석사 학위논문

Master's Thesis

특정 사용자로의 영향력을 최대화하는 친구 추천 방법

Maximizing Influence over a Target User through Friend Recommendation

김 선 동 (金 先 東 Sundong Kim)

산업 및 시스템 공학과

Department of Industrial & Systems Engineering

KAIST

2015

특정 사용자로의 영향력을 최대화하는 친구 추천 방법

Maximizing Influence over a Target User through Friend Recommendation

Maximizing Influence over a Target User through Friend Recommendation

Advisor : Professor Kyoung-Kuk Kim

by

Sundong Kim
Department of Industrial & Systems Engineering
KAIST

A thesis submitted to the faculty of KAIST in partial fulfillment of the requirements for the degree of in the Department of Industrial & Systems Engineering . The study was conducted in accordance with Code of Research Ethics[1].

2014. 12. 05.

Approved by

Professor Kyoung-Kuk Kim

[Advisor]

[1]Declaration of Ethical Conduct in Research: I, as a graduate student of KAIST, hereby declare that I have not committed any acts that may damage the credibility of my research. These include, but are not limited to: falsification, thesis written by someone else, distortion of research findings or plagiarism. I affirm that my thesis contains honest conclusions based on my own careful research under the guidance of my thesis advisor.

특정 사용자로의 영향력을 최대화하는 친구 추천 방법

김 선 동

위 논문은 한국과학기술원 석사학위논문으로
학위논문심사위원회에서 심사 통과하였음.

2014년 12월 05일

심사위원장 김 경 국 (인)

심사위원 이 재 길 (인)

심사위원 James Morrison (인)

특정 사용자로의 영향력을 최대화하는 친구 추천 방법

Maximizing Influence over a Target User through Friend Recommendation

MIE
20133101

김 선 동. Sundong Kim. Maximizing Influence over a Target User through Friend Recommendation. 특정 사용자로의 영향력을 최대화하는 친구 추천 방법. Department of Industrial & Systems Engineering . 2015. 31p. Advisor Prof. Kyoung-Kuk Kim. Text in English.

ABSTRACT

Given a graph, with a source node that needs to maximize its influence over a specific target node, a set of nodes need to be determined that will enable information flow from the source to the target node. By solving this problem, we can enable a user to receive more attention from a specific target user by forming useful connections between the two. Current friend recommendation algorithms focus on suggesting nodes that are similar to the user, but these algorithms are not designed to tackle the maximum influence issue. Based on the observation that information propagation on online social networks is enabled by the sharing activity, we define the influence of a source node over target as a variation of Katz centrality. In addition, we model the reluctance, to account for awkwardness between two users. With these measures, we propose k-node suggestion problem. However, the optimization problem of recommending a node set that maximizes influence is NP-hard. Therefore, in this paper, we suggest an algorithm to find a node set sequentially. Next, we model reluctance between two nodes, which is inversely related to node similarity. Reluctance is based on the intuition that people not related with the user, might not accept the user's online connection request. By considering reluctance, we could identify the nodes that have a higher probability of refusing the connection request. We address this problem by proposing an Incremental Katz Approximation (IKA) algorithm that is designed to search the top-k nodes with high accuracy and has the ability to run on a large graph. The IKA algorithm can handle large graphs by approximating influence measure using Monte-Carlo simulation. This algorithm updates information diffusion incrementally to find the node that can maximize influence. We discuss the complexity of our algorithm and the evaluation results demonstrate the performance and scalability of our algorithm. We also report interesting behavior of the sequential recommendation result. In summary, our algorithm is effectively recommend friends such that they can improve influence of the source user on the target user.

Keywords - Network with Target; Friend Recommendation; Maximizing influence; Information Propagation; Incremental Algorithm for Katz Centrality;

Contents

List of Tables

List of Figures

Chapter 1. Introduction

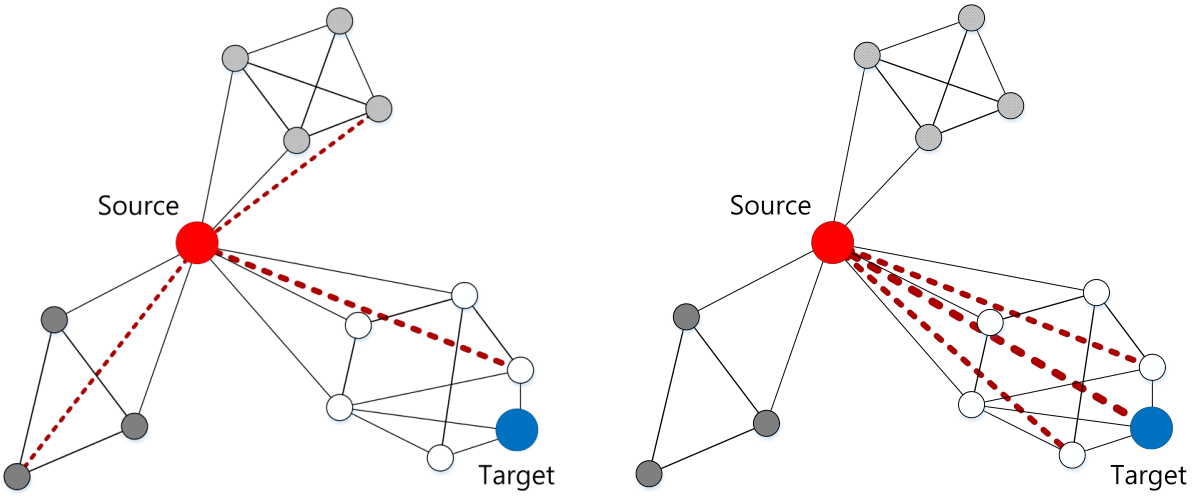
People use social networking services such as Facebook, Twitter, and Instagram extensively. Almost a billion users are active on such applications, on a daily basis. Several of these users send friend requests to other users. One of the most important features of any social networking service is the ability to suggest friends[37]. Each application uses its own algorithm to suggest probable friends to a user[12]. For example, an application might suggest people belonging to a group recently joined by the user, or people related to the user's current job or school.

Although these recommendations help users form connections within their groups, and suggest people that the users probably know, the algorithm does not recommend people the users want to connect with, or groups they want to join. For example, a graduate student specializing in the subject area of data mining may want to connect with a famous professor in the same field. The student might send a friend request to the professor directly, but there is a high probability that the professor might not respond because he/she might not have heard of the student. What if there is an application that enables the user to select a target user, and then suggests friends accordingly, so that the user is noticed by the target? The student can then successfully build a connection with the professor. Based on the various studies and research on how people create social ties, maximizing exposure to a specific target can be one way of enabling people to create social ties. In addition to the online social network, the concept of maximizing exposure to the specific target user can be extended to targeting particular advertisements to a user.

In this paper, we propose a friend recommendation algorithm for scenarios where the source node wants to connect with a specific target node. Based on past observations, we know that a post on a social network can propagate through all possible paths between two vertices. Therefore, we intend to suggest friends who can facilitate information flow from the source to the target. For example, if a source node and a target node are not directly connected, suggesting intermediate nodes can help facilitate communication between the two nodes. Additionally, by suggesting the target's neighboring nodes to the source node, we can increase the possibility of a direct connection. Figure 1.1 shows the probable changes to friend suggestions when a source picks a specific target.

Before tackling the recommendation problem, we modeled the process of information diffusion through social network. We considered common social network scenarios such as when an article is shared with the direct neighbor of an author, and propagates over the corresponding two-hop neighbor if the author's direct neighbor shares or comments on the post. In our work, we refer to all conveying actions as "sharing" and define the sharing probability as a constant. We also specified a setting that each user in the network has a homogeneous behavior of posting. In this setting, the number of posts that a user can receive can be determined solely by their topological location in the network. Among the posts that the target node receives, there might be some posts that may have originated from the source node, which we defined as the source node's influence over the target node.

Next, we modeled reluctance between two nodes, which is inversely related to node similarity. Reluctance is based on the intuition that people not related with the user, might not accept the user's online connection request.



(a) Current friend recommendation algorithm suggests node from each cluster which is not connected with the user

(b) When the user has a target node 2 to maximize its influence, first we can suggest the node 1 which shorten the distance to the target. And then we enables to connect direct with the target, finally add the node 3 which boosts information transfer

Figure 1.1: Friend recommendation in online social network

By considering reluctance, we could identify the nodes that have a higher probability of refusing the connection request.

We define this scenario as the k-node suggestion problem in online social networks. This problem is an optimization problem, where an existing source node wants to maximize its exposure to a certain target node. By drawing a graph using the source node and target node as inputs, we recommend k-nodes to the source user, where k is unknown. The main objective was to maximize influence gain by having k new edges and each recommendation should meet the reluctance threshold. The expected result is a set of k users for recommendation.

However, we found that determining a k-node that maximizes the source node's influence is intractable. First, the benefit of having a new friend can vary depending on the sequence of suggestions, and all $\binom{n}{k}$ possible cases need to be calculated. Therefore, k-nodes should be found one-by-one using a greedy algorithm. However, this involves complex calculations.

First, the influence score needs to be represented as a division of two Katz centralities. This concept will be discussed further in the Section 3.3 of this paper. To calculate a Katz centrality precisely, matrix inversion needs to be performed, which requires extensive computation. Furthermore, a node that maximizes the influence needs to be found, which requires calculating the Katz centrality for a new graph by changing the additional edge. This process is completed when there are no beneficial nodes left. In addition, no previous research regarding the calculation of Katz centrality on a streaming graph could be found.

To solve this problem, we proposed our algorithm—Incremental Katz Approximation (IKA). In this algorithm, we first decreased the candidate set, and then applied Monte-Carlo simulation to approximate the Katz centrality. The influence score was calculated by using the results of the two simulations. Using the simulation results, we could update the random diffusion and calculate the influence by taking into account the effect of a new edge.

We calculated the algorithm complexity, and supported our calculations experimentally by measuring the performance of the proposed algorithm against non-incremental greedy algorithm using various sizes and structures of synthetic network and real-world network. Additionally, we found that even short random diffusions are sufficient to find the top-k node recommendations. Based on our results, we concluded that our algorithm is superior compared to topology-based recommendation algorithms that use various node-to-node similarity measures. Finally, we analyzed the characteristics of a recommendation set by changing the topological location of the source and target node.

We summarize our contribution as follows:

- The problem: We undertake a friend recommendation problem in online social networks when a user wants to maximize his/her influence over a specific target user.
- Design a new measure: We define an influence measure and analyze the effect of having new connection
- Performance: We design the IKA algorithm, which incrementally approximates Katz centralities and proves its performance over topology-based recommendation algorithms.
- Discovery: We experiment with diverse networks and interpret the characteristics of recommended nodes.

The rest of the paper is organized as follows.

- In Chapter 2, we examine related works about friend recommendation and incremental algorithms.
- In Chapter 3, we introduce our social network model and formulate k-node suggestion problem to derive the best recommendation result.
- In Chapter 4, we introduce greedy algorithms for sequential node suggestion.
  - In the first section, we present an exact algorithm and two variations that focus on candidate reduction.
  - In the next section, we present our IKA algorithm using Monte-Carlo simulation.
  - In the last section, we discuss the complexity of both algorithms.
- In Chapter 5, we describe the experiments conducted using synthetic and real networks.
- In Chapter 6, we conclude this thesis.

Chapter 2. Related Work

In this paper, we propose a personalized friend recommendation problem when there is a specific target for user. So in this section, we provide related works on various research topics such as friend recommendation, link prediction, approximation and incremental algorithm for centrality, and analysis of evolving social network.

