Reducing Seed Noise in Personalized PageRank

Noname manuscript No.
(will be inserted by the editor)

Reducing Seed Noise in Personalized PageRank

Shengyu Huang* · Xinsheng Li* · K. Selçuk Candan · Maria Luisa Sapino

Received: date / Accepted: date

Abstract Network based recommendation systems leverage the topology of the underlying graph and the current user context to rank objects in the database. Random-walk based techniques, such as PageRank, encode the structure of the graph in the form of a transition matrix of a stochastic process from which the significances of the nodes in the graph are inferred. Personalized PageRank (PPR) techniques complement this with a seed node set which serves as the *personalization context*. In this paper, we note (and experimentally show) that PPR algorithms that do not differentiate among the seed nodes may not properly rank nodes in situations where the seed set is *incomplete* and/or *noisy*. To tackle this problem, we propose alternative *robust personalized PageRank* (RPR) strategies, which are insensitive to noise in the set of seed nodes and in which the rankings are not overly biased towards the seed nodes. In particular, we show that novel *teleportation discounting* and *seed-set maximal PPR* techniques help eliminate harmful bias of individual seed nodes and provide effective seed differentiation to lead to more accurate rankings. We also show that the proposed techniques lead to efficient implementations, where existing approximation algorithms and/or parallel implementations for computing the PPR scores can be easily leveraged. Moreover, the proposed formulations are *reuse-promoting* in the sense that, it is possible to divide the work relative to individual seed nodes and cache the intermediary results

This paper is the extended version of *Shengyu Huang, Xinsheng Li, K. Selçuk Candan, Maria Luisa Sapino. "Can you really trust that seed?" : Reducing the Impact of Seed Noise in Personalized PageRank. International Conference on Advances in Social Network Analysis and Mining (ASONAM). Beijing, China. 2014. "*" indicates student authors with equal contributions.*

Shengyu Huang, Xinsheng Li, K. Selçuk Candan
Arizona State University
E-mail: {shengyu.huang,lxinshen,candan}@asu.edu

Maria Luisa Sapino
University of Torino
E-mail: marialuisa.sapino@unito.it

obtained during the computation, and especially in systems with large query throughputs, it may be possible to cluster queries based on the partial overlaps between the seed sets and reduce the overall robust PPR computation costs. Experiment results show that the proposed techniques are efficient and highly effective in improving recommendations and eliminating unwanted bias due to imperfections in the seed set.

1 Introduction

How a given pair of nodes in a social network are related to each other reflects the underlying network topology. Recommendation systems often use the analysis of the structure of a given data and/or social graph, relative to the user's current context, to generate rankings and recommendations [22].

Significance of a node in a graph needs to reflect both the topology of the graph and the application semantics and measures: The *betweenness* measure [43] for example aims to quantify whether deleting the node would disconnect or disrupt the graph. The *centrality/cohesion* [9] measures quantify how close to a clique the given node and its neighbors are. Other *authority*, *prestige*, and *prominence* measures [5, 9, 10] measure the significance of the node in the graph through eigen-analysis or random walks. For example, the well-known PageRank algorithm [10] associates an importance score to each node relying on random walks: Let us consider a weighted, directed graph $G(V, E)$, where the weight of the edge $e_j \in E$ is denoted as $w_j (\geq 0)$ and $\sum_{e_j \in outedge(v_i)} w_j = 1.0$. The PageRank score of nodes V is the stationary distribution of a random walk on G, denoted with a vector $\mathbf{p}$:

$$\mathbf{p} = (1 - \beta)\mathbf{T}_G \times \mathbf{p} + \beta\mathbf{v}, \tag{1}$$

where $\mathbf{T}_G$ denotes the transition matrix corresponding to the graph G (and the underlying edge weights), $\mathbf{v}$ is a so-called *teleportation* vector, where all entries are $\frac{1}{\|V\|}$ and β, transportation probability, is probability of the random walk with teleporting.

An early attempt to contextualize the PageRank scores was the *topic sensitive PageRank* [24] approach which adjusts the PageRank scores of the nodes by assigning the *teleportation* probabilities in vector $\mathbf{j}$ in a way that reflects the graph nodes' degrees of match to a given search topic. In many applications, however, the context relevant to the recommendation is defined not through an explicit query, but a subset of the nodes (often referred to as the "*seed nodes*") in the graph. Personalized PageRank (PPR) [5, 14], for example, reflects a user's interest by modifying the teleportation vector taking into account a given set of *important* nodes which are the target of the random jumps: given a set of nodes $S \subseteq V$, instead of jumping to a random node in V with probability β, the random walk jumps to one of the nodes in the seed set, S, given by the user. More specifically, if we denote the *Personalized PageRank* (PPR) scores of the nodes in V with a vector $\boldsymbol{\pi}$, then

$$\boldsymbol{\pi} = (1 - \beta)\mathbf{T}_G \times \boldsymbol{\pi} + \beta\mathbf{s}, \tag{2}$$

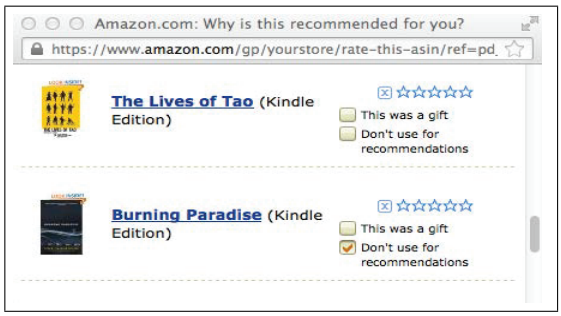


Fig. 1 An example interface enabling the user to explicitely eliminate outliers (e.g., a book purchased as a gift to a friend) in generating recommendations; such explicit corrections are not feasible in all applications of social networks

Rank Statistics of 9 Noisy Seeds w/o Seed Differentiation (out of 49 Seeds in 2500 Movies)		
Best Rank	*Avg. Rank*	*Worst Rank*
18	42.9	52

Table 1 Bias and lack of seed differentiation in PPR scores: most of the 49 seeds (out of 2500 movies) have very high PPR ranks, even if they are outliers in the seed set

where $\mathbf{s}$ is a re-seeding vector, such that if $v_i \in S$, then $\mathbf{s}[i] = \frac{1}{\|S\|}$ and $\mathbf{s}[i] = 0$, otherwise.

1.1 Problem - Noisy Seed Sets

The above formulation of PPR *assumes that all seeds are equally important* in characterizing the user's interest. This, however, may not always be the case, since in practice, the user feedback is often incomplete and noisy [11] (Figure 1). Unfortunately unless (a) each *individual* seed node is a good representative for the entire seed set, (b) the user/system was successful in including all seed nodes relevant for defining the current user context, and (c) most importantly, the user/system did not include any outlier nodes in the seed set, the resulting rankings might be biased and might contain undesirable artifacts, such as movies with low ratings, having high PPR rankings. Consider the following example:

Example 1 (Impact of the Noise in the Seed Set) Table 1 shows the PPR scores and ranks of the noisy seed nodes in a movie graph (see Section 5 for more details about this data set) for a sample user. In this example, we construct an *imperfect seed set* as follows: we select a random user and include in the seed set 40 movies rated "↑" (for "like") and 9 movies rated "↓" (for "dislike") by this user (out of 81 movies rated ↑ and 25 movies rated ↓ by this user).

Table 1 studies the relationship between the original user rating and the PPR ranking. As we see here, while as expected the average PPR rankings of

Rank Statistics of 9 Noisy Seeds with Seed Differentiation (out of 49 Seeds in 2500 Movies)		
Best Rank	*Avg. Rank*	*Worst Rank*
94	1109.8	1568

Table 2 Seed differentiation and bias elimination through *robust personalized PageRank* (RPR) scores: in the same situation as in Table 1, the average rating of the noisy seeds is greater than 1000 out of 2500 movies

the 9 noisy seed movies is poor among the rankings of the seed movies (~ 43 out of 52), it is also true that all the seed movies (including the 9 $\downarrow$-rated outlier movies) rank highly (i.e., better than 53 out of 2500). $\diamond$

Note that, while we often do not care about the ranks of the seed movies, high PPR-ranking of $\downarrow$-rated seeds implies that those movies neighboring these noisy seeds are also likely to be ranked highly, which is generaly undesirable.

1.2 Our Contributions: Robust Personalized PageRank (RPR)

Intuitively, a *noisy seed* (provided by the user or selected by the system) does not properly reflect the user's focus in that it does not fit in the context defined by the seed set as a whole. The above example illustrates that *a poor seed may overestimate the rankings of its (equally poor) neighbors in the final ranking*. This is primarily due to *(a) Teleportation-bias:* Firstly, as discussed in Section 1, the teleportation-vector based seeding algorithms jump on the seed set, S, relatively often (the common transportation probability, β, is 0.15) and the size of the seed set is often much smaller relative to the size of the data graph: as a result, the PPR value of the least significant seed node will be at least $\frac{\beta}{|S|}$, which is likely to be much higher than the PPR of non-seed nodes in the graph. *(b) Need for seed differentiation:* Secondly, as we commented above, the negative impact of the teleportation-bias can be alleviated if the teleportation rates to the seeds can be *differentiated*, but this requires a way to identify which seeds are truly better than others as this information is not available *a priori*[1]:

In this paper, we first propose *techniques to eliminate teleportation-bias and/or provide seed differentiation*:

- First and foremost, we discuss how the node rankings can be negatively affected by possible incompleteness and/or imperfection in the seeds set, and we experimentally establish that the conventional PPR metrics might not properly differentiate seed nodes in a graph.
- Secondly, we propose alternative *Robust personalized PageRank* (RPR) strategies, (a) which are insensitive to noise in the set of seed nodes (and thus differentiate seeds well) and (b) in which the rankings are not overly biased towards the seed nodes (Table 2).

[1] Otherwise the seed set would have been constructed differently

In particular, we propose novel *teleportation discounting* and *seed-set maximal PPR* techniques that can help eliminate harmful bias of individual seed nodes and provide effective seed differentiation to lead to more accurate rankings. Moreover, we also show that

- these can be obtained highly efficiently, if necessary, leveraging existing approximation algorithms [2, 4, 14, 17, 21, 23, 41] and/or parallel implementations [3, 32] for computing the PPR scores,
- the proposed formulations are *reuse-promoting* in the sense that, it is possible to divide the work relative to individual seed nodes and cache the intermediary results obtained during the computation – these cached results can then be *reused* for future queries sharing seed nodes, and
- especially in systems with large query throughputs, it may be possible to cluster queries based on the partial overlaps between the seed sets and, thus, significantly reduce the overall robust PPR computation costs.

1.3 Structure of this Paper

In the next section, we discuss the related work. In Section 3, we first formally introduce the problem and then present our solutions for seed differentiation, seed-bias elimination, and *Robust personalized PageRank* (RPR) computation. We propose four methods to solve this problem: PPR with global seed ranking (PPR-G), teleportation-discounted PPR (RPR-1), seed-set maximal PPR (RPR-2) and teleportation-discounted and seed-set maximal PPR(RPR-3). In Section 4, we discuss optimization and parallelization opportunities. First, we optimize the seed set maximal PPR problem by formulating it into a set of linear equations. We also discuss how to re-use intermediate results for efficient incremental computations. We next show that RPR queries can not only be approximated by relying on existing approximate PPR solutions, but (as we later show it experimentally) the proposed teleportation-discounting strategy can further improve the robustness of the RPR against inaccuracies due to graph partitioning. In this section, we also propose a multi-query strategy for high-throughput systems where RPR results are computed collectively for a set of PPR queries together. In Section 5, we evaluate the proposed robust personalized PageRank schemes for different data sets under different scenarios in terms of recommendation effectiveness, seed differentiation power, seed-bias elimination and efficiency. In Section 6, we conclude the paper.

2 Related Works

2.1 Node Significance

As described in the introduction, *significance* of a node in a graph needs to reflect both the topology of the graph and the application semantics. Measures like *betweenness* [43], *centrality/cohesion* [9], and others (including *authority*,

prestige, and *prominence* measures [5,9,10]) help quantify how *significant* any node is on a given graph under different assumptions of significance.

Many of these measures recognize that a node is more significant if it is reachable from many nodes in the graph through many short paths. Since enumerating all paths among the graph nodes would require exponential time in the size of the graph, Random-walk based techniques encode the structure of the network in the form of a transition matrix of a stochastic process from which the node significance can be inferred. For example, the well-known PageRank algorithm [10] associates an importance score to each node relying on random walks: the PageRank score of nodes of the graph is the stationary distribution of a random walk on the graph. Another example is the Katz index where the significance vector, $\mathbf{r}$, can be computed by solving $\mathbf{r} = \beta \mathbf{A}_G(\mathbf{r}+\mathbf{1})$, given a binary adjacency matrix, $\mathbf{A}_G$, and an attenuation factor, β [19,26].

