IMRank: Influence Maximization via Finding Self-Consistent Ranking

Suqi Cheng, Huawei Shen, Junming Huang, Wei Chen, Xueqi Cheng
Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China
{chengsuqi, shenhuawei, huangjunming, chenwei2012, cxq}@ict.ac.cn

ABSTRACT

Influence maximization, fundamental for word-of-mouth marketing and viral marketing, aims to find a set of seed nodes maximizing influence spread on social network. Early methods mainly fall into two paradigms with certain benefits and drawbacks: (1) Greedy algorithms, selecting seed nodes one by one, give a guaranteed accuracy relying on the accurate approximation of influence spread with high computational cost; (2) Heuristic algorithms, estimating influence spread using efficient heuristics, have low computational cost but unstable accuracy.

We first point out that greedy algorithms are essentially finding a self-consistent ranking, where nodes' ranks are consistent with their ranking-based marginal influence spread. This insight motivates us to develop an iterative ranking framework, i.e., IMRank, to efficiently solve influence maximization problem under independent cascade model. Starting from an initial ranking, e.g., one obtained from efficient heuristic algorithm, IMRank finds a self-consistent ranking by reordering nodes iteratively in terms of their ranking-based marginal influence spread computed according to current ranking. We also prove that IMRank definitely converges to a self-consistent ranking starting from any initial ranking. Furthermore, within this framework, a last-to-first allocating strategy and a generalization of this strategy are proposed to improve the efficiency of estimating ranking-based marginal influence spread for a given ranking. In this way, IMRank achieves both remarkable efficiency and high accuracy by leveraging simultaneously the benefits of greedy algorithms and heuristic algorithms. As demonstrated by extensive experiments on large scale real-world social networks, IMRank always achieves high accuracy comparable to greedy algorithms, while the computational cost is reduced dramatically, about $10 - 100$ times faster than other scalable heuristics.

Categories and Subject Descriptors

F.2.2 [**Analysis of Algorithms and Problem Complexity**]: Non-numerical Algorithms and Problems; D.2.8 [**Software Engineering**]: Metrics—*complexity measures, performance measures*

General Terms

Algorithms, Experiments, Performance

Keywords

influence maximization; social network analysis; viral marketing; iterative method; self-consistent ranking

1. INTRODUCTION

The prosperity of online social networks and social media invokes a new wave of research on social influence analysis [18, 8]. Finding influential individuals is an important problem for many applications such as expert finding, online advertising and marketing. Influence maximization is identified as a fundamental problem for viral marketing in the area of online marketing. It aims to find a fixed-size set of seed nodes in social network to maximize their *influence spread*, i.e., the expected number of activated nodes triggered by the seed nodes. Ever since being formalized by Kempe et al. [11], influence maximization has attracted much research attention from various fields, including social network analysis, data mining and marketing.

Early methods for influence maximization mainly use greedy framework, iteratively selecting the node with the largest marginal influence spread as seed node. with an accurate estimation of influence spread,the greedy framework provides a $(1 - 1/e)$ approximation to the optimal solution of influence maximization [11], guaranteed by the submodularity and monotonicity properties of influence spread as a function of seed node set. According to the ways of estimating the influence spread, these methods roughly fall into two paradigms: greedy algorithms [11, 13, 4, 7, 5] and heuristic algorithms [12, 3, 20, 10]. Greedy algorithms provide a $(1 - 1/e - \epsilon)$ approximation by approximating influence spread through Monte Carlo simulation. However, they have high computation cost, for the calculation of marginal influence spread invokes estimating the influence spread of nodes from scratch, using time-consuming Monte Carlo simulation. The latter, in contrast, resorts to estimate the influence spread via efficient heuristic methods. The scalability of these heuristics generally outperforms the greedy algorithms by several orders of magnitude. Yet, their high scalability is gained with the pain of unguaranteed accuracy and unreliable performance on various scenarios. To the best of our knowledge, we lack an efficient and accurate algorithm of influence maximization for applications to large scale social networks in real world.

In this paper, we propose an efficient and accurate algorithm to solve influence maximization problem under the widely-adopted independent cascade model [11]. This algorithm is motivated by the key insight that greedy algorithms are essentially finding a self-consistent ranking, where nodes' ranks are consistent with their ranking-based marginal influence spread. We prove that such self-consistent ranking can be obtained directly using an iterative ranking framework, i.e., IMRank, proposed in this paper. Starting from an initial ranking, e.g., one obtained from efficient heuristic algorithm, IMRank efficiently finds a self-consistent ranking by reordering nodes iteratively in terms of their ranking-based marginal influence spread computed according to current ranking. Different from greedy algorithms computing marginal influence spread from scratch, IMRank conducts the computation of ranking-based marginal influence spread via an efficient last-to-first allocating strategy. As a result, IMRank achieves both high efficiency and high accuracy by leveraging simultaneously the benefits of greedy algorithms and heuristic algorithms.

To evaluate the performance of IMRank, we conduct extensive experiments on large-scale social networks with hundreds of thousands of edges to millions of edges. Experimental results demonstrate that IMRank achieves high accuracy comparable to greedy algorithms with computational cost reduced dramatically.

Our main contributions are summarized as follows:

- We propose a novel framework IMRank, which unifies the estimation of marginal influence spread and the selection of seed nodes. IMRank achieves both remarkable efficiency and high accuracy by exploiting the interplay between the calculation of ranking-based marginal influence spread and the ranking of nodes.

- We prove that IMRank, starting from any initial ranking, definitely converges to a self-consistent ranking in a finite number of steps. This indicates that IMRank is efficient at solving the influence maximization problem via finding a final self-consistent ranking.

- We design an efficient last-to-first allocating strategy to approximately estimate the ranking-based marginal influence spread of nodes for a given ranking, further improving the efficiency of IMRank.

- We conduct extensive experiments on several real-world networks under different types of independent cascade model. Through comparing two instances of IMRank with both greedy algorithm and existing state-of-the-art heuristics, we show that IMRank always achieves comparable accuracy to the greedy algorithm, while runs $10 - 100$ times faster than other heuristics with comparable accuracy.

2. RELATED WORK

Influence maximization problem was first studied by Domingos and Richardson from algorithmic perspective [6, 16]. Kempe et al. then formulated it as a combinatorial optimization problem of finding a set of seed nodes with maximum influence spread [11]. They proved that this problem is NP-hard and proposed a greedy algorithm which can guarantee a $(1 - 1/e - \epsilon)$ approximation ratio. Here, ϵ is caused by the inaccurate estimation of influence spread. The biggest problem suffered by Kempe's greedy algorithm is its low scalability, limiting it to social networks with small or moderate size.

Many efforts have been made to improve the scalability of Kempe's greedy algorithm for influence maximization. "cost-effective lazy forward" (CELF) optimization strategy [13] and CELF++ [7] are proposed to reduce the times of influence spread estimation in Kempe's greedy algorithm by exploiting the submodularity property of influence spread function. To reduce the number of Monte Carlo simulations, Chen et al. [4] proposed NewGreedy algorithm and MixedGreedy algorithm. The NewGreedy algorithm reusing the results of Monte Carlo simulations in the same iteration to calculate marginal influence spread for all candidate nodes. Yet, it increases the computational cost for a single Monte Carlo simulation because the simulation is now conducted globally rather than locally as done in Kempe's greedy algorithm. As a remedy, the MixedGreedy algorithm was developed, integrating the CELF strategy into the NewGreedy algorithm. Recently, Sheldon et al. [17] proposed a sample average approximation approach from stochastic optimization for maximizing the spread of cascades under budget restriction. Cheng et al. [5] proposed a StaticGreedy algorithm, remarkably reducing the number of Monte-Carlo simulations through strictly guaranteeing the submodularity and monotonicity properties of influence spread function. The above two works further improve the scalability of greedy algorithm effectively. These improvements can speedup the original greedy algorithm in several orders of magnitude, however, scalability is still a challenge for greedy algorithms.

