 have the same attribute value $a \in A$. If there are multiple paths connecting s_1 and s_2, then they are close. On the other hand, if there are very few or no paths between s_1 and s_2, then they are far apart.

Definition 1 (Transition Probability Matrix): Let S be the service set, P be the provider set, and A be the set of associated attribute vertices, the transition probability matrix T of a heterogeneous service network G is defined as follow.

$$T = \begin{bmatrix} T_{SS} & T_{SP} & T_{SA} \\ T_{PS} & T_{PP} & T_{PA} \\ T_{AS} & T_{AP} & T_{AA} \end{bmatrix} \quad (1)$$

where T_{SS} is a $|S| \times |S|$ matrix representing the transition probabilities between service vertices; a $|P| \times |P|$ matrix T_{PP} specifies the transition probabilities between provider vertices; T_{SP} or T_{PS} represents the transition probabilities between services and providers; T_{SA} or T_{AS} denotes the transition probabilities between services and attributes; T_{PA} or T_{AP} represents the transition probabilities between providers and attributes; and T_{AA} is a $|A| \times |A|$ matrix with all 0s since there is no edge between attribute vertices. Here, $|S|$, $|P|$ and $|A|$ represent the cardinalities of the service set S, the provider set P and the attribute set A, respectively.

Since each type of structure and attribute edges may have different degrees of contribution in random walk distance, we assign each type of edges an individual weight. T_{SS}, T_{SP}, T_{PS} and T_{PP} correspond to four kinds of structure edges, and the corresponding structure weights are defined as α_{SS}, α_{SP}, α_{PS} and α_{PP}, respectively. Notice that α_{SP} is equal to α_{PS} since edges have no orientation in an undirected service network. On the other hand, there are m associated attributes with object vertices in the service network. The attribute edges connected to attribute vertices $v_{i1}, \ldots, v_{in_i}$ corresponding to attribute a_i are assigned to an attribute weight β_i. We proposed a dynamic weight tuning method [11] to produce a local optimal weight assignment for various types of links. Based on this weight assignment, each submatrix in T is defined as follow.

$$T_{SS}(i,j) = \begin{cases} \alpha_{SS} e_{ij}, & if (v_i, v_j) \in E_{SS} \\ 0, & otherwise \end{cases} \quad (2)$$

where e_{ij} denotes the number of providers that service i and service j co-shared. When service i and service j have different types or are from different categories, e_{ij} is usually equal to 0 since they often do not have the same providers.

$$T_{PP}(i,j) = \begin{cases} \alpha_{PP} e_{ij}, & if (v_i, v_j) \in E_{PP} \\ 0, & otherwise \end{cases} \quad (3)$$

where e_{ij} is the number of services that provider i and provider j co-provided.

$$T_{PS}(i,j) = \begin{cases} \alpha_{PS} e_{ij}, & if (v_i, v_j) \in E_{PS} \\ 0, & otherwise \end{cases} \quad (4)$$

where e_{ij} is the number of service j that provider i provided. e_{ij} has a value of 0 or 1 since provider i may or may not provide service j in the original dataset. As the relationship between services and providers is symmetric, $T_{PS}(i,j)$ is equal to $T_{SP}(j,i)$ due to $\alpha_{SP} = \alpha_{PS}$.

$$T_{SA}(i,J) = \begin{cases} \beta_j e_{ijk}, & if (v_i, v_{jk}) \in E_{SA} \\ 0, & otherwise \end{cases}, \; J = k + \Sigma_{l=1}^{j-1} n_l \quad (5)$$

where e_{ijk} denotes whether service i takes the k^{th} value on attribute a_j. It also has a value of 0 or 1.

$$T_{AS}(I,j) = \begin{cases} e_{ijk}, & if (v_j, v_{ik}) \in E_{SA} \\ 0, & otherwise \end{cases}, \; I = k + \Sigma_{l=1}^{i-1} n_l \quad (6)$$

where e_{ijk} specifies whether the k^{th} value on attribute a_i is taken by service j. The weight factor is ignored since each row in T_{AS} corresponds to the same attribute vertex v_{ik}.

Since both services and providers are the subclass of objects, T_{PA} and T_{SA} have the similar definitions and so have T_{AP} and T_{AS}.

$$T_{PA}(i,J) = \begin{cases} \beta_j e_{ijk}, & if (v_i, v_{jk}) \in E_{PA} \\ 0, & otherwise \end{cases}, \; J = k + \Sigma_{l=1}^{j-1} n_l \quad (7)$$

where e_{ijk} denotes whether provider i takes the k^{th} value on attribute a_j.

$$T_{AP}(I,j) = \begin{cases} e_{ijk}, & if (v_j, v_{ik}) \in E_{PA} \\ 0, & otherwise \end{cases}, \; I = k + \Sigma_{l=1}^{i-1} n_l \quad (8)$$

where e_{ijk} specifies whether the k^{th} value on attribute a_i is taken by provider j.

