Influence Maximization From Cascade Information Traces in Complex Networks in the Absence of Network Structure

Naimisha Kolli and Balakrishnan Narayanaswamy

***Abstract*— Influence maximization refers to the problem of selecting the most influential source set of a given size in a diffusion network. Most of the existing literature assumes the knowledge of the underlying network and the knowledge of the diffusion model of propagation to solve the influence maximization problem. However, both the real-world information networks and their diffusion models are not easy to determine in practice. In this article, an influence maximization algorithm based on the observed cascades has been proposed. A novel idea that a good subset of nodes for influence maximization should be composed of nodes that are not only active and strong or influential by themselves but also independent of each other has been proposed. Based on this premise, the problem has been cast as a quadratic integer programming problem with constraints. Furthermore, this problem has been shown to be equivalent to a particular class of Max-GP problem, namely, max-not-cut with size *k* (MNC), for which the state-of-the-art semidefinite programming (SDP) solution techniques exist with guaranteed performance ratios, although the original problem is NP-hard. The new SDP-based method presented in this article is tested on a number of synthetic as well as real-world data sets and has been shown to perform better than the state-of-the-art influence maximization algorithms which work with the apriori knowledge of the network and assume a particular diffusion model. The results demonstrate its practical applicability in applications using Twitter, blogosphere, and other social networks which are being increasingly used to target influential customers for viral marketing.**

***Index Terms*— Complex networks, influence maximization, online social networks semidefinite programming (SDP).**

I. Introduction

INFLUENCE maximization refers to the problem of selecting the most influential source set of a given size in a diffusion network [1]–[4]. In other words, the primary goal is to automatically select the target set of nodes or entities such that the effectiveness of a campaign is maximized. A diffusion process that starts in such an influential set of nodes is most likely to reach the largest number of nodes in the network [3]. Whether it is blogosphere, or viral marketing, or epidemiology, the influence maximization problem reduces to choosing a set of initial sources that together are expected to influence the largest percentage of the network in the shortest time. In all these cases, the solution of the influence maximization problem helps toward developing effective media campaigns, maximizing product adoptions or for optimal immunization policies.

The influence maximization problem as first proposed by Richardson and Domingos [5] and later by Kempe *et al.* [1] is as follows: we are given a network G with pairwise user influence probabilities (edge weights) and a parameter k, and want to find a set A of k users (nodes) such that the expected spread of influence is maximized [6]. There are two fundamental assumptions that have been made in the above-referenced work to solve the influence maximization problem. Most of the existing literature [1]–[4], [7] assumes the knowledge of the underlying network and the diffusion model and have developed algorithms only when the network G is known and assuming a particular diffusion model of how the influence propagates over the edges of this network.

However, in most real-world applications, both these assumptions are not appropriate. The network over which the information propagates is usually unknown and unobserved. Most social networks do not provide full information about their underlying network structure. Commonly, one only observes the times when particular people post or adopt a piece of information but does not observe who explicitly influenced them, i.e., only the information propagation traces called cascades are observed. For example, Twitter does not provide direct access to its follower–followee network through its application programming interface (API). It provides access to a stream of time-stamped tweets which only show when users tweet, but not who influenced them to do so. Also, with growing concerns about Facebook's privacy policy, the general preference of most Facebook users is to keep their friendship profiles private. Furthermore, complete network information may not be available because of the costs of collecting it. The exponential growth in volumes of these social networks, and the growing concerns of users over privacy-related issues, will only exacerbate the problem of knowing the true underlying network over time. Also, in practice, as only the cascades are observed, a particular diffusion model should be assumed in order to model the available cascade data. Again, this assumption that the diffusion model is known may not be appropriate for real-world applications as pointed out by Du et al. [8]. They have argued that the real-world information diffusion model is not easy to determine in practice, and the success

Manuscript received May 6, 2019; revised July 16, 2019; accepted August 26, 2019. This work was supported by the Ministry of Electronics and Information Technology of the Government of India through a sponsored project. *(Corresponding author: Naimisha Kolli.)*

The authors are with the Indian Institute of Science, Bengaluru 560012, India (e-mail: naimisha.k@gmail.com; balki@iisc.ac.in).

Digital Object Identifier 10.1109/TCSS.2019.2939841

of the developed algorithms is susceptible to diffusion model misspecifications. When the diffusion model is also hidden, not only the hidden network structure has to be learned but also the parameters of the diffusion model. This complicates the model inference problems further.

In this article, unlike previous approaches, the influence maximization problem has been tackled directly from the cascade data with no knowledge of the network structure or the diffusion model. This generalization of the influence maximization problem makes it applicable to a wider class of situations when the cost of collecting the complete network information is prohibitive. A novel idea proposed in this article is that a good subset of nodes for influence maximization should be composed of nodes that are not only active and strong or influential by themselves, but also independent of each other. In other words, the nodes should be influential in different parts of the network for them to affect a large percentage of the overall network in a short time. Based on this premise, the problem has been cast as a quadratic integer programming problem with constraints. We have derived its equivalence to a particular class of Max-GP problem, namely, max-not-cut with size k (MNC) which is known to be NP-hard and for which very good approximate algorithms based on the state-of-the-art semidefinite programming (SDP) solution technique exist. Using this derived equivalence, we propose an approximate SDP algorithm (SDPA)-based algorithm to efficiently solve the influence maximization problem.

The new SDPA-based method presented in this article is tested on a number of synthetic as well as real-world data sets and has been shown to perform better than the state-of-the-art two-stage influence maximization algorithms namely, GREEDY [1] and INFLUMAX [3], both of which work with the apriori knowledge of the network and assume a particular diffusion model and also with INFLULEARNER [8] which makes no assumptions on the structure of the network or diffusion model.