2.2 Context-Sensitive Node Significance and Personalization

As also briefly discussed in the introduction, an early attempt to contextualize the PageRank scores was the *topic sensitive PageRank* [24] approach: here, the PageRank scores of the nodes were contextualized by varying the probabilities underlying the stochastic process in a way that reflects the nodes match to a given search topic.

In many applications, the context relevant to the recommendation is defined through a subset of the nodes (often referred to as the "*seed nodes*") in the graph and one needs to measure how *related* a given node in the graph is to the seed nodes. Naturally, how two nodes are related to each other reflects the topology of the graph G. Path-length based definitions, such as those proposed by [7, 16, 34, 42, 44, 45] help capture the *relatedness* of a pair of nodes solely based on the properties of the nodes and edges on the *shortest* path between the pair. [12] and [13] were among the first works which recognized that random-walks can also be used for measuring the *significance* of the graph nodes relative to a given *seed node set*, $S \subseteq V$: authors observed that, if one constructs a random-walk graph such that transition probabilities represent the separation between the seed nodes in the graph then the random-walk would spend more time on nodes that are closer to the seed nodes in S. More specifically, in [12] the authors proposed to construct a transition matrix, $\mathbf{T}_S$, where edges leading away from the seed nodes are weighted less than those edges leading towards the seed nodes. Consequently, in this scheme, a node with a high connectivity but separated from the seed nodes in too many hops may be less significant within the context of seed nodes. In fact, a node which both has a high connectivity and few hops away from the seed nodes will have the highest convergence score. In other words, the convergence probabilities of the nodes capture both (a) the separations among the seeds and the graph nodes and (b) the connectivity of the graph nodes relative to nodes in S.

Note that there are other random walk based approaches to personalization: [15, 20, 33, 37], for example, rely on a *hitting time* based approach, where

the hitting time is defined as the expected number of steps a random walk from the source vertex to the destination vertex will take, for query suggestion. Another approach to contextualizing PageRank scores is to use the PPR techniques [5, 14, 25, 39, 40] discussed in Section 1. One key advantage of this teleportation vector modification based approach over modifying the transition matrix, as in [12], is that the term β (in Equation 2) can be used to directly control the *degree of seeding (or personalization)* of the PPR score. In fact, these personalized random-walk and PageRank based measures of node significance have been shown to be highly effective in many prediction and recommendation applications [1, 29], where it is necessary to rank the nodes of the graph relative to a given set of (seed) nodes that represent the user's interest. Therefore, in this paper, we rely on this PPR based formulation of personalized, context-sensitive node significance.

2.2.1 Obtaining PageRank and Personalized PageRank Scores

Despite its effectiveness in prediction and recommendation applications, however, the use of personalized PageRank for large graphs is difficult due to its high computation cost.

One way to obtain PageRank and Personalized PageRank scores is to mathematically solve for $\mathbf{p}$ and $\boldsymbol{\pi}$ in Equations 1 and 2, respectively, mathematically [6]. Alternatively, PowerIteration [27] or using iterative approximations [14, 30], which explicitly simulate the dissemination of probability mass by repeatedly applying the transition process to an initial distribution $\boldsymbol{\pi}_0$ until a convergence criterion is satisfied. Recent advances on PPR computation include top-k and approximate personalized PageRank algorithms [2, 4, 14, 17, 21, 23, 41] and parallelized implementations on MapReduce or Pregel based systems [3, 32, 36, 38]. The FastRWR algorithm [41], for example partitions the graph into subgraphs and indexes partial intermediary solutions. Given a seed node set S then relevant intermediary solutions are combined to quickly solve for approximate PPR scores. Naturally, there is a trade-off between the number of partitions created for the input graph G and the accuracy: the higher the number of partitions, the faster the runtime execution (and smaller the memory requirement), but the higher the drop in accuracy. Recently, [31] proposed a fast Personalized PageRank algorithm: firstly the graph is decomposed into two parts: a *core* part which behaves like an expander graph and thus making the convergence of an iterative method very fast, and an *almost a tree* part. Authors suggest to rely on LU decomposition [18], which is a matrix factorization of the form A = LU, where L is a lower triangular matrix with unit diagonals and U is an upper triangular matrix. For the tree part and the result is used as a precondition which accelerates iterative solution on the rest of graph significantly. Finally they rely on the generalized minimal residual method to solve the problem for the entire graph. Also recently, [8] proposed a sublinear time Personalized PageRank(PPR) computation which identifies a set of nodes that, with high probability, contains all nodes with PageRank score at least Δ. Their approach

has two main parts: firstly they propose a local algorithm for approximating PPR values based on a careful simulation of random walks from the restart nodes; they also propose a multi-scale matrix sampling algorithm on the PPR result matrix. [28] proposed an alternative *locality-sensitive*, *re-use promoting*, *approximate Personalized PageRank* (LR-PPR) algorithm for efficiently computing the PPR values relying on the *localities* of the given seed nodes on the graph: The LR-PPR algorithm is (a) *locality sensitive* in the sense that it reduces the computational cost of the PPR computation process by focusing on the local neighborhoods of the seed nodes and is (b) *re-use promoting* in that instead of performing a *monolithic* computation for the given seed node set using the entire graph, LR-PPR divides the work into localities of the seeds and *caches* the intermediary results obtained during the computation. The robust personalized PageRank formulations proposed in this paper are also re-use promoting in the same sense, though they rely on a different mechanism for dividing the work relative to individual seed nodes to support caching and reuse of the intermediary results obtained during the computation.

3 Robust Personalized PageRank

Before presenting the proposed solutions for seed-bias elimination and *Robust personalized PageRank* (RPR) computation, let us briefly recall that the *personalized PageRank* scores of the nodes in G are captured using a vector $\boldsymbol{\pi}$,

$$\boldsymbol{\pi} = (1-\beta)\mathbf{T}_G\ \boldsymbol{\pi} + \beta\mathbf{s},$$

where $\mathbf{s}$ is a re-seeding vector, such that if $v_i \in S$, then $\mathbf{s}[i] = \frac{1}{\|S\|}$ and $\mathbf{s}[i] = 0$, otherwise. Note that the above stationary state equation can be rewritten as

$$(\mathbf{I} - (1-\beta)\mathbf{T}_G)\ \boldsymbol{\pi} = \beta\mathbf{s},$$

which (since the $\mathbf{s}$ is known *a priori*) corresponds to a set of linear equations, which can be solved for $\boldsymbol{\pi}$, either mathematically [6] or using iterative approximations [14,30]. Intuitively, since at each step, the random-walk has a non-zero probability of jumping back to the seed nodes from its current position in the graph, the nodes closer to the nodes in S will have larger stationary scores than they would have had if, during teleportation, the random walk jumped randomly in the entire graph.

As we discussed in Sections 1.1 and 1.2, PPR solutions cannot properly handle noise that may exist among the seed nodes and this may negatively affect the node rankings in situations where the seed set is (a) *incomplete* and/or (b) *imperfect/noisy*. Since the seed nodes are visited uniformly during the teleportation process, the conventional PPR scores rely on an inherent assumption that all seed nodes are equally good, at least in terms of being candidates for restarting points of the random-walk. In scenarios where we cannot assume that the seed set is noise free, however, the assumption that all seed nodes are equally good candidates for teleportation may not be valid.

Therefore, addressing the problem of noisy seeds through non-uniform teleportation to the seed nodes requires a mechanism to distinguish among the nodes in the seed set, S.

3.1 PPR-G: PPR with Global Seed Ranking

A *first (and as we see later, mostly ineffective) attempt* to differentiate among the seeds might be to consider the global properties of the nodes in the seed set. One relatively straightforward way to achieve this is to first measure the significance of the individual seed nodes in the overall graph, G, (using for example PageRank) and, then, modulate the teleportation rates onto the seed nodes based on the relative significance values of the seeds; i.e., instead of selecting a seed node uniformly randomly when transporting, we could select the seed in such a way that those seeds that are more significant in S have a higher likelihood of being chosen as the restart points.

In other words, we can compute the *PPR scores with global seed ranking* (also referred to as PPR-G scores) as follows: Given a graph, $G(V, E)$, and a seed node set S, let $\mathbf{p}$ be the PageRank scores computed by solving Equation 1. Then, the PPR-G scores, $\mathbf{g}$, are obtained by solving

$$\mathbf{g} = (1 - \beta)\mathbf{T}_G\ \mathbf{g} + \beta\mathbf{s_2},$$

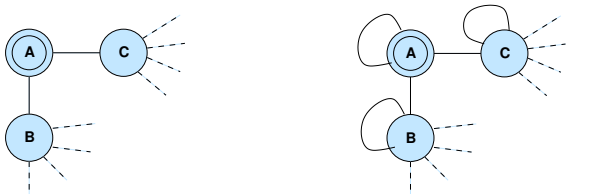
where $\mathbf{s_2}$ is a re-seeding vector, such that for each $v_i \notin S$, $\mathbf{s_2}[i] = 0$, and for each $v_i \in S$, $\mathbf{s_2}[i] = \frac{\mathbf{p}[i]}{\sum_{v_j \in S} \mathbf{p}[j]}$.

As we later see in Section 5, however, in many cases, simply modifying the teleportation rates of the seed nodes based on the global significances of the nodes in the seed set does not properly eliminate seed-bias.

3.2 RPR-1: Teleportation-Discounted PPR

In the PPR formulation, the seed and non-seed nodes have different contributors to their final scores. *For the non-seed nodes*, the only contributor to their PPR scores is the number of times they are visited during the regular random-walk process. On the other hand, *for the seed nodes*, both (a) the number of times they are visited during the regular random-walk process and (b) the number of times they are selected as a teleportation destination for random-walk restart contribute to the PPR scores; we refer to these as the *random-walk contribution* (`rw-PPR`) and *teleportation contribution* (`t-PPR`), respectively. As described above, for non-seed nodes, the value of `t-PPR` score is 0.0.

Our first observation is that the `t-PPR` score of a seed node is exactly $\beta/|S|$ for each seed and, thus, a seed node will have at least $\beta/|S|$ overall PPR score, even if it is an outlier in the overall context defined by the seed set, S. Therefore, the first proposal to increase robustness of the PPR scores against noise in the the seed set S is to discount the teleportation contributions from the PPR scores.



(a) A is under-accounted relative to B and C if we discount teleportations

(b) A is not under-accounted when teleportation scores are discounted

Fig. 2 (a) Discounting teleportation scores would cause under-accounting of A relative to B and C. (b) If we add self-loops to nodes, when A is selected as a re-start point, A will not be under-accounted relative to its neighbors

3.2.1 Teleportation-Discounting

Based on this observation, we define *teleportation-discounted PPR* scores (also referred to as RPR-1 scores) as follows: Given a graph, G, and a seed node set S, let $\boldsymbol{\pi}$ be the PPR scores as defined earlier:

- for each $v_i \notin S$, the corresponding RPR-1 score, $\boldsymbol{\rho_1}[i]$, is defined as $\boldsymbol{\rho_1}[i] = \frac{\boldsymbol{\pi}[i]}{1-\beta}$;
- in contrast, for each $v_i \in S$, the corresponding RPR-1 score, $\boldsymbol{\rho_1}[i]$, is defined as $\boldsymbol{\rho_1}[i] = \frac{\boldsymbol{\pi}[i]-\frac{\beta}{|S|}}{1-\beta}$.

PPR-1 scores as defined above do not alter the relative ordering of the non-seed nodes; instead (as discussed above) they aim to allow us to discover the significance of the seed nodes themselves relative to the non-seed nodes within the overall context defined by the seed set, S.

3.2.2 Preventing Under-Accounting of Seeds

One potential problem with the teleportation-discounting is that this time the seed nodes may in fact be *under-accounted*: neighbors of a seed node may get higher amounts of random-walk traffic (i.e, `rw-PPR`) than the seed node itself, simply because once you teleport to a seed node, you need to visit one of its neighbors but not vice versa. In order to prevent this *under-accounting*, we modify the input graph G by inserting a self-loop to each node in the network (Figure 2). In the resulting graph G', every node is a neighbor of itself and thus a seed node, v, will not get a lesser amount of random-walk traffic than its neighbors when v is selected as a re-start point.

3.3 RPR-2: Seed-Set Maximal PPR

RPR-1 reduces the teleportation bias, but the seed nodes are still being teleported to with the same, undifferentiated rate. The PPR-G scheme we discussed in Section 3.1 aimed to address this, but the key disadvantage of the PPR-G score is that the *a priori* significance of the individual seed nodes (used

for modulating the teleportation rates) is decided based on the global structure of the graph, without considering the context provided by the other seed nodes.