Heuristic algorithms, in contrast, mainly reduce the complexity of Kempe's greedy algorithm through computing influence spread heuristically. DegreeDiscount, designed for uniform independent cascade model, only computes direct influence [4]. Community-based greedy algorithm conducted Monte Carlo simulation within each community rather than on the whole network [20]. SPM/SP1M algorithms [12] estimated influence spread according to shortest paths, while PMIA algorithm [3] [19] used maximum influence paths. SP1N algorithm employed the concept of Shapley value from the cooperative game theory [15]. IRIE algorithm [10] efficiently estimated marginal influence spread through an iterative method. Besides the above heuristics using greedy approach, Jiang et al. proposed a simulated annealing approach with several heuristics [9], and Mathioudakis et al. suggested to speed up influence maximization using a simplified influence network [14]. However, these heuristics cannot give rise to guaranteed accuracy and their performance is unstable on different networks and diffusion models.

Taken together, in existing algorithms for influence maximization, the estimation of influence spread and the ranking of nodes are studied separately. On one hand, without leveraging the ranking of nodes, greedy algorithms estimate the influence spread of nodes from scratch, causing high computational cost. On the other hand, lacking a reliable estimation of influence spread, heuristic algorithms have no guaranteed accuracy. Hence, in this paper, we improve the state-of-the-art solution of influence maximization problem by exploiting the interplay between marginal influence spread and the ranking of nodes.

3. SELF-CONSISTENT RANKING

For influence maximization on a social network $G = (V, E)$, *influence spread function* $I(S)$ of a node set $S \subseteq V$ is defined as the expected number of nodes in G eventually activated by S under certain diffusion model. The function $I(\cdot)$ is nonnegative, monotone, and submodular, satisfying

Table 1: Notations.

Notation	Description
v_i	a node with index i
r_i	the index of node with rank i with respect to a given ranking r
$S = \{v_1, v_2, \ldots, v_n\}$	a set of nodes
$I(S)$	expected number of nodes eventually activated by set S
$M(v\|S)$	marginal influence spread by adding node v into a seed set S
$M_r(v_{r_i})$	ranking-based marginal influence, short for $M(v\|\{v_{r_1}, v_{r_2}, \ldots, v_{r_{i-1}}\})$
$p(v_i\|\{v_1, v_2, \ldots, v_{i-1}\})$	probability that v_i is activated given that a collection of nodes $\{v_1, v_2, \ldots, v_{i-1}\}$ are already activated
$\eta_r(v_i, v_j)$	influence score that node v_i sends to node v_j with respect to a given ranking r
$d(v_j, v_i)$	a simple path starting from v_j and ending at v_i, i.e., $\{w_1 = v_j, w_2, \ldots, w_n = v_i\}$
$d_r(v_j, v_i)$	influence path, which is a simple path where v_j is the only node ranked higher than v_i on the path
$\rho_r(v_i, v_j)$	probability that v_i is activated by v_j through any influence path, with respect to a given ranking r
l	maximal length of all influence paths to account into

- Nonnegative: $I(S) \geq 0$;

- Monotone: $I(S) \leq I(T)$, if $S \subseteq T \subseteq V$;

- Submodular: $I(S \cup \{v\}) - I(S) \geq I(T \cup \{v\}) - I(T)$, for all $v \in V$ and $S \subseteq T \subseteq V$.

These properties guarantee that a fair approximation to the optimal solution of influence maximization can be obtained by greedy algorithms, iteratively selecting the node with maximum marginal influence spread as seed node.

DEFINITION 1. *Marginal influence spread:* *Given a node set $S \subseteq V$ and a node $v \in V$, the marginal influence spread of v upon S is defined as $M(v\|S) = I(S\cup\{v\}) - I(S)$.*

However, the influence spread function is not extensive, i.e., $I(S \cup \{v\}) \neq I(S) + I(\{v\})$ if $v \notin S$, since the nodes activated by S may overlap with the nodes activated by v. Therefore, one has to compute the marginal influence spread by computing both $I(S)$ and $I(S\cup\{v\})$ from scratch, resulting in huge computation cost. To remedy this problem, we further analyze the property of the set of seed nodes obtained by greedy algorithms. Indeed, greedy algorithms implicitly give a ranking of nodes, where nodes are ranked in decreasing order of their marginal influence spread. Meanwhile, their marginal influence spread are computed based on their ranks in the implicit ranking. Hence, greedy algorithms obtain a *self-consistent ranking* of nodes.

Before formally defining self-consistent ranking, we first introduce several related notations for clarity. Without loss of generality, we index all the nodes into $\{v_1, v_2, \cdots, v_n\}$ where $n = |V|$. A ranking of nodes, determined by a permutation $(r_1, r_2, \cdots, r_n)$ with $r_i \in \{1, 2, \cdots, n\}$ denoting the index of node with rank i, is denoted as $r = \{v_{r_1}, v_{r_2}, \cdots, v_{r_n}\}$. With these notations, for convenience, we now define the ranking-based marginal influence spread of node with respect to a ranking r as $M_r(v_{r_i}) = M(v_{r_i}\|\{v_{r_1}, v_{r_2}, \cdots, v_{r_{i-1}}\})$. In addition, for clarity, Table 1 lists all important notations used in this paper.

DEFINITION 2. *Self-consistent ranking:* *A ranking r is a self-consistent ranking iff $M_r(v_{r_i}) \geq M_r(v_{r_j}), \forall 1 \leq i < j \leq n$.*

For the set of seed nodes obtained by greedy algorithms, there exists an interplay between the ranks of nodes and their marginal influence spread. On one hand, these nodes are ranked in descending order of their marginal influence spread. On the other hand, the marginal influence spread of nodes is calculated with respect to the ranks of nodes. Indeed, the set of seed nodes obtained by greedy algorithms forms a self-consistent ranking.

THEOREM 1. *Greedy algorithms for influence maximization gives a self-consistent ranking.*

PROOF. Greedy algorithms iteratively select the node with maximum marginal influence spread as seed node. With a ranking r denoting the order seed nodes are selected, we have $M(v_{r_i}\|\{v_{r_1}, v_{r_2}, \cdots, v_{r_{i-1}}\}) \geq M(v_{r_j}\|\{v_{r_1}, v_{r_2}, \ldots, v_{r_{i-1}}\})$, for $i < j$. In addition, the submodularity of influence spread function implies that $M(v_{r_j}\|\{v_{r_1}, v_{r_2}, \cdots, v_{r_{i-1}}\}) \geq M(v_{r_j}\|\{v_{r_1}, v_{r_2}, \cdots, v_{r_{j-1}}\})$. Using transitivity, we complete the proof with $M_r(v_{r_i}) = M(v_{r_i}\|\{v_{r_1}, v_{r_2}, \cdots, v_{r_{i-1}}\}) \geq M(v_{r_j}\|\{v_{r_1}, v_{r_2}, \cdots, v_{r_{j-1}}\}) = M_r(v_{r_j})$. □

For a given social network, however, there are multiple self-consistent rankings besides the one obtained by greedy algorithms. Hence it is critical to develop effective algorithms to achieve a desired self-consistent ranking which is either the very ranking obtained by greedy algorithms or comparable to it from the point of influence maximization.