2.1. Friend Recommendation

Friend recommendation is one of the biggest research field in social networks [2]. By recommending friend, users can make new connections and expand their ego network [30]. There are mainly two friend recommendation approaches; topology-based approach and content-based approach. Topology-based approach exploits properties from the network structure and calculates node-to-node similarities. Recommendation node will be the node with the highest similarity. Jaccard similarity [39] and SimRank [17] are well-known node-to-node similarity measures. Zhao et al [48] proposed P-Rank, which is the general version of structural similarity. And Leicht et al [23] proposed a similarity measure viewed as a weighted count of number of paths having possible length between two vertices. Content-based approach tries to recommend item similar to those a given user has liked before. Collaborative filtering is widely used in content-based approach. Comprehensive knowledge of content-based approach and collaborative filtering is covered in [29, 43]. In addition to that, Liu et al suggest hybrid method [27] which refined content-based similarity [32] and topology-based approach [41], and combined them together to get better recommendation from their Twitter dataset. Different types of algorithms are used in different contexts. Lo et al [28] developed topology-based model to estimate relationship strength by exploiting real message interaction and Armentano et al [3] developed an unsupervised model in twitter environment to identify users who can be considered as good information sources. In this paper, we apply a topology-based method where every user has homogeneous posting behavior. And we consider the general social network environment where people share their neighbor's article to his/her own neighbor. Similar to [23], we assume that the target user can get information through all different paths between source and target. And we define the influence of source node as the proportion of source node's post covered in target node's newsfeed. Final recommendation is done by selecting a node one at a time, the node which maximizes the source nodes influence over the target.

2.2. Link Prediction

Link prediction problem is considered similarly as friend recommendation problem. Main goal of this problem is to predict implicit interactions, based on observed links. Nowell and Kleinberg [25] formalized the question of inferring which new interaction would occur in the near future, given network snapshot. And they uses different predictors such as graph distance, common neighbors, Jaccard [39], SimRank [17], Adamic-Adar [1], Katz [20]. From their experiment, performance of Adamic-Adar [1] is overall satisfactory compared to other predictors. This

implies that we can represent the connecting behavior in real network by Adamic-Adar coefficient. Leskovec et al [24] predicted the sign of link in online social networks. In our paper, we designed reluctance between user and recommended node as a negative exponential to Adamic-Adar coefficient. Subsequently, maximum influence with minimum reluctance node can be suggested to the user.

2.3. Approximation and Incremental Algorithm for Centrality

By finding the node with maximum influence, we need to calculate the influence in accordance with adding new edge to the original graph. Approximation and incremental algorithm might increase computation speed significantly. Kas [19] dealt with incremental algorithm on closeness, betweenness, and k-centrality. In order to compute those centralities incrementally, the only information needed is the all-pair shortest path. And newly generated edge will affect the nodes whose shortest path lies on. And Okamoto et al [35] combined existing method on calculating exact value and approximate value of close centrality and efficiently find top-k vertices. However, influence measure in our paper is modeled as Katz centrality. In order to calculate the Katz centrality, we need an information of all walks between two nodes. Even a single edge enables to affect a large portion of the length of walk, exact calculation through partial update is almost impossible. Also, method in [35] cannot be adapted to Katz centrality since it is a variant of eigenvector centrality. So the calculation requires computing all centrality values for all vertices unavoidably, even if we only need top-k values. However, Bahmani et al [4] analyzed the efficiency of Monte-Carlo methods for incremental computation of PageRank [36], personalized PageRank [18] on evolving social networks. This work is meaningful for us since PageRank is a variant of eigenvector centrality. From their work, we incarnated to approximate Katz centrality by Monte-Carlo simulation and incrementally updating the influence value by expanding random diffusion.

2.4. Analysis of Evolving Social Network

Social scientists has discussed about how individuals create and break off the relationship and form social networks [22]. For example, homophily is well-known tendency that similar individuals interact each other [31]. However, people cannot have much information about the other part of global social network. So they tend to make a relation from their surroundings. For example, individual makes new connection with someone who are friend of friends and this is known as triadic closure [42], and form locally dense cluster [5]. However, there is also a theory of structural hole [21] that individuals who seek new information or resources may benefit from across to bridge, which also emphasizes the strength of weak tie [9]. In our networks, individual forms a social connection by increasing its influence over target. If the distance between source node and target node is long, then the user makes connections to series of nodes which decreases the distance between two vertices, this action can be interpreted as having benefit from its weak ties.

Chapter 3. Problem Definition

In this chapter, we introduce our social network model and formulate the k-node suggestion problem to derive the best recommendation result. In Section 3.1, we introduce our information propagation model in online social network. And we modeled influence and reluctance between two vertices. In Section 3.2, we introduce k-node suggestion problem. Main goal of this problem is to find the set of nodes which maximize exposure to specific target node.

3.1. Information Propagation Model

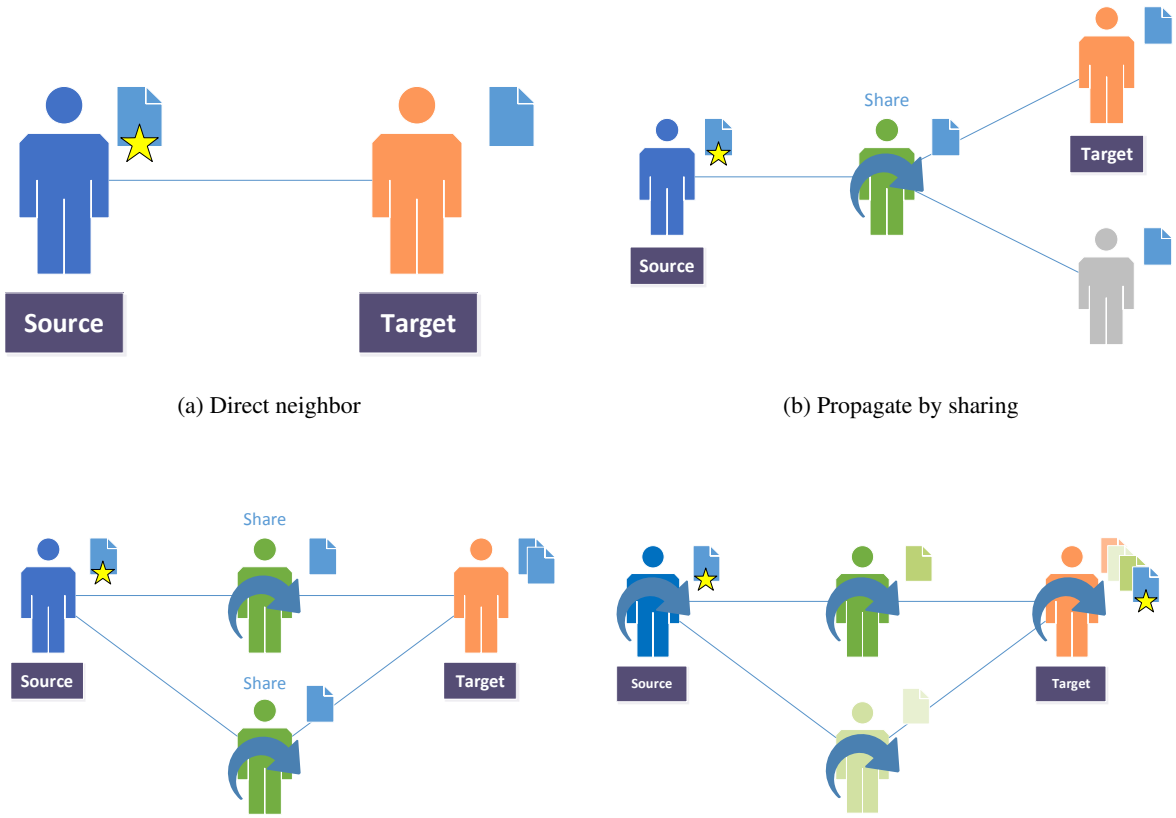
In this section, we model the process of information propagation through a social network. We focus on the online social network environment where each individual has its own feed which displays the recent articles of its neighbor. As shown in Figure 3.1(a), that article uploaded by source node appears to its neighboring node's feed without any action. Second, this article enables to propagate on friend of friend, and this process is done by "sharing" actitities such as share and reply to someone's post. Figure 3.1(b) describes that source node's article is transferred to its two-hop neighbor due to the sharing activity by intermediate node. In our model, we fix the sharing probability as a constant. In online-social network, people can receive an article several times due to multiple sharing by its neighbor. For example, in Figure 3.1(c), target node will receive same article twice if there are two intermediate nodes shared the article. For easier explanation, in Figure 3.1(c) we ignore the case that source node received its own article again by sharing activity. To put it simply, the article by source node can be transferred through any walks between source node and target node. Figure 3.1(d) shows that every node acts as a source node and intermediary node at the same time. In this example, target node's feed is filled with the post that four different individual uploaded. And here we are interested in the ratio of source node's post on target node's feed. We denote that ratio as influence of source node over target.

Proposition 3.1 *Assume that the only individual who upload its post is source node n_s. Let S is the set of all walks from n_s to target node n_t, and $length_w$ is a length of each walk w. By having fixed sharing probability p_s, then the expected number of article that n_t received is:*

$$r_{st} = \sum\nolimits_{w \in S} p_s^{length_w - 1} \tag{3.1}$$

Definition 3.2 *(Influence) Let r_{st} is number of article that n_t received from the network with single uploading node n_s. By considering that every node has uploading behavior, then the probability of n_s's article covered in n_t's feed is:*

$$I_{st} = \frac{r_{st}}{\sum_s r_{st}} \tag{3.2}$$



(a) Direct neighbor

(b) Propagate by sharing

(c) Message can be duplicated

(d) Every node acts as a source node

Figure 3.1: Information propagation in online social network

Remark Consider all walks in graph G. If $N(dist = i)$ is the number of walks which length is i from n_t. Then influence can be rewritten as

$$I_{st} = \frac{r_{st}}{\sum_s r_{st}} = \frac{r_{st}}{\sum_i N(dist = i) * p^{i-1}} \tag{3.3}$$

Corollary 3.3 *From the network where every node posts an article, Total number of article that n_t received can be represented as Katz centrality of node n_t.*

Proof Total number of article that n_t received throughout the network is represented as $\sum_s r_{st}$. In remark, we introduce that this can be represented as the weighted summation according to the distance. Let A be the adjacency matrix of a network, then Katz centrality for node t is

$$C_{Katz}(t) = \sum_{k=1}^{\infty} \sum_{s=1}^{n} \alpha^k (A^k)_{st} \tag{3.4}$$

By considering attenuation factor α as a sharing probability p, $\sum_s r_{st} = \frac{C_{Katz}(t)}{\alpha}$. Denominator α appears since we assume that direct neighbor received an article without any action. In order to guarantee the convergence, attenuation factor α has to be smaller than the largest eigenvalue of the adjacency matrix.

Remark In a similar way, the number of source node's article that n_t received can be represented as Personalized Katz centrality of node n_t. In the case of single uploader n_s, the number of article by that n_t received throughout the network is represented as r_{st}. And we can represent it as

$$r_{st} = \frac{C_{PKatz}(t)}{\alpha} = \sum_{k=1}^{\infty} \alpha^{k-1} (A^k)_{st} \tag{3.5}$$

Katz centrality is represented as matrix inversion.

$$\overrightarrow{C}_{katz} = ((I - \alpha A^T)^{-1} - I)\overrightarrow{I} \tag{3.6}$$

We are able to compute the Katz centrality numerically by power method. We can get the Katz centrality for node i by this equation.

$$x_i = \alpha \sum\nolimits_j A_{ij} x_j + \beta \tag{3.7}$$

where A is the adjacency matrix of graph G with eigenvalue λ. In order to guarantee convergence, $\alpha < \frac{1}{\lambda_{max}}$, and parameter β control the initial centrality. $\overrightarrow{\beta} = \mathbf{1}$ for Katz centrality and $\overrightarrow{\beta} = \mathbf{1}_s$ for Personalized Katz centrality.

Remark We can represent influence I_{st} by using Katz centrality.

$$I_{st} = \frac{C_{PKatz}(t)}{C_{Katz}(t)} \tag{3.8}$$

We observed that influence of source node over target can be represented using Katz centrality. However, modeling influence by Katz centrality also have some weaknesses. First, Katz centrality is eigenvector centrality so we need to compute for whole network to get the value for specific node. Second, incremental algorithm to calculate Katz centrality is not known.

Then, we modeled the reluctance between two individuals. The concept of reluctance is awkwardness between two nodes. When a system suggests a new node to source node, source node might not request a friend to suggested node since they do not know each other. Even though he sends a request to suggested node, it might not accept the friend request from source node. To sum up this idea, we define reluctance as negative exponentially proportional to Adamic-Adar similarity [1].