Since each row of the transition probability matrix should sum to 1, we then perform the row-wise normalization for T. Entries $T_{SS}(i,j)$, $T_{SP}(i,j)$ and $T_{SA}(i,j)$ are normalized by dividing them by the sum of row entries, i.e., $\Sigma_{l=1}^{|S|} T_{SS}(i,l) + \Sigma_{l=1}^{|P|} T_{SP}(i,l) + \Sigma_{l=1}^{|A|} T_{SA}(i,l)$. Similarly, elements $T_{PS}(i,j)$, $T_{PP}(i,j)$ and $T_{PA}(i,j)$ are normalized by $\Sigma_{l=1}^{|S|} T_{PS}(i,l) + \Sigma_{l=1}^{|P|} T_{PP}(i,l) + \Sigma_{l=1}^{|A|} T_{PA}(i,l)$. Entries $T_{AS}(i,j)$ and $T_{AP}(i,j)$ are normalized by $\Sigma_{l=1}^{|S|} T_{AS}(i,l) + \Sigma_{l=1}^{|P|} T_{AP}(i,l)$ since T_{AA} is a zero matrix.

A random walk on a heterogeneous service network G is performed in the following way. Suppose a particle starts at a certain vertex v_0 and walks to a vertex v_s in the s^{th} step and it is about to move to one of the neighbors of v_s, denoted as $v_t \in N(v_s)$, with the transition probability $T(s,t)$, where $N(v_s)$ contains all neighbors of vertex v_s. The vertex sequence of the random walk is a Markov chain. The probability of going from v_i to v_j through a random walk of length l can be obtained by multiplying the transition probability matrix l times.

Definition 2 (Unified Neighborhood Random Walk Distance): Let T be the transition probability of a heterogeneous service network G, l be the length that a random walk can go, and $c \in (0,1)$ be the restart probability, the unified neighborhood random walk distance $d(u,v)$ from vertex u to vertex v in G is defined as follow.

$$d(u,v) = \sum_{\substack{\tau: u \rightsquigarrow v \\ length(\tau) \leq l}} p(\tau) c (1-c)^{length(\tau)} \quad (9)$$

where τ is a path from u to v whose length is $length(\tau)$ with transition probability $p(\tau)$. $d(u,v)$ reflects the vertex closeness based on multiple types of links among services, providers and attributes.

The matrix form of the unified distance is given as follow.

$$R = \sum_{\gamma=1}^{l} c(1-c)^{\gamma} T^{\gamma} \quad (10)$$

For ease of presentation, R is rewritten as the following form by using the similar representation in Eq.(1).

$$R = \begin{bmatrix} R_{SS} & R_{SP} & R_{SA} \\ R_{PS} & R_{PP} & R_{PA} \\ R_{AS} & R_{AP} & R_{AA} \end{bmatrix} \quad (11)$$

IV. Service Influence Based Clustering

In this section, we propse an innovative service influence based probabilistic clustering framework considering social interactions among service and provider disjoint clusters.

A. Clustering Matching Process

Most existing clustering methods such as *k-Means* [12] and *k-Medoids* [13] only offer disjoint clusters. We can not directly apply them to our heterogeneous service network, otherwise they will lead to two issues: (1) service clusters and provider clusters are independent of each other; and (2) each object is partitioned into a single cluster. To address the first issue, the clustering matching process is designed to produce a one-to-one matching between service clusters and provider clusters based on some similarity measures.

The inter-cluster similarity with the same type is defined as follow.

$$d(\bar{c}_L) = \frac{1}{k(k-1)} \sum_{i=1}^{k} \sum_{j=1, j\neq i}^{k} d(c_L^i, c_L^j),\ L \in \{S, P\} \quad (12)$$

where c_L^x represents the centroid of cluster L_x and L specifies the service set (or the provider set). $d(c_L^i, c_L^j)$ is the unified distance from c_L^i to c_L^j. $S = \cup_{i=1}^{k} S_i$ $(S_i \cap S_j = \emptyset, \forall i \neq j)$ (or $P = \cup_{i=1}^{k} P_i$ $(P_i \cap P_j = \emptyset, \forall i \neq j)$) is a disjoint clustering of services (or providers). A clustering of services (or providers) with a small inter-cluster similarity is considered as a good clustering based on this criterion.

The inter-cluster similarity across different types is designed to quantitatively measure the similar extent between a service cluster and a provider cluster. It is defined as follow.

$$d(X_i, Y_j) = \frac{1}{|X_i||Y_j|} \sum_{x \in X_i, y \in Y_j} d(x, y)\ (X, Y \in \{S, P\}, X \neq Y) \quad (13)$$

where X is a disjoint clustering of services (or providers) and Y is a disjoint clustering of providers (or services).