4. IMRANK

In this section, we develop an efficient iterative framework IMRank to solve the influence maximization problem through finding a desired self-consistent ranking. IMRank distinguishes itself from greedy algorithms in one key point: in each iteration, IMRank efficiently estimates the marginal influence spread of all nodes based on current ranking, while greedy algorithms compute the marginal influence spread from scratch with high computational cost.

4.1 IMRank: iterative framework

IMRank aims to find a self-consistent ranking from any initial ranking. It achieves the goal by iteratively adjusting current ranking as follows:

- Compute the ranking-based marginal influence spread of all nodes M_r with respect to the current ranking r;

- Obtain a new ranking by sorting all nodes according to M_r.

This iterative process is formally described in Algorithm 1. It definitely converges to a self-consistent ranking, starting from any initial ranking (see Section 4.3 for proof). Intuitively, IMRank iteratively promotes influential nodes to top positions in the ranking, always increasing the influence spread of top-k nodes during the process until it converges to a self-consistent ranking. Indeed, different initial rankings could make IMRank converge to different self-consistent rankings. We leave the discussion about initial ranking to Section 4.4.

Algorithm 1 IMRank (r)

1: $r^{(0)} = r$
2: $t \leftarrow 0$
3: **repeat**
4:     $t \leftarrow t + 1$
5:     Calculate $M_{r^{(t)}}$ with respect to the ranking r
6:     Generate a new ranking $r^{(t)}$ by sorting nodes in decreasing order according to $M_{r^{(t)}}$
7: **until** $r^{(t)} = r^{(t-1)}$
8: output the self-consistent ranking $r^{(t)}$

Algorithm 2 Calculate $\mathbf{M_r}(r)$

1: **for** $i = 1$ to n **do**
2:     $M_r(v_{r_i}) \leftarrow 1$
3: **end for**
4: **for** $i = n$ to 2 **do**
5:     **for** $j = 1$ to i **do**
6:         $M_r(v_{r_j}) \leftarrow M_r(v_{r_j}) + p(v_{r_j}, v_{r_i}) \times M_r(v_{r_i})$
7:         $M_r(v_{r_i}) \leftarrow \left(1 - p(v_{r_j}, v_{r_i})\right) \times M_r(v_{r_i})$
8:     **end for**
9: **end for**
10: output M_r

4.2 Calculate ranking-based marginal influence spread

The core step in IMRank is the calculation of ranking-based marginal influence spread. One straightforward way is to directly compute $M_r(v_{r_i}) = M(v_{r_i}|\{v_{r_1}, v_{r_2}, \cdots, v_{r_{i-1}}\})$ using Monte Carlo simulation, as done by greedy algorithms. However, prohibitively high computational cost makes it impractical for IMRank. To combat this problem, we propose a *Last-to-First Allocating* (LFA) strategy to efficiently estimate M_r, leveraging the intrinsic interdependence between ranking and ranking-based marginal influence spread. We develop the LFA strategy under the independent cascade model [11]. For the independent cascade model, when a node u is activated, it has one chance to independently activate its neighboring nodes with a propagation probability $p(u, v)$ if v has not been activated yet. Each node can be activated for only once.

The LFA strategy is based on the following fact: by definition, the ranking-based marginal influence spread $M_r(v)$ is equal to the expected number of nodes activated by v, given that when all nodes ranked higher than it have finished the propagation of their influence. This implies two basic rules under the calculation of $M_r(v)$:

1. Each node can only be activated by nodes ranked higher than it in the given ranking;

2. When a node could be activated by multiple nodes, higher-ranked node has higher priority to activate it.

Following the two basic rules, the LFA strategy is described as follows:

- Given a ranking r, the initial value of $M_r(v_{r_i})$ of each node is setted to be 1, satisfying the fact that the sum of $M_r(v_{r_i})$ over all nodes is equal to the number of nodes, since each node can only be activated once.

- Scanning the ranking from the last node to the top one, a fraction of $M_r(v_{r_i})$ is delivered to the nodes ranked higher than v_{r_i}, reflecting the first rule;

- The delivered influence score of $M_r(v_{r_i})$ is allocated among the nodes $v_j (j < i)$ in terms of their ranks, reflecting the second rule.

  Specifically, with $\eta(v_{r_j}, v_{r_i})$ denoting the fraction of influence score delivered to node v_{r_j} from node v_{r_i}, we have $\eta(v_{r_j}, v_{r_i}) =$

$$
\begin{cases}
M_r(v_{r_i})p(v_{r_j}, v_{r_i}) \displaystyle\prod_{k:1 \le k < j} \left(1 - p(v_{r_k}, v_{r_i})\right), & j < i, \\
0, & \text{otherwise.}
\end{cases}
$$

$$(1)$$

where $p(v_{r_j}, v_{r_i})$ is the propagation probability that node v_{r_j} directly activates node v_{r_i}, known as *a priori* for the independent cascade model.

The calculation of the ranking-based marginal influence spread M_r is completed after all nodes are scanned. The LFA strategy is formally depicted in Algorithm 2.

Now we use an example to illustrate the LFA strategy. In Figure 1, v_k denotes the node with rank k for convenience, and $p_{i,j}$ is the propagation probability along edge $\langle v_i, v_j \rangle$. Here, the ranking is simply $r = \{v_1, v_2, v_3, v_4, v_5\}$. Solid lines represent the edges where influence could propagate, while dashed lines depict the edges where influence score is delivered when nodes are scanned. The lack of dashed line from node v_3 to node v_2 reflects that node v_2 is ranked higher than node v_3. For this case, the LFA strategy computes the ranking-based marginal influence spread as follows:

1. Initially, $M_r(v_i) = 1, 1 \le i \le 5$.

2. Node v_5 is then scanned as the last node in the ranking. According to Equation(1), v_5 delivers $p_{3,5}M_r(v_5) = p_{3,5}$ to v_3 and $p_{4,5}(1 - p_{3,5})$ to v_4 respectively. Accordingly, $M_r(v_5)$ becomes $(1 - p_{4,5})(1 - p_{3,5})$.

3. Then node v_4 is scanned. Since $M_r(v_4)$ is now $1 + p_{4,5}(1 - p_{3,5})$, v_4 delivers $p_{2,4} + p_{2,4}p_{4,5}(1 - p_{3,5})$ to v_2. Note that the second item characterizes the influence of v_2 to v_5 through the path $\langle v_2, v_4, v_5 \rangle$, reflecting that the LFA strategy could effectively capture the indirect influence among nodes. After v_4 is scanned, the final value of $M_r(v_4)$ is $(1 - p_{2,4})(1 + p_{4,5}(1 - p_{3,5}))$.