Definition 3.4 *(Reluctance) Let ρ_{ij} is the reluctance between the two nodes n_i and n_j. Then ρ_{ij} is defined as:*

$$\rho_{ij} = e^{-sim(i,j)} \tag{3.9}$$

where

$$sim(i,j) = \sum\nolimits_{n \in \Gamma(i) \cap \Gamma(j)} \frac{1}{\log|\Gamma(n)|} \tag{3.10}$$

$\Gamma(n)$ refers a set of neighbors of node n and $|\Gamma(n)|$ is the size of $\Gamma(n)$. And the range of Adamic-Adar similarity is between 0 to ∞.

3.2. K-node Suggestion Problem

In this thesis, we consider the network with specific target where source node n_s wants to maximize its influence over a specific target n_t. To solve this problem, we suggest relevant friend recommendation to increase the information flow from n_s to n_t. More specifically, we want our suggestion will control the portion of source node's article among articles that target received. However, increasing influence by recommending nodes is not a trivial problem. Intuitively, we can think that I_{st} increases by having another connections from n_s to intermediate node n_t. Although the number of routes between n_s to n_t increases, the number of routes from other nodes to n_t also increases. That makes our problem non-trivial. And yet there is another issue. If we only consider the maximization of influence, then the friend recommendation algorithm might suggest n_t directly or only suggests the nodes which is topologically located next to n_t. However, those nodes are not related with n_s. So we have to consider that the suggestions are at least relevant to source node, which means that we need to consider the reluctance between n_s and our suggestion. In this section, we formalize a node suggestion problem to maximize ΔI_{st}. The basic formulation of the k-node suggestion problem can be represented as follows:

$$\begin{aligned} &\text{maximize} \quad I_{st}(G') - I_{st}(G) \\ &\text{subject to} \quad \rho_{si} < 1 \qquad\qquad i = 1, ..., k \end{aligned} \tag{3.11}$$

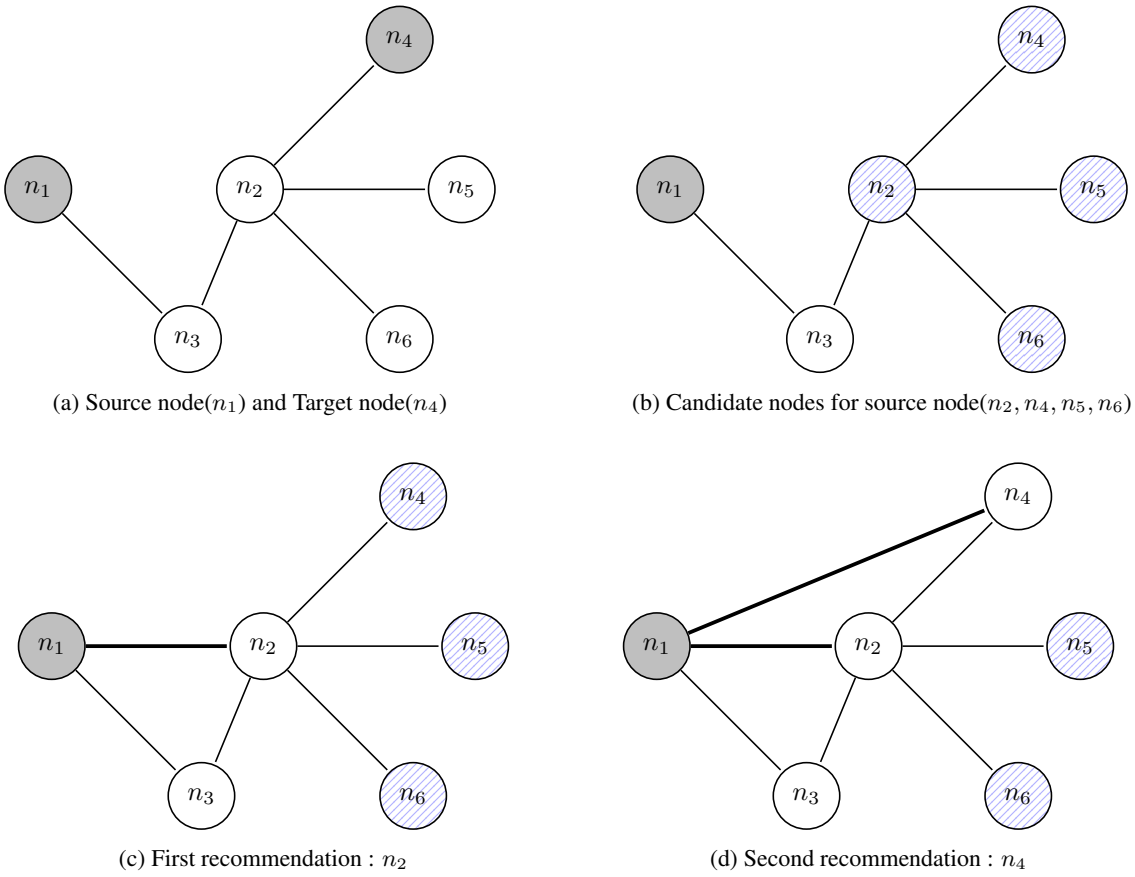
Symbols	Description
G	Undirected Network, $G = (V, E)$
G'	Network after adding edges, $G' = (V, E + E_S)$
S	Ordered Set of k suggested nodes, $S = \{n_1, n_2, ..., n_k\}$
E_S	Ordered Set of new connections by S, $E_S = \{e(n_s, n_1), e(n_s, n_2), ..., e(n_s, n_k)\}$
I_{st}(G)	Influence of n_s over n_t in graph G
ρ_{si}	Reluctance between n_s and n_i in graph G

Table 3.1: Symbols for the problem

Here we want to maximize the influence by having k new connections. And the constraint is a reluctance which means we need to guarantee that the source node and a suggested node need to have at least single mutual friend. We consider the case that the node is recommended sequentially.

We explain our problem with toy graph in Figure 3.2. We have undirected graph with $|V| = 6, |E| = 5$. $n_s = n_1$, and $n_t = n_4$ that n_1 wants to maximize its influence. Initially, n_4 can get n_1's article by the unique path, shared by n_3 and n_2 consecutively. For source node n_1, there exists a set of four nodes $S = \{n_2, n_4, n_5, n_6\}$ which can be a candidate of future connection. In order to give the best recommendation, we can calculate our objective function for those candidate nodes. Here we consider the case that $k = 2$, with sharing probability $p_s = 0.5$. The calculation result is given in Table 3.2. In this example, first recommended node becomes n_2 to source node n_1. And the second recommendation is direct connection with target n_4. After connecting those two

nodes, there aren't any beneficial node left, so $S = \{n_2, n_4\}$. However, calculating value for all possible edges is computationally expensive. Hence we propose heuristic algorithm to get k suggested friends.



(a) Source node(n_1) and Target node(n_4)

(b) Candidate nodes for source node(n_2, n_4, n_5, n_6)

(c) First recommendation : n_2

(d) Second recommendation : n_4

Figure 3.2: K-node suggestion problem on toy graph

New edge	ΔI_{st}	r	*Selected*
$e(n_1, n_2)$	0.0658	0.0361	Yes
$e(n_1, n_4)$	0.3732	1	No
$e(n_1, n_5)$	0.0071	1	No
$e(n_1, n_6)$	0.0071	1	No
$e(n_1, n_4)$	0.3078	0.2391	Yes
$e(n_1, n_5)$	0.0062	0.2391	No
$e(n_1, n_6)$	0.0062	0.2391	No
$e(n_1, n_4)$	-0.0122	0.2391	No
$e(n_1, n_5)$	-0.0122	0.2391	No

Table 3.2: Processing result of toy example

Chapter 4. Proposed Algorithm

In this chapter, we discuss the algorithms designed for searching nodes which maximize source node's influence over target node. Algorithm presented in this paper follows a greedy approach which selects the nodes sequentially. In Section 4.1 we describe three greedy algorithms. Main difference between three algorithms is the candidate set size when searching the best recommendation at each step. Algorithm 3 limits the candidate set as two-hop neighbor from the source node. In algorithm 4, we gradually cut off the candidate nodes which decreases influence by having new connection.

In Section 4.2, we propose *Incremental Katz Approximation*(IKA). IKA applies Monte-Carlo simulation to approximate influence value, and incrementally update by extending the random diffusion. In the stage of finding each recommendation, IKA save the status of random diffusion and only calculate the part that is affected by new connection. We then discuss about the complexity of each algorithm in Section 4.3.

4.1. Non-Incremental Exact Algorithm

4.1.1 Naïve Greedy Algorithm

Our greedy approach suggests the node sequentially which maximizes the influence each step. The algorithm halts if there does not exist any node which increases the influence. In order to find k suggestion sequentially, we need to repeat computing influence value for all unconnected candidate nodes k times. Algorithm 1 is the pseudo-code of greedy algorithm. Here, V_C represents candidate set, and $N_G[n_s]$ represents close neighborhood of n_s.

Algorithm 1 GREEDY FINDER

Input: Graph $G = (V, E)$, source node n_s, target node n_t
Output: Set of nodes $S = \{n_{i_1}, n_{i_2}, ..., n_{i_k}\}$

Calculate I_{st}
$S = \{\}$
$V_C = V - N_G[n_s]$ // Candidate set
while $\Delta I_{st} > 0$ **do**
  $n_m = \underset{n_c \in V_C, \rho_{sc} < 1}{\operatorname{argmax}} \Delta I_{st}$ // Finding the node which maximizes influence
  if $\Delta I_{st} > 0$ **then**
    $G = G + e(n_s, n_c)$
    $S = S + n_c$
    Update V_C // Remove n_c from the candidate set
  end if
end while
return S

Calculation procedure of influence is explained in Algorithm 2. For the purpose of calculating Katz centrality, we used NetworkX[14] package in Python. It is numerically calculated by power method iteration $x_i = \alpha \sum_j A_{ij} x_j + \beta$ to find the eigenvector corresponding to the largest eigenvalue of the adjacency matrix of G. And the attenuation factor α is strictly less than $1/\lambda_{max}$ to guarantee the convergence of Katz centrality value. And β is a weight attributed to the immediate neighborhood so that we can adjust how much each node will affect Katz centrality value. By setting $\beta = \mathbf{1}$, we are able to get the total number of post that target node receive from whole network. And by setting only s^{th} element 1 and otherwise 0, we can get the personalized Katz centrality of which denotes the number of post that target received through all paths, consider that only source node put a post online.

Algorithm 2 CALCULATE INFLUENCE

Input: Graph $G = (V, E)$, source node n_s, target node n_t

Output: I_{st}

1: Find λ_{max} of G

2: Calculate Katz Centrality $\vec{C}_{Katz}$ of G ($\alpha = 0.9/\lambda_{max}$, $\beta = \mathbf{1}$)

3: Calculate Personalized Katz Centrality $\vec{C}_{PKatz}$ of G ($\alpha = 0.9/\lambda_{max}$, $\beta_s = 1$)

4: $I_{st} = C_{PKatz}(t)/C_{Katz}(t)$ // (3.8)

5: **return** I_{st}

4.1.2 Two-Hop Neighbor Greedy Algorithm

As experiment network sizes becomes larger, the candidate size also grows proportional to the size of the network. Therefore, we need to shrink the size of candidate set in order to lessen the total running time of the algorithm. One possible solution is to restrict the candidate set as two-hop neighbor of source node. From the point that the node set which has a reluctance value smaller than 1 is exactly same as the two-hop neighbor of the source node, because Adamic-Adar value is nonzero when two vertices has at least single mutual neighbor. This also matches with the theory that individual makes a connection with their friend of friend known as triadic closure. This reduces the size of the searching set from $O(n)$ to $O(d^2)$, which gives huge benefit dealing with a sparse graph. Algorithm 3 describes the pseudo-code of this algorithm. Here, notation ${N_G}^2[n_s]$ represents the set of two-hop neighbor of n_s.

4.1.3 Candidate Reduction in Greedy Algorithm

We applied further candidate reduction after each recommendation, by removing candidate nodes that do not increase ΔI_{st}. Since we only require single best node on each recommendation, there are only little possibility that those garbage candidates are selected at the next recommendation among all candidates. Once the node is removed from the candidate set, it does not appear until it is newly selected as the two-hop neighbor.