An alternative to fixing the teleportation significance of the individual seed nodes *a priori*, without considering the context provided by the other seed nodes (as in the PPR-G scheme we discussed in Section 3.1), is to discover the teleportation significance of the individual seed nodes, relying on a novel *seed-set maximality* principle that would tie the teleportation rates of the seeds to their contributions to the overall personalized PageRank score of the seed set.

Principle 1 (Seed-Set Maximal PPR Scores) Let $G(V, E)$ be a graph and let $S \subseteq V$ be a set of seed nodes. Given an overall teleportation rate β, the re-seeding/re-start vector, $\mathbf{s}$, should be selected such that the overall PageRank scores of the nodes in S will be maximal.

Based on this principle, the *seed-set maximal PPR scores* (also referred to as the RPR-2 scores) are computed as follows: Given a graph $G(V, E)$, a teleportation probability, β, and a seed set, S, the re-start vector $\mathbf{s}$ should be such that

$$\boldsymbol{\rho_2} = (1-\beta)\mathbf{T}_G\ \boldsymbol{\rho_2} + \beta\mathbf{s},$$

$$\sum_{v_i \in V} \mathbf{s}[i] = 1, \forall_{v_i \in V} 0 \leq \mathbf{s}[i] \leq 1, \sum_{v_i \in V} \boldsymbol{\rho_2}[i] = 1, 0 \leq \boldsymbol{\rho_2}[i] \leq 1,$$

and the following term is *maximized*:

$$seed\_set\_significance = \sum_{v_i \in S} \boldsymbol{\rho_2}[i].$$

◇

Intuitively, this principle requires that the seed set as a whole must score highly, but does not require that the individual seed nodes themselves have high scores. As we see in Section 5.4, this provides an effective mechanism through which the impact of noisy seed nodes in the seed set are discounted.

3.4 RPR-3: Teleportation-Discounted, Seed-Set Maximal PPR

As we have seen in Section 3.2, one disadvantage of the use of standard PPR scores is that the teleportation contribution `t-PPR` score of a seed node (which is exactly $\frac{\beta}{|S|}$ for each seed node) may not capture how significant the node is within the context defined by the entire seed set S. This is especially true in the case of RPR-2 scores, where the set, S_{crit}, of seed nodes selected for re-start is very small. In fact, when S_{crit} is singleton (as most likely), the only seed node in S_{crit} will have at least β overall RPR-2 score. Therefore, when computing the *teleportation-discounted, seed-set maximal PPR scores* (or RPR-3 scores, also denoted as $\boldsymbol{\rho_3}$), we replace the use of $\boldsymbol{\pi_i}$ vectors for each $v_i \in S$, with the RPR-1 vectors $\boldsymbol{\rho_{1,i}}$ as introduced in Section 3.2.

4 Efficient Computation of Seed-Set Maximal PPR

In the previous section, we proposed a *seed-set maximality* principle that ties the teleportation rates of the seeds to their contributions to the overall personalized PageRank score of the seed set: intuitively, this principle requires that the seed set as a whole must score highly, whereas it allows individual seed nodes themselves to have low scores. As we experimentally show in Section 5.6, this principle helps discover the teleportation significance of the individual seed nodes, thereby differentiating among them, without having to set them *a priori*.

To recap, given a graph, $G(V, E)$, and a seed set, $S \subseteq V$, the *seed-set maximal PPR scores*, $\boldsymbol{\rho_2}$, are obtained using a re-start vector $\mathbf{s}$ such that the overall PageRank scores of the nodes in S will be maximal. More specifically, $\mathbf{s}$ should be selected such that the term $seed\_set\_significance = \sum_{v_i \in S} \boldsymbol{\rho_2}[i]$ is *maximized*, where

$$\boldsymbol{\rho_2} = (1-\beta)\mathbf{T}_G\ \boldsymbol{\rho_2} + \beta\mathbf{s},$$

$$\sum_{v_i \in V} \mathbf{s}[i] = 1, \forall_{v_i \in V} 0 \leq \mathbf{s}[i] \leq 1, \sum_{v_i \in V} \boldsymbol{\rho_2}[i] = 1, 0 \leq \boldsymbol{\rho_2}[i] \leq 1.$$

Naturally, one possible concern with this new formulation is that it might potentially increase the computation cost. In this section, we show that RPR-2 scores can be cheaply computed.

4.1 Seed only Re-Starts for RPR-2

According to the above formulation of seed-set maximal PPR scores, the seed nodes in S are not necessarily the only targets for re-starts. However, we can show that, for any traversal that re-starts at a non-seed node, there is a traversal that starts only at the seed nodes, but has a higher seed node traversal rate.

Proof 1 (Seed Only Re-Starts) *Given a graph $G(V, E)$, let $\mathbf{s}$ be an optimal re-start vector, such that $\exists v_i \notin S$ and $\mathbf{s}[i] > 0$. Now consider the traversals that start only from a $v_i \notin S$ and let $\boldsymbol{\alpha_i}$ be a vector describing the average portion of time the random walk spends on the graph nodes in V before the next teleportation. We can then define two quantities (that add up to 1.0):*

$$seed\_ratio_i = \sum_{v_j \in S} \boldsymbol{\alpha_i}[j], \text{ and } non\_seed\_ratio_i = \sum_{v_j \notin S} \boldsymbol{\alpha_i}[j].$$

Note that, since the traversal starts at a non-seed node, we have $non\_seed\_ratio_i > 0$; moreover, we can split this term into two, $non\_seed\_ratio_{i,before}$ and $non\_seed\_ratio_{i,after}$; i.e., the amount of time spent on non-seed nodes before and after a seed node is met during the random-walk, respectively. Let us also define a vector, $\mathbf{f_i}$, where for a given $v_j \in S$, the value

of $\mathbf{f_i}[j]$ *is the likelihood of* v_j *being the first seed met during the random-walk starting at node* v_i*.*

Now consider an alternative re-start vector $\boldsymbol{\sigma}$ *such that (a)* $\boldsymbol{\sigma}[i] = 0$*, (b)* $\forall_{v_j \notin S}$*, if* $j \neq i$*, then* $\boldsymbol{\sigma}[j] = \mathbf{s}[j]$*, and (c)* $\forall_{v_j \in S}$*,* $\boldsymbol{\sigma}[j] = \mathbf{s}[j] + \mathbf{s}[i]\ \mathbf{f_i}[j]$*. It is easy to see that the random-walks resulting when using the restart vector* $\boldsymbol{\sigma}$ *are similar to the random-walks resulting when using* $\mathbf{s}$*, except that the value of* $non\_seed\_ratio_{i,before}$ *is equal to* 0*. This means some of this time will be spent on the seed nodes contradicting the initial premise that* $\mathbf{s}$ *was an optimal re-start vector, maximizing the total amount of time spent on seed nodes.* □

Based on the above proof, we can reformulate the equation set for seed-set maximal PPR scores in a way that limits the transportation targets, as follows:

$$\boldsymbol{\rho_2} = (1-\beta)\mathbf{T}_G\ \boldsymbol{\rho_2} + \beta\mathbf{s}, \quad \sum_{v_i \in V} \boldsymbol{\rho_2}[i] = 1, \quad 0 \leq \boldsymbol{\rho_2}[i] \leq 1,$$
$$\sum_{v_i \in S} \mathbf{s}[i] = 1, \quad \forall_{v_i \in s} 0 \leq \mathbf{s}[i] \leq 1, \quad \forall_{v_i \notin s} \mathbf{s}[i] = 0,$$

and $\quad seed\_set\_significance = \sum_{v_i \in S} \boldsymbol{\rho_2}[i] \quad$ is *maximum.*

4.2 Constraining the Size of the Re-Start Set

We can further show that in practice the restart set, S_{crit}, is a singleton; i.e., with very high likelihood, $|S_{crit}| = 1$.

Proof 2 (Singleton Re-Start Set) *Given a graph* $G(V, E)$*, let* $\mathbf{s}$ *be an optimal re-start vector, such that* $\exists v_i, v_j \in S$ *and* $\mathbf{s}[i] > 0$ *and* $\mathbf{s}[j] > 0$*. Now consider all the traversals that start only from* v_i *and let* $\boldsymbol{\alpha_i}$ *be a vector describing the average portion of time the random walk that starts at* v_i *spends on the nodes in* V *before the next teleportation. Similarly, let* $\boldsymbol{\alpha_j}$ *be a vector describing the average portion of time a random walk that starts at* v_j *spends on the nodes in* V *before the next teleportation. Given* $\boldsymbol{\alpha_i}$ *and* $\boldsymbol{\alpha_j}$*, let us define two quantities,*

$$seed\_ratio_i = \sum_{v_k \in S} \boldsymbol{\alpha_i}[k] \ \text{ and } \ seed\_ratio_j = \sum_{v_k \in S} \boldsymbol{\alpha_j}[k],$$

and let us assume that $seed\_ratio_i > seed\_ratio_j$ *(a similar argument holds when* $seed\_ratio_j > seed\_ratio_i$*).*

Now consider an alternative re-start vector $\boldsymbol{\sigma}$ *such that (a)* $\forall_{v_k \notin \{v_i, v_j\}}$*,* $\boldsymbol{\sigma}[k] = \mathbf{s}[k]$*, (b)* $\boldsymbol{\sigma}[j] = 0$*, and (c)* $\boldsymbol{\sigma}[i] = \mathbf{s}[i] + \mathbf{s}[j]$*. It is easy to see that in the random-walks resulting when using the restart vector* $\boldsymbol{\sigma}$ *are similar to the random-walks resulting when using* $\mathbf{s}$*, except that all transportations to* v_j *are replaced with transportations to* v_i *(with the higher seed_ratio value among the two); thus, overall, more time will be spent on seed nodes when using* $\boldsymbol{\sigma}$ *instead of* $\mathbf{s}$*. It follows that when* $seed\_ratio_i > seed\_ratio_j$*, an optimal re-start vector cannot contain both* v_i *and* v_j*, contradicting the initial premise that* $\mathbf{s}$ *is an optimal re-start vector, such that* $\mathbf{s}[i] > 0$ *and* $\mathbf{s}[j] > 0$*.* □

In other words, since in practice it is highly unlikely that *seed_ratio* values will be equivalent for different seed nodes, the subset S_{crit} of S is likely to contain the one and only node, v_i, which has the highest $seed\_ratio_i$.

4.3 Efficient and Re-Use Promoting Computation of RPR-2

Given a graph, $G(V, E)$, a seed set, S, and a teleportation probability, β, one way to obtain the RPR-2 scores is to solve the linear optimization problem

$$\texttt{maximize} \quad \sum_{v_i \in S} \boldsymbol{\rho_2}[i]$$

subject to the constraints

$$\boldsymbol{\rho_2} = (1-\beta)\mathbf{T}_G\ \boldsymbol{\rho_2} + \beta\mathbf{s}, \sum_{v_i \in V} \boldsymbol{\rho_2}[i] = 1, \quad 0 \leq \boldsymbol{\rho_2}[i] \leq 1,$$

$$\sum_{v_i \in S} \mathbf{s}[i] = 1, \quad \forall_{v_i \in s} 0 \leq \mathbf{s}[i] \leq 1, \quad \forall_{v_i \notin s} \mathbf{s}[i] = 0.$$

While there are many efficient linear solvers that one can use to obtain a solution to the above optimization problem, there are two issues to consider: (a) in general, solving the optimization problem is more expensive than simply solving the linear equations for a given re-start vector $\mathbf{s}$, and (b) when the seed set S changes (even if the change is small, say one new seed node is considered or one of the seed nodes is dropped) the linear optimization problem needs to be reformulated and solved anew. In this subsection, we note that we can avoid treating the problem as an optimization problem (thereby reducing its cost) and, in the meantime, also support the re-use of existing solutions, by converting the problem into a set of single-seed PPR computations.

4.3.1 Converting the Problem into a Set of Linear Equations

Given a graph, $G(V, E)$, a seed set, S, and an overall teleportation probability, β, we reformulate the problem (relying on the observation in Section 4.2) as follows:

- *Step 1.* for each $v_i \in S$, solve the linear equation $\boldsymbol{\pi_i} = (1-\beta)\mathbf{T}_G\ \boldsymbol{\pi_i} + \beta\mathbf{s_i}$, where $\mathbf{s_i}$ is a re-start vector such that $\mathbf{s_i}[i] = 1$ and $\forall_{j \neq i}\ \mathbf{s_i}[j] = 0$;
- *Step 2.* Next, for each $v_i \in S$, compute $\Pi(v_i) = \sum_{v_j \in S} \boldsymbol{\pi_i}[j]$;
- *Step 3.* Let S_{crit} be the (small) subset of S, where $S_{crit} = \{v | v = argmax_{v_i}(\Pi(v_i))\}$;
- *Step 4.* If S_{crit} is singleton (i.e., $S_{crit} = \{v_i\}$) then $\boldsymbol{\pi_i}$ gives the RPR-2 scores; i.e., $\rho_2 = \boldsymbol{\pi_i}$; else (i.e., if S_{crit} is not a singleton), then $\rho_2 = \frac{1}{|S_{crit}|} \sum_{v_i \in S_{crit}} \boldsymbol{\pi_i}$.