The disjoint clustering matching algorithm is presented in Algorithm 1. It first chooses the clustering with the minimal inter-cluster similarity as the basic clustering X and the clustering with the maximal inter-cluster similarity as the clustering Y to be matched. The algorithm then calculates the inter-cluster similarity between X's clusters and Y's clusters and matches each cluster of Y to the cluster of X with the maximal inter-cluster similarity. It places the potential conflict cluster labels of Y and the corresponding cluster label of X into Q. During each "Dequeue" iteration, the algorithm rematches the one of two conflict clusters Y_m and Y_n, which has smaller inter-cluster similarity with cluster X_i, and puts new possible conflict tripes into Q until the queue is empty. Finally, it updates the cluster labels of Y with the matched cluster labels respectively.

Algorithm 1 Disjoint Clustering Matching

Input: a heterogeneous service network $G = (V, A, E)$, a disjoint clustering $S_1, \ldots, S_k, P_1, \ldots, P_k$, a unified random walk distance matrix R, and a queue Q containing triples of conflict cluster labels.
Output: a $1 \times k$ vector CL containing new cluster label for the matched clusters of S or P.

1: $X = argmin_{L\in\{S,P\}} d(\bar{c}_L)$; $Y = argmax_{L\in\{S,P\}} d(\bar{c}_L)$;
2: $CL = (0, 0, \cdots, 0)$; $Q = \phi$;
3: Calculate $d(X_i, Y_j)$ for $\forall i, j = 1, \cdots, k$;
4: **for** $j = 1, \cdots, k$
5: $\quad CL(j) = argmax_{i\in\{1,\cdots,k\}} d(X_i, Y_j)$;
6: $EnQueue(Q, (i, m, n))$ for $\forall CL(m) = CL(n) = i, m < n$;
7: **while** $QueueEmpty(Q) == False$
8: $\quad DeQueue(Q, (i, m, n))$;
9: $\quad$ **if** $CL(m) == CL(n)$
10: $\quad\quad j = argmin_{l\in\{m,n\}} d(X_i, Y_l)$;
11: $\quad\quad CL(j) = argmax_{l\in\{1,\cdots,k\}-\{i\}} d(X_l, Y_j)$;
12: $\quad\quad EnQueue(Q, (i', m', j))$ for $\forall CL(m') = CL(j) = i'$ $m' < j$;
13: $\quad\quad EnQueue(Q, (i', j, n'))$ for $\forall CL(j) = CL(n') = i'$ $j < n'$;
14: Update each Y_j's cluster label as $CL(j)$.

B. Service Influence Propagation

The process of service influencing one another in a service network is very similar to the heat diffusion process [14], [15]. Heat diffusion is a physical phenomenon that heat always flows from an object with high temperature to an object with low temperature. In the context of a heterogeneous service network, an early service often influences other late services through links. For example, some "IT" company currently provides an early service of "Datacenter" but doesn't support a late service of "Cloud Computing". If there exists a direct connection between "Datacenter" and "Cloud Computing", then it is very possible that "Cloud Computing" will be provided by this company in the future. On the other hand, a provider may also transfer its influence to services provided by it. For instance, "IBM" provides both "Datacenter" and "Business Strategy" so that these two services have an indirect connection through the common provider "IBM". The spread of service influence resembles the heat diffusion phenomenon. Early choice of a provider (or a service) with many peers and services (or providers) in a cluster may act as heat sources, transfer its heat to its peers and services (or providers) and diffuse their influences to other majority. Finally, at a certain time, heat is diffused to the margin of the service network.

We use our unified random walk distance as the heat diffusion kernel since it captures the service (or provider) influence through both direct and indirect links. The heat diffusion kernel H is a submatrix of R in Eq.(11).

$$H = \begin{bmatrix} R_{SS} & R_{SP} \\ R_{PS} & R_{PP} \end{bmatrix} \quad (14)$$

We utilize the matched clusters of services and providers from the disjoint clustering process as the heat (influence) source. A $(|S|+|P|) \times k$ initial heat matrix $sc(0)$ represents the heat source from which the heat kernel starts its diffusion process. The initial heat column vector $sc(0)(:, j)(j = 1, \cdots, k)$ of the service network at time 0 is defined below.

$$sc(0)(:, j) = (p_{1j}, \cdots, p_{|S|j}, p_{(|S|+1)j}, \cdots, p_{(|S|+|P|)j})^T \quad (15)$$

where p_{ij} is the probability of the i^{th} $(1 \leq i \leq |S|)$ service vertex in cluster S_j or the $(i-|S|)^{th}$ $(|S|+1 \leq i \leq |S|+|P|)$ provider vertex in cluster P_j. Due to the disjoint clustering,

each initial p_{ij} is equal to 0 or 1 and $\sum_{j=1}^{k} p_{ij} = 1, \forall i$. Thus, $sc(0)$ is given as follow.

$$sc(0) = [sc(0)(:,1), sc(0)(:,2), \cdots, sc(0)(:,k)] \quad (16)$$

where $sc(0)$ essentially denotes the clustering distribution of services and providers after the clustering matching process.