The rest of this article is organized as follows. Section II discusses the related work. Section III presents the problem formulation, and in Section IV, the proposed framework is discussed. In Section V, illustrative examples that demonstrate the efficiency of the proposed SDPA-based algorithm using both synthetic as well as real data are presented. The results are shown to compare favorably with those reported in the literature. The conclusions are summarized in Section VI.

II. Related Work

In this section, we present a brief survey of influence maximization algorithms and network inference algorithms proposed in the literature.

As mentioned previously, the influence maximization problem was first proposed by Richardson and Domingos [5]. Richardson and Domingos [5], [7] model the social network based on Markov random fields and developed methods to calculate the customer's network value based on the effect that marketing to such a person has on the rest of the network. Kempe *et al.* [1] were the first to study the problem of influence maximization as a discrete optimization problem and investigated its computational issues under two diffusion models, namely, linear threshold (LT) and independent cascade (IC) models. They proposed a greedy approach based on the submodularity of the influence function ($\sigma(.)$). Leskovec *et al.* [9] proposed a "cost-effective-lazy-forward" (CELF) strategy in choosing new seeds, which significantly reduced the number of influence spread evaluations. Goyal *et al.* [6] proposed an optimized version of CELF by exploiting the submodularity property of social influence function and named it as CELF++. Chen *et al.* [2] proposed the maximum influence arborescence (MIA) model of influence propagation and proposed greedy algorithms for selecting a seed set based on this diffusion model. Rodriguez and Schölkopf [3] and Du *et al.* [4] propose the continuous-time IC (CIC) diffusion model and study the influence maximization problem in this setting. All the above works determine an effective seed set when a particular diffusion model is assumed and the underlying the network G is given.

For the instances where the underlying network G is unknown and needs to be inferred, there exist algorithms in the literature for inferring the network G^* from the cascades [10]–[13]. Some of the approaches infer only the network structure [10], while others infer the strength of the edges in the network [11]–[13]. Rodriguez *et al.* [10] (NETINF) use a generative probabilistic model for inferring the network connectivity based on the submodular optimization [10]. Myers and Leskovec [11] (CONNIE) infer not only the connectivity but also a prior probability of infection for every edge using a convex program. However, the cascade probability between all nodes is fixed and not inferred in these works. In contrast, Rodriguez *et al.* [12] (NETRATE) allows transmission at different rates across different edges to infer temporally heterogeneous interactions within a network. They have applied survival theory to develop general additive risk models under which the network inference problems can be solved efficiently by exploiting their convexity. A few methods [13], [14] consider both temporal information and additional nontemporal features. All the above methods assume a particular diffusion model, the IC model, the LT model, or the CIC model in order to infer the network structure from the cascade data. Once the network structure has been inferred, the existing influence maximization algorithm [1]–[4] can be used to find the optimal set of nodes on the inferred network G^*. In these works, the inferred network G^* relies on the assumed diffusion model.

Another two-stage solution is proposed in the literature by Du *et al.* [8]. An influence function learning framework called INFLULEARNER to learn the influence functions directly from the cascade data when the network structure as well as the diffusion model is unknown has been proposed. The greedy algorithm framework of Nemhauser *et al.* [15] on the influence functions learned from INFLULEARNER has then been applied for solving influence maximization problem. Although it does not assume a diffusion model or a network structure explicitly, this is still a two-stage approach of first learning the influence function of all the nodes and then selecting an influential subset of users.

III. Problem Formulation

The problem is formally described as the problem where many different cascades diffuse over an unknown static network, and for each of them, one observes the times when nodes participated but not who influenced them to participate. In the information diffusion setting, each cascade corresponds to a different piece of information that spreads over the social network and only the times when particular nodes mentioned that particular information are available. Given a collection of cascades $C = \{C_1, C_2, \ldots, C_m\}$ where every cascade C_c is a collection of node and timestamp pairs of the form $< v, t_v >$ denoting when (t_v) the particular node v participated in the cascade C_c, the goal is to find the optimal set of source nodes A of size k in a diffusion network that maximizes the spread of information and influence. However, no apriori knowledge of the network or diffusion model is assumed.

A. Construction of the Cascade Matrix C

We record the trajectory of each cascade c as a directed graph from the cascade information traces in the matrix C_c as follows:

$$\begin{aligned} C_c(u,v) &= 1, \text{ if } u \neq v \text{ and } t_u^c < t_v^c \\ &= 1, \text{ if } u = v \text{ and } t_u^c < \infty \\ &= 0, \text{otherwise} \end{aligned} \tag{1}$$

where t_u^c and t_v^c are the times when node u and node v participated in the cascade c. The above matrix is constructed for each cascade and summed over and normalized by the number of cascades M to form a matrix C

$$C = \frac{1}{M} \sum_{c \varepsilon C} C_c. \tag{2}$$

The diagonal terms of C $(C(u,u))$ denote the fraction of cascades that each node u participated in while the off-diagonal terms $(C(u,v))$ denote the fraction of cascades in which u participated in a cascade before v and hence could have influenced v.

For the selection of a good subset of nodes for influence maximization, it is proposed in this article that the set should compose of nodes that have two major characteristics: 1) they should be active or influential by themselves and 2) be independent of each other. This would ensure that the selected nodes affect information propagation in different parts of the network and also eliminate more correlated influential nodes from the selection. In this article, the first characteristic of nodes being influential by themselves is called individual influence or strength and the second characteristic of independence of the members of the influential set is called the heterogeneity. Maximizing both strength and heterogeneity of the selected subset would lead to the solution to the problem of predicting an optimal set of nodes A. Therefore, there must be a tradeoff between the strength and the heterogeneity of an influential subset. A subset of nodes with the best tradeoff so that the generalized influence spread is optimal would yield the solution. In fact, this kind of formulation has been used for problems such as feature selection [16], ensemble pruning [17], and kernel optimization [18]. The construction of an influence matrix Q based on these two characteristics is shown in Section III-B.