4. When node v_3 is scanned. it delivers $p_{1,3}(1 + p_{3,5})$ to node v_1, with $(1 - p_{1,3})(1 + p_{3,5})$ remained.

5. Finally, node v_2 is scanned. After v_2 is scanned, the final scores of $M_r(v_2)$ and $M_r(v_1)$ are $(1 - p_{1,2})(1 + p_{2,4} + p_{2,4}p_{4,5}(1 - p_{3,5}))$ and $1 + p_{1,2}(1 + p_{2,4} + p_{2,4}p_{4,5}(1-p_{3,5})) + p_{1,3}(1 + p_{3,5})$ respectively. The term $p_{1,2}p_{2,4}p_{4,5}(1 - p_{3,5})$ in $M_r(v_1)$ captures the indirect influence from v_1 to v_5 through the path $\langle v_1, v_2, v_4, v_5 \rangle$, indicating that the LFA strategy does collect influence with multiple intermediate nodes on the path. Note that it is not necessary to scan node v_1 since it does not delivery influence to other nodes

The above illustration tells us that the LFA strategy efficiently calculates the ranking-based marginal influence spread for all nodes, scanning each node only once. Meanwhile, with indirect influence propagation being effectively captured, the LFA strategy provides a good delegate to calculate ranking-based marginal influence spread. We show the numerical results of the LFA strategy and 20,000 times
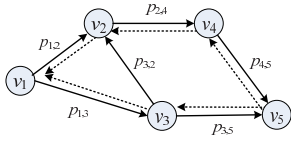
Figure 1: Illustration of the LFA strategy.

Table 2: Estimation on ranking-based marginal influence spread. MC indicates Monte Carlo simulation, and LAF indicates the LAF strategy.

	v_1	v_2	v_3	v_4	v_5
MC	1.29846	1.38800	0.77941	0.89406	0.64007
LAF	1.24000	1.42400	0.76800	0.92800	0.64000

Monte Carlo simulations in the case of setting $p_{u,v} = 0.2$ for all edges as done in uniform independent cascade model. As shown in Table 2, our strategy offers very close results to the time-consuming Monte Carlo simulations.

Finally, we sum up the LFA strategy by explaining why it works remarkably. First, it achieves high efficiency by exploiting the interdependence between ranking and ranking-based marginal influence spread, avoiding the adoption of Monte Carlo simulations done in greedy algorithms. Second, it employs the intermediate nodes as delegates, in a last-to-first manner, to capture both direct and indirect influence propagation among nodes. In this way, ranking-based marginal influence spread could be efficiently calculated via scanning all nodes only once. In addition, the LFA strategy only offers one effective approximation rather than exact calculation of influence spread. This is partly caused by the restriction that lower-ranked nodes only deliver influence score to higher-ranked neighboring nodes. In Section 5, we further improve the LFA strategy via relaxing this restriction.

4.3 Convergence of IMRank

In this section, we first theoretically prove the convergence of IMRank. Then we illustrate the quick convergence of IMRank empirically using a real-word network as example.

THEOREM 2. **Starting from any initial ranking of nodes, IMRank converges to a self-consistent ranking after a finite number of iterations.**

PROOF. We first prove that, for any k, the influence spread of the top-k nodes, denoted as $I(k)$ for convenience, is nondecreasing in the iterative process of IMRank.

After each iteration of IMRank, a ranking r is adjusted to another ranking r'. Since IMRank adjusts all nodes in decreasing order of their current ranking-based influence spread $M_r(v)$, the values of $M_r(v_{r'_i})(1 \le i \le k)$ are the largest k values among all the $M_r(v)$. Hence, there is $I_r(k) = \sum_{1 \le i \le k} M_r(v_{r_i}) \le \sum_{1 \le i \le k} M_r(v_{r'_i})$. Moreover, $I_r(k) = \sum_{1 \le i \le k} M_r(v_{r'_i})$ iff the sets of top-k nodes in ranking r and r' are the same, otherwise $I_r(k) < \sum_{1 \le i \le k} M_r(v_{r'_i})$.

Now let's consider a new ranking r'' obtained from just reordering the top-k nodes of ranking r' in decreasing order of their ranks in ranking r, and keeping the ranks of other nodes still. Apparently, the sets of top-k nodes are the same between ranking r' and r'', thus $I_{r'}(k) = I_{r''}(k)$. Then, for each node $v_{r'_i}$, the set of nodes ranked higher than it in ranking r'' is definitely a subset of the set of
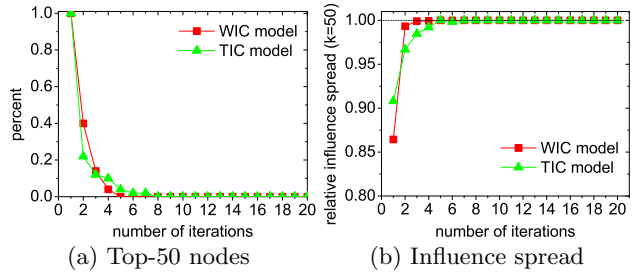

(a) Top-50 nodes   (b) Influence spread

Figure 2: Convergence of IMRank

nodes ranked higher than it in ranking r. According to the submodularity of influence spread function, we can obtain $M_r(v_{r'_i}) \le M_{r''}(v_{r'_i})$ for each node $v_{r'_i}$ $(1 \le i \le k)$. Thus, there is $\sum_{1 \le i \le k} M_r(v_{r'_i}) \le \sum_{1 \le i \le k} M_{r''}(v_{r'_i}) = I_{r''}(k)$. Note we have proved $I_r(k) \le \sum_{1 \le i \le k} M_r(v_{r'_i})$ and $I_{r'}(k) = I_{r''}(k)$. Taken together, we can obtain $I_r(k) \le I_{r'}(k)$, and the equal-sign is tenable iff the sets of the top-k nodes in ranking r and r' are the same, otherwise $I_r(k) < I_{r'}(k)$.

Based on the above conclusion, as long as the current ranking is not a self-consistent ranking, in each iteration all the values of $I(k)(1 \le k \le n)$ are nondecreasing, and at least one $I(k)$ increases. Since each $I(k)$ has an upper bound (i.e., n), IMRank eventually converges to a self-consistent ranking within a finite number of iterations, starting from any initial ranking. □

We now empirically illustrate the quick convergence of IMRank, using a scientific collaboration network, namely HEPT, extracted from the "High Energy Physics-Theory" section of the e-print arXiv website arXiv.org. This network is composed of $15K$ nodes and $59K$ edges. We run IMRank to select 50 seed nodes. Figure 2(a) shows the percent of different nodes in two successive iterations. For two widely-used models, weighted independent cascade (WIC) model [11] and trivalency independent cascade (TIC) model [3], the set of top-50 nodes becomes unchanged after 5 and 8 iterations respectively. Clearly, IMRank converges much quicker than greedy algorithms, which requires k iteration for selecting k seed nodes. Figure 2(b) depicts the influence spread of top-50 nodes. We employ the relative influence spread, i.e., the ratio of the obtained influence spread in each iteration to the obtained influence spread when IMRank converges. IMRank only takes 3 and 5 iterations to achieve a stable and high influence spread under the two models respectively. The influence spread of top-k nodes seems always converges with smaller number of iterations than the convergence of the set of top-k nodes. Therefore, one can stop IMRank safely in practice by checking the change of top-k nodes between two successive iterations.