We treat each $p_{ij} > 0$ as an initial influence source to execute the service influence propagation in each cluster until convergence, i.e., influence equilibrium. We argue that a service's partitioning not only depends on the cluster label of peer services but also lies on the cluster label of its providers. During the following influence propagation, a service continuously obtains the influences from its providers and peer services about the cluster selection until it arrives at the margin of the service network.

$$\begin{aligned} sc(1) &= H \times sc(0) \\ &\cdots \\ sc(t) &= H \times sc(t-1) \end{aligned} \quad (17)$$

Therefore, the service network's thermal capacity at time t, denoted by $sc(t)$, is defined as follow.

$$sc(t) = H^t \times sc(0) \quad (18)$$

where $sc(t)$ corresponds to a probabilistic clustering result, i.e., each entry $sc(t)(i,j)$ represents the probability of the i^{th} $(1 \leq i \leq |S|)$ service or the $(i-|S|)^{th}$ $(|S|+1 \leq i \leq |S|+|P|)$ provider in the j^{th} cluster. We then normalize each $sc(t)(i,j)$ so that $\Sigma_{l=1}^{k} sc(t)(i,l) = 1$.

$$sc(t)(i,j) = \frac{sc(t)(i,j)}{\Sigma_{l=1}^{k} sc(t)(i,l)} \quad (19)$$

V. Service Influence Based Ranking

When ranking services over the global service network without considering which clusters they belong to, it is not clear to the user, from where to start looking at the results, since users usually prefer seeing a ranked result list in some area what they are interested in rather than a global ranked list. We require ranking functionality to present services to the user in a meaningful way i.e. by ranking them in each cluster based on a good clustering result. However, it is still not enough for a local ranking method to help the user efficiently distinguish the returned service list. We argue that the influence propagation among services and providers can provide more informative views of ranking result. The ranking of an early service usually influences other late peer services through the direct connection. On the other hand, a provider may transfer its influence to the ranking of services provided by it. Their influences are finally diffused to the entire service network through the influence propagation.

The following heuristics rules give us initial ideas.

- Highly ranked services in each cluster have connections with many highly ranked services within the same cluster.
- Highly ranked services in each cluster are provided by many highly ranked providers within the same cluster.
- Highly ranked providers in each cluster own many highly ranked peer providers within the same cluster.
- Highly ranked providers in each cluster provide many highly ranked services within the same cluster.

We still utilize H in Eq.(14) as the propagating heat-diffusion kernel in the ranking process since it captures the direct connections between different types of objects based on the above four ranking rules. We use a $(|S|+|P|) \times k$ influence matrix $sr(0)$ to represent the initial ranking of services and providers in each cluster. The initial ranking score of the i^{th} object vertex in j^{th} cluster is defined below.

$$sr(0)(i,j) = \begin{cases} \Sigma_{l=1, l\neq i, sc(t)(l,j)>0}^{|S|+|P|} H(i,l), & if\ sc(t)(i,j) > 0 \\ 0, & otherwise \end{cases} \quad (20)$$

where $sr(0)(i,j)$ is the sum of the unified random walk distance from the i^{th} $(1 \leq i \leq |S|)$ service or the $(i-|S|)^{th}$ $(|S|+1 \leq i \leq |S|+|P|)$ provider to other objects partitioned into cluster $sc(t)(:,j)$. If a service (or a provider) in a cluster is connected to many peers and many providers (or services) in the same cluster, then it will achieve a higher initial ranking score in this cluster. We then normalize each $sr(0)(i,j)$ so that $\Sigma_{l=1}^{|S|+|P|} sr(0)(l,j) = 1$.

$$sr(0)(i,j) = \frac{sr(0)(i,j)}{\Sigma_{l=1}^{|S|+|P|} sr(0)(l,j)} \quad (21)$$

Thus, we use the influence propagation to iteratively refine ranking scores based on the above heuristics rules.

$$\begin{aligned} sr(t)(:,j) &= H \times \begin{bmatrix} sr(t-1)(1:|S|,j) \\ sr(t-1)(|S|+1:|S|+|P|,j) \end{bmatrix}, \\ sr(t)(i,j) &= \frac{sr(t)(i,j)}{\Sigma_{l=1}^{|S|+|P|} sr(t)(l,j)} \end{aligned} \quad (22)$$

where $sr(t)(:,j)$ specifies the ranking score of each service (or provider) in cluster $sc(t)(:,j)$ at time t. During the influence propagation, a service's ranking is continuously refined by the ranking of its direct peers and providers as well as the ranking of these neighbors' neighbors. When the influence propagation reaches equilibrium, the final ranking score of a service in a cluster is jointly decided by the ranking scores of the services and providers in the same cluster which have at least one feasible path to this service.

The clustering and ranking based similarity is defined as follow.

$$\begin{aligned} e_{ij} = 1 - \frac{\sum_{l=1}^{k} |sc(t)(i,l)sr(t)(i,l) - sc(t)(j,l)sr(t)(j,l)|}{\sum_{l=1}^{k} sc(t)(i,l)sr(t)(i,l) + sc(t)(j,l)sr(t)(j,l)}, \\ \forall i,j \in \{1,\ldots,|S|+|P|\} \end{aligned} \quad (23)$$

We then substitute this new e_{ij} for the old one in T and recalculate the matrices T and R to further improve the performance of both clustering and ranking.