4.2. Incremental Approximation Algorithm

In this section, we will explain our proposed algorithm Incremental Katz Approximation(IKA). First, we applied Monte-Carlo simulation for approximating influence value in large network. Then we show our way of

Algorithm 3 TWO-HOP NEIGHBOR GREEDY FINDER

Input: Graph $G = (V, E)$, source node n_s, target node n_t

Output: Set of nodes $S = \{n_{i_1}, n_{i_2}, ..., n_{i_k}\}$

Calculate I_{st}
$S = \{\}$
$V_C = {N_G}^2[n_s] - N_G[n_s]$ // Candidate set is two-hop neighbor
while $\Delta I_{st} > 0$ **do**
  $n_m = \underset{n_c}{\operatorname{argmax}} \Delta I_{st}$
  if $\Delta I_{st} > 0$ **then**
    $G = G + e(n_s, n_c)$
    $S = S + n_c$
    Update V_C // Update V_C by considering two-hop neighbor of n_c
  end if
end while
return S

Algorithm 4 CANDIDATE REDUCTION ON TWO-HOP NEIGHBOR GREEDY FINDER

Input: Graph $G = (V, E)$, source node n_s, target node n_t

Output: Set of nodes $S = \{n_{i_1}, n_{i_2}, ..., n_{i_k}\}$

Calculate I_{st}
$S = \{\}$
$V_R = \{\}$
$V_C = {N_G}^2[n_s] - N_G[n_s]$
while $\Delta I_{st} > 0$ **do**
  $V_R = \{n_c | \Delta I_{st} < 0\}$
  $n_m = \underset{n_c}{\operatorname{argmax}} \Delta I_{st}$
  if $\Delta I_{st} > 0$ **then**
    $G = G + e(n_s, n_c)$
    $S = S + n_c$
    $V_C = V_C - V_R$ // Remove nonbeneficial nodes from the candidate set
    Update V_C // Formerly removed nodes in $N_G(n_c)$ can be added to the candidate set
  end if
end while
return S

updating this value when additional edge is added to the graph. Using this technique, we can incrementally update the influence by having new recommendation.

4.2.1 Influence Approximation

In order to approximate information transfer ratio, we do two separate Monte-Carlo simulation. We found the possibility of Monte-Carlo simulation on Katz centrality from the paper which analyzes the efficiency of approximating personalized PageRank on evolving graph[4]. First we initialize R_1 articles starting from each node of the network with size n. Unlike PageRank, articles can spread or disappear to multiple nodes in our settings. At the first iteration each article diffuses over its direct neighbors, since an article is directly transmitted to neighbor's web feed in online social network. From the second iteration, each article spreads over its neighboring node if a sharing condition is met. An article from any node can totally disappear if everyone who got the post does not share at all. Average number of article that target received can be approximated by counting the articles that pass through target node divided by initial number of articles in the whole network. This simulation result approximates Katz centrality we discussed before.

Given target node n_t, X_t is the total number of times that random walk passes through n_t. If R_1 random walk starts at each node, we approximate the Katz centrality or information transfer rate from all nodes to target with:

$$\sum\nolimits_s \tilde{r}_{st} = X_t / nR_1 \tag{4.1}$$

Then, we approximate the average number of source node's article that target node received. This is done by initializing articles solely from the source node. Unlike the previous case, a sufficiently large number of articles should be initialized from the source node to approximate it. This becomes the Monte-Carlo way of getting personalized Katz centrality. Assume for target node n_t, Y_t is the total number of times that random walk visits target node. If R_2 random walks starts from source node, we can approximate r_{st}(Proposition 3.1) or personalized Katz centrality with:

$$\tilde{r}_{st} = Y_t / R_2 \tag{4.2}$$

Our measure for recommendation, influence of source node over target, I_{st} can be approximated by using (4.1) and (4.2) .

$$I_{st} = \frac{r_{st}}{\sum_s r_{st}} \approx \frac{nY_tR_1}{X_tR_2} \tag{4.3}$$

In order to regenerate the same result, pseudo code for approximating Katz centrality is described in Algorithm 5. Approximating personalized Katz centrality can be done by initializing articles on single source node.

4.2.2 Incremental Update

In order to recommend a node, we have to recalculate influence value when graph has an additional edge. And huge amount of calculation overlap occurs if we recalculate Katz centrality from the beginning. Here we refer the previous random diffusion result, and only calculate the extra part that is affected by a new edge. Having

Algorithm 5 MONTE-CARLO APPROXIMATION OF KATZ CENTRALITY

Input: $G = (V, E)$, Source node n_s, Target node n_t, Sharing probability p_s, Number of article initialized for each node : R, Diffusion length $nIter$

Output: r_{st}, Diffusion information $(Count(n_i), j)$: Number of article located in each node at each step j

Katz : Initialize R articles on each node
Personalized Katz : Initialize R articles on n_s
for node n_i in V **do**
  for each article in node n_i **do**
    Diffuse to its neighbor $N_G(n_i)$
  end for
end for
for Each iteration j from 2 to $nIter$ **do**
  for node n_i in V **do**
    for each article in node n_i **do**
      if $r \sim U[0, 1] < p_s$ **then**
        Diffuse to its neighbor $N_G(n_i)$ // $Count(N_G(n_i)) + 1$, for next iteration
      end if
    end for
  end for
end for
Sum the total number of diffusion arrive in target node r_t.
return $C_{Katz}(t) \approx r_t/R$, $(Count(n_i), j)$ // We also track how many articles are in n_i at iteration j

another edge means that information from source node can flow to new neighbors if sharing condition is met. So at each simulation step, we check an article located at the end of new edge. If sharing condition is met, another diffusion is starting from those two endpoints. Detailed algorithm is described in Algorithm 6. First part of the algorithm is initializing the new diffusion if there exists articles available on two nodes affected by new edge. Second part is generating additional transmission for articles which arrives at two nodes at each iteration. And the last part is continuing the random diffusion initiated by the effect of the new edge.

Algorithm 6 UPDATE KATZ CENTRALITY

Input: Graph $G = (V, E)$, Source node n_s, Target node n_t, Intermediate node n_i, Sharing probability p_s, Diffusion length $nIter$, Diffusion information from Algorithm 5 : d_1 = $(Count(n_i), j)$

Output: $\Delta C_{Katz}(t)$, Updated part of diffusion : $d_2 = (Count(n_i), j)$

Load diffusion information d_1
Set new diffusion information d_2 = $(Count(n_i), j)$ for updating part
for Each seeds in node n_i and n_s in $d_1(j = 1)$ **do**
    Random walk spread to opposite node
    Save the changes in $d_2(j = 2)$
end for // Check the initial case which is affected by the new connection
for Each iteration j from 2 to $nIter$ **do**
    for Each random walk in node n_i and n_s in $d_1(j)$ **do**
        if $r \sim U[0, 1] < p_s$ **then**
            Random walk spread to opposite node
            Update d_2 for iteration $j + 1$
        end if
    end for // Check affected nodes and give chances to propagate
    for Each random walk in node i according to d_2 **do**
        Set S = $N_G(i)$
        for node n in S **do**
            if $r \sim U[0, 1] < p_s$ **then**
                Random walk spread to opposite node
                Update d_2 for iteration $j + 1$
            end if
        end for
    end for // Process random diffusions by previous steps
end for
Count Δr_t from dictionaries
return $\Delta C_{Katz}(t) = \Delta r_t / R$, d_2 // Track the updated diffusion processes

In order to find the node which maximizes the influence, now we only need several small sizes of calculation for every new connection. By the same way, we can update the value when new edges are added to the original graph rather than re-calculating. Our algorithm IKA combined incremental Monte-Carlo approximation on greedy algorithm with candidate reduction technique. Since we can control the number of articles for simulation, this algorithm becomes scalable to large size network. Pseudo code of IKA is described in Algorithm 7.

Algorithm 7 INCREMENTAL KATZ APPROXIMATION(IKA)

Input: Graph $G = (V, E)$, n_s, n_t, p_s, R_1, R_2, $nIter$

Output: Set of nodes $S = \{n_{i_1}, n_{i_2}, ..., n_{i_k}\}$

Approximate Katz centrality $C_{Katz}(t)$: Initialize R_1 articles from each node // Algorithm 5
Approximate Personalized Katz centrality $C_{PKatz}(t)$: Initialize R_2 articles from source node // Algorithm 5
Calculate I_{st} using $C_{Katz}(t)$ and $C_{PKatz}(t)$ // (4.3)
$S = \{\}$
$V_R = \{\}$
$V_C = {N_G}^2[n_s] - N_G[n_s]$ // Two-hop neighbor
while $\Delta I_{st} > 0$ **do**
  $V_R = \{n_c | \Delta I_{st} < 0\}$
  Find $n_m = \underset{n_c \in V_C}{\operatorname{argmax}} \Delta I_{st}$ using Algorithm 6 // Effectively recommend node by updating I_{st}
  if $\Delta I_{st} > 0$ **then**
    $G = G + e(n_s, n_c)$
    $S = S + n_c$
    $V_C = V_C - V_R$ // Remove nonbeneficial nodes
    Update Random Diffusion
    Update V_C
  end if
end while
return S

4.3. Discussion on Algorithm Complexity

Next, we dicuss the time complexities of the proposed algorithms in Section 4.1 and Section 4.2. GREEDY FINDER algorithm runs in while loop until there exists no candidate that increases influence or candidate set size becomes k. The complexity inside the while loop depends on the size of the candidate set. In algorithm GREEDY FINDER, first candidate set is equal to every node which is not directly connected to source node. And it gets smaller by one as source node makes new connection with suggested node. In order to recommend k nodes, GREEDY FINDER requires kn candidates. And for each candidate node, influence score from n_s to n_t should be calculated. In order to calculate influence score, proportion of two Katz centrality - numerator is the source node's personalized version of Katz centrality over target, and the denominator is the Katz centrality of target node utilizing global network. For the purpose of calculating Katz centrality matrix inversion process is required. However direct matrix inversion does not work since the adjacency matrix of the network is always singular because at least two vertices have same neighboring set. We used NetworkX module which computes the Katz centrality by power iteration until it converge, and complexity of it is also $O(n^3)$. Hence, the overall complexity of recommending k node by GREEDY FINDER is $O(kn^4)$.

And in TWO-HOP NEIGHBOR GREEDY FINDER, we reduce the candidate set from all unconnected node to two-hop neighbor. Consider that the average degree is d, average number of two-hop neighbor of single node is bounded by $d(d-1)$ in random network if considering the worst case that there is no overlap between friend of friend. However in scale-free network, there is high chance to make connection with one of the hub nodes, and

the size of two-hop neighbor is affected by the degree of the hub node. If we consider the worst case which is star network, then the size of two-hop neighbor becomes n, so we can deduce that the number of candidate until algorithm finds k nodes can be estimated by $\sum_{i=1}^{k} d(d+i)$ in the best case. Overall algorithm complexity becomes $O(d(d+k^2)n^3)$ from $O(kn^4)$.

Algorithm 4 further reduce the candidate set in every iteration. If an edge from n_s to n_k does not increase any influence in iteration i, we remove it from the candidate set. Even though source node get many new connections, there is low possibility that node n_k will get suggested after another influence calculation. We cannot exactly calculate how many node could be dropped from the list at this moment, so we leave it as one of our future work.

Next, we discuss the algorithm complexity of IKA. First we approximate influence score by two Monte-Carlo simulations. First simulation is for the Katz centrality that all nodes affect, and another is the Katz centrality that only the walks from source node affect. Only difference between those two simulations is the number of seeds generated before starting the diffusion process. Since the number of node is big enough in real graph, we generate single seeds on each node for calculating Katz centrality. And for personalized Katz centrality we generate enough large number of seeds (smaller than n on large graph) on source node.

Initializing a seed on each node takes $O(n)$ time. And then duplicating the post to its neighbor takes $O(dn)$. After the first round, post is successfully move to its neighbor if random variable meets sharing probability. Ratio of average number of post for consecutive round is pd, since only chosen post can diffuse to the node's next neighbor. By continuing the simulation until all posts disappear, the time complexity of Algorithm 5 becomes $\Theta\left(n + \sum_{k=0}^{\infty} nd(dp)^k\right) = \Theta\left(n + \frac{nd}{1-dp}\right) = O(nd)$. Note that personalized Katz centrality does not affect to the total complexity of approximating influence if number of seed on source node $R \approx n$.