B. Construction of the Influence Matrix Q

From cascade matrix C, the influence matrix Q is constructed. The diagonal terms of matrix Q $(Q(u,u))$ should reflect the total number of nodes that node u can possibly influence while the off-diagonal terms of matrix Q. $(Q(u,v))$ reflect the heterogeneity of nodes that can be influenced by node pairs u and v. The fraction of nonzero entries of the uth row of matrix C is used to populate the diagonal elements of Q. Thus, $Q(u,u)$ captures the influence strength of a node u. Furthermore, Jaccard coefficient [19] or similarity is calculated between all the pairs of nodes or rows of matrix C. This measures the overlap of nodes that are influenced by node pairs u and v. To measure the heterogeneity, the Jaccard distance (the complement of the Jaccard coefficient found by subtracting the Jaccard coefficient from one) has been used to populate the off-diagonal elements $Q(u,v)$ as follows:

$$\begin{aligned} Q(u,v) &= \frac{1}{N} \sum_{j=1}^{N} (I_{(C(u,j)>0)})) \text{ if } u = v \\ &= d_J(u,v) = (1 - J(u,v)) \text{ if } u \neq v \end{aligned} \tag{3}$$

where $I_{(C(u,j)>0)}$ is an indicator function that is 1 when $C(u,j)$ is nonzero and 0 otherwise, $d_J(u,v)$ and $J(u,v)$ are the Jaccard distance and the Jaccard coefficient between u and v, respectively [19]. The constructed Q matrix captures both the individual influence (along with diagonal elements) and the pairwise heterogeneity (along with off-diagonal elements) between all the nodes. It follows naturally that for a good influential subset, all diagonal elements as well as all off-diagonal elements of this matrix should be large. Intuitively, the sum of the diagonal elements measures the overall influence strength of the set of the nodes while the sum of the off-diagonal elements measures the dissimilarity. A combination of these two terms should be a good approximation of the influence spread. All elements of the Q matrix are between 0 and 1.

C. Influential Subset Selection Problem

The influential subset selection problem defined in Section III-B can now be formulated as a quadratic integer programming problem. Essentially, the optimal solution is to look for a subset of nodes of size k, with the sum of the corresponding diagonal elements in the Q matrix and also the sum of the corresponding off-diagonal elements representing heterogeneity maximized. Finally, to choose the optimal subset A, the above objective can be mathematically expressed as follows:

$$\begin{aligned} &\max_x x^{\mathrm{T}} Q x \\ &\text{s.t. } \sum_i x_i = k \\ &\quad x_i \in \{0,1\}. \end{aligned} \tag{4}$$

The binary variable x_i represents whether the ith node will be chosen. If $x_i = 1$, which means that the ith node is included in the selected set A, its corresponding diagonal and off-diagonal elements will be accounted for in the objective function. Note that the cardinality constraint $\sum_i x_i = k$ is mathematically important because without it, there is only one trivial solution to the problem with all of the nodes picked. In addition, it gives control over the size of the selected subset.

IV. Proposed Framework

Equation (4) is a quadratic integer programming problem of the standard 0–1 optimization problem. In Sections IV-A–IV-C, it is shown that the problem is equivalent to MNC. The advantage of this equivalence is that MNC belongs to the general class of Max-GP problems that are known to have good approximate solutions based on the state-of-the-art SDP solution technique although the original problem is NP-hard [20]. Given an undirected graph $G = (V, E)$ and $|V| = N$, nonnegative weights w_{ij} on edges $(i, j) \in E$, and an integer $k(1 < k < N)$, the goal of general Max-GP problems is to determine a subset $A \subset V$ of k nodes such that an appropriate objective function $w(A)$ is maximized. Based on this derived equivalence, we propose an SDP relaxation to (4) in which each binary variable x_i is replaced by a vector-valued decision variable, resulting in a convex optimization problem that can be solved to a prescribed accuracy in polynomial time.

A. Equivalence to Max-Not-Cut With Size k

As mentioned above, our problem formulation is close to the MNC problem, in which one partitions the vertices of an edge-weighted graph into two sets, one of which has size k, so that the total weight of edges noncrossing edges between the partition is maximized [20]. This is formulated as follows:

$$\begin{aligned} &\max_y \frac{1}{2}\sum_j \sum_{i<j} w_{ij}(1+y_i y_j) \\ &\text{s.t. } \sum_i y_i = N - 2k \\ &\quad y_i \in \{-1, 1\} \end{aligned} \tag{5}$$

where N is the number of the nodes in the graph. This is equivalent to partitioning the nodes via the assignment of $y_i = 1$ or $y_i = -1$ to each node i in the graph and maximizing the sum, $\sum w_{ij} y_i y_j$. Here, the interaction term $y_i y_j$ equals 1 when i and j are in the same set of the partition, and maximizing contributes to the maximization of dissimilarity within the partition. Equation (5) can be written in quadratic form (ignoring the coefficients) as follows:

$$\begin{aligned} &\max_y y^{\mathrm{T}} W y \\ &\text{s.t. } y^{\mathrm{T}} I y = N - 2k \\ &\quad y_i \in \{-1, 1\} \end{aligned} \tag{6}$$

where W is the edge-weight matrix and N is the number of the nodes in the graph. In (4) $x_i \in \{0, 1\}$, and in (6) (for MNC) $y_i \in \{-1, 1\}$. Therefore, a transformation of variables can be done by letting

$$x_i = \left(\frac{v_i + 1}{2}\right) \text{ and } v_i \in \{-1, 1\}. \tag{7}$$

Now, the objective function in (4) can be rewritten as follows:

$$\frac{1}{4}(v + e)^{\mathrm{T}} Q (v + e) \tag{8}$$

where e is a column of vector of all 1s. The original cardinality constraint in (4) can be written as follows:

$$x^{\mathrm{T}} I x = k \tag{9}$$

where I is the identity matrix. After the variable transformation, this constraint becomes $(v + e)^{\mathrm{T}} I (v + e) = 4k$.