By assembling different parts, our ranking algorithm, ServiceRank, is presented in Algorithm 2. Steps 1 and 2 figure out a local optimal weight assignment for various types of links. Steps 3 calculates the transition probability matrix T and the unified random walk distance matrix R. Step 4 performs the clustering method, *k-Medoids*, to produce disjoint clusters of services and providers. The clustering matching process is then executed between service clusters and provider clusters in step 5. Steps 7-10 repeatedly run the service influence based clustering and ranking and iteratively update the clustering and ranking based similarity so that T and R are continuously refined to generate better ranking result until convergence.

Algorithm 2 ServiceRank

Input: a heterogeneous service network G, a length limit l of random walk paths, a restart probability c, a cluster number k.

Output: service ranking $sr(t)(:, j)$ in each cluster.

$\alpha_{SS} = \alpha_{SP} = \alpha_{PS} = \alpha_{PP} = \beta_1 = \ldots = \beta_m = 1.0$;
Generate the local optimal weights through the algorithm in [11];
Calculate T in Eq.(1) and R in Eq.(11);
Execute *k-Medoids* to produce clusters $S_1, \ldots, S_k$, $P_1, \ldots, P_k$;
Match $S_1, \ldots, S_k$ to $P_1, \ldots, P_k$ through Algorithm 1;
Repeat until convergence:
  Get probabilistic clusters $sc(t)(i, j)$ in Eq.(19);
  Get local ranking $sr(t)(i, j)$ in Eq.(22);
  Update all e_{ij}s in Eq.(23);
Recalculate T and R.

VI. Experimental Evaluation

We have performed extensive experiments to evaluate the performance of ServiceRank on both synthetic and real service datasets.

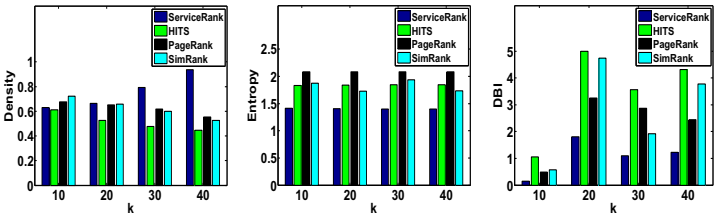
A. Experimental Datasets

BSBM Dataset: we modify the BSBM data generator [16] and create a dataset with $246,161$ triples where "Provides" is used to model the relationship between "Service" and "Provider"s providing them, while an instance of "Service" has multiple instances of properties "Capability", "Function" and "Type", and an instance of "Provider" contains multiple instances of properties "Feature" and "Type". There are totally $10,000$ "Service" instances and $3,628$ "Provider" instances with 10 "Type" instances and 5 instances of "Capability", "Function" and "Feature", respectively.

IBM Service Dataset: the dataset contains a total of $41,292$ triples with $2,450$ "Service" instances and 928 "Provider" instances. Each "Service" instance may have three types of properties: "Type", "Category" and "Capability", respectively. On the other hand, each "Provider" instance may contain two kinds of properties: "Type" and "Category". We build a service network where object vertices represent services or providers, attribute vertices denote their properties, object edges represent the relationship between object vertices, attribute edges specify the relationship between object vertices and their associated attributes.

B. Comparison Methods and Evaluation

We compare **ServiceRank** with three representative ranking algorithms, **HITS** [9], **PageRank** [10] and **SimRank** [17]. Since the last three ranking algorithms can not directly handle a heterogeneous network, we treat services, providers and attributes as homogeneous vertices with the same type and rank homogeneous vertices in each cluster. We choose three disjoint clustering methods of *k-Means* [12], *k-Means++* [18] and *k-Medoids* [13] for these three ranking algorithms, respectively. ServiceRank integrates various types of entity, link and attribute information into a unified distance measure with the local optimal weights on a heterogeneous service network. Service influence based clustering and ranking are iteratively performed to mutually enhance the performance of both of them.



(a) density (b) entropy (c) DBI

Figure 2. Cluster Quality Comparison on BSBM 13,628 Vertices

Evaluation Measures We use three measures of *density*, *entropy* and *Davies-Bouldin Index* (*DBI*) to evaluate the quality of clusters $\{S_i\}_{i=1}^k$ and $\{P_i\}_{i=1}^k$ generated by different methods. The metrics are defined as follows.

$$density(\{S_i \cup P_i\}_{i=1}^k) = \sum_{i=1}^{k} \frac{\sum_{v_p, v_q \in S_i \cup P_i, (v_p, v_q) \in E} min(\pi_{pi}, \pi_{qi})}{|E|} \tag{24}$$

where π_{pi}, π_{qi} represent the probabilities of vertex v_p and vertex v_q in the i^{th} cluster respectively. $min(\pi_{pi}, \pi_{qi})$ is equal to 1 for disjoint clustering methods but may be less than 1 for our probabilistic clustering.