In terms of the complexity, the advantage of the Algorithm 6 is significant. In previous algorithms, we have to re-calculate Katz centrality whenever n_s connects to new candidate node. But here we only need to check the possibility of diffusion when random seed is on the each side of the new edge. Checking the whole process needs $O(d)$ and if new diffusion is generated we have to calculate until we finish the process. It takes $\Theta\left(\frac{d}{1-dp}\right) = O(d)$. Lastly expected number of diffusion newly generated is $\Theta\left(\frac{\sum_{k=0}^{\infty} nd(dp)^k}{\frac{2}{n}}\right) = O(d)$. So when original influence score is given, and there is new candidate node, we can calculate the change of influence score on $O(d^2)$. With two-hop neighbors candidate set with approximation, finding a single node which maximizes the influence takes $O(d^4)$. And in the same way, complexity of finding a k-node set becomes $O(kd^4)$ which signifinantly improved from $O(k^2d^2n^3)$.

Chapter 5. Experiment

In this section, we evaluate our algorithm and compare the recommendation result with existing algorithms. And we aim to answer the following questions:

- How well IKA can solve our problem?
- What patterns and behavior does the recommended node of IKA has?
- How different our result compare to existing network growth model or friend recommendation model?

5.1. Implementation Details

The code for IKA has been written in Python. For synthetic network generation and implementation we used Python NetworxX module [14]. All experiment were performed on a PC with Intel Core i5-4670 3.4GHz processor, 8 GB of main memory and a 128GB SSD drive. Graphs we use in experiments along with their descriptions are summarized in Table 5.1.

Topology	Node	Edges	Description
Scale-Free [16]	100 1,000 10,000 100,000 1,000,000	99 999 9,999 99,999 999,999	m=1 (m : number of edges to attach from a new node to existing nodes)
	200	398	m=2
	100 1,000	297 2,997	m=3
	100 1,000	990 9,990	m=10
Erdős-Rényi Random [10]	100 1,000	100 1,000	Average degree = 1

Table 5.1: Synthetic graph description

5.2. Results

5.2.1 Time Comparison

In order to compare the performance between exact influence calculation and Monte-Carlo approximation, we measured the time for recommending ten consecutive nodes. We setup the first five synthetic graph on Table 5.1, and set the source node as the last node, and target node as the first node of the graph. In our graph generation module, target node becomes one of the leaf nodes on the graph, and the source node becomes one of the hub node having a largest degree. For approximating influence on IKA, we set the number of random seeds($R_1 = 1$, $R_2 = 10,000$) regardless of the network size. By the result, we found our algorithm IKA is much faster and scalable compared to the greedy algorithms with exact influence calculation. Figure 5.1 shows the time comparison between three methods algorithm 1, 3 and 7. All other algorithms fails to compute influence score from $n = 10,000$ except IKA since matrix calculation on Numpy module fails because of memory error when the matrix size gets larger. Here IKA also shows the time increasing according to the network size. But it's inevitable since total time is proportional to candidate set size, which becomes larger as number of nodes increases.

Then, we experimented our algorithm on graphs having different densities. Figure 5.2 shows the time comparison between those graphs. Interestingly, we found that the running time on random graph is faster than the power-law graph.

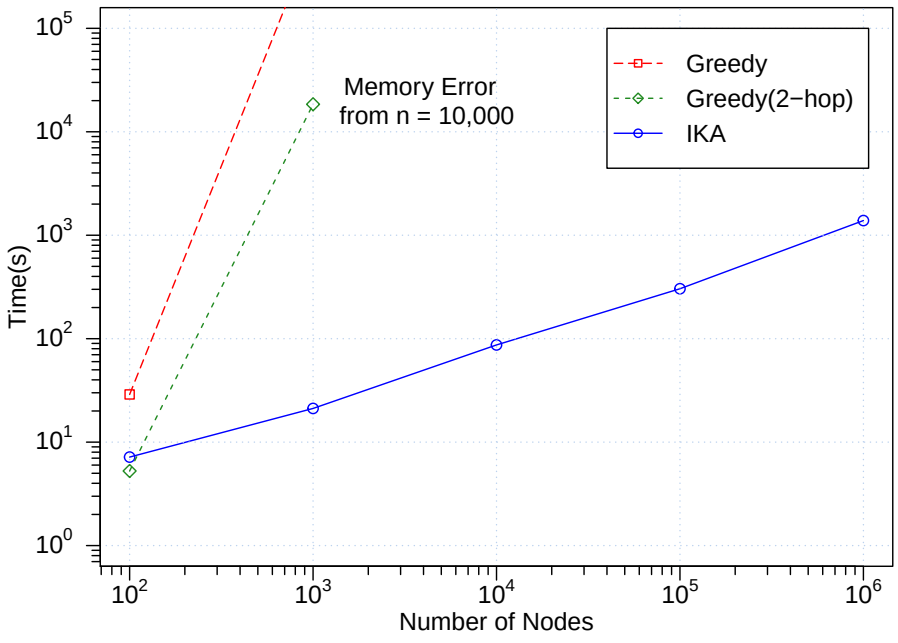


Figure 5.1: Time comparison according to the number of nodes

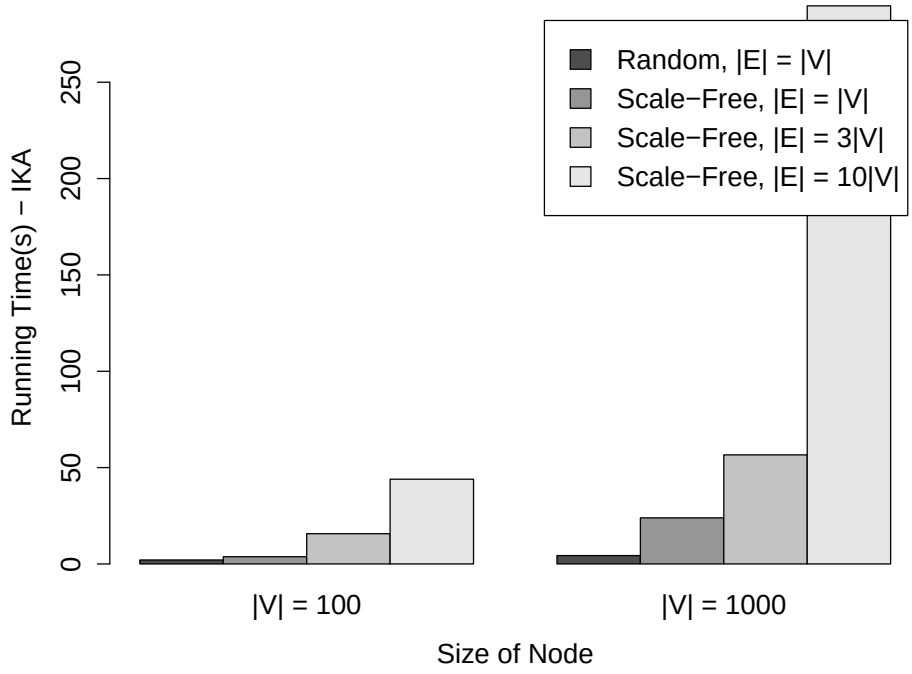


Figure 5.2: Time comparison according to network density

5.2.2 Error Analysis

Second we found that our random diffusion simulation results in similar influence score comparing with the result calculated by power iteration of Katz centrality. For this experiment, we used scale-free graph with $|V| = 100, |E| = 100$. Here we measure the influence after the first recommendation, and compare the error between the exact value and approximated value. We experimented 30 times to get the mean of approximated influence value. And we control the number of articles initializing from each node for approximation. R_1 refers the number of node initialized on each node for approximating Katz centrality and R_2 refers the number of node initialized on source node for approximating personalized Katz centrality. We use $R_1 = \{1, 10, 100, 1000\}, R_2 = \{100, 1000, 10000, 100000\}$. And the relative error is measured as

$$\frac{|I_{st} - E[\tilde{I}_{st}]|}{I_{st}}$$

Figure 5.3 shows the relative error between IKA and GREEDY FINDER as increasing total number of articles nR_1(y-axis) and number of personalized random seed R_2(x-axis) increases. Green color means accurate approximation on this heat map. We can find that approximation becomes accurate with large R_1 and R_2. Interestingly, the size of R_2 affects more than the size of R_1, because R_2 is responsible to the value of r_{st}.

And Figure 5.4 shows the variance of approximated influence $\tilde{I}_{st}$ calculated 30 times. As we know that, more simulations by using more random seeds guarantee the smaller variance between the results.

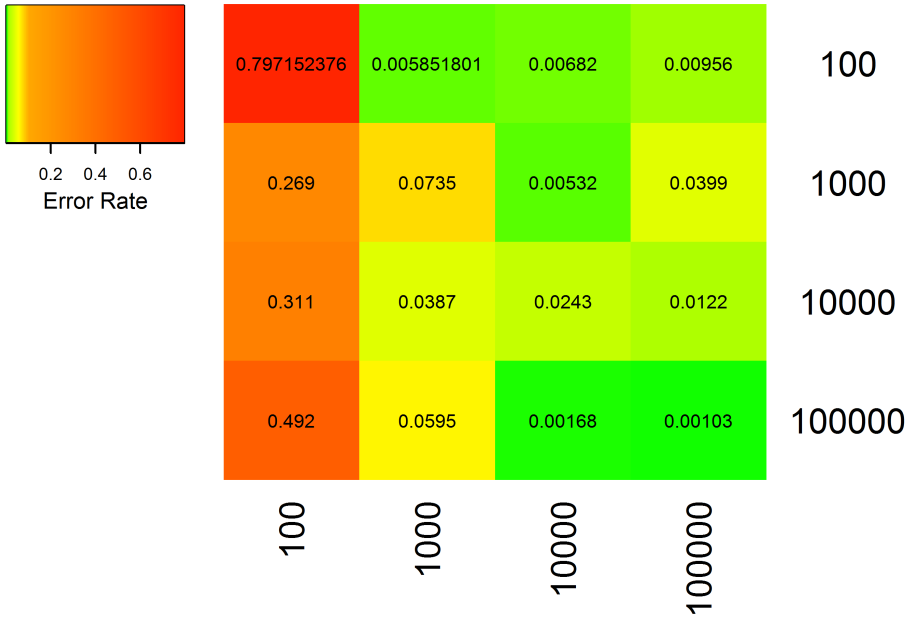


Figure 5.3: Relative error between exact and approximated influence

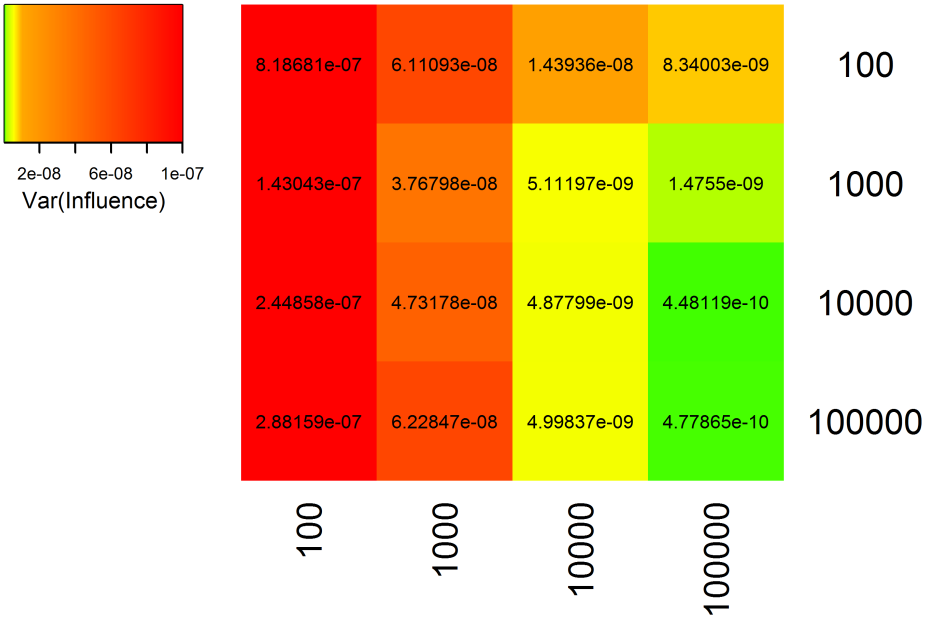


Figure 5.4: Variance of approximated influence

5.2.3 Performance Comparison and Interpretation

We compared the performance of our algorithm with existing topology-based recommendation algoriths with various node-to-node similarity measures. Performance measure of this experiment is cumulative influence gain by having new connections. For the comparison, We implemented variation of node similarity measure such as

common neighbor, Jaccard [39], SimRank [17].

$$
\begin{aligned}
Common\ neighbor(x,y) &= |\Gamma(x) \cap \Gamma(y)| \\
Jaccard(x,y) &= \frac{|\Gamma(x) \cap \Gamma(y)|}{|\Gamma(x) \cup \Gamma(y)|} \\
SimRank(x,y) &= \gamma \cdot \frac{\sum_{a \in \Gamma(x)} \sum_{b \in \Gamma(y)} 1_{a=b}}{\Gamma(x) \cdot \Gamma(y)}
\end{aligned}
\tag{5.1}
$$

And we consider the two cases, one is consecutively finding the most similar node with n_s, and another is finding the most similar node with n_t. For example of Jaccard(Target), we recommend the node that has the highest Jaccard similarity with n_t from the candidate set. Figure 5.5 to 5.8 shows the performance of IKA and existing topology-based algorithm with node-to-node similarity measure. We used scale-free graph with $|V| = 200, |E| = 396$ for four different experiment settings.