In sum, we have theoretically and empirically demonstrated the convergence of IMRank. Indeed, the convergence of IMRank could be affected by the accuracy of marginal influence spread estimation. Extensive experiments further show IMRank with the LFA strategy always quickly converges to desirable rankings in Section 6.

4.4 Analysis of initial ranking

Since IMRank is guaranteed to converge to a self-consistent ranking from any initial ranking, it is necessary to extend the discussion to its dependence on the initial ranking: does an arbitrary initial ranking results in a unique convergence? If

not, what initial ranking corresponds to a better result? We explore those questions by empirically simulating IMRank with five typical initial rankings as follows,

- **Random**: Nodes are initially ranked randomly;

- **Degree**: Nodes are initially ranked in descending order of degrees (undirected networks) or out-degrees (directed networks);

- **InversedDegree**: Nodes are initially ranked in ascending order of degrees (undirected networks) or out-degrees (directed networks);

- **Strength**: Nodes are initially ranked in descending order of node strengths (undirected networks) or node out-strengths (directed networks). The node strength is the sum of all weights on its edges. The node out-strength is the sum of all weights on its out-edges;

- **PageRank**: Nodes are initially ranked in descending order of PageRank scores [2], with the default value 0.15 for the damping factor parameter.

Empirical results on the HEPT dataset under the WIC model are reported in Figure 3, to compare the performance of IMRank with different initial rankings, as well as the performance of those rankings alone. We also report the performance of classic greedy algorithm for comparison, implemented with CELF optimization [13]. Performance of IMRank with Random initial ranking and Random ranking alone are averaged over 50 trials.

With the empirical results we conclude:

- With different initial rankings, IMRank could converge to different self-consistent rankings. However, IMRank consistently improves the initial rankings in terms of obtained influence spread.

- Comparable with the greedy algorithm, IMRank with a "good" initial ranking such as Degree, Strength, and PageRank shows indistinguishable performance, shown in a single curve in Figure 3(a). A good initial ranking prefers nodes with high influence;

- IMRank with a "neural" initial ranking such as Random ranking also shows fair performance, slightly poorer than the greedy algorithm and IMRank with a good initial ranking. A neural initial ranking prefers no nodes;

- IMRank with a "bad" initial ranking such as Inversed-Degree ranking shows remarkably improvements upon the initial ranking alone but is dominated by the greedy algorithm. A bad initial ranking prefers nodes with low influence.

Therefore, IMRank is robust to the selection of initial ranking, and IMRank works well with an initial ranking prefering nodes with high influence, which could be obtained efficiently in practice. A possible explanation is the priori bias that, a high-ranked node earns more allocated influence score than a low-ranked node, even with the same topological circumstance. Thus, it helps IMRank to converge to a good ranking if influential nodes are initially ranked high.

Among the three "good" initial rankings with indistinguishable performance, Degree offers a good candidate of initial ranking, since computing the initial ranking consumes a large part in the total running time of IMRank, as shown in Figure 3(b). Moreover, Figure 3(b) also shows that, IMRank runs more than 4 orders of magnitude faster than traditional greedy algorithm.
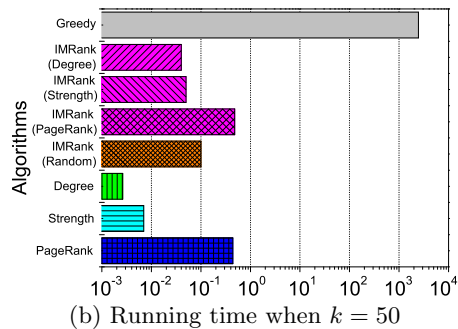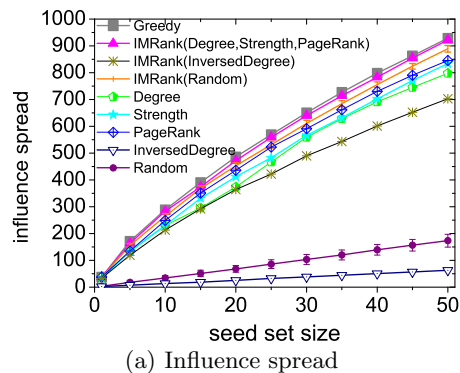

(a) Influence spread

(b) Running time when $k = 50$

Figure 3: Comparison between IMRank with different initial rankings under the WIC model.

5. ADVANCED IMRANK

In the LFA strategy, a node v_{r_i} is only allowed to allocate its influence score to a higher ranked neighboring node v_{r_j}, implying an assumption that a node can only be activated by higher ranked neighbors. That assumption ignores the possibility that, a lower ranked node v_{r_j} activates a higher ranked node v_{r_i} by playing the role of an intermediate agent of another node v_{r_k} with $k < i$. Take the path $\langle v_1, v_3, v_2 \rangle$ in Figure 1 for example. When v_1 is selected as a seed, it is possible that it activates v_3 and then v_3 as an intermediate agent activates v_2.

To combat the above problem, we propose a generalized LFA strategy that trades a slight increase in running time for better accuracy in estimating M_r, and therefore improves the performance of IMRank on influence spread. The generalized LFA strategy relaxes the above assumption and explores *influence paths* instead of higher ranked neighbors, which is introduced below to avoid duplicated computation that a path is contained in another path.

DEFINITION 3. ***Influence path:*** *Given a ranking r, a simple path $d_r(v_{r_j}, v_{r_i}) = \langle v_{r_j}, \cdots, v_{r_i} \rangle$ is an influence path if v_{r_j} is the only node along the path that is ranked higher than v_{r_i}.*

LEMMA 1. *A directed edge $\langle v_{r_j}, v_{r_i} \rangle$ is an influence path if $j < i$.*

LEMMA 2. *A node v_{r_i} allocates influence score to another node v_{r_j} only along an influence path $d_r(v_{r_j}, v_{r_i})$.*

PROOF. Consider a path $d_r(v_{r_j}, v_{r_i})$. It is not negligible only when $j < i$, since v_{r_j} has no chance to trigger a cascade to activate v_{r_i} immediately or eventually if $j > i$. Besides, v_{r_j} has no chance to activate v_{r_i} along this path

if there exists an intermediate node v_{r_k} with $k < j$, since v_{r_k} is triggered earlier. The path should also be neglected if there exists an intermediate node v_{r_k} with $j < k < i$ to avoid duplicated computation, since the influence score allocated from v_{r_k} to v_{r_j} already contains the fraction that v_{r_j} activates v_{r_i}, as discussed in Section 4.2. Therefore, influence score is allocated though a simple path $d_r(v_{r_j}, v_{r_i})$ if and only if v_{r_j} is the only node along the path that is ranked higher than v_{r_i}. $\square$

We denote $\rho_r(v_{r_i}, v_{r_j})$ to the probability that v_{r_i} is activated by v_{r_j} through any influence path. It is equal to the probability that at least one influence path $d_r(v_{r_j}, v_{r_i})$ has all its nodes activated, minus the probability that v_{r_i} is already activated before v_{r_j} attempts. $\rho_r(v_{r_i}, v_{r_j})$ can be obtained as follows,

$$\rho_r(v_{r_i}, v_{r_j}) =$$
$$\left(1 - \prod_{d_r \in D_r(v_{r_j}, v_{r_i})} (1 - p(d_r)) \right) \prod_{1 \le k < i} (1 - \rho_r(v_{r_i}, v_{r_k})),$$
$$(2)$$

where $p(d_r) = \prod_{\langle v_{r_x}, v_{r_y} \rangle \in d_r} p_{v_{r_x}, v_{r_y}}$ is the joint probability that v_{r_j} activates all nodes on a path d_r, and $D_r(v_{r_j}, v_{r_j})$ denotes the set of all influence paths from v_{r_j} to v_{r_i}.