$$entropy(\{S_i \cup P_i\}_{i=1}^k) = \sum_{i=1}^{m} \frac{\beta_i}{\sum_{l=1}^{m} \beta_l} \sum_{j=1}^{k} \frac{|S_j|}{|S|} entropy(a_i, S_j) + \frac{|P_j|}{|P|} entropy(a_i, P_j) \tag{25}$$

where $entropy(a_i, S_j) = -\sum_{l=1}^{|S|} p_{ijl} log_2 p_{ijl}$ and p_{ijl} is the percentage of services in cluster S_j which have value a_{il} on attribute a_i. $entropy(\{S_i \cup P_i\}_{i=1}^k)$ measures the weighted entropy from all attributes over k clusters.

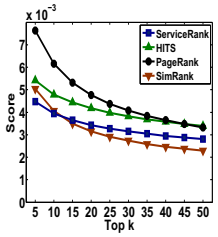
$$DBI(\{S_i \cup P_i\}_{i=1}^k) = \frac{1}{2k} \sum_{i=1}^{k} max_{j \neq i}\left(\frac{d(c_S^j, c_S^i)}{\sigma_S^j + \sigma_S^i}\right) + max_{j \neq i}\left(\frac{d(c_P^j, c_P^i)}{\sigma_P^j + \sigma_P^i}\right) \tag{26}$$

where c_S^x $(1 \leq x \leq k)$ is the centroid of cluster S_x, $d(c_S^i, c_S^j)$ is the random walk distance between centroids c_S^i and c_S^j, and σ_S^x $(1 \leq x \leq k)$ is the average similarity of all elements in cluster S_x to centroid c_S^x. $entropy(a_i, P_j)$, c_P^x, $d(c_P^i, c_P^j)$ and σ_P^x corresponding to providers P have the similar meanings. A cluster with high intra-cluster similarity and low inter-cluster similarity will have a low *DBI* value.

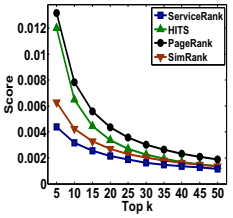
C. Cluster Quality Evaluation

Figure 2 (a) shows the density comparison on BSBM $13,628$ Vertices by varying the number of clusters $k = 10, 20, 30, 40$. The density values by ServiceRank obviously higher than that other three methods except $k = 10$. They remain in the range of 0.63 or above even when k is increasing. This demonstrates that ServiceRank can find densely connected components. Different from the disjoint clustering methods, the density value by our service influence based probabilistic clustering keeps increasing when k is increasing since each vertex is partitioned into more clusters with a larger k.

Figure 2 (b) presents the entropy comparison between four methods on BSBM $13,628$ Vertices. ServiceRank achieves the lowest entropy than other three methods. Entropy by ServiceRank is as low as less than 1.32 while entropy by other three approaches is still above 1.83 since they

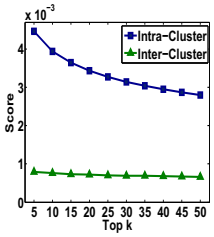


(a) BSBM 13,628 Vertices

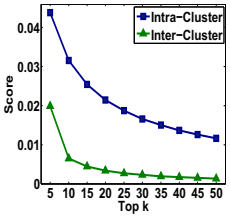


(b) IBM 3,652 Vertices

Figure 3. Ranking Quality Comparison

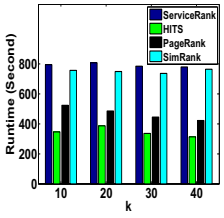


(a) BSBM 13,628 Vertices

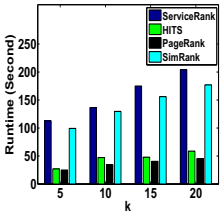


(b) IBM 3,652 Vertices

Figure 4. Ranking Accuracy Comparison



(a) BSBM 13,628 Vertices



(b) IBM 3,652 Vertices

Figure 5. Efficiency Evaluation

partitions the service network without considering different degrees of importance for multiple types of links. As shown in Figures 2 (a) and (b), the performance of ServiceRank is the best in terms of both density and entropy. This is because its distance function integrates the local optimal weight assignment for multiple types of structural and attribute links, thus it achieves a good performance on both criteria. On the other hand, other three methods do not differentiate the links between different kinds of objects. It is equivalent to combine multiple types of structural and attribute similarities with the equal weighting factor of 1, which usually does not produce a good clustering.

Figures 2 (c) shows the DBI comparison on BSBM $13,628$ Vertices with different k values by four methods. ServiceRank has the lowest DBI of around $0.15 - 1.80$, while PageRank, HITS and SimRank have a much higher DBI than ServiceRank. This demonstrates that ServiceRank can achieve both high intra-cluster similarity and low inter-cluster similarity.