- Figure 5.5 is the case where n_s(leaf node) and n_t(hub node) are initially unconnected
- Figure 5.6 is the case where n_s(leaf node) and n_t(leaf node) are initially unconnected
- Figure 5.7 is the case where n_s(leaf node) and n_t(hub node) are initially connected
- Figure 5.8 is the case where n_s(leaf node) and n_t(leaf node) are initially connected

There are several things to check in these figures. First, how cumulative influence changes depending on first few recommendation. In detail, how many step is needed to get the large jump on influence gain. In the case of 5.5 and 5.6, IKA connects to the most influential node in the first few steps. As we know that big jump of influence value occurs by connecting to the n_t directly, we can understand that recommendation before the big jump is the process of indirectly exposing myself to target. And the recommendation result after the big jump is the way of connecting to the target node's neighbor. In the case of 5.7, we cannot find the big jump on both ideal recommendation and our experimental result. Second, absolute value of cumulative influence difference depends on each situation. Absolute value of cumulative influence is larger where n_s and n_t are both leaf nodes. We can think that n_t receive an article from the small number of peoples, therefore influence of n_s can change rapidly by the recommendation result.

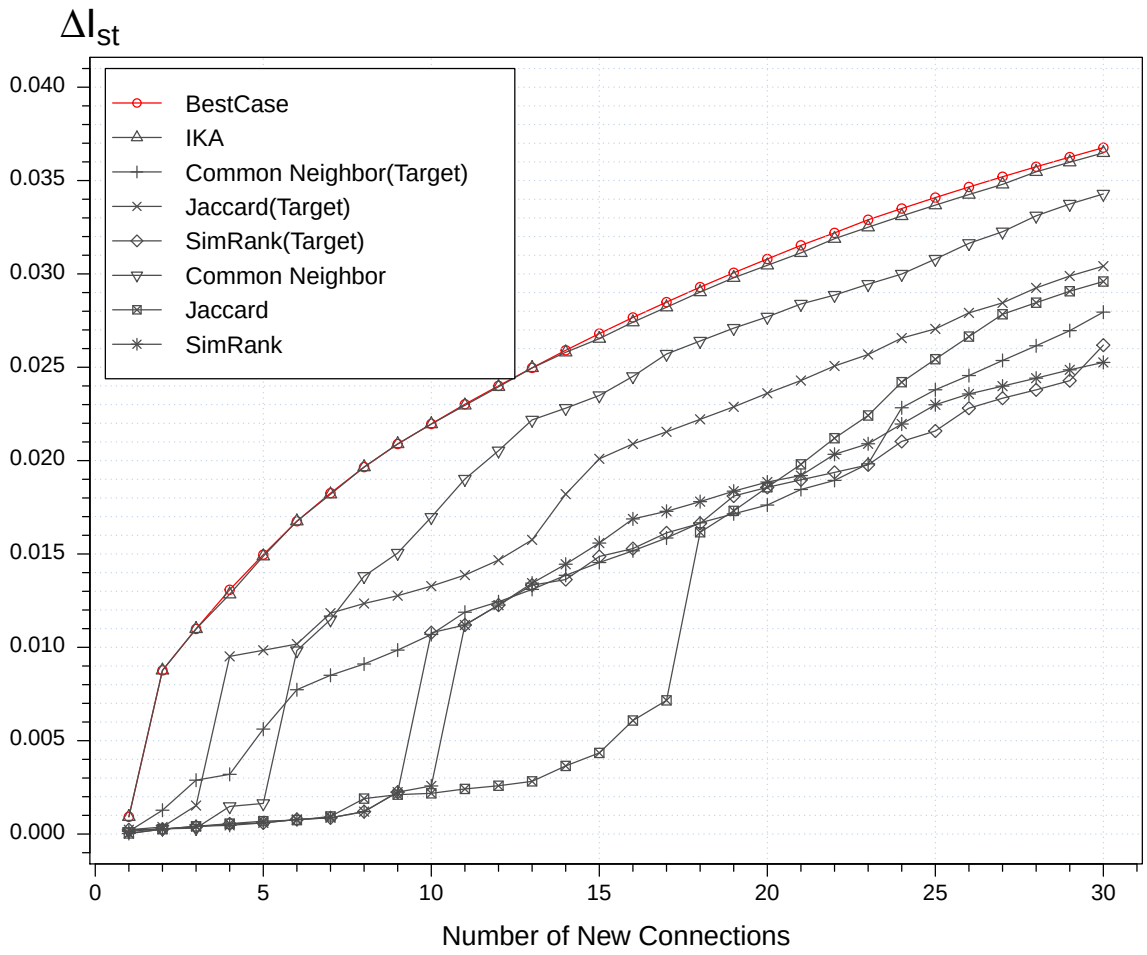


Figure 5.5: Friend recommendation result (n_s, n_t = not connected, n_s = leaf, n_t = center)

In 5.5 we are able to find interesting result. Only IKA suggests different nodes at the first recommendation, which brings to big influence gain at the second step. This shows that node-to-node similarity measures do not work effectively on the influence maximization problem.

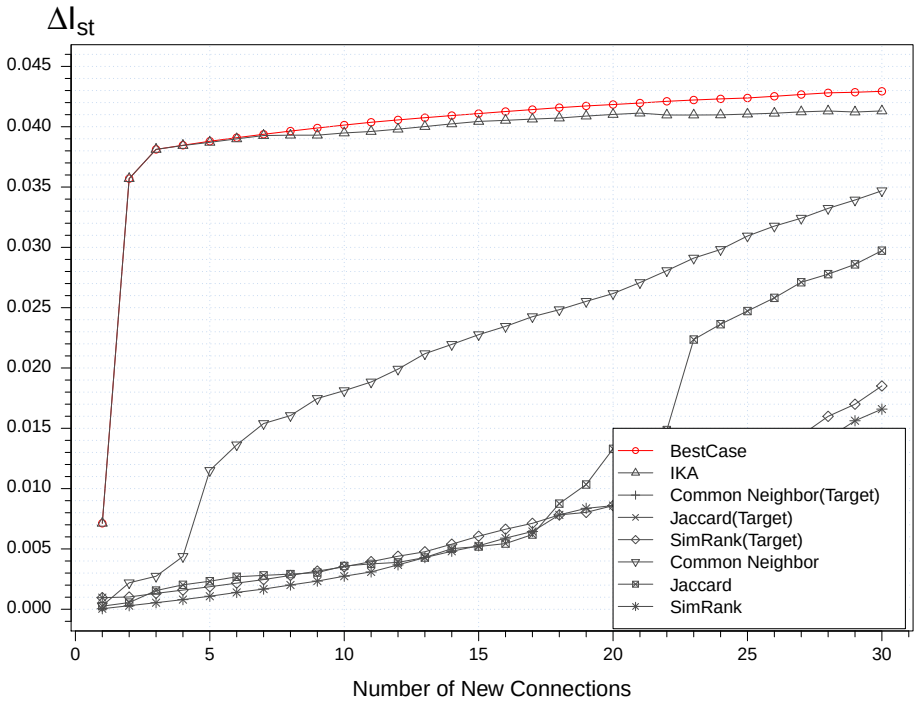


Figure 5.6: Friend recommendation result (n_s, n_t = not connected, n_s = leaf, n_t = leaf)

Figure 5.6 shows the case where two leaf nodes are not connected. In the real world, this is the main case where an individual wants to express him/herself to another leaf node. Here, IKA find relevant nodes faster than any other node-to-node recommendation algorithms using diverse similarity measures.

In Figure 5.7, we can describe the case where n_s is a fan of a well-known individual n_t, and they are already connected. As long as n_t maintains hubness, any connection from n_s to other node guarantees influence increasing. Of course, performance of IKA is superior against other recommendation methods.

Last experiment is a special case where recommendation is refrained, since two leaf nodes n_s and n_t are already connected. Generating more edges around two users adversely affect on their information transfer. Unlike other experiments, IKA stops after recommending nine nodes. After the second recommendation, each recommendation result actually decreases the influence, however it seems that some new diffusions gave positive effect on our approximation. Most algorithms using similarity measure fail to increase I_{st} because node-to-node similarity is not directly directed to influence maximization, and two initial leaf nodes are connected from the beginning.

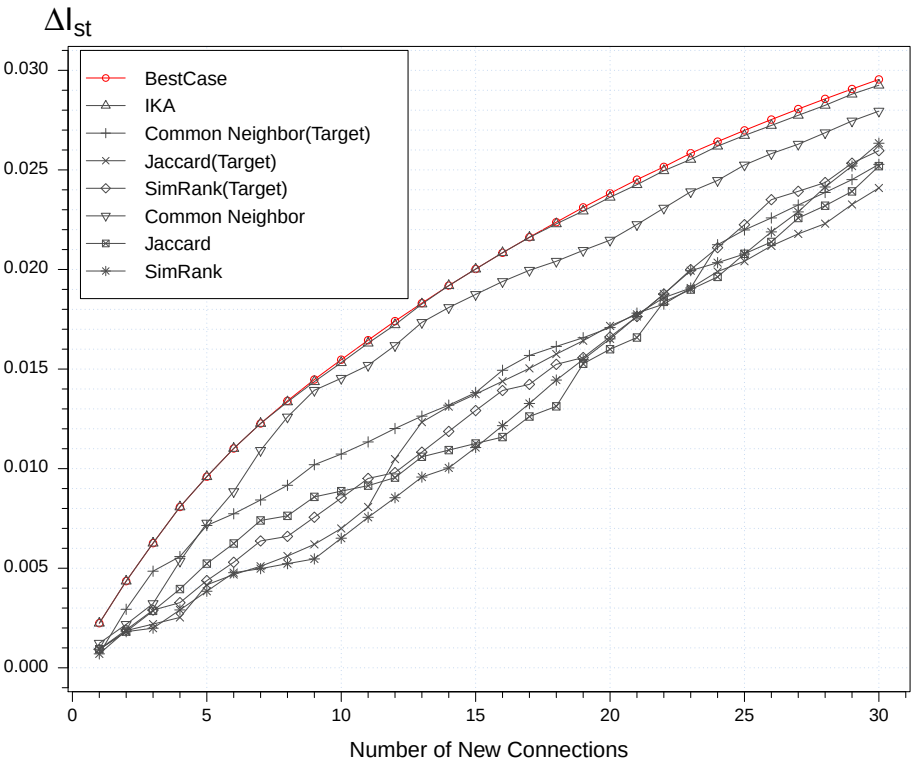


Figure 5.7: Friend recommendation result (n_s, n_t = connected, n_s = leaf, n_t = center)