Note that, since in general the seed set S includes relatively few nodes, the above formulation requires the solution of a small number of single-seed PPR problems. This is especially advantageous when G is large as we can leverage any of the highly effective approximation algorithms [2, 4, 14, 17, 21, 23, 41] or

parallelized implementations [3, 32] for computing these PPR scores. Most importantly, the first step of the algorithm (where we solve a linear equation independently for each seed node) can be trivially parallelized by assigning each node to a different computation unit.

4.3.2 Solution Re-Use for Incremental Computation

Given a graph, $G(V, E)$, a seed set, S, and an overall teleportation probability, β, assume that we have already computed $\boldsymbol{\pi_i}$ and $\Pi(v_i)$ for all $v_i \in S$. Let S_{new} be a new seed set, let $\Delta S^+ = S_{new} \setminus S$ denote the new nodes in the seed set and $\Delta S^- = S \setminus S_{new}$ denote the set of nodes dropped from the seed set. We can incrementally compute the RPR-2 scores as follows:

- *Step 1.* For each $v_i \in \Delta S^+$, solve the linear equation $\boldsymbol{\pi_i} = (1-\beta)\mathbf{T}_G\ \boldsymbol{\pi_i} + \beta\mathbf{s_i}$, where $\mathbf{s_i}$ is a re-start vector such that $\mathbf{s_i}[i] = 1$ and $\forall_{j \neq i}\ \mathbf{s_i}[j] = 0$;
- *Step 2.* Next, for each $v_i \in \Delta S^+$, compute $\Pi_{new}(v_i) = \sum_{v_j \in S_{new}} \boldsymbol{\pi_i}[j]$;
- *Step 3.* Also, for each $v_i \in S_{new} \cap S$, compute $\Pi_{new}(v_i) = \Pi_{new}(v_i) + \sum_{v_j \in \Delta S^+} \boldsymbol{\pi_i}[j] - \sum_{v_j \in \Delta S^-} \boldsymbol{\pi_i}[j]$;
- *Step 4.* Given these, once again, let S_{crit} be the (small) subset of S, where $S_{crit,new} = \{v | v = argmax_{v_i}\ (\Pi_{new}(v_i))\}$, and compute the ρ_2 scores as $\rho_2 = \frac{1}{|S_{crit,new}|} \sum_{v_i \in S_{crit,new}} \boldsymbol{\pi_i}$.

It is easy to see that, when ΔS^+ and ΔS^- are small, the RPR-2 computations can be done very fast (if necessary, leveraging approximation algorithms [2,4,14,17,21,23,41] and/or parallel implementations [3,32] for computing the new PPR scores). Once again, the first step of the algorithm (where we solve a linear equation independently for each new seed node) can be trivially parallelized by assigning each node to a different computation unit.

4.4 Multi-Query Robust Personalized PageRank

Note that the above formulation also leads to efficient computations of the RPR-2 scores in systems with large PPR query throughputs – i.e., in systems where PPR based recommendations are continuously executed for 100s or 1000s of users, which may share certain interests while also having their personal preferences.

Consider a graph, $G(V, E)$, a set, $\mathcal{S} = \{S_1, S_2, \ldots, S_s\}$, of seed sets, and a teleportation probability, β. We refer to the task of identifying a (robust-) personalized PageRank vector, $\boldsymbol{\rho}_j$ for each $s_j \in \mathcal{S}$ as the multi-query (robust-) personalized PageRank problem. The naive way of executing a multi-query robust personalized PageRank (m-RPR) task would be to go over the set $\mathcal{S}$ and process each RPR query separately, from scratch. It is, however, easy to see that this would lead to significant amount of redundant work, especially if there are seed sets with large degrees of overlaps (i.e., users who share interests).

If we are computing RPR-2 scores, on the other hand, a clearly better alternative is to leverage caching opportunities discussed in the previous section by caching $\boldsymbol{\pi_i}$ for all v_i's that have been considered and re-using them, from the cache, for each subsequent $S_j \in S$, where $v_i \in S_j$: More specifically, for each $S_j \in S$,

- *Step 1.* For each $v_i \in S_j$ but $\boldsymbol{\pi_i} \notin cache$, solve the linear equation $\boldsymbol{\pi_i} = (1-\beta)\mathbf{T}_G\ \boldsymbol{\pi_i} + \beta \mathbf{s_i}$, where $\mathbf{s_i}$ is a re-start vector such that $\mathbf{s_i}[i] = 1$ and $\forall_{j \neq i}\ \mathbf{s_i}[j] = 0$;
- *Step 2.* For each $v_i \in S_j$, compute $\Pi_j(v_i) = \sum_{v_k \in S} \boldsymbol{\pi_i}[k]$;
- *Step 3.* Given these, let S_{crit} be the (small) subset of S, where $S_{crit,j} = \{v | v = argmax_{v_i}(\Pi_j(v_i))\}$, and compute the ρ_2 scores as $\rho_2 = \frac{1}{|S_{crit,j}|} \sum_{v_i \in S_{crit}} \boldsymbol{\pi_i}$.

This, however, may have a key disadvantage: If the cache is limited, it is possible that some $\boldsymbol{\pi_i}$ will be ejected from the cache, only to be recomputed later for a subsequent seed set that also contains it. Therefore, a potentially more advantageous approach is to first identify clusters of seed sets in $\mathcal{S}$ that have large overlaps and process each cluster together to ensure that the cache is maximally utilized. Note that the larger the overlaps are, the smaller the amount of redundant work and the larger the expected gains.

4.5 Approximate RPR with Fast Random Walk

At their core, all RPR formulations presented in Section 3 rely on random walk with restart (RWR). However, it is well known that a straightforward implementation of RWR does not scale for large graphs since the naive implementation includes finding the inverse of a very large matrix.

As discussed in Section 4.4, there are various solutions proposed to address this challenge. One such solution is to pre-compute the matrix inverse for the whole graph and re-use it for each query. However, since matrix inverses tend to be dense, the matrix inverse of a large graph may require large storage. A second possible approach is to compute matrix inverses only for parts of the graph and combine these matrix inverses, on-demand, during query processing.

An example of this approach is the *fast random walk (FRW)* scheme presented in [41]. FRW partitions the graph into subgraphs and indexes partial intermediary solutions. Given a seed node set S the relevant intermediary solutions are combined to quickly solve for approximate PPR scores. Let us consider the term

$$\mathbf{r} = (1-\beta)\mathbf{T}_G\mathbf{r} + \beta\mathbf{s},$$

which forms the core of the PPR and the various RPR measures. [41] rewrites $\mathbf{T}_G$ as

$$\mathbf{T}_g = \mathbf{B} + \mathbf{X},$$

where $\mathbf{B}$ is a block diagonal matrix obtained by partitioning, off-line, the graph G into t partitions and keeping only in-partition links and $\mathbf{X}$ contains all cross-partition links. Then, again during the off-line phase, for each partition, l, the

matrix inverse, $\mathbf{Q}_l^{-1}$, is computed for the corresponding block, $\mathbf{B}_l$:

$$\mathbf{Q}_l^{-1} = (I - (1-\beta)\mathbf{B}_l)^{-1}.$$

Also, during the off-line phase, a low-rank approximate decomposition of $\mathbf{X}$ is obtained; i.e., for a small decomposition rank, r,

$$\mathbf{X} \sim \mathbf{U_r}\mathbf{\Sigma_r}\mathbf{V_r}.$$

This low rank approximation is then used for obtaining, still in the off-line phase, a small $r \times r$ matrix, $\mathbf{\Lambda}$, as

$$\mathbf{\Lambda} = \left(\mathbf{\Sigma_r}^{-1} - (1-\beta)\mathbf{V_r}\mathbf{Q}^{-1}\mathbf{U_r}\right)^{-1},$$

where $\mathbf{Q}^{-1}$ is a block-diagonal matrix constructed by combining $\mathbf{Q}_1^{-1}$ through $\mathbf{Q}_t^{-1}$ at the diagonal.

The key idea of [41] is that, relying on the Sherman-Morisson lemma [35], one can solve for $\mathbf{r}$, during the on-line phase, by computing

$$\mathbf{r} = \beta(\mathbf{Q}^{-1}\mathbf{s} + (1-\beta)\mathbf{Q}^{-1}\mathbf{U_r}\mathbf{\Lambda}\mathbf{V_r}\mathbf{Q}^{-1}\mathbf{s}).$$

Naturally, there is a trade-off between the number of partitions created for the input graph G and the accuracy: The higher the number of partitions, the faster the run-time execution and smaller the memory requirement. This is because, (assuming that the partitions are roughly same size),

- the number of non-zero entries in $\mathbf{Q}^{-1}$ is $\sim t \times \left(\frac{|V|}{t}\right)^2 = \frac{|V|^2}{t}$
- U_r and V_r are $|V| \times r$ and $r \times |V|$ matrices, respectively, and
- $\mathbf{\Lambda}$ is a small $r \times r$ matrix.

In this paper, we note that this formulation is especially suitable for the seed-set maximal RPR (i.e., RPR-2) computation, since (as we discussed in Section 4.3.1), given a seed set S, we can convert the seed-set maximal RPR problem into a set of $|S|$ linear equations, where for each $v_i \in S$, we need to solve $\boldsymbol{\pi}_\mathbf{i} = (1-\beta)\mathbf{T}_G\,\boldsymbol{\pi}_\mathbf{i} + \beta\mathbf{s_i}$, where $\mathbf{s_i}$ is a re-start vector such that $\mathbf{s_i}[i] = 1$ and $\forall_{j \neq i}\ \mathbf{s_i}[j] = 0$. Once the block-diagonal $\mathbf{Q}^{-1}$, $\mathbf{\Lambda}$, $\mathbf{U_r}$, and $\mathbf{V_r}$ are obtained during the on-line phase, we can simultaneously obtain (approximate) solutions for the $|S|$ linear equations, by

$$\mathbf{P} = \beta(\mathbf{Q}^{-1}\mathfrak{S} + (1-\beta)\mathbf{Q}^{-1}\mathbf{U_r}\mathbf{\Lambda}\mathbf{V_r}\mathbf{Q}^{-1}\mathfrak{S}),$$

where

- each column of $\mathfrak{S}$ is a re-start vector, $\mathbf{s_i}$, corresponding to the node $v_i \in S$ and
- each column of $\mathbf{P}$ is the solution vector, $\boldsymbol{\pi}_\mathbf{i}$, for the linear equation corresponding to node $v_i \in S$.

As we experimentally show in Section 5.13, the robust PPR measures proposed in this paper can effectively leverage this block-diagonal solution to significantly reduce memory needed to maintain the inverse matrices. Thanks to the possibility of converting (as discussed in Section 4.3.1) a given seed-set maximal PPR computation task from an expensive optimization problem into a set of linear equations of the form $\boldsymbol{\pi_i} = (1-\beta)\mathbf{T}_G\ \boldsymbol{\pi_i}+\beta\mathbf{s_i}$, the proposed RPR measures can also easily leverage other random-walks with restart approximation techniques, such as [31]: one only needs to replace the computation of $\boldsymbol{\pi_i}$ with the selected approximate PPR technique to obtain an approximation of $\boldsymbol{\pi_i}$.

Most importantly, though, the experimental results show that *teleportation-discounting* not only increases robustness of PPR scores against noise in the seed set S but also significantly improves robustness against noise introduced due to approximate computation. This is because, the graph-partitioning (and block diagonalization) scheme, which essentially distorts the graph structure due to the low-rank approximation of cross-partition edges and teleportations cause frequent cross-partition jumps. *Teleportation-discounting* (does not completely eliminate, but) carefully pulls down the overall contribution of the teleportation process and, consequently, provides robustness against noise introduced during fast random walk computation.

5 Experimental Evaluations

In this section, we present experimental results assessing the effectiveness of the proposed alternative robust Personalized Page Rank (RPR) schemes in handling noise in the seed set. We ran the experiments on a quad-core Intel(R) Core(TM)i5-2400 CPU @ 3.10GHz machine with 8.00GB RAM. All codes are implemented in Matlab and run using Matlab 7.11.0 (2010b).