D. Ranking Quality Evaluation

Figures 3 (a) and (b) plot the average ranking scores of top-k vertices in each cluster on two datasets, respectively. The score curve of ServiceRank with respect to k value is most stable among four ranking algorithms. Especially, the curve is almost a stable horizontal line when the k value stands in between 20 and 50. This is because ServiceRank achieves both the highest intra-cluster similarity and the lowest inter-cluster similarity. In addition, the iteratively updated ranking-based similarity between services makes the generated clusters get more cohesive intra-cluster structure and more homogeneous vertex properties. As a result, services within clusters are connected to similar services, providers and attributes so that they achieve similar ranking scores in terms of our heuristics ranking rules. The quality by PageRank and HITS is the worst since the resulted clusters without considering the weight assignment and the alternative iterations of clustering and ranking have a rather random distribution of services in terms of both structure and attribute similarities. Although SimRank also doesn't consider the weight assignment and the ranking iterations, the quality of SimRank stands in between since it iteratively updates the pairwise similarities.

Figures 4 (a) and (b) present the average ranking scores of services within clusters and services outside clusters by ServiceRank, respectively. We rank all services in each cluster and treat a service with a positive probability in cluster as the service within cluster. The score curve of services within clusters is much higher than that of services outside clusters since services within clusters are connected to many services, providers and attributes in clusters but there are few links between cluster and services outside cluster. During each iteration of clustering and ranking, the clustering iteratively refines the weights of multiple types of links in terms of their contribution. The updated ranking-based similarity with the refined weights makes services with the similar cluster distribution and the similar ranking scores become more similar so that they will have a higher probability to be partitioned into the same clusters in the next iteration.

E. Efficiency Evaluation

The running time by each algorithm for BSBM $13,628$ Vertices and IBM $3,378$ Vertices is summarized in Figures 5 (a) and (b), respectively. As we can observe, HITS and PageRank are the most efficient as they execute both clustering and ranking algorithms only once on the homogeneous service network (i.e., without running the clustering matching and the alternative iterations of clustering and ranking). SimRank is usually $1.5 - 4$ times slower than HITS and PageRank since it needs to iteratively update the pairwise similarities on the entire service network. Although ServiceRank does not need to update the pairwise similarities, it is about $1.05 - 1.16$ times slower than SimRank. As ServiceRank needs to iteratively adjust the weights of multiple types of structure and attribute links, it iteratively computes the random walk distance matrix from scratch on the service network. In addition, service influence based clustering and ranking are iteratively performed so that the total cost of ServiceRank has been greatly increased. Although ServiceRank is more expensive, the iterative refinement improves the ranking quality a lot, as demonstrated in Figures 3 and 4.

VII. Related Work

Web service discovery and management has been a heated topic in recent years [1], [5], [6], [8]. Skoutas et al. [3] proposed a methodology for ranking and clustering the relevant web services based on the notion of dominance, which apply multiple matching criteria without aggregating the match scores of individual service parameters. Xiao et al. [2] proposed a context modeling approach which can dynamically handle various context types and values. Based on the relations among context values, the algorithm can capture the potential services that the user might need. Almulla et

al. [4] presented a web services selection model based on fuzzy logic and proposed a fuzzy ranking algorithm based on the dependencies between proposed quality attributes. Liu et al. [7] proposed a heuristic social context-Aware trust network discovery algorithm, H-SCAN, by adopting the K-Best-First Search (KBFS) method and some optimization strategies.

Graph ranking is one of the core tasks in social networks. Most of existing graph ranking techniques [9], [10], [19], [20], [21] compute ranking scores by only resorting to graph structure information. Jeh and Widom [17] designed a measure called SimRank, which defines the similarity between two vertices in a graph by their neighborhood similarity. DivRank [22], based on a reinforced random walk in an information network, can automatically balances the prestige and the diversity of the top ranked vertices in a principled way. Tong et al. [23] defined a goodness measure to capture both the relevance and the diversity for a given ranking list.

Graph clustering has attracted active research in the last decade. Most of existing graph clustering techniques have focused on the topological structures based on various criteria, including normalized cuts [24], modularity [25], structural density [26]. Some recent works, SA-Cluster [27] and BAGC [28], perform clustering based on both structural and attribute similarities to partition the collaboration graph with single link type and single attribute type into k clusters. Sun et al. [29] proposed GenClus to cluster general heterogeneous information networks with different link types and different attribute types.

VIII. Conclusion

In this paper, we have presented a unified neighborhood random walk distance measure integrating various types of link and attribute information with the local optimal weight assignment on a heterogeneous service network. We present a reinforcement algorithm that is developed to tightly integrate ranking and clustering by mutually and simultaneously enhancing each other such that the performance of both can be improved. We propose an additional greedy strategy to efficiently execute clustering matching process to align clusters for each type of objects in the heterogeneous service network. Our extensive evaluation on both synthetic and real service networks demonstrates the power of our method in terms of the quality of both clustering and ranking.

Acknowledgment

The first two authors acknowledge the partial support from grants under NSF CISE NetSE program and SaTC program.