To summarize, the generalized LFA strategy replaces the allocating method: a node v_{r_i} delivers a fraction of its influence score to each higher ranked node reachable with an influence path, instead of each higher ranked neighbor, while replacing $p(\cdot, v_{r_i})$ with $\rho(\cdot, v_{r_i})$.

In practice, we limit the search range within influence paths no longer than l hops, since long paths are expensive to count but contribute little due to low probability to propagate influence. The LFA strategy is a special case of the generalized LFA strategy with $l = 1$.

The time and space complexity of IMRank with the generalized LFA strategy is low. Its space complexity is $O(n)$, storing the value of $M_r(v)$ for each node. Its time complexity mainly depends on l. We denote d_{max} to the largest number of paths end in an arbitrary node with length no more than l. The time required for scanning each node is $O(d_{max} \log d_{max})$, including the time used for searching candidate nodes, sorting candidate nodes by their ranks, and allocating influence. Finally, the time complexity of IMRank is $O(nTd_{max} \log d_{max})$, where T is the number of iterations IMRank takes before convergence. According to extensive experiment results, T is always significantly smaller than k. Besides, d_{max} is usually much smaller than n, e.g. d_{max} equals to the largest indegree among all nodes when $l = 1$. Therefore, the running time of IMRank is affordable.

Figure 4 shows the impact of l on the performance of IMRank, measured with the influence spread on the NEPT network with WIC model and $k = 50$ for example. We report the results of IMRank with Degree and Random initial rankings and the results for other initial rankings are similar. When l increases from 1 to 2, there is an obvious increase on the performance. It supports that a larger l leads to more accurate estimation of marginal influence spread, and thus a better solution. With $l > 2$, the performance increases little, because the propagation probabilities of long paths decrease exponentially with their length, resulting in little necessity to count long paths. On the other hand, the running time of IMRank increases rapidly with l, as shown in the inset figure of Figure 4. In that case $l = 2$ is good enough and fast, while in future practice a suitable l should
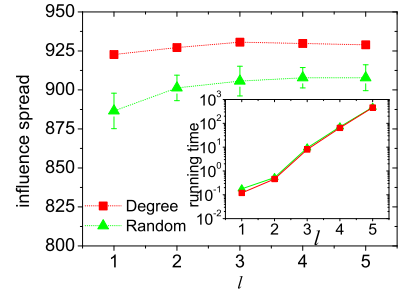

Figure 4: Impact of l on the performance of IMRank.

be carefully selected according to the expected accuracy and affordable computational cost.

6. EXPERIMENTS

In this section, we evaluate IMRank on real-world networks by comparing IMRank with state-of-the-art influence maximization algorithms.

6.1 Experimental Setup

6.1.1 Diffusion models

Experiments are conducted under two widely-used independent cascade models:

- **Weighted independent cascade (WIC) model** [11]: Each edge (u, v) is assigned a propagation probability $p(u, v) = 1/d_v$, where d_v is the indegree of node v.

- **Trivalency independent cascade (TIC) model** [3]: Each edge is assigned a propagation probability selected from $\{0.1, 0.01, 0.001\}$ in a uniform random manner, indicating high, medium and low levels of influence.

6.1.2 Baseline algorithms

The compared algorithms include two implementations of IMRank and two state-of-the-art heuristic algorithms, i.e., PMIA and IRIE. Details are as follows:

- **IMRank1**: This is the IMRank with Degree as initial ranking and $l = 1$. According to the analysis of section 4.3, we set its stopping criteria as when the sets of top-k nodes are the same during two successive iterations or the iteration runs 10 rounds.

- **IMRank2**: This is the IMRank with Degree as initial ranking and $l = 2$, with the same stopping criteria as IMRank1.

- **PMIA**: This heuristic algorithm estimates influence spread based on maximum influence paths [3]. We use the recommended parameter setting $\theta = 1/320$.

- **IRIE**: This heuristic algorithm integrates influence ranking with influence estimation [10]. Its parameters α and θ are setted to be 0.7 and $1/320$ respectively, and the maximum times of iterations for initial round and subsequent rounds are 20 and 5 respectively as recommended.
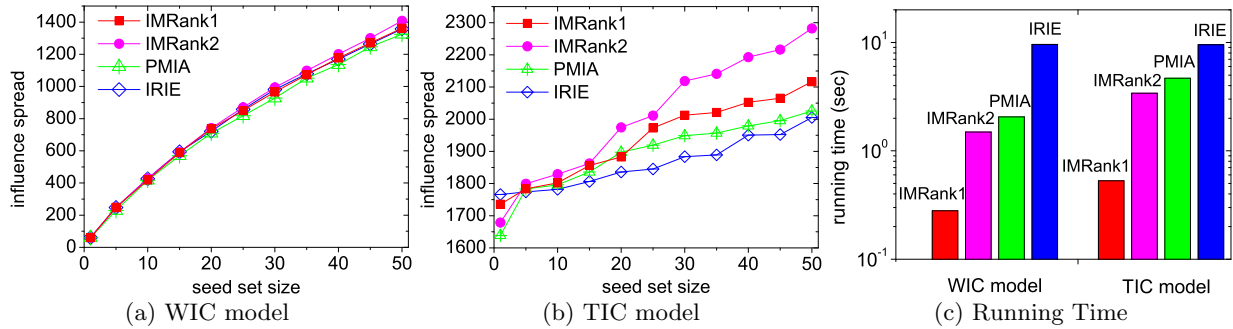
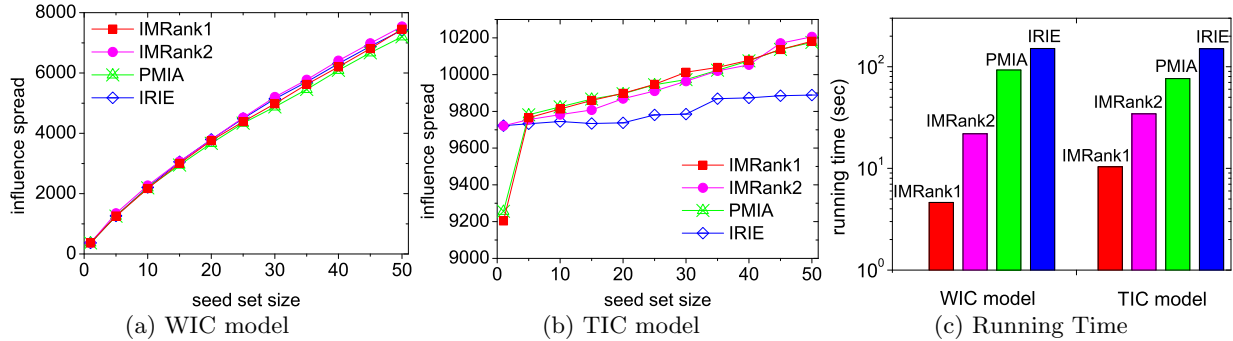Figure 5: Influence spread and running time on the PHY dataset