5.1 Data Sets

For comparing the different RPR alternatives' performances against conventional PPR, we used the IMDB and MovieLens datasets available from [46,47]. This dataset contains metadata (e.g. actors, directors) about 1681 movies, a total of 100K ratings (between 1, for "dislike" (↓), to 5, for "like" (↑)) provided by 943 users and big dataset containing about one million ratings relationship provided by 6040 users. From this graph, we have constructed three data and social graphs, with distinct semantics and topological properties:

- *Metadata Graph:* In the metadata graph, nodes represent the data elements (such as movies) and the edges represent relationships between these data elements (such as an actor playing in a movie). The data graph contains 1272 nodes and 60K relationship edges (with average node degree of ∼47).
- *User-Movie (UM) Graph:* In the UM graph, nodes represent users and movies. There is an edge between a user-movie pair if the user has watched

Parameter	Range	Default Value
# of seeds	{10, 40}	10
% of noise in the seed set	{0%, 10%, 20%}	10%

Table 3 Personalized PageRank evaluation parameters

the movie (indicated by the existence of a rating). This graph has 1682 movie nodes, 943 user nodes, and 200K directional (user to movie) edges (with average node degree of ∼76).
– *Ratings Graph:* The ratings graph consists of the same nodes and edges as the user-movie (UM) graph. However, each ratings edge $u_i \rightarrow m_j$ has an associated numeric weight between 1 and 5, reflecting user u_i's preference for movie m_j.

In addition to these, in order to help observe the impact of the data size, we have also considered larger versions of the UM and ratings graphs, with $\sim 10K$ nodes and $\sim 1M$ edges:

– *User-Movie II (UM-II) Graph:* In the UM2 graph, once again, nodes represent users and movies. There is an edge between a user-movie pair if the user has watched the movie (indicated by the existence of a rating). This graph has 3952 movie nodes, 6040 user nodes, and 1 million directional (user to movie) edges. In this graph, the average node degree is around 200.
– *Ratings-II Graph:* This ratings graph consists of the same nodes and edges as the UM-II graph. However, each ratings edge $u_i \rightarrow m_j$ has an associated numeric weight between 1 and 5, reflecting user u_i's preference for movie m_j.

Note that the *metadata graph* captures no knowledge about users and their preferences. The *user-movie (UM,UM-II) graphs*, which can be seen as a rich social network of users and movies, capture which users judged (rated) which movies, but does not capture the value of the rating. The *ratings* and *ratings-II* graphs also capture users' declared preferences in the form of edge weights. In these experiments, we set the default value of β to 0.15 as is commonly done. Results for other β values are similar.

5.2 Evaluation Strategies

5.2.1 Effectiveness

In order to measure effectiveness of the scores computed for ranking movies, we rely on the following criteria:

– **Relevance:** If we select a subset of a user's ↑-rated movies as seeds, the scores should be such that the user's remaining ↑-rated movies should rank well, whereas user's ↓-rated movies should rank poorly.

– **Robustness:** Moreover, if the scores are robust, then even if the seed set contains some small number of ↓-rated movies, this should not negatively affect the rankings significantly.

As shown in Table 3, we consider seed sets of different sizes (10 and 40). For each configuration, we select those users who have sufficient ↑-rated movies (e.g., if the target seed set size is 10, then we pick those users with at least 10 ↑-rated movies):

– For the UM and Ratings graphs, there are 313 users for seed set size 10 and 64 users for seed set size 40.
– For the UM-II and Ratings-II graphs, there are 1959 users for seed set size 10 and 516 users for seed set size 40.
– For the Metadata graph, there are 139 users when the seed set size is 10 and 42 users when the seed set size is 40.

For each user, we created random seed sets with different degrees of noise (see Table 3 for the experiment parameters). Each seed set consists of a number of movies rated "↑" by the user (for measuring relevance) and a smaller number of movies rated "↓" by the same user. The "↓" movies included in the seed set act as noise (and serve for measuring robustness). For each configuration, we have considered 10 different (randomly picked) seed sets. For each seed set, we treated the rest of the movie ratings by this user as the **ground truth** to help measure the following effectiveness criteria based on the transition probabilities implied by the underlying graph:

– *Recommendation Effectiveness – Average rank for non-seed ↑-rated movies ($AvgRank_{(N\uparrow)}$):* Movies that we know (from the ground truth) that the user would like, but are not included in the seed set are *expected to rank well; i.e., have small average rank values.*
– *Seed Differentiation Power – Average rank for ↓-rated seed movies ($AvgRank_{(S\downarrow)}$):* "Noise" in the seed set (i.e., movies we know from the ground truth that the user does not like, but nevertheless included in the seed set) are *expected to have large average rank values*, even though they are in the seed set. Note that, since a well ranked ↓-rated seed would imply that *the movies neighboring this noisy seed* would also be well ranked, the average rankings of the seed nodes help us (indirectly) quantify the impact of noise on the neighbors of the seeds.
– *Seed-Bias Elimination – Average rank for ↑-rated seed movies ($AvgRank_{(S\uparrow)}$):* Movies that the user likes and are included in the seed set are *expected to rank well and have small average rank values.*

5.2.2 Efficiency

In addition to the above consideration of effectiveness, we also consider the efficiency. In particular, we computed matrix algebra based formulations of PPR and RPR using Matlab. The RPR-2 and RPR-3 schemes, which seek seed-maximal solutions, are implemented by converting the maximization problem

to a set of linear equations (as discussed in Sections 4.1 through 4.3): these enable the same evaluation mechanism for PPR, PPR-G, and RPR-1 to be applicable also for RPR-2 and RPR-3, making the accuracy and execution time comparisons straightforward.

5.3 Approximation Schemes

As we discussed in Section 4.5, computation of PPR and RPR can be expensive for large graphs and alternative approximation algorithms (such as block diagonalization/partitioning based techniques [41]) may be used to reduce computational cost. To assess the impact of such approximation on the effectiveness and efficiency of RPR schemes, we also implemented partitioning-based fast random walk based PPR and RPR schemes as discussed in Section 4.5. For partitioning the graph, we used the multi-level based graph partitioner of the METIS tool set [48]. In the experiments reported in this section, we considered different graph partitions, t, and low-rank approximation, r for different graphs (as a function of the graph size). In particular, for the meta-data graph, $t = 12$, $r = 12$; for UM and Ratings graphs, $t = 26$, $r = 26$; and for the UM-II and Ratings-II graphs, $t = 99$, $r = 99$. It is important to note that these are intentionally low values, to help us observe the performance of robust personalized PageRank (RPR) schemes under significant quality degradations.

5.4 Effectiveness Evaluations

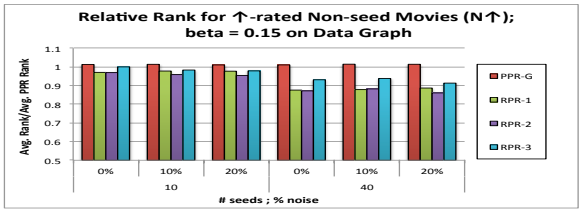
Figures 3 through 8 compare and evaluate the effectiveness of the different ranking algorithms described in this paper, based on the three effectiveness criteria $C = \{N\uparrow, S\uparrow, S\downarrow\}$ listed above. For each of these three criteria, we compare the rankings returned by algorithm A to the rankings returned by the conventional PPR (i.e., PPR with uniform teleportation probabilities) using the measure

$$relative\_rank(A, C) = (AvgRank_C \text{ by } A)/(AvgRank_C \text{ by } PPR).$$

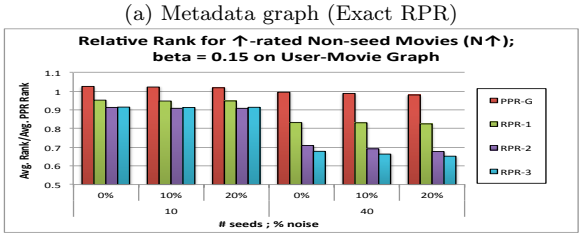
This measure helps us observe how algorithm A handles seed noise relative to PPR with no seed differentiation.

5.4.1 Recommendation Effectiveness

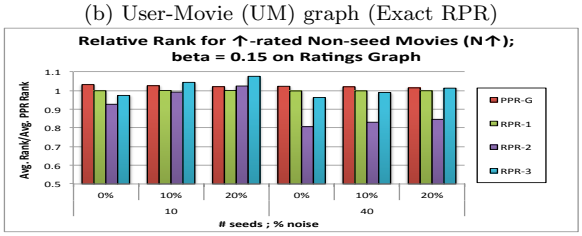
Firstly, let us consider Figure 3 which compares the average user rankings of $\uparrow$-rated movies that were not included in the seed set. Since the primary goal of a movie ranking system is to locate non-seed movies that the user would enjoy ($N\uparrow$) and rank them earlier than the other movies, *the smaller the value of the* $relative\_rank(A, N\uparrow)$*, the better would be the recommendations returned by the algorithm* A.



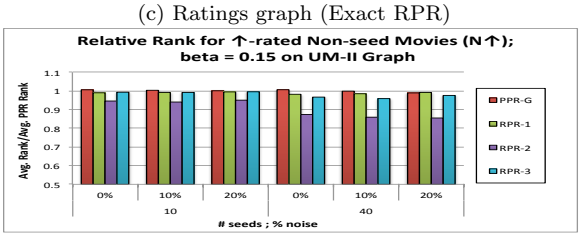
(a) Metadata graph (Exact RPR)



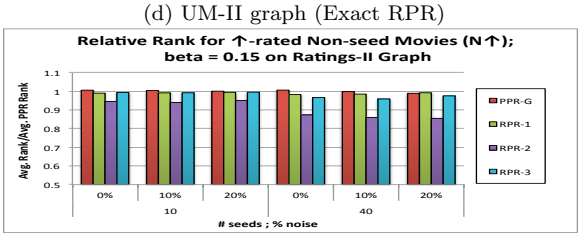
(b) User-Movie (UM) graph (Exact RPR)



(c) Ratings graph (Exact RPR)



(d) UM-II graph (Exact RPR)



(e) Ratings-II graph (Exact RPR)

Fig. 3 Average exact RPR ranks of non-seed ↑-rated movies relative to their ranks returned by conventional exact PPR (the ratio for exact PPR itself is 1.0) : the lower is the ratio, the better is the measure

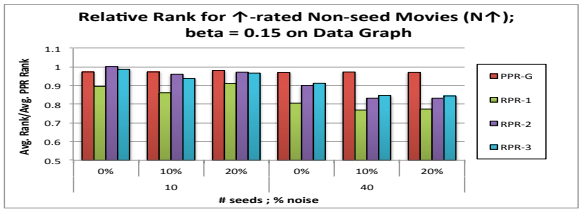
– As we see in the Figure 3, PPR-G does not provide any significant advantages over pure PPR, unless the teleportation rate is very small (i.e., not much significance is given to the seed set) or the seed set contains large amounts of noise.

- Figures 3 show that the proposed RPR schemes lead to significant improvements in the rankings, with RPR-2 providing the most consistent improvements.
- *Teleportation-discounting* (used in RPR-1 and RPR-3) is effective in (metadata and UM) graphs, which do not properly capture user preferences. *Seed-set maximization* (used in RPR-2), however, provides benefits for all graphs, including the Ratings graph, which reflects the user preferences in the transition probabilities. In fact as the data size grows (UM-II and Ratings-II graphs) *seed-set maximization* strategy provides the single most dominant gain in recommendation effectiveness.
- It is important to note that the RPR techniques provide better recommendation rankings even in situations where the seed set contains 0% artificially introduced noise, confirming that RPR provides better personalization given user history.