References

[1] K. Elgazzar, A. E. Hassan, and P. Martin, "Clustering wsdl documents to bootstrap the discovery of web services," in *ICWS*, 2010, pp. 147–154.

[2] H. Xiao, Y. Zou, J. Ng, and L. Nigul, "An approach for context-aware service discovery and recommendation," in *ICWS*, 2010, pp. 163–170.

[3] D. Skoutas, D. Sacharidis, A. Simitsis, and T. Sellis, "Ranking and clustering web services using multi-criteria dominance relationships," *TSC*, 3(3), pp. 163–177, 2010.

[4] M. Almulla, K. Almatori, and H. Yahyaoui, "A qos-based fuzzy model for ranking real world web services," in *ICWS*, 2011, pp. 203–210.

[5] M. Aznag, M. Quafafou, N. Durand, and Z. Jarir, "Multiple representations of web services: Discovery, clustering and recommendation," in *ICWS*, 2011, pp. 748–749.

[6] S. Dasgupta, S. Bhat, and Y. Lee, "Taxonomic clustering and query matching for efficient service discovery," in *ICWS*, 2011, pp. 363–370.

[7] G. Liu, Y. Wang, M. A. Orgun, and H. Liu, "Discovering trust networks for the selection of trustworthy service providers in complex contextual social networks," in *ICWS*, 2012.

[8] H. Q. Yu, X. Zhao, S. Reiff-Marganiec, and J. Domingue, "Linked context: A linked data approach to personalised service provisioning," in *ICWS*, 2012, pp. 376–383.

[9] J. M. Kleinberg, "Authoritative sources in a hyperlinked environment," *J. ACM*, vol. 46, pp. 604–632, 1999.

[10] S. Brin and L. Page, "The anatomy of a large-scale hypertextual web search engine," in *WWW*, 1998, pp. 107–117.

[11] Y. Zhou and L. Liu, "Clustering social networks with entity and link heterogeneity," in *GT Tech Report*, Atlanta, 2011.

[12] S. P. Lloyd, "Least squares quantization in PCM" *IEEE Transaction on Information Theory*, 28(2), pp. 128–137, 1982.

[13] L. Kaufman and P. J. Rousseeuw, "Clustering by means of medoids," *Statistical Data Analysis based on the L1 Norm*, pp. 405–416, 1987.

[14] H. Ma, H. Yang, M. R. Lyu, and I. King, "Mining social networks using heat diffusion processes for marketing candidates selection," in *CIKM*, 2008, pp. 233–242.

[15] D. Kempe, J. Kleinberg, and E. Tardos, "Maximizing the spread of influence through a social network," in *KDD*, 2003.

[16] C. Bizer and A. Schultz, "The berlin sparql benchmark," *IJSWIS*, 5(2), pp. 1–24, 2009.

[17] G. Jeh and J. Widom, "SimRank: a measure of structural-context similarity," in *KDD*, 2002, pp. 538–543.

[18] D. Arthur and S. Vassilvitskii, "K-means++: The Advantages of Careful Seeding," in *SODA*, 2007, pp. 1027–1035.

[19] B. Bahmani, A. Chowdhury, and A. Goel, "Fast incremental and personalized pagerank," *PVLDB*, 4(3), 173–184, 2010.

[20] B. Gao, T.-Y. Liu, W. Wei, T. Wang, and H. Li, "Semi-supervised ranking on very large graph with rich metadata," in *KDD*, 2011, pp. 96–104.

[21] B. Bahmani, R. Kumar, M. Mahdian, and E. Upfal, "Pagerank on an evolving graph," in *KDD*, 2012, pp. 24–32.

[22] Q. Mei, J. Guo, and D. Radev, "Divrank: the interplay of prestige and diversity in information networks," in *KDD'10*.

[23] H. Tong, J. He, Z. Wen, R. Konuru, and C.-Y. Lin, "Diversified ranking on large graphs: an optimization viewpoint," in *KDD*, 2011, 1028–1036.

[24] J. Shi and J. Malik, "Normalized cuts and image segmentation," *TPAMI*, 22(8), 2000, pp. 888–905.

[25] M. E. J. Newman and M. Girvan, "Finding and evaluating community structure in networks," *Phys. Rev. E69, 026113*, 2004.

[26] X. Xu, N. Yuruk, Z. Feng, and T. A. J. Schweiger, "Scan: a structural clustering algorithm for networks," in *KDD*, 2007.

[27] Y. Zhou, H. Cheng, and J. X. Yu, "Graph clustering based on structural/attribute similarities," in *VLDB*, 2009, pp. 718–729.

[28] Z. Xu, Y. Ke, Y. Wang, H. Cheng, and J. Cheng, "A model-based approach to attributed graph clustering," in *SIGMOD*, 2012, 505–516.

[29] Y. Sun, C. C. Aggarwal, and J. Han, "Relation strength-aware clustering of heterogeneous information networks with incomplete attributes," *PVLDB*, 5(5), pp. 394–405, 2012.