Figure 6: Influence spread and running time on the DBLP dataset

Table 3: Statistics of test networks

Datasets	#Nodes	#Edges	Directed?
PHY	37K	231K	undirected
DBLP	655K	2M	undirected
EPINIONS	76K	509K	directed
DOUBAN	552K	22M	directed
LIVEJOURNAL	4M	69M	directed

6.1.3 Datasets

Experiments are conducted on five real-world networks, two undirected scientific collaboration networks and three directed online social networks. Table 3 gives basic statistics of those networks. One of the two scientific collaboration networks, denoted as PHY, is obtained from the complete list of papers of the Physics section of the e-print arXiv website. The other one, denoted as DBLP, is extracted from the DBLP Computer Science Bibliography [1]. The three online social networks are EPINIONS, DOUBAN, and LIVEJOURNAL [2], respectively extracted from the websites of epinions.com, douban.com and livejournal.com. In the EPINIONS dataset, an edge between two users u and v, denoted as $\langle u, v \rangle$, represents that user u trusts user v. In the DOUBAN dataset [8], an edge between two users u and v represents that user u follows user v. In the LIVEJOURNAL network [1], an edge between two users u and v represents that user u declares user v as his/her friend. We choose

these five networks based on the consideration that these networks possess various kinds of relationships and different sizes ranging from hundreds of thousands edges to millions of edges. Actually we test these algorithms on many other networks, and the results are similar. Limited by space, results on these networks are not included in this paper.

All experiments are conducted on a server with 1.9GHz Quad-Core AMD Opteron(tm) Processor 8347HEx4 and 64G memory.

6.2 Experimental results

We evaluate IMRank on real-world networks by comparing it with state-of-the-art algorithms. Evaluation metrics include influence spread and running time. For the comparison of obtained influence spread, we test the cases of $k = 1, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50$. For the comparison of running time, we focus on the typical case $k = 50$. Each figure of Figures 5-8 shows the results on a certain network. The first two subfigures give the results of influence spread under the WIC model and the TIC model respectively, and the last one gives the results of running time.

Figure 5 shows the experimental results on the PHY dataset. Under the WIC model, IMRank2 achieves the best influence spread, followed by IMRank1, outperforming PMIA and IRIE. The distinguished accuracy of IMRank2 is attributed to the fact that, IMRank2 explores more influence paths to accurately estimate marginal influence spread. PMIA exhibits the worst performance, 6.3% lower influence spread than IMRank2 when $k = 50$. Under the TIC model, as shown in Figure 5(b), similar results are obtained, and the gaps between those algorithms become more visible. For influence spread, IMRank2 and IMRank1 are the top two algorithms while PMIA slightly outperforms IRIE. The influence spread obtained by IMRank2 is 13.8% and 12.7%
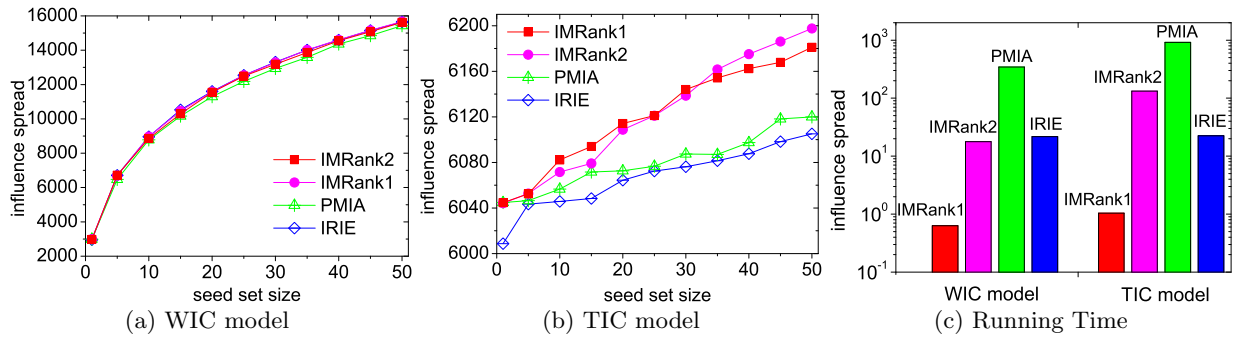
Figure 7: Influence spread and running time on the EPINIONS dataset
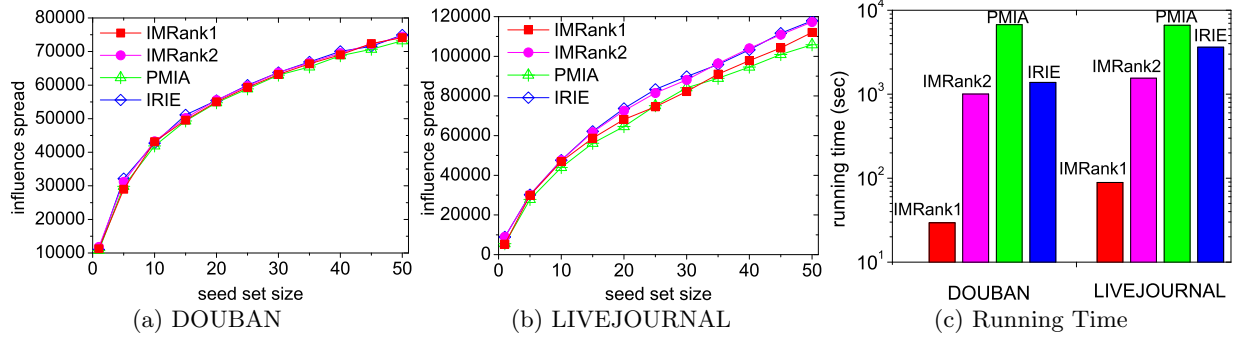

Figure 8: Influence spread and running time on the DOUBAN and LIVEJOURNAL datasets

higher than that obtained by IRIE and PMIA respectively. Moreover, as shown in Figure 5(c), IMRank1 and IMRank2 run faster than the competing algorithms under both the two models. IMRank1 is the fastest one followed by IMRank2 ,while PMIA takes the third place and IRIE runs slowest. In particular, the running times of IRIE and PMIA are 30 times and 10 times longer than the running time of IMRank1 under the WIC model respectively, and 18 times and 9 times longer than that of IMRank1 under the TIC model. With the running time dramatically reduced, IMRank1 still achieves better influence spread which is about 5.5% and 4.5% higher than that of IRIE and PMIA respectively. The consistent performance of IMRank1 and IMRank2 demonstrates the effectiveness of IMRank. The inconsistent performance of PMIA and IRIE under the two diffusion models illustrates that both PMIA and IRIE are unstable.

Figure 6 shows the results on DBLP dataset. The four algorithms performs similar on this dataset as on the PHY dataset. For the WIC model, IMRank2 achieves the highest influence spread and IMRank1 is the fastest one. In particular, when $k = 50$, the highest influence spread is achieved by IMRank2 and its running time is less than PMIA and IRIE. IMRank1 obtains similar influence spread to PMIA and its running time is one order of magnitude smaller than that of PMIA. For the TIC model, IMRank1, IMRank2 and PMIA achieve very similar influence spread, which is significantly higher than the influence spread achieved by IRIE. Moreover, IMRank1 runs nearly 8 times and 13 times faster than PMIA and IRIE.