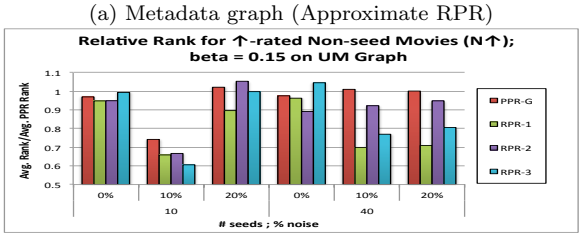
5.5 Recommendation Effectiveness under Approximate Computation

As discussed before, in order to observe the impact of such approximation on the effectiveness and efficiency of RPR schemes, we also implemented partitioning-based fast random walk based PPR and RPR schemes. Figure 4 compares the recommendation effectiveness of the RPR schemes obtained under a fast random walk based approximation scheme against the recommendation effectiveness of the PPR schemes also obtained under the same fast random walk based approximation. As before, *the smaller the value of the* $relative\_rank(A, N\uparrow)$*, the better is the recommendations returned by the algorithm* A:

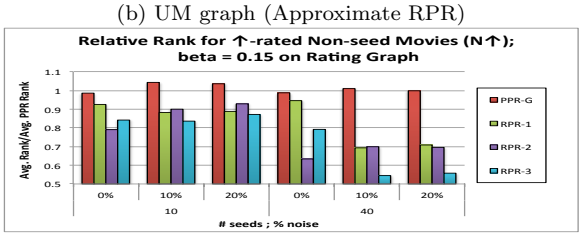
- As before, fast random walk based PPR-G does not provide any significant advantages over fast random walk based PPR.
- Also as before, as the graph size becomes larger (i.e., UM-II and Ratings-II graphs) the effectiveness performance of RPR schemes over the PPR scheme becomes more consistent.
- Again, as before, RPR techniques provide better recommendation rankings even in situations where the seed set contains 0% artificially introduced noise, confirming that RPR provides better personalization given user history even under approximate computation.
- The most interesting (and, at the first look, surprizing) result, however, is that, when using graph partitioning based approximations, RPR techniques that leverage teleportation-discounting (RPR-1 or RPR-3) provide the best overall gains in recommendation effectiveness.
  This shows that the graph-partitioning (and block diagonalization) scheme, which essentially distorts the graph structure due to the low-rank approximation of cross-partition edges, cannot properly account for teleportation-contributions (`t-PPR`, Section 3.2) which may involve frequent jumps across partitions. While teleportation-discounting increases robustness of PPR



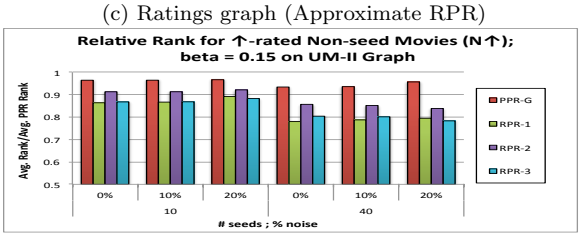
(a) Metadata graph (Approximate RPR)



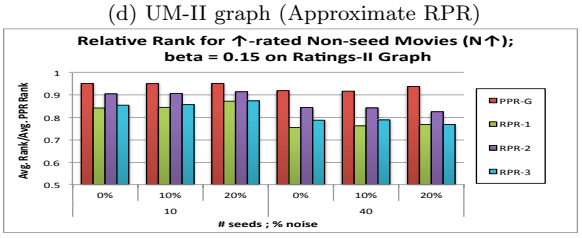
(b) UM graph (Approximate RPR)



(c) Ratings graph (Approximate RPR)



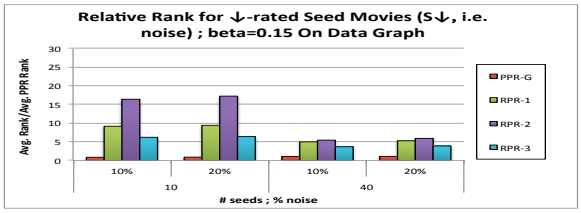
(d) UM-II graph (Approximate RPR)



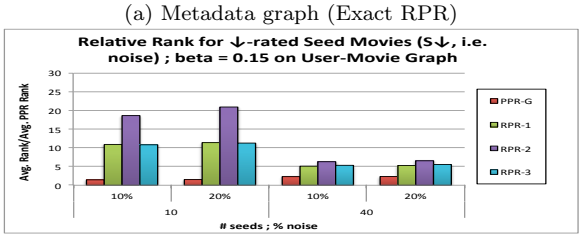
(e) Ratings-II graph (Approximate RPR)

Fig. 4 Average approximate RPR ranks of non-seed ↑-rated movies relative to their ranks returned by approximate PPR: the lower is the ratio, the better is the measure

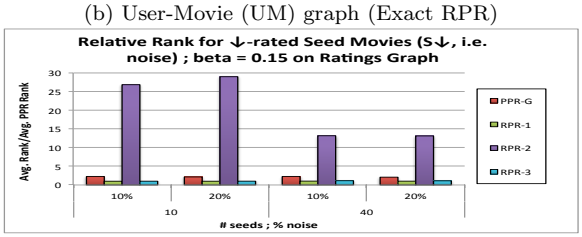
scores against noise in the seed set S by carefully discounting the teleportation contributions, we see that this technique also improves robustness against noise introduced due to graph-partitioning and block diagonalization inherent in many approximation schemes.



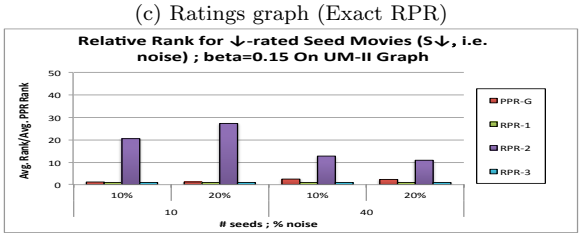
(a) Metadata graph (Exact RPR)



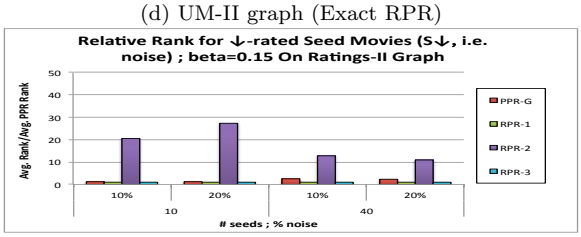
(b) User-Movie (UM) graph (Exact RPR)



(c) Ratings graph (Exact RPR)



(d) UM-II graph (Exact RPR)



(e) Ratings-II graph (Exact RPR)

Fig. 5 Average exact RPR ranks of ↓-rated movies included in the seed set relative to their ranks returned by conventional exact PPR (the ratio for exact PPR itself is 1.0): the higher is the ratio, the better is the measure

5.6 Seed Differentiation Power

Next, let us consider the effectiveness of the different personalized PageRank measures in differentiating the noise in the seed set ($S\downarrow$). Figure 5 compares the average user rankings of ↓-rated movies that were included in the seed

set. In this case, the *higher the relative rank, the better is the algorithm in differentiating the noise in the seed set*:

- As we have seen in the figure, once again, PPR-G does not provide any significant advantages over pure PPR;
- The proposed RPR algorithms, on the other hand push the ↓rated seed nodes (i.e., noise) significantly further down in the overall ranking relative to conventional PPR, indicating that RPR algorithms are effective in eliminating seed-bias; and
- As was the case with recommendation effectiveness, *teleportation-discounting* (used in RPR-1 and RPR-3) is effective in (metadata and UM) graphs, which do not capture user preferences; but *seed-set maximization* (used in RPR-2), provides benefits for all graphs. In fact, as the graph size becomes larger, as in UM-II and Ratings-II graphs, seed-set maximization based RPR-2 becomes the single most beneficial approach in terms of seed-set differentiation.

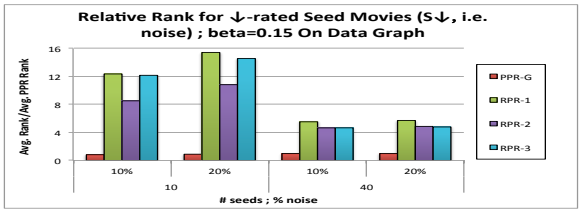
5.7 Seed Differentiation Power under Approximate Computation

In order to observe the impact of approximation schemes on the seed differentiation effectiveness, we also experimentally compared partitioning-based fast random walk based PPR and RPR schemes. Figure 6 compares the average user rankings of ↓-rated movies that were included in the seed set for fast random walk algorithm. Again, the *higher the relative rank, the better is the algorithm in differentiating the noise in the seed set*:

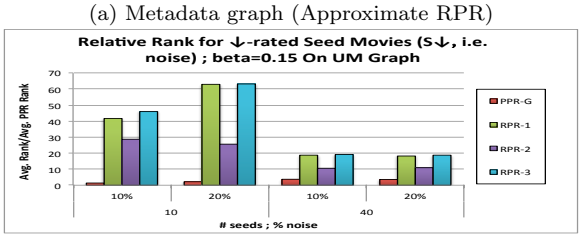
- As we have seen in the figure, once again, PPR-G does not provide significant advantages over pure fast random walk PPR.
- Also, under the use of approximations, RPR algorithms are able to push the ↓rated seed nodes (i.e., noise) significantly further down in the overall ranking relative to conventional (approximate) PPR, indicating that RPR algorithms are effective in seed differentiation even when approximation schemes are used for efficiency.
- Confirming our observation in 5.5, also in the case of seed-set differentiation, when using graph partitioning based approximations, RPR techniques that leverage teleportation-discounting (RPR-1 or RPR-3) are more effective than solely seed-set maximization based scheme (RPR-2). This is again because teleportation-discounting strategy not only improves robustness of PPR scores against noise in the seed set S, but also improves robustness against noise introduced due to graph-partitioning and block diagonalization inherent in approximation schemes.

5.8 Seed-Bias Elimination

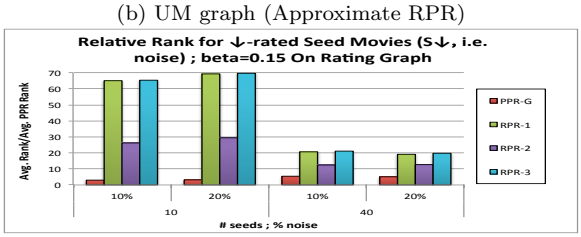
As we discussed in Section 3, PPR assumes that all the nodes in the seed set are very important and thus they tend to rank better than most (if not



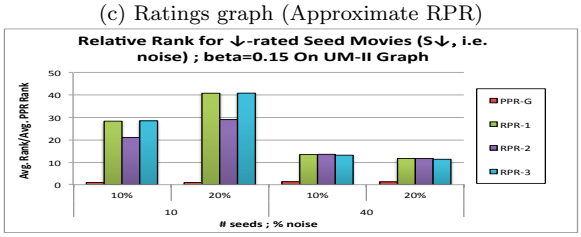
(a) Metadata graph (Approximate RPR)



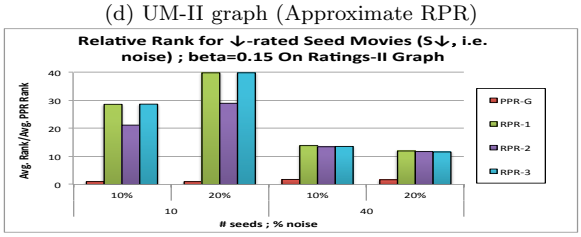
(b) UM graph (Approximate RPR)



(c) Ratings graph (Approximate RPR)



(d) UM-II graph (Approximate RPR)

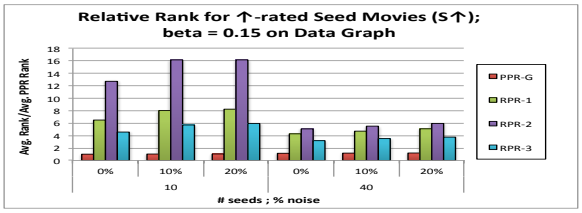


(e) Ratings-II graph (Approximate RPR)

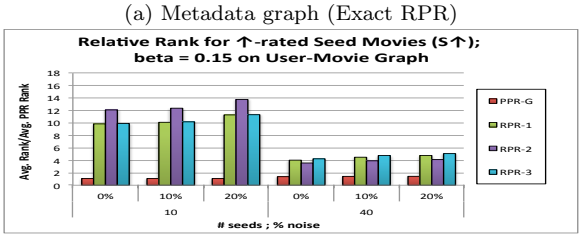
Fig. 6 Average approximate RPR ranks of ↓-rated movies included in the seed set relative to their ranks returned by approximate PPR: the higher is the ratio, the better is the measure

all) non-seed nodes. However, *we expect that a seed-bias eliminating ranking system would push some of the highly rated seeds further down in the rankings to bring up those movies that are good, but not used as seeds*:

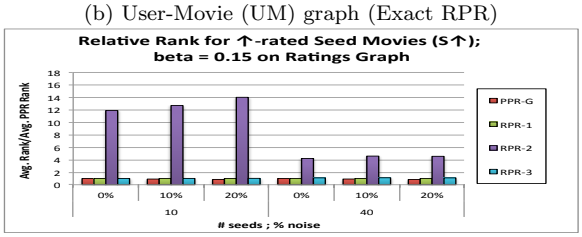
– In Figure 7, we see that this is indeed true for the proposed RPR schemes: as we would expect from a good seed bias eliminating algorithm, *teleportation-discounting* (for metadata and UM graphs) and *seed-set max-*



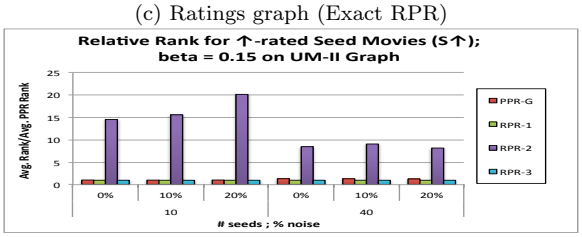
(a) Metadata graph (Exact RPR)