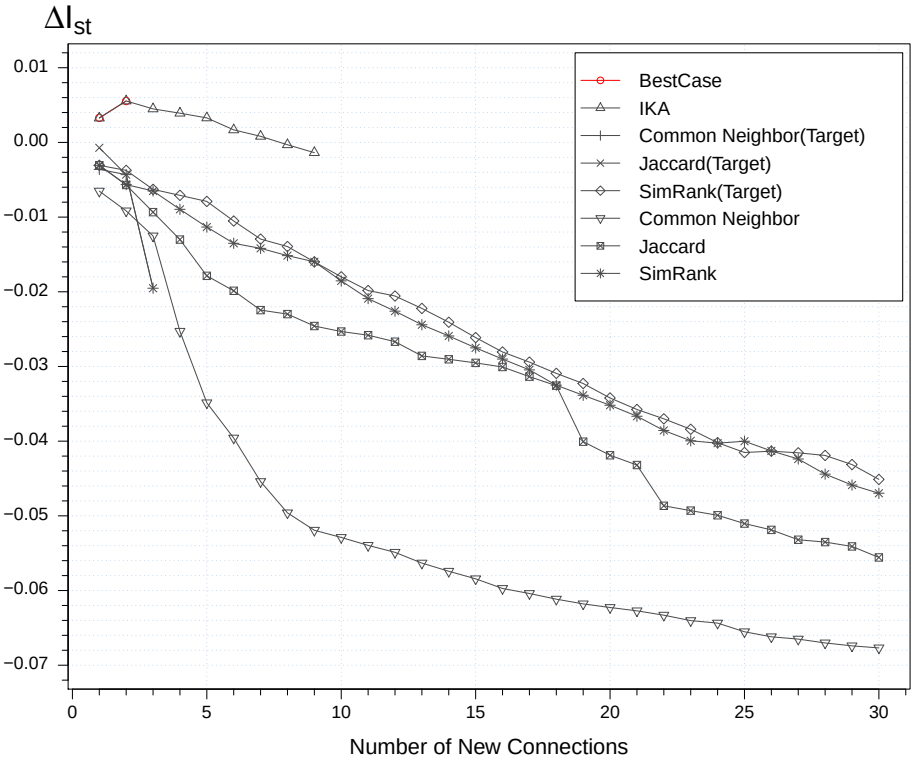


Figure 5.8: Friend recommendation result (n_s, n_t = connected, n_s = leaf, n_t = leaf)

Chapter 6. Conclusion and Future Directions

6.1. Summary of Findings

In this paper, we proposed the social network environment when the user has a specific target node to maximize its influence. In our problem, maximizing influence is indirectly done by having relevant friend suggestion to source node, since the new friend will share the post and increase the probability that the post will arrive to the target node. We defined the influence as how much effect that source node's post has on target node's feed. And we formulated our problem as suggesting a node one-by-one which maximizes the influence score.

The purpose of our friend recommendation is maximizing influence over specific target, rather than suggesting a similar node with the user. So traditional friend recommendation algorithms may not guarantee the good result. So we modeled our environment from the scratch, and designed our algorithm.

One finding is that the influence score matches with the definition of Katz centrality. The amount of effect that source node's post on target node's feed can be calculated, when we estimate how many posts that the target node can receive from source node, and among the network. Since we assumed that each person share any post with deterministic probability, and the post can be transferred to the neighbors of it, the total number of post that target node can receive is equal to Katz centrality of target node. And the number of post from source node can be also measured by considering the effect of all paths between two vertices.

However, we met with huge computational cost. In order to recommend k-node that maximizes influence, we need to calculate influence for exponential amount of time. Inevitably we allowed greedy algorithm which recommend node set one-by-one which maximizes influence. Although our approach changed into heuristic way, computation issue haven't solved at once. In order to recommend a single node, we need to calculate influence for all nodes except the neighbors in the network. The total algorithm complexity becomes $O(kn^4)$ in this case.

To tackle this problem, we reduced the size of candidate set. First, we only consider the second-hop neighbor of the source node which has at least single mutual friend. This condition also guarantees that all candidate node has positive node-to-node similarity values. However, as source node is getting more connection, size of the candidate set became larger. Considering the power-law graph, there is high probability that the node can make a connection with one of the hub nodes which can cause scalability issue again. So, second we decide to cut down nonbeneficial nodes from the candidate set. Nonbeneficial nodes are the nodes which decreases influences over target when source node makes new connection with one of them. This reduction is possible since there is little possibility that nonbeneficial node will be suggested among all candidate nodes.

With candidate reduction, we engaged in approximation and incremental calculation of influence value. Katz centrality computation for influence needs matrix inversion, which cause memory shortage and huge computational burden. So we applied Monte-Carlo simulation for approximation. Since we only store the number of posts exists in each node for every period, huge benefit occurs for both computation speed and memory storage. In addition to single approximation, we incrementally update the influence value on the case of having new

connection. For incremental update, we can independently consider the post which located at the end of new connection, adding the new random post diffusion starting from those nodes. In this way we can also suggest k nodes incrementally, by significantly reducing the computation overlap.

We validated the performance of our algorithm using both synthetic graphs and real graphs. First, we showed our Monte-Carlo approach is much faster than computing influence in scalable graph, and we analyzed the error rate according to the size of articles for simulation. Second, we presented our algorithm IKA works superior than current topology-based friend recommendation algorithms in influence maximization problem.

6.2. Future Work

- **More Reduction on Algorithm Efficiency**

  Although we scaled down the algorithm complexity, still we have some place to boost the algorithm. For example, each iteration we calculate the influence score for all candidate nodes. After having new connections, we have to calculate the influence score again for those nodes plus new neighbors. However our algorithm does not save previous values for next recommendation. If we find the way to save all diffusions efficiently, we can also improve this part by incrementally calculating the effect of a new edge.

- **Unsolved Scalability Issue**

  Although we approximate the calculation of Katz centrality and incrementally apply IKA into larger network, our algorithm still have serious time complexity when the network becomes denser. Main reason is that candidate set sizes grows faster when there exists more edges, and also the degree of the hub node is much larger.

- **Extend to Multiple Targets**

  In this thesis, we proposed network growing mechanism on the view of single source node and a target node. We can extend the setting into multiple targets, especificly a group. We can suggests a set of node to maximize the influence over target group.

- **Apply More Realistic Settings**

  We can generalize or make our problem more realistic considering the fact that every user in social network has different posting behavior and sharing probability. By applying it to real data, we can suggest a new recommendation platform.

- **Asymmetric behavior in a network**

  The main idea of this thesis comes from the contemplation of hidden asymmetric behavior in a network. Connecting to the set of nodes is the one possible solution to resolve this asymmetry and we suggest an algorithm to find those set. Another potential future work direction is to design the diverse way to analyze asymmetric relationship in a network.

References

[1] Adamic, L., Adar, E. (2003) Friends and Neighbors on the Web. *Social Networks* **25.** : 211–230.

[2] Aggarwal, C. (2011) An Introduction to Social Network Data Analytics. *Springer*.

[3] Armentano, M. G., Godoy, D., Amandi, A. (2012) Topology-Based Recommendation of Users in Micro-Blogging Communities. *Journal of Computer Science and Technology* **27. 3.** : 624–634.

[4] Bahmani, B., Chowdhury, A., Goel, A. (2010) Fast Incremental and Personalized PageRank. *Proceedings of the VLDB Endowment* **4. 3.** : 173–184.

[5] Blau, P. M., Schwartz, J. E. (1997) Crosscutting Social Circles: Testing a Macrostructural Theory of Intergroup Relations. : Academic Press.

[6] Borgatti, S. (2005) Centrality and Network Flow. *Social Networks* **27** : 55–71.

[7] Burton, S., Giraud-Carrier, C. G. (2013) Discovering Social Circles in Directed Graphs. *ACM Transactions on Knowledge Discovery from Data* : In Submission

[8] De Meo, P., Ferrara, E., Fiumara, G., and Ricciardello. A. (2013) A Novel Measure of Edge Centrality in Social Networks. *Knowledge Based Systems* **30** : 136–150.

[9] Easley, D., Kleinberg, J. (2010) Networks, Crowds, and Markets: Reasoning about a Highly Connected World. Chapter 3, Strong and Weak Ties *Cambridge University Press*.

[10] Erdős, P., Rényi, A. (1959) On Random Graphs. 1 *Publicationes Mathematicae* **6.** : 290–297.

[11] Gilbert, E., and Karahalios. K. (2009) Predicting Tie Strength with Social Media. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'2009)* : 211–220.

[12] Golder, S. A., Yardi, S. (2010) Structural Predictors of Tie Formation in Twitter: Transitivity and Mutuality. *2010 IEEE Second International Conference on Social Computing* : 88–95.

[13] Guille, A., Hacid, H., Favre, C., Zughed, D. A. (2013) Information Diffusion in Online Social Networks: A Survey. *ACM SIGMOD Record* **42. 2.** : 17–28.

[14] Hagberg, A. A., Schult, D. A., Swart, P. J. (2008) Exploring Network Structure, Dynamics, and Function using NetworkX. *Proceedings of the 7th Python in Science Conference (SciPy2008)* : 11–15.

[15] Hangal, S., Maclean, D., Lam, M. S., Heer, J. (2010) All Friends are Not Equal: Using Weights in Social Graphs to Improve Search. *The 4th Workshop on Social Network Mining and Analysis (SNAKDD'2010)*

[16] Holme, P., Kim, B. J. (2012) Growing Scale-Free Networks with Tunable Clustering. *Physics Review E***65.** : 026135

[17] Jeh, G., Widom, J. (2002) SimRank: A Measure of Structural-Context Similarity. *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'2002)* : 538–543.

[18] Jeh, G., Widom, J. (2003) Scaling Personalized Web Search. *Proceedings of the 12th International Conference on World Wide Web(WWW'2003)* : 271–279.

[19] Kas, M. (2013) Incremental Centrality Algorithms for Dynamic Network Analysis. *Ph.D. Thesis, Dept. of Electrical and Computer Engineering, Carnegie Mellon University*

[20] Katz, L. (1953) A New Status Index Derived from Sociometric Analysis. *Psychometrika* **18. 1.** : 39–43.

[21] Kleinberg, J., Suri, S., Tardos, E., Wexler, T. (2008) Strategic Network Formation with Structural Holes. *ACM SIGecom Exchanges* **7. 3.** : Article No.11.

[22] Kossinets, G., Watts, D. J. (2009) Empirical Analysis of an Evolving Social Network. *Science* **311.** : 88–90.

[23] Leicht, E. A., Holme, P., Newman, M. (2005) Vertex Similarity in Networks. *Physics Review E* **73.** : 026120.

[24] Leskovec, J., Huttenlocher, D., Kleinberg, J. (2010) Predicting Positive and Negative Links in Online Social Networks. *Proceedings of the 19th International Conference on World Wide Web (WWW'2010)* : 641–650.

[25] Liben-Nowell, D., Kleinberg, J. (2004) The Link Prediction Problem for Social Networks. *Journal of the American Society for Information Science and Technology* **58. 7.** : 1019–1031.

[26] Lindelauf, R., Borm, P., Hamers, H. (2008) The Influence of Secrecy on the Communication Structure of Covert Networks. *Social Networks* **31. 2.** : 126–137.

[27] Liu, J., Meng, Z., Hu, Y., He, Y., Shiu, S., Cho, V. (2014) A Hybrid Algorithm for Recommendation Twitter Peers. *Proceedings of 29th Annual ACM Symposium on Applied Computing(SAC'2014)* : 644–649.

[28] Lo, S., Lin, C. (2006) WMR-A Graph-based ALgorithm for Friend Recommendation. *Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence* : 121–128.

[29] Lops, P., Gemmis, M., Semeraro, G. (2011) Content-based Recommendation Systems: State of the Art and Trends. *Recommender Systems Handbook* Springer : 73–105.

[30] McAuley, J., Leskovec, J. (2014) Discovering Social Circles in Ego Networks. *ACM Transactions on Knowledge Discovery from Data* **8. 1** : Article 4.

[31] McPherson, M., Smith-Lovin, L., Cook, J. M. (2001) Birds of a feather: Homophily in Social Networks. *Annual Review of Sociology* **27.** : 415–444.