Figure 7 gives the results on EPINIONS dataset. For the WIC model, IMRank1 and IMRank2 run faster than PMIA and IRIE. In particular, compared to PMIA, IMRank1 runs two orders of magnitudes faster and IMRank2 runs one

order of magnitude faster. For the TIC model, IMRank2 achieves the best influence spread and IMRank1 takes the second place. Both IMRank1 and IMRank2 significantly outperform PMIA and IRIE. Moreover, the running time of IMRank1 is only 0.1% of the running time of PMIA and 5% of that of IRIE. With similar running time, IMRank2 achieves significant higher influence spread than that of PMIA and IRIE.

Figure 8 shows the results on the DOUBAN and LIVE-JOURNAL datasets. The number of edges of DOUBAN and LIVEJOURNAL is 22 millions and 69 millions respectively. Here we only give the results under the WIC model. On the DOUBAN network, the four algorithms achieve comparable influence spread. However, IMRank1 runs more than two orders of magnitude faster than PMIA and more than one order of magnitude faster than IRIE. On the LIVEJOURNAL network, IMRank2 and IRIE have similar influence spread, while IMRank1 follows and PMIA achieves the lowest influence spread. Note that IMRank2 runs faster than IRIE, and IMRank1 runs much faster than PMIA. We do not show the results under the TIC model since no visible difference is observed among the four tested algorithms. This is due to the fact that selecting one influential node always achieves a very large influence spread on DOUBAN and LIVEJOURNAL networks, and no increase of influence spread can be gained by adding a new seed. Such phenomenon has been observed and discussed in [11] and [3]. The possible reason is that the influence networks generated by the TIC model on the two networks have a relatively large strongly connected component.

These experiments show that, in different scenarios, IMRank consistently perform well while PMIA and IRIE perform unstable. IMRank1 always runs more than one order

of magnitude faster than PMIA and IRIE, and achieves similar influence spread as them. IMRank2 consistently provides better influence spread than PMIA and IRIE, and runs faster than them. In addition, under the two different diffusion models, IMRank shows similar improvements on influence spread from the relative improvement angle. However, the improvements of IMRank seems more visible under the TIC model. With respect to this model, the links between influential nodes have high probability to be assigned a relative high propagation probability($p = 0.1$), which counts against heuristic methods to accurate estimate influence spread. Hence, we speculate that, IMRank has more advantages than PMIA and IRIE to handle such cases.

7. CONCLUSIONS

In this paper, we investigated influence maximization from a novel ranking perspective. We proposed an efficient iterative framework IMRank to explore the benefits of accurate greedy algorithms and efficient heuristic estimation of influence spread. This framework effectively tunes any initial ranking into a self-consistent ranking in an iterative manner through fully leveraging the interplay between the ranking of nodes and their ranking-based marginal influence spread. A last-to-first allocating strategy is further proposed to efficiently estimate the ranking-based marginal influence spread under the independent cascade model. This strategy is elaborately designed according to the characteristics of the independent cascade model and the ranking-based marginal influence spread. We further generalize the last-to-first allocating strategy in order to achieve more accurate estimation. We also prove the convergence of IMRank and analyze the impact of initial ranking. Moreover, IMRank always works well with simple heuristic rankings, such as degree, strength. Extensive experiments on large scale real-world social networks demonstrate the efficiency of IMRank. Its scalability outperforms the state-of-the-art heuristics while its accuracy is comparable to the greedy algorithms.

For future work, we will try to analyze the accuracy of IMRank theoretically. Moreover, we believe our proposed iterative framework is of generality for some cases which greedy algorithm is suitable for, and it can also be adapted in some re-ranking applications. We will try to extend it to other problems beyond influence maximization, such as diversity problem in information retrieval.

8. ACKNOWLEDGMENTS

9. REFERENCES

[1] L. Backstrom, D. Huttenlocher, J. Kleinberg, and X. Lan. Group formation in large social networks: membership, growth, and evolution. In *KDD'06*, pages 44–54, 2006.

[2] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. In *WWW'98*, pages 107–117, 1998.

[3] W. Chen, C. Wang, and Y. Wang. Scalable influence maximization for prevalent viral marketing in large-scale social networks. In *KDD'10*, pages 1029–1038, 2010.

[4] W. Chen, Y. Wang, and S. Yang. Efficient influence maximization in social networks. In *KDD'09*, pages 199–208, 2009.

[5] S. Cheng, H. Shen, J. Huang, G. Zhang, and X. Cheng. Staticgreedy: Solving the scalability-accuracy dilemma in influence maximization. In *CIKM'13*, 2013.

[6] P. Domingos and M. Richardson. Mining the network value of customers. In *KDD'01*, pages 57–66, 2001.

[7] A. Goyal, W. Lu, and L. V. Lakshmanan. Celf++: optimizing the greedy algorithm for influence maximization in social networks. In *WWW'11*, pages 47–48, 2011.

[8] J. Huang, X.-Q. Cheng, H.-W. Shen, T. Zhou, and X. Jin. Exploring social influence via posterior effect of word-of-mouth recommendations. In *WSDM'12*, WSDM '12, pages 573–582, 2012.

[9] Q. Jiang, G. Song, C. Gao, Y. Wang, W. Si, and K. Xie. Simulated annealing based influence maximization in social networks. In *AAAI'11*, 2011.

[10] K. Jung, W. Heo, and W. Chen. Irie: Scalable and robust influence maximization in social networks. In *ICDM'12*, pages 918–923, 2012.

[11] D. Kempe, J. Kleinberg, and E. Tardos. Maximizing the spread of influence through a social network. In *KDD'03*, pages 137–146, 2003.

[12] M. Kimura, K. Saito, R. Nakano, and H. Motoda. Extracting influential nodes on a social network for information diffusion. *Data Mining and Knowledge Discovery*, 20(1):70–97, 2010.

[13] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. VanBriesen, and N. Glance. Cost-effective outbreak detection in networks. In *KDD'07*, pages 420–429, 2007.

[14] M. Mathioudakis, F. Bonchi, C. Castillo, A. Gionis, and A. Ukkonen. Sparsification of influence networks. In *KDD'11*, pages 529–537, 2011.

[15] R. Narayanam and Y. Narahari. A shapley value-based approach to discover influential nodes in social networks. *IEEE Transactions on Automation Science and Engineering*, 8(1):130–147, 2011.

[16] M. Richardson and P. Domingos. Mining knowledge-sharing sites for viral marketing. In *KDD'02*, pages 61–70, 2002.

[17] D. Sheldon, B. Dilkina, A. N. Elmachtoub, R. Finseth, A. Sabharwal, J. Conrad, C. P. Gomes, D. Shmoys, W. Allen, O. Amundsen, and W. Vaughan. Maximizing the spread of cascades using network design. In *UAI'10*, pages 517–526, 2010.

[18] J. Tang, J. Sun, C. Wang, and Z. Yang. Social influence analysis in large-scale networks. In *KDD'09*, pages 807–816, 2009.

[19] C. Wang, W. Chen, and Y. Wang. Scalable influence maximization for independent cascade model in large-scale social networks. *Data Mining and Knowledge Discovery*, 25(3):545–576, 2012.

[20] Y. Wang, G. Cong, G. Song, and K. Xie. Community-based greedy algorithm for mining top-k influential nodes in mobile social networks. In *KDD'10*, pages 1039–1048, 2010.