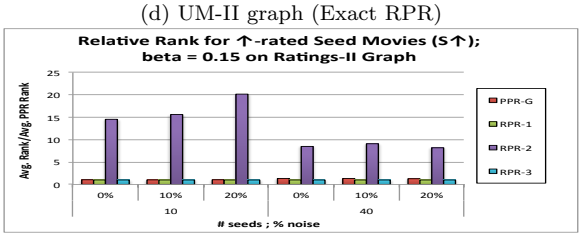
(b) User-Movie (UM) graph (Exact RPR)



(c) Ratings graph (Exact RPR)



(d) UM-II graph (Exact RPR)

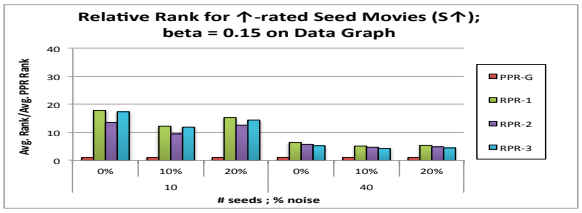


(e) Ratings-II graph (Exact RPR)

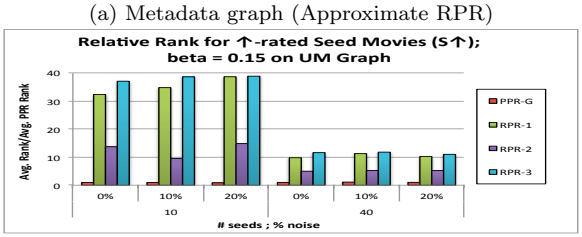
Fig. 7 Average exact RPR ranks of ↑-rated movies included in the seed set relative to their ranks returned by conventional exact PPR (the ratio for exact PPR itself is 1.0): the lower is the ratio, the better is the measure

imization (for all graphs) approaches help increase the relative rankings of the ↑-rated seed nodes, for accommodating the better rankings of good, but not seed nodes – as we have already seen in Figure 3.

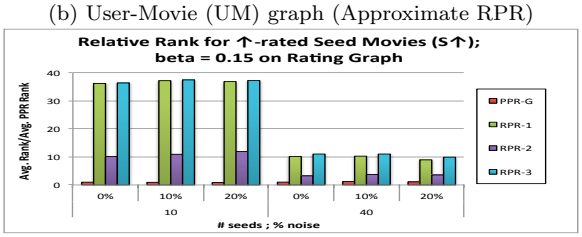
– As also observed before, the performance is especially consistent for large (UM-II and Ratings-II) graphs.



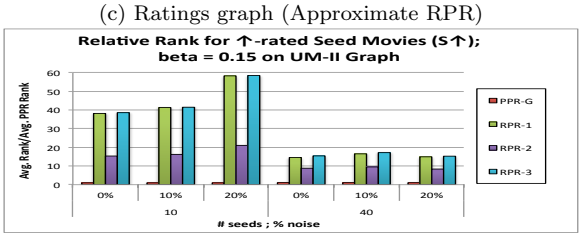
(a) Metadata graph (Approximate RPR)



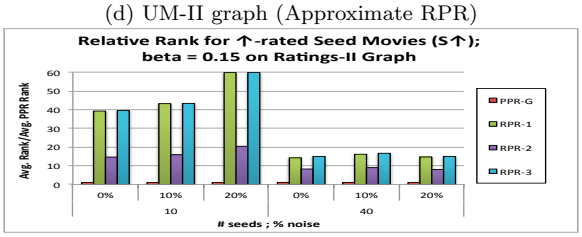
(b) User-Movie (UM) graph (Approximate RPR)



(c) Ratings graph (Approximate RPR)



(d) UM-II graph (Approximate RPR)



(e) Ratings-II graph (Approximate RPR)

Fig. 8 Average approximate RPR ranks of ↑-rated movies included in the seed set relative to their ranks returned by approximate PPR: the lower is the ratio, the better is the measure

5.9 Seed-Bias Elimination under Approximate Computation

Figure 8 compares seed-bias in RPR and PPR schemes under graph partitioning-based fast random walk approximation. As in the previous subsection, *we expect that a seed-bias eliminating ranking system would push some*

of the highly rated seeds further down in the rankings to bring up those movies that are good, but not used as seeds:

- In Figure 8, we see that this is indeed true for the proposed fast random walk based RPR schemes: as we would expect from a good seed bias eliminating algorithm, *teleportation-discounting* and *seed-set maximization* increase the relative rankings of the ↑-rated seed nodes, for accommodating the better rankings of good, but not seed nodes.
- As also observed in other measures of effectiveness, when using graph-partitioning based approximation schemes, *teleportation-discounting* provides better gains than *seed-set maximization* as it reduces the impact of noise caused by teleportations which may involve frequent jumps across partitions.

5.10 Effectiveness Summary

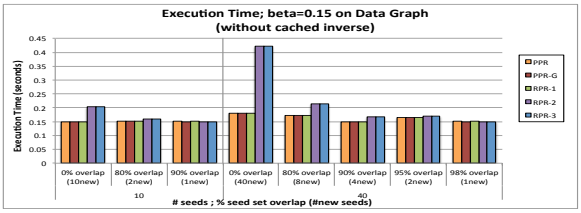
The above experiments have shown that the proposed RPR schemes are effective in improving ranking quality. When the underlying equations are solved exactly, the *seed-set maximization* technique performs well for all graphs (including those that already capture user preferences) and noise scenarios and thus should be the preferred ranking technique (according to the results, even in situations where the noise is 0%).

When graph-partitioning based approximations are used however, *teleportation-discounting* provides the highest gains since teleportation-discounting increases robustness of PPR scores against both noise in the seed set S as well as against noise introduced due to cross-partition teleportations needed under graph-partitioning.

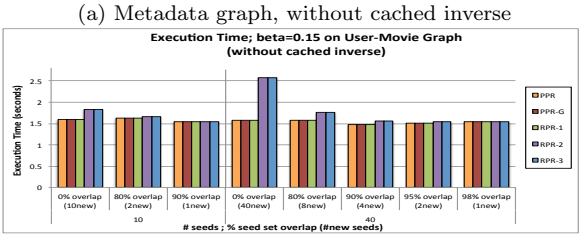
5.11 Efficiency Evaluations

In Figures 9 and 10, we consider the execution times of the different personalized PageRank algorithms considered in this paper (without explicit parallelization). In particular, we consider three scenarios: Fresh graph, fresh seed set: In the first scenario, we are given a graph and a fresh set of seeds and the computation starts from scratch (Figures 9(a), (b) and (c), columns corresponding to 0% overlap). Cached graph, fresh seed set: In the second, the graph is fixed and the matrix inverse has already been computed and cached; given a fresh seed set, this cached inverse is used for computing the scores and rankings (Figures 10(a), (b) and (c), columns corresponding to 0% overlap). Fresh/cached graph, overlapping (i.e., cached) seed set: In the third scenario (Figures 9 and 10, columns corresponding to more than 0% overlap), the new seed set overlaps with seed sets considered in the past and this overlap is leveraged as described in Section 4.3.

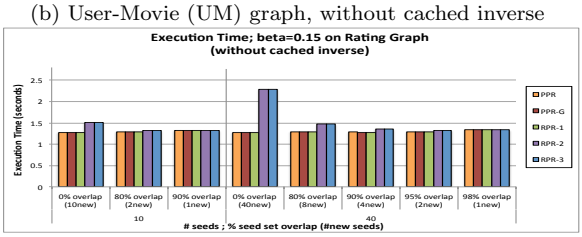
- As we see in Figures 10(a) and (b), when a cached inverse of the transition matrix is available and the seed set overlap is low, RPR-2 and RPR-3



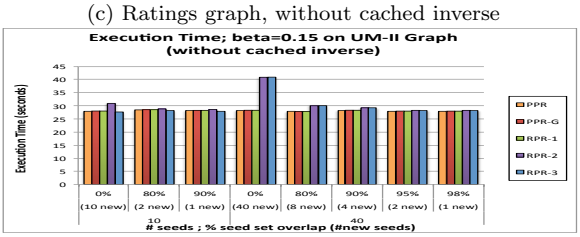
(a) Metadata graph, without cached inverse



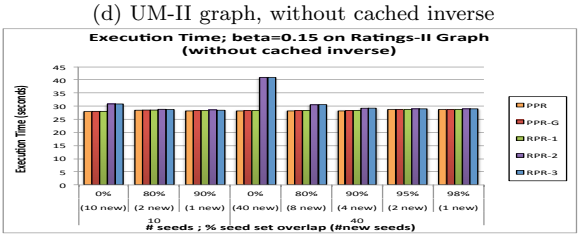
(b) User-Movie (UM) graph, without cached inverse



(c) Ratings graph, without cached inverse



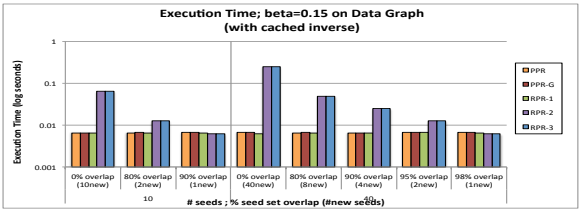
(d) UM-II graph, without cached inverse



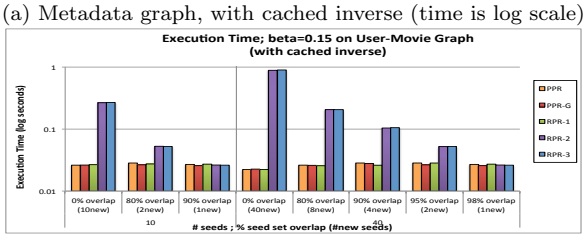
(e) Ratings-II graph, without cached inverse

Fig. 9 Execution times for different measures (w/o explicit parallelization) – personalized PageRank computation starts from scratch. For each configuration, we consider different rates of updates to the seed set.

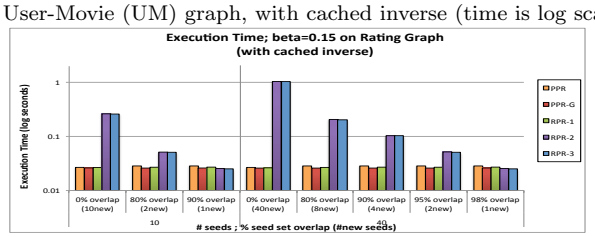
schemes (which need to solve multiple linear equations per Section 4.3) are slower than PPR, PPR-G, and RPR-1 schemes. Thus, when cached inverse is available and seed overlaps are small, we recommend using the



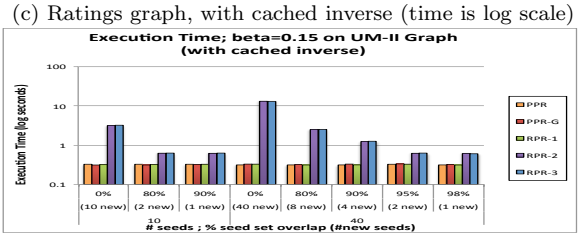
(a) Metadata graph, with cached inverse (time is log scale)



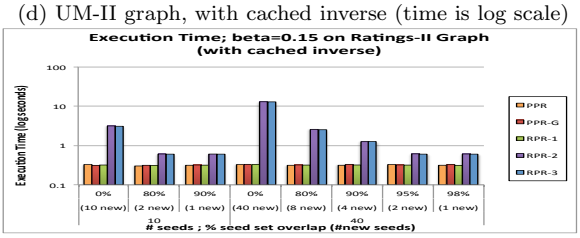
(b) User-Movie (UM) graph, with cached inverse (time is log scale)



(c) Ratings graph, with cached inverse (time is log scale)



(d) UM-II graph, with cached inverse (time is log scale)



(e) Ratings-II graph, with cached inverse (time is log scale)

Fig. 10 Execution times for different measures (w/o explicit parallelization) – personalized PageRank computation leverages cached matrix inverses. For each configuration, we consider different rates of updates to the seed set.

RPR-1 scheme which (as we have seen earlier) is more robust than PPR and PPR-G and, also, as fast.

– Figures 9 and 10 also show that the difference between the various strategies is small or non-existent (a) when the graph itself is fresh (i.e., no

	Execution time in seconds (with cached inverse)		
	10 seeds per unit	20 seeds per unit	40 seeds per unit
UM	0.26	0.51	1.02
Ratings	0.26	0.51	1.03
UM-II	3.63	7.44	14.48
Ratings-II	3.61	7.38	14.51

Table 4 The execution time of the RPR-II scheme grows linearly with the number of seeds

	Required Cache Size	
	UM and Ratings Graphs	UM-II and Ratings-II Graphs
Exact RPR	52.57MB	761.72MB
Approximate RPR	4.54MB	29.13MB

Table 5 Required cache size for exact and approximate PPR/RPR schemes

cached inverse is available), (b) the seed set is small, or (c) the overlap between the current seed set and the seeds considered in the past is large (i.e., cached solutions for individual seeds can be reused per Section 4.3). In these cases, RPR-2 and RPR-3 are competitive in execution times and should be used where the accuracy provided by the *seed-set maximization strategy* is critical.