[32] Mitchell, M. (2006) Complex Systems: Network Thinking. *Artificial Intelligence* **170. 18.** : 1194–1212.

[33] Moricz, M., Dosbayev, Y., Berlyant, M. (2010) PYMK: Friend Recommendation at Myspace. *Proceedings of the 2010 ACM SIGMOD International Conference on Management of Data* : 999–1002.

[34] Newman, M. (2003) The Structure and Function of Complex Networks. *SIAM Review* **45.** : 167–256.

[35] Okamoto, K., Chen, W., Li, X. -Y. (2008) Ranking of Closeness Centrality for Large-Scale Social Networks. *Proceedings of the 2nd annual international workshop on Frontiers in Algorithmics* : 186–195.

[36] Page, L., Brin, S., Motwani, R., Winograd, T. (1999) The PageRank Citation Ranking: Bringing Order to the Web. *Technical Report, Stanford InfoLab*.

[37] Roth, M., Ben-David, A., Deutscher, D., Flysher, G., Horn, I., Leichtberg, A., Leiser, N., Matias, Y., Merom, R. (2010) Suggesting Friends Using the Implicit Social Graph. *Proceedings of the 16th ACM SIGKDD Conference on Knowledge Discovery and Data Mining* **25.** : 233–242.

[38] Rubin, F. (1978) Enumerating All Simple Paths in a Graph. *IEEE Transactions on Circuits and Systems* **25. 8.** : 641–642.

[39] Salton, G., McGill, M. J. (1983) Introduction to Modern Information Retrieval. *McGraw-Hill*.

[40] Sharma, A., Gemini, M., Cosley, D. (2011) Friends, Strangers, and the Value of Ego Networks for Recommendation. *Proceedings of the Seventh International AAAI Conference on Weblogs and Social Media (ICWSM'2013)* : 721–724.

[41] Silva, N. B., Tsang, I. -R., Cavalcanti, G., Tsang, I. -J. (2010) A Graph-Based Friend Recommendation System Using Genetic Algorithm. *2010 IEEE Congress on Evolutionary Computation* : 1–7.

[42] Simmel, G. (1950) The Sociology of Georg Simmel. Free Press, New York.

[43] Su, X., Khoshgoftaar, T. M. (2009) A Survey of Collaborative Filtering Techniques, *Advances in Artificial Intelligence* **2009.** : Article 4.

[44] Tian, X., Song, Y., Wang, X., Gong, X. (2012) Shortest Path Based Potential Common Friend Recommendation in Social Networks. *2012 Second International Conference on Cloud and Green Computing* : 541–548.

[45] Will. U. K., Gnaidek, J., Memon, N. (2010) Measuring Link Importance in Terrorist Networks. *2010 Internation Conference on Advances in Social Networks Analysis and Mining* : 225–232.

[46] Xiang, R., Neville, J., Rogati, M. (2010) Modeling Relationship Strength in Online Social Networks. *Proceedings of the 19th International Conference on World Wide Web (WWW'2010)* : 981–990.

[47] Zhao, J., Wu, J., Feng, X., Xiong, H., Xu, K.(2012) Information Propagation in Online Social Networks: A Tie-Strength Perspective. *Knowledge Information System* **32.** : 589–608.

[48] Zhao, P., Han, J., Sun, Y. (2009) P-Rank: a Comprehensive Structural Similarity Measure over Information Networks. *Proceedings of the 18th ACM Conference on Information and Knowledge Management(CIKM'2009)* : 553–562.

[49] Zhou, T., Lu, L., Zhang, Y. -C. (2009) Predicting Missing Links via Local Information. *Europian Physics Journal. B* **71.** : 623–630.

요 약 문

특정 사용자로의 영향력을 최대화하는 친구 추천 방법

본 논문에서는 소셜 네트워크 상에서 영향력을 확장하고 싶은 특정 사용자가 존재할 때, 그 영향력을 최대화할 수 있는 친구를 한 명씩 추천하는 알고리즘(IKA)을 제안한다. 소셜 네트워크의 뉴스피드 등에서 우리는 직접 연결된 사용자들의 소식 뿐만이 아니라 그들이 관심 있어 공유한 다양한 글들을 접하게 된다. 이런 네트워크 상에서 특정한 사용자에게 자신의 영향력을 극대화하고 싶을 때, 자신의 정보의 전파에 도움이 되는 새로운 친구 추천이 가능하다면 다양한 이익을 가져올 수 있을 것이며 이러한 생각으로부터 연구를 시작하게 되었다.

기존의 친구 추천 알고리즘은 네트워크 구조 및 프로필의 유사도가 높은 사용자를 추천해 주지만, 특정 사용자에게 나의 영향력을 증가시키는 목적과는 무관하다. 우리가 제시한 문제를 해결하기 위해 소셜 네트워크 상에서 문제 정의를 하였다. 먼저 소셜 네트워크 상에 업로드한 게시물은 직접 연결된 친구들의 게시판에 올라가게 되고, 친구가 해당 게시글을 공유하거나 답하는 등의 행동을 통해 그의 이웃으로 전파되는 모델을 설정하고, 사용자 A가 B에 미치는 영향력은 B가 받는 모든 게시글 중 A로부터 도착한 게시글의 비율로 정의하였다. 한 사용자가 받는 게시글의 개수는 우리가 설정한 모델 내에서 Katz centrality의 정의와 부합함을 확인하였다. 영향력을 최대화하는 연결 조건에 더불어 어떤 사용자에게 친구 요청을 하였을 때 수락하지 않을 확률을 두 노드 사이의 유사도를 이용하여 모델링하였고, 추천 과정에 함께 고려하였다.

친구를 추천을 위해서는 네트워크 상에서 직접 연결되지 않은 모든 사용자들에 대해, 각각 연결되었을 때 증가하는 영향력을 계산하는 과정을 거쳐야 하는데 이 경우 불필요한 연산이 매우 많아진다. 따라서 본 논문에서는 몬테 카를로 시뮬레이션을 통하여 Katz centrality를 근사하였고 새로운 연결이 추가 될 때, 특정 사용자에게 내가 미치는 영향력를 처음부터 계산하는 대신 추가된 연결로 인해 게시글이 네트웍상에 퍼지는 부분만 따로 추가하여 알고리즘의 복잡도를 크게 개선하였다.

다양한 크기와 구조를 가진 합성 네트워크와 실제 소셜 네트워크 데이터를 통해 기존 알고리즘에 비해 제안한 알고리즘 IKA의 속도가 월등하고, 커다란 그래프를 처리할 수 있음을 보였다. 또한 정확한 값의 영향력을 계산하였을 때와 근사 알고리즘의 결과 역시 유사함을 보였다.

핵심 단어 - 특정 사용자가 존재하는 네트워크; 친구 추천 알고리즘; 영향력 최대화; 소셜 네트워크에서의 정보 전파; Katz Centrality의 근사 방법

감 사 의 글

연구실을 정하지 못하고 있는 저를 받아 주시고, 석사 과정 동안 제가 하고 싶은 연구를 진행할 수 있게 도와주신 지도교수 김경국 교수님께 감사합니다. 또한 소셜 네트워크 분야에 눈을 뜨게 해 주시고 매주 미팅을 하며 아이디어를 구체화하는데 도움을 주신 이재길 교수님께 감사합니다.

석사 과정동안 보내면서 일 년간 연구실에서 함께 생활한 우리 리스크 연구실 식구들께 먼저 감사의 말씀을 드립니다. 같은 시기에 프로포절을 준비하며 밤을 같이 샌 현석이형, 부드러운 카리스마 랩짱 승균이형, 여행 얘기 많이 나눌 수 있었던 소정누나, 유자차 건환이형, 시크하지만 틈틈히 많은 조언 해준 등건이형, INFORMS 베스트 페이퍼를 수상하여 랩을 빛낸 동영이형, 항상 챙겨주려 노력하는 도원이, 항상 좋은 말동무가 되준 희랑누나. 일년 간 같은 공간에서 함께 하면서 편하게 지내게 해 준 것에 감사하고, 연구실 생활을 하면서 네트워크, 에너지, 금융 상품 헤징까지 다양한 연구 주제를 이야기하는 등 견문을 많이 넓힐 수 있었습니다.

또한 앞으로의 박사 과정동안 함께 하게 될 이재길 교수님 연구실 분들, 데이터 마이닝 세미나에 참여할 기회를 주신 강유 교수님 연구실 분들 모두 오래동안 알고 지내고 싶습니다. 특히 학부 시절부터 친하게 지내온 산업 및 시스템 공학과 동기이자, 이제는 카이스트를 떠나 사회에 첫 발을 내딛는 세훈이형과 주영이형 각자의 길에서 최선을 다하길 기원합니다. 마지막 학기 논문을 쓰는 기간 함께 공감하며 힘을 북돋아 주웠던 덕종이형도 석사 졸업을 축하합니다.

학과 밖에서 알게 된 분들께도 감사를 전하고 싶습니다. 허물 없이 놀아주고 피아노 앙상블의 재미를 알게 해 준 카이스트 피아노 동아리 PIAST 친구들, 함께 하는 운동의 재미를 알게 해 준 대학원 수영 동호회 수달 멤버들, iPodia 조교를 함께 하며 만나게 된 용일이형과 이경민 선생님, 모리슨 교수님과 5개국 많은 학생들, 그리고 최근에 알게 된 스윙 피버 식구들. 이 분들과 함께한 다양한 경험이 있었기에 지겨울 법도 한 카이스트 캠퍼스에서 7년째 보내면서도 항상 웃음을 잃지 않았습니다. 그리고 내 베프, 이젠 취업 준비하느랴 바쁜 은홍이와 성희, 내가 비록 대전에 있어서 항상 같이 놀지는 못하지만 계속 친하게 지내줘서 고맙다.

끝으로 대전에 지내는 기간이 길어질수록 집에 소홀해 졌는데, 멀리서나마 항상 응원해 주신 부모님과 동생에게 감사합니다.

사람과 사람 사이의 관계에서 필연적으로 발생하는 서로간의 비대칭성에 대해서 생각해보고, 이를 수치화하기 위한 방법이 무엇이 있을까 고민하였습니다. 위의 비대칭성을 깨고 특정한 사용자에게 주목을 받고 싶은 한 유저의 입장에서 적절한 친구 추천은 간접적으로 영향력을 극대화할 수 있는 좋은 어플리케이션이 될 것이라 생각했고, 특정 사용자가 존재하는 네트워크에서 친구를 추천하는 방법을 제안하였습니다. 어떤 문제를 제시하고 좀더 나은 방법으로 해결하는 하나의 사이클을 이번 논문을 작성하면서 경험해 볼 수 있어서 보람찼습니다. 아직은 미진한 수준이지만 이 작은 결실이 앞으로 진행할 연구에 큰 원동력이 되길 기대합니다.

Curriculum Vitae

Name : Sundong Kim

Date of Birth : February 11, 1991

E-mail : sundong.kim@kaist.ac.kr

Educations

2006. 3. – 2008. 2. Sungil High School

2008. 2. – 2013. 2. Industrial & Systems Engineering, KAIST (B.S.)

2013. 3. – 2015. 2. Industrial & Systems Engineering, KAIST (M.S.)

Career

2013. 2. – 2013. 6. IE481 Principles and Practices of Global Innovation (iPodia) T.A

2014. 2. – 2013. 6. IE481 Principles and Practices of Global Innovation (iPodia) T.A

2013. 8. – 2014. 8. Research on Ontology Boostrapping module in Exobrain project

Academic Activities

1. **Sundong Kim**, Minseo Kang and Jae-Gil Lee, A Method of Automatic Schema Evolution on DBpedia Korea, *Korea Information Processing Society*, Spring 2014, *Oral Presentation*.
2. Minseo Kang, Jaesung Kim, **Sundong Kim** and Jae-Gil Lee, Ontology Visualization Tool for Evolved Ontology, *Korea Computer Congress* 2014, *Poster Presentation*.
3. Minseo Kang, Jaesung Kim, **Sundong Kim** and Jae-Gil Lee, Building a Schema of the Korean DBpedia Ontology, *The 26th Annual Conference on Human and Cognitive Language Technology 2014*, *Poster Presentation*.