By expanding the variable $v = (v_1, v_2, .., v_n)$ into $v = (v_0, v_1, v_2, .., v_n)$ and let $v_0 = 1$, both the transformed objective function and the cardinality constraint can be put back into a nice quadratic form. Following the variable expansion, a new square matrix $H(N + 1\text{dimension})$ is constructed as follows:

$$H = \begin{pmatrix} e^{\mathrm{T}} Q e & e^{\mathrm{T}} Q \\ Q e & Q \end{pmatrix}. \tag{10}$$

So that the objective function is now equivalent to $v^{\mathrm{T}} H v$. Similarly, a new square matrix $D(N + 1\text{dimension})$ is also constructed as follows:

$$D = \begin{pmatrix} N & e^{\mathrm{T}} \\ e & 1 \end{pmatrix}. \tag{11}$$

The cardinality constraint is $v^{\mathrm{T}} D v = 4k$. Rewriting the original problem in (4) after applying transformations from (7)–(11), the formal equivalence between (6) and (12) can be shown as follows:

$$\begin{aligned} &\max_v v^{\mathrm{T}} H v \\ &\text{s.t. } v^{\mathrm{T}} D v = 4k \\ &\quad v_0 = 1 \\ &\quad v_i \in \{-1, 1\}, \quad \forall i \neq 0. \end{aligned} \tag{12}$$

B. Transforming to an SDP

The advantage of the derived equivalence between the MNC problem in (6) and our problem defined in (12) in Section IV-A is that the MNC problem can be solved by efficient approximate algorithms based on SDP with guaranteed performance ratios. The key to applying an SDP relaxation for solving (12) is to relax each binary variable into a unit vector. Equation (12) can be written as follows:

$$\begin{aligned} &\max_v H \cdot v v^{\mathrm{T}} \\ &\text{s.t. } D \cdot v v^{\mathrm{T}} = 4k \\ &\quad v_0 = 1 \\ &\quad v_i \in \{-1, 1\}, \quad \forall i \neq 0 \\ &\quad \text{where } A \cdot B = \sum_{ij} A_{ij} B_{ij}. \end{aligned} \tag{13}$$

To construct the relaxation, the constraint $v_0 = 1$ can be relaxed to $v_o \in \{-1, 1\}$ without changing the problem since $-v$ is feasible for the remaining constraints if and only if v is feasible and since $Hvv^{\mathrm{T}} = H(-v)(-v)^{\mathrm{T}}$. Next, rewriting the constraints $v_i \in \{-1, 1\}, i = 0, 1, ..., n$ as the single, collective constraint $\mathrm{diag}(vv^{\mathrm{T}}) = e$, (13) can be transformed to the following formulation:

$$\begin{aligned} &\max_v H \cdot vv^{\mathrm{T}} \\ &\text{s.t. } D \cdot vv^{\mathrm{T}} = 4k \\ &\quad \mathrm{diag}(vv^{\mathrm{T}}) = e. \end{aligned} \tag{14}$$

Next, substitute $V = vv^{\mathrm{T}}$, and note that V can be expressed as vv^{T} if and only if $V \geq 0$ with $\mathrm{rank}(V) = 1$, which gives

$$\begin{aligned} &\max_v H \cdot V \\ &\text{s.t. } D \cdot V = 4k \\ &\quad \mathrm{diag}(V) = e \\ &\quad V \geq 0 \text{ and } \mathrm{rank}(V) = 1. \end{aligned} \tag{15}$$

The SDP relaxation is now obtained by dropping the rank constraint, which yields the following (convex) SDP:

$$\begin{aligned} &\max_v H \cdot V \\ &\text{s.t.} D \cdot V = 4k \\ &\quad \mathrm{diag}(V) = e \\ &\quad V \geq 0. \end{aligned} \tag{16}$$

Now the original NP-hard problem is relaxed into a convex SDP problem which can be solved in polynomial time. The approximation with the SDP relaxation along with a method to round the optimal solution of the SDP relaxation is discussed in Section IV-C.

C. Solving SDP Relaxation

The primal-dual interior-point method (PDIPM) [21] is the preferred polynomial time algorithm for SDPs, since it solves large-scale SDPs with numerical stability and accuracy. PDIPM solves the primal and the dual problems simultaneously keeping the positive definiteness of the variable matrices and reduces the duality gap until the approximate optimal solution of primal and dual is found. In this article, the SDP relaxation of (16) is solved by using the publicly available package SDPA-C [22]. The SDPA-C [22] implements a variant of the PDIPM using the positive definite matrix completion technique for solving the SDP which takes advantage of the sparsity.

Furthermore, an improved method to round the optimal solution of the SDP relaxation introduced by Han *et al.* [20] is implemented. This rounding technique is related to the well-known rounding method introduced by Goemans and Williamson [23] in their seminal work for MAX-CUT. As argued by Han *et al.* [20], the difference between the MAX-GP problem and the MAX-CUT problem is that two objectives are sought in MAX-GP. They are: 1) the objective value of $w(A)$ and 2) the size of A. Ideally, both a high $w(A)$ and a small difference between $|A|$ and k are preferred; otherwise, adding or subtracting nodes from A needs to be done, resulting in a deterioration of $w(A)$ at the end. The randomized rounding algorithm proposed by Han *et al.* [20] is built upon this need for balance and tries to balance the (expected) quality of $w(A)$ and the (expected) size of A. This technique known as r-approximation technique has a performance guarantee of better than 0.5 for MNC and MC (MAX-CUT) for a wide range of k and in fact has a performance guarantee of better than 0.845 for $k = 3n/5$ for max vertex cover (MVC). For MNC problems with $k = n/2$, the performance guarantee is better than 0.6026.

V. Experiments

The proposed SDPA based algorithm is evaluated on: 1) three synthetic networks and 2) two real-world networks. Here, the same outline of experiments as followed by INFLUMAX [3] has been carried out for both synthetic and real data. The performance of our algorithm has been compared to: 1) INFLUMAX [3]; 2) GREEDY [1], both of which require the knowledge of the network; 3) NETRATE based, which first infers the network from the available cascades using NETRATE [12] and then uses INFLUMAX to choose the subset of sources; and 4) RANDOM algorithm which randomly chooses a source subset. The performance of our algorithm has also been compared with INFLULEARNER [8] on real networks, which first learns the influence functions with no knowledge of the network or the diffusion model and then applies a greedy algorithm proposed by Nemhauser *et al.* [15] to find the subset k of most influential nodes.

Given a diffusion process that started in the set of source nodes A, $N(A; T)$ is the number of nodes infected up to time T and then the influence function $\sigma(A; T)$ is the average total number of nodes infected upto time T, i.e.,

$$\sigma(A;T) = E[N(A;T)]. \tag{17}$$

The reference for this can be found in (2) in INFLUMAX [3]. This measure has been used to evaluate the performance in all the experiments. For all methods considered, the influence achieved has been computed by evaluating the above influence function for the set of sources selected by the respective method on the respective networks.

In order to evaluate the quality of the solution in (17), it is necessary to provide the network and it assumes the CIC model of diffusion. Wherever network information is available (synthetic networks), the actual network is used to calculate the influence function. In cases where the actual network is not available, the network inferred from the public implementation of NETRATE [24] has been used to calculate the influence function. Though our proposed SDPA solution does not require any apriori knowledge of the network or the diffusion model, it has been done purely for comparison purposes. Since the network and the assumed diffusion model (CIC) are identical to the inputs given to INFLUMAX, the comparison is expected to be biased toward INFLUMAX.

A. Experiments and Results on Synthetic Data

Experimental Setup: Experiments have been carried out on two models of synthetic networks that mimic the structure

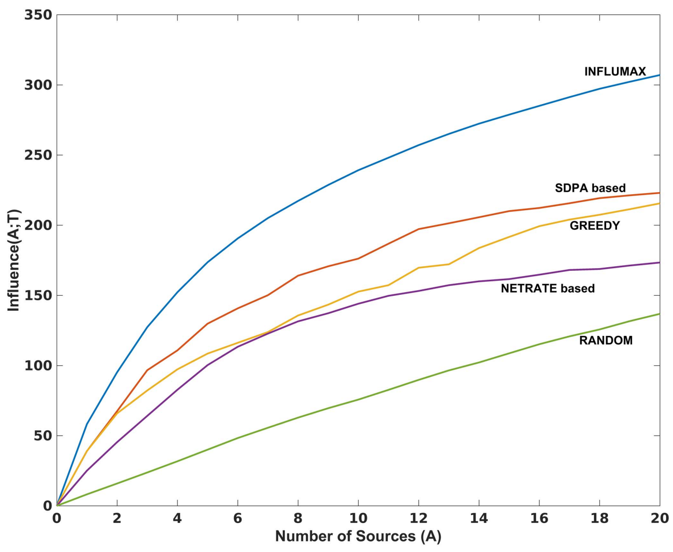

Fig. 1. Comparison of influence function of different algorithms versus the number of sources for synthetic 512-node random Kronecker network at $T = 1$.

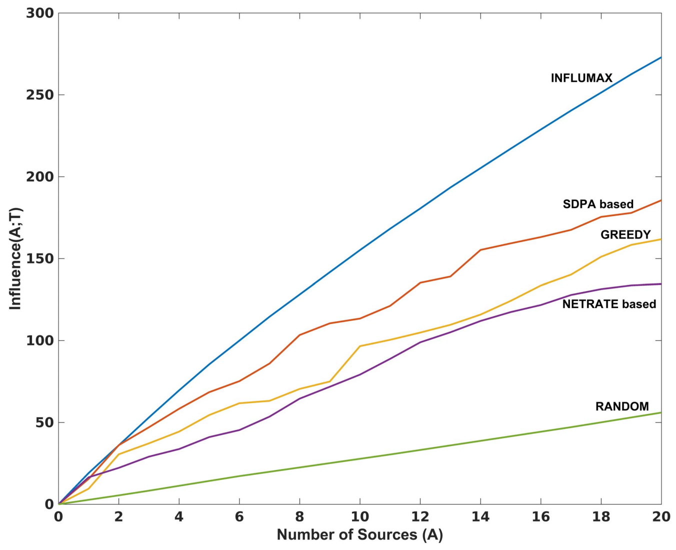

Fig. 2. Comparison of influence function of different algorithms versus the number of sources for synthetic 1024-node hk network at $T = 1$.

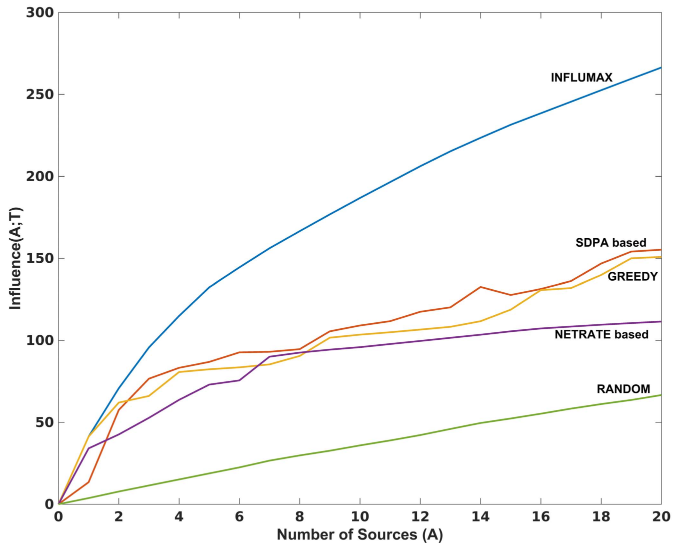

Fig. 3. Comparison of influence function of different algorithms versus the number of sources for synthetic 1024-node Forest Fire network at $T = 1$.

of real-world social networks: Kronecker [25] and Forest Fire (scale-free) [26] networks. We consider two types of Kronecker network models with very different structures: random (parameter matrix $[0.5, 0.5; 0.5, 0.5]$, 512 nodes) and hierarchical [27] ($[0.9, 0.1; 0.1; 0.9]$, 1024 nodes). The Forest Fire network [26] has 1024 nodes. All three generated networks G have approximately two edges in average per node. The time horizon is set to $T = 1.0$, and the transmission rates are drawn from a uniform distribution $\alpha \sim U(0.01, 1)$. The transmission rate for an edge (i, j) models how fast the information spreads from node i to node j in the social network. Then, around 6000 cascades are generated over G based on the CIC diffusion model (CTC). Root nodes of cascades are chosen uniformly at random.

Results: The comparison of results of the influence function with the number of sources for random Kronecker, hierarchical Kronecker (hk), and Forest Fire networks are given in Figs. 1–3, respectively. It can be seen from the results that the RANDOM algorithm performs worst as expected. INFLUMAX produces a performance better than all the other algorithms as it has both the knowledge of the network as well as the correct diffusion model (CIC). However, as has been discussed earlier, its practical utility is limited due to its assumptions on network and diffusion model. GREEDY algorithm has the knowledge of the network, but it assumes the IC model of diffusion, and it only performs marginally better than the NETRATE-based algorithm. The performance of two-stage approach used in the NETRATE-based algorithm lies between that of GREEDY and RANDOM. The SDPA-based method proposed in this article performs better than the GREEDY, NETRATE-based, and RANDOM algorithms still retaining the advantages of having no knowledge of the network or the diffusion model and being a single-stage approach.

It is seen in Figs. 1–3, which exhibit a similar trend, the SDPA-based algorithm proposed in this article provides superior performance to popular two-stage approaches and influence maximization algorithms with known network information. This shows that significantly better results can be obtained by explicitly considering the dynamics of information cascades in a network, rather than relying solely on the network structure of the graph.

As explained previously, the observed advantage of INFLUMAX in these graphs is due to the choice of the performance metric. It is worth noting that in the case of Forest Fire network, our algorithm is only marginally better than GREEDY.

B. Experiment and Results on Real Data

Experimental Setup: The publicly available MemeTracker data set [28] has been used, which contains more than

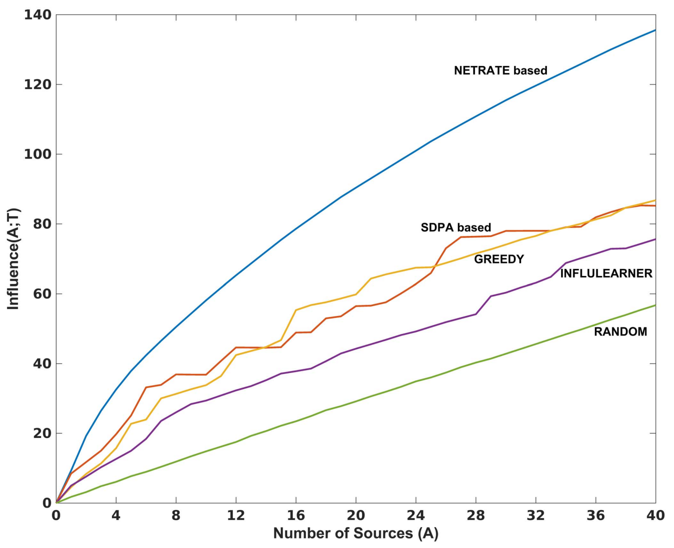

Fig. 4. Comparison of influence function of different algorithms versus the number of sources for real-world MemeTracker network at $T = 1$.

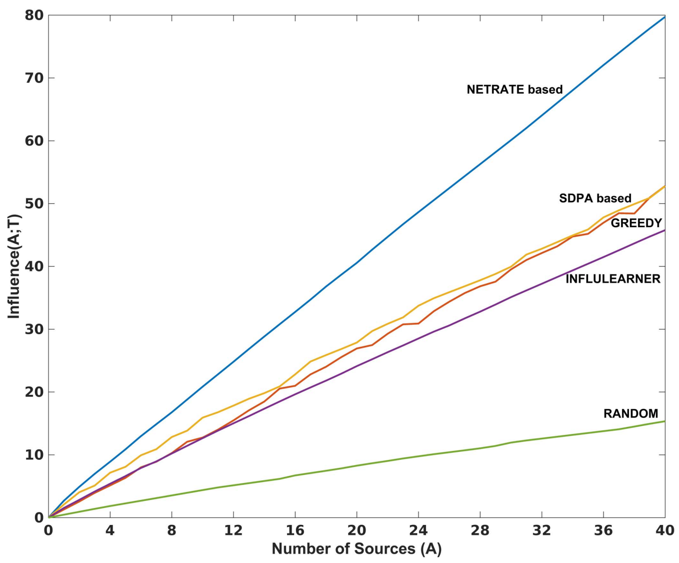

Fig. 5. Comparison of influence function of different algorithms versus the number of sources for real-world Hyperlinks network at $T = 1$.

172 million news articles and blog posts. Information is traced in two different ways and then two different diffusion networks have been inferred using NETRATE [12]. Again, this is the same outline of experiments followed in INFLUMAX [3].

MemeTracker Data Set: The phrase-cluster data set provided by Snap [28], [29] has been used. It contains clustered phrases that aggregate different textual variants of the same phrase and 12 000 largest clusters are considered. Each phrase cluster is a MemeTracker cascade. Each cascade consists of a collection of time-stamps when sites (in blog posts) first mentioned any phrase in the cluster. From the MemeTracker cascades, an underlying diffusion network with the top (in terms of phrases) 1000 media sites and blogs has been inferred using NETRATE [12]. Then, the network is further sparsified by keeping the 1000 fastest edges since it has been shown that in the context of influence maximization, computations on sparsified models give up little accuracy, but improves scalability [30].

Hyperlink Cascade Data Set: Each hyperlink cascade consists of a collection of time-stamped hyperlinks between sites (in blog posts) that refer to closely related pieces of information. First, more than 100 000 hyperlink cascades in the MemeTracker data set are extracted. From the hyperlink cascade data, an underlying diffusion network with the top (in terms of hyperlinks) 1000 media sites and blogs has been inferred using NETRATE [12].

Results: The comparison results of the influence function with the number of sources for MemeTracker Cascades network and Hyperlink Cascades network are given in Figs. 4 and 5, respectively. It can be seen from the results that the RANDOM algorithm performs worst as expected. In these real-world data sets, knowledge of the underlying network is not available, and hence, this is one of the example scenarios where INFLUMAX cannot be applied directly limiting its practical applicability. For experiments on real-world networks, the inferred network given by NETRATE has been used as the ground truth network on which the solution quality of all the algorithms considered has been evaluated and hence observed the superior performance of the NETRATE-based algorithm. Another algorithm compared for real networks is INFLULEARNER [8], discussed previously. Our SDPA-based algorithm gives marginally better performance than INFLULEARNER as well in both the real-world networks considered. In both these cases, GREEDY- and SDPA-based algorithms give similar superior performance consistent with our results on synthetic networks. However, the advantage of SDPA-based algorithm for applications to a wider class of problems is a great asset.

C. Computational Complexity and Memory

The SDP relaxation of (16) is solved by using the publicly available package SDPA-C [22]. The SDPA-C is a C++ software package which solves SDP problems very efficiently (in computation time and memory). This package implements an efficient PDIPM method using the positive definite matrix completion technique. The positive definite matrix completion enables the SDPA-C to store only the sparse matrices and perform matrix computations taking advantages of the sparsity. Nakata *et al.* [31] present numerical experiments to evaluate the positive definite completion method on randomly generated SDPs for max-cut problems. The numerical results run on a DEC Alpha Station (CPU Alpha 21164-600 MHz with 1024 MB) show that this completion method takes 320.9 s of CPU time and uses 19 MB of memory for problems of size $n = 1000$ and $m = 1000$. The n and m correspond to the sizes of the matrices and the number of equality constraints, respectively, for the original SDP in the standard form.

D. Performance Versus Cascade Coverage

Since the cascades have been used as input to our experiments, an important criterion would be to estimate the number

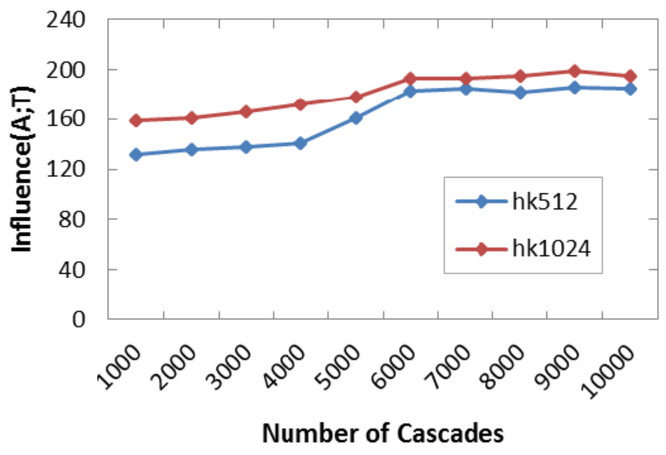

Fig. 6. Dependence of influence function on the number of cascades for hk networks with 512 (hk512) and 1024 (hk1024) nodes.

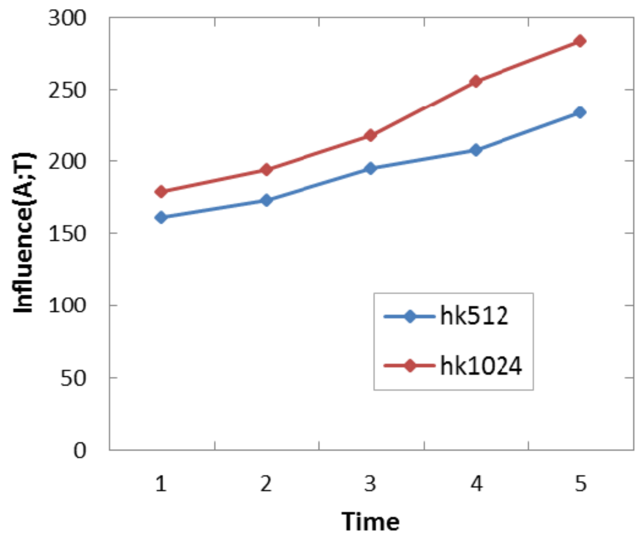

Fig. 7. Dependence of influence function on time for hk networks with 512 (hk512) and 1024 (hk1024) nodes.

of cascades needed to arrive at a solution. This has been studied in this article by studying the convergence of influence function in the synthetic network, namely, hk with parameter matrix [0.9, 0.1; 0.1; 0.9] and two different sizes, namely, 1024 nodes (hk1024) and 512 nodes (hk512). It can be seen in Fig. 6 that around 6000 cascades are sufficient to achieve a good performance in both the cases.

E. Performance Versus Time Horizon

Running simulations over longer periods of time compared to $T = 1$ used in the above experiments for both hk1024 and hk512 networks are shown in Fig. 7. It can be seen from the graphs that the influence function only marginally increases beyond the time $T = 3$ as is to be expected. Furthermore, Fig. 8 shows the comparison of the influence function with time horizon T for all the algorithms considered for an hk network with 512 nodes (hk512) with a source set of cardinality $|A| = 10$. It can be observed that the influence gain achieved during smaller horizons (say $T = 1$) tends to persist over subsequent times. The trend remains the same

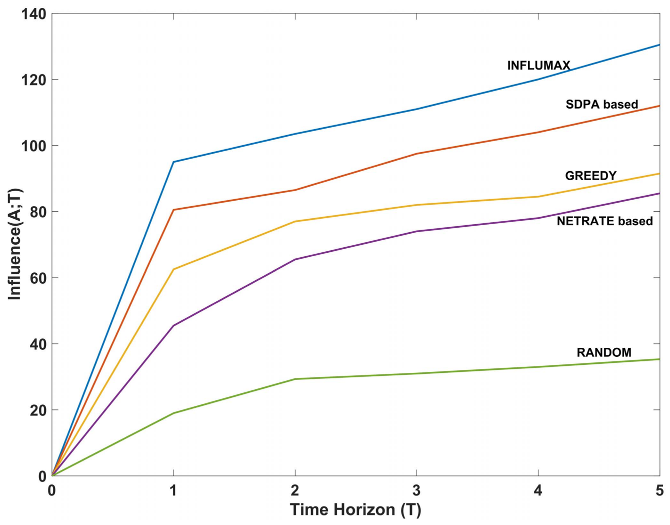

Fig. 8. Comparison of influence function of different algorithms versus time horizon for synthetic hk network with 512 nodes (hk512) and source set cardinality |A| = 10.

across different algorithms compared and hence points to the significant role that temporal dynamics play in the information diffusion especially for smaller horizons irrespective of the algorithm applied.

VI. Conclusion

In this work, a novel algorithm that does not require the knowledge of the network and the diffusion model has been proposed. As opposed to the existing techniques reported in the literature, our method uses a single-stage approach. The novel feature of this algorithm is to cast the influence maximization problem in the absence of apriori knowledge of the network and diffusion model into a quadratic integer program. This is then transformed into a convex SDP that has been shown to have good approximate solutions and thereby making it feasible to find a solution to otherwise an NP-hard problem. The performance of our algorithm has been compared with influence maximization algorithms that work with the knowledge of the network (INFLUMAX and GREEDY) and also with two-stage approaches that first infer the network and then use the existing influence maximization algorithms (NETRATE-based and INFLULEANER). In contrast, the proposed algorithm is able to give comparable or better performance albeit not knowing the network and also making no assumptions on the diffusion model. Experiments have been run on both the synthetic as well as the real-world networks.

The results demonstrate its practical applicability in scenarios where the ground truth network is not known. The proposed algorithm is directly applicable in a variety of everyday applications like Twitter and blogosphere which are being increasingly used to target influential customers for viral marketing where budgets and time constraints limit the availability of the ground truth network.

References

[1] D. Kempe, J. Kleinberg, and É. Tardos, “Maximizing the spread of influence through a social network,” in *Proc. 9th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2003, pp. 137–146.

[2] W. Chen, C. Wang, and Y. Wang, "Scalable influence maximization for prevalent viral marketing in large-scale social networks," in *Proc. 16th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2010, pp. 1029–1038.

[3] M. G. Rodriguez and B. Schölkopf, "Influence maximization in continuous time diffusion networks," 2012, *arXiv:1205.1682*. [Online]. Available: https://arxiv.org/abs/1205.1682

[4] N. Du, L. Song, M. Yuan, and A. J. Smola, "Learning networks of heterogeneous influence," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 2780–2788.

[5] M. Richardson and P. Domingos, "Mining the network value of customers," in *Proc. 7th Int. Conf. Knowl. Discovery Data Mining*, 2001, pp. 57–66.

[6] A. Goyal, W. Lu, and L. V. S. Lakshmanan, "CELF++: Optimizing the greedy algorithm for influence maximization in social networks," in *Proc. ACM 20th Int. Conf. Companion World Wide Web*, 2011, pp. 47–48.

[7] M. Richardson and P. Domingos, "Mining knowledge-sharing sites for viral marketing," in *Proc. 8th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2002, pp. 61–70.

[8] N. Du, Y. Liang, M.-F. Balcan, and L. Song, "Influence function learning in information diffusion networks," in *Proc. Int. Conf. Mach. Learn.*, 2014, pp. 2016–2024.

[9] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. VanBriesen, and N. Glance, "Cost-effective outbreak detection in networks," in *Proc. 13th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2007, pp. 420–429.

[10] M. Gomez-Rodriguez, J. Leskovec, and A. Krause, "Inferring networks of diffusion and influence," *ACM Trans. Knowl. Discovery Data*, vol. 5, no. 4, pp. 21-1–21-37, Feb. 2012.

[11] S. Myers and J. Leskovec, "On the convexity of latent social network inference," in *Proc. Adv. Neural Inf. Process. Syst.*, 2010, pp. 1741–1749.

[12] M. G. Rodriguez, D. Balduzzi, and B. Schölkopf, "Uncovering the temporal dynamics of diffusion networks," 2011, *arXiv:1105.0697*. [Online]. Available: https://arxiv.org/abs/1105.0697

[13] L. Wang, S. Ermon, and J. E. Hopcroft