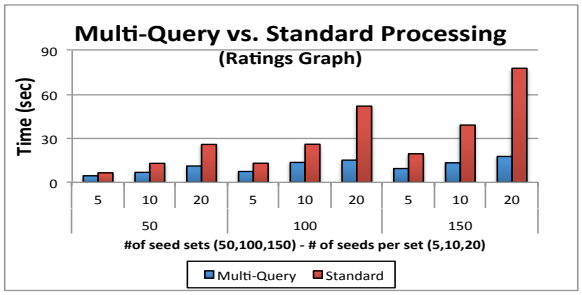
As discussed in Section 4.3, RPR is trivially parallelizable by mapping new seeds such that each computation unit processes only one (or few) new seeds. The impact of processing *only one seed per processing unit* can be seen by considering the cases marked "**1new**" in Figures 9 and 10, which illustrate the cost of performing RPR computation when each seed is processed by a dedicated processing unit.

5.12 Parallelization Opportunities

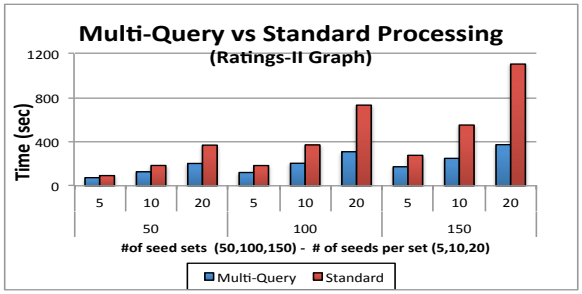
In Table 4, we study the parallelization opportunities provided by the proposed RPR-II scheme, discussed in Section 4.3.1, where the seed-set maximal optimization problem is converted into a set of linear optimization problems, each (or a subset)can be assigned to a different computation unit. The table confirms that the execution time (with a cached inverse that can be shared across computation units) grows linearly with the number of seeds assigned to each unit. This indicates that, as argued in Section 4.3.1, RPR-II is perfectly parallelizable across multiple computation units.

5.13 Space Consumption of RPR under Graph-Partitioning based Approximate Computation

As we see in Table 5, in these experiments, the sizes of cached inverses for exact RPR on Ratings graph and Rating-II graph are 52.57 MB and 761.72 MB, respectively. Size of cached inverses for approximate RPR on Ratings



(a) Execution times for multi-query RPR-2 and standard RPR-2 on the Ratings graph



(b) Execution times for multi-query RPR-2 and standard RPR-2 on the Ratings-II graph

Fig. 11 Execution times for multi-query RPR-2 and standard RPR-2 on the Ratings and the, larger, Ratings-II graphs (leveraging cached inverses)

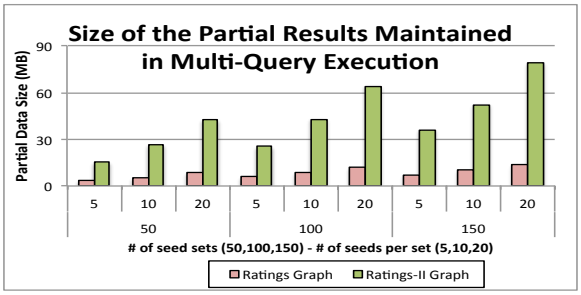


Fig. 12 Sizes of the partial results maintained during multi-query RPR-3 processing for Ratings and Ratings-II graphs

graph and Rating-II graph are 4.54 MB and 29.13 MB. Cache size is the same for different users and seeds settings because it is only related with graph size.

These results indicate, along with the accuracy results presented in the previous subsections, that RPR can benefit from approximate computations to reduce the resource requirements and *teleportation-discounting* can help improve the robustness of the algorithms to the noise introduced due to inherent graph partitioning.

5.14 Multi Query Evaluation

Finally, in Figure 11, we study the usefulness of multi-query processing scheme proposed in Section 4.4 in scenarios with high query throughput: the algorithm first groups different users' shared seeds, then conducts partial computation on shared seed sets, and finally obtains RPR scores for the individual users by combining relevant partial computations.

In this experiment, we have considered the Ratings and Rating-II graphs (of different sizes) and varied the number of seed sets for which we need to compute RPR-2 scores as 50, 100, and 150. These seed sets were obtained by selecting random users from the system, for which we generate recommendations. For each randomly selected user, we considered 5, 10, and 20 seeds. In this figure, the *standard* scheme involves running an RPR-2 task separately for each user (i.e., seed set).

As we see in Figure 11, multi-query processing significantly lowers the query execution time needed to process large number of RPR queries. The gains are especially significant when the graph size, number of seeds sets, and the number of seed per set are large: $\sim$ 1100 seconds in standard RPR-2 processing on the Ratings-II graph for 150 random queries with 20 seed sets vs. only $\sim$ 370 seconds in multi-query RPR-2 processing on the same scenario. These large time savings in multi-query processing are due to the partial results shared by the different seed-sets (Figure 12). These savings are especially impressive when we consider that, in these experiments, the users (to whom the seed sets belong) are selected randomly.

6 Conclusions

In this paper, we have shown that conventional personalized PageRank (PPR) algorithms associate *unnecessarily high bias* to the seed nodes and this negatively affects the node rankings when the seed set is *incomplete* and/or *noisy*. To deal with this problem, we propose three alternative *robust personalized PageRank* (RPR) algorithms that eliminate the potential noise in the seed set. We have shown that a novel *teleportation discounting* technique ensures that rankings are not overly biased towards the seed nodes and a novel *seed-set maximal PPR principle* helps differentiate among the seeds by considering the overall context defined by the seed set. Experiment results showed that the resulting robust Personalized PageRank (RPR) techniques are efficient and highly effective in improving recommendations and eliminating unwanted bias due to imperfections in the seed set. The RPR measures also have efficient implementations, where existing approximation algorithms and/or parallel implementations for computing the PPR scores can be easily leveraged. In fact, experiment results confirm that the seed-set maximal approach is *reuse-promoting* in that it is possible to divide the work relative to individual seed nodes and teleportation discounting technique provides additional robustness

against noise introduced during graph-partitioning (and block diagonalization) based approximate random walk computation processes.

References

1. Andersen R, et al. (2008) Trust-based recommendation systems: an axiomatic approach. WWW, pages 199-208, 2008.
2. Avrachenkov K, et al. (2011) Quick Detection of Top-k Personalized PageRank Lists. WAW, pages 50-61, 2011,
3. B. Bahmani., K. Chakrabarti, D. Xin, Fast personalized PageRank on MapReduce. In SIGMOD, pages 973-984, 2011.
4. Bahmani B., et al. Fast incremental and personalized PageRank. PVLDB. 4, 3, pages 173-184, 2010.
5. Balmin A., et al. ObjectRank: Authority-based keyword search in databases. VLDB, pages 564-575, 2004.
6. Berkhin P. Bookmark-coloring approach to personalized pagerank computing. Internet Mathematics, 3(1), 2007.
7. Boldi P, Rosa M, Vigna S (2011) HyperANF: Approximating the neighbourhood function of very large graphs on a
8. Borgs C, Brautbar M, Chayes J, et al. Multiscale Matrix Sampling and Sublinear-Time PageRank Computation[J]. Internet Mathematics, 10(1-2): 20-48, 2014
9. Borgatti MG., et al. Network measures of social capital. Connections 21(2):27-36, 1998.
10. Brin S., et al. The anatomy of a large-scale hypertextual Web search engine. Computer Networks and ISDN Systems 30: 107-117, 1998.
11. Buckley C., Voorhees E.M. Retrieval evaluation with incomplete information. SIGIR, pages 25-32, 2004.
12. Candan K.S. and Li W.S. Using random walks for mining web document associations. PAKDD, pages 294-305, 2000.
13. Candan K.S., et al. Reasoning for Web document associations and its applications in site map construction. Data Knowl. Eng. 43(2): 121-150, 2002.
14. Chakrabarti S. Dynamic personalized pagerank in entity-relation graphs. WWW, pages 571-580, 2007.
15. Chen M., et al. Clustering via random walk hitting time on directed graphs. AAAI, pages 616-621, 2008.
16. Cohen E, Halperin E, Kaplan H, Zwick U (2003) Reachability and distance queries via 2-hop labels. SIAM, 2003.
17. Csalogany K., et al. Towards Scaling Fully Personalized PageRank: Algorithms, Lower Bounds, and Experiments Internet Math. 2,3, pages 333-358, 2005.
18. Davis T. A., et al. Direct methods for sparse linear systems. SIAM, 2006.
19. Foster K.C., Muth S.Q., Potterat J.J., and Rothenberg R. B., A Faster Katz Status Score Algorithm, Comput. and Math. Organ. Theo., 7(4):275-285, 2001.
20. Fouss F., et al. Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation. TKDE, pages 1041-4347, 2007.
21. Fujiwara Y., et al. Fast and exact top-k search for random walk with restart. PVLDB. 5, 5, pages 442-453, 2012.
22. Guan Z., et al. Personalized tag recommendation using graph-based ranking on multi-type interrelated objects. SIGIR, 540-547, 2009
23. Gupta M., et al. Fast algorithms for Top-k Personalized PageRank Queries. WWW, pages 1225-1226, 2008.
24. Haveliwala T.H. Topic-sensitive PageRank. WWW, 517-526, 2002.
25. Jeh G., Widom J. Scaling personalized web
26. Katz L., "A new status index derived from sociometric analysis. Psychometrika, 18:39-43, 1953.
27. Kamver S.D.,Haveliwala T., Manning C.D., Golub G. Extrapolation methods for accelerating PageRank computations. Proceedings of the 12th international conference on World Wide Web (WWW'03). ACM, New York, NY, USA, 261-270. 2003.

28. Kim H. J., Candan K. S., Sapino M. L., LR-PPR: Locality-Sensitive, Re-use Promoting, Approximate Personalized PageRank Computation. CIKM'13, 2013
29. Kim H.N, El-Saddik A. Personalized PageRank vectors for tag recommendations: inside FolkRank. RecSys, 45-52, 2011.
30. Langville A.N., Meyer C.D. Updating pagerank with iterative aggregation. WWW (Alternate Track Papers & Posters)'04. , 2004.
31. Maehara T., Akiba T., et al. Computing personalized PageRank quickly by exploiting graph structures. Proceedings of the VLDB Endowment, 7:10231034, 2014.
32. Malewicz G., et al. Pregel: a system for large-scale graph processing. SIGMOD, pages 135-146, 2010.
33. Mei Q., et al. Query suggestion using hitting time. CIKM, 2008.
34. Palmer C, Gibbons P, Faloutsos C (2002) Anf: a fast and scalable tool for data mining in massive graphs. KDD'02, 2002.
35. Piegorsch W, Casella G.E (1990) Inverting a sum of matrices. In SIAM Review, 1990.
36. Perozzi B. , McCubbin C. , Halbert J.T.. Scalable graph clustering with parallel approximate PageRank. Social Network Analysis and Mining, 4:179-189, 2014
37. Sarkar P., et al. Fast incremental proximity search in large graphs. ICML, pages 896-903, 2008.
38. Sarma A.D, Molla A.R, Pandurangan G, Upfal E. Fast Distributed PageRank Computation. ICDCN, pages 11-26, 2013
39. Tong H, Faloutsos C (Center-piece subgraphs: problem definition and fast solutions. In *KDD* '06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining}, pp. 404–413, New York, NY, USA, 2006. ACM.
40. Tong H, Faloutsos C, Koren Y (2007) Fast direction-aware proximity for graph mining. *KDD*, pp. 747–756, 2007.
41. Tong H., et al. Fast Random Walk with Restart and Its Applications. ICDM, pages 613-622, 2006.
42. Wei F (2010) Tedi: efficient shortest path query answering on graphs. SIGMOD'10.
43. White D.R., et al. Betweenness centrality measures for directed graphs. Social Networks, 16, pages 335-346,1994.
44. Xiao Y, Wu W, Pei J, Wang W, He Z (2009) Efficiently indexing shortest paths by exploiting symmetry in graphs. EDBT, 2009.
45. Zhou L, Chen L, Ozsu M.T (2009) Distance-join: pattern match query in a large graph, VLDB, 2009.
46. http://www.imdb.com/
47. http://www.grouplens.org/
48. http://www.cs.umn.edu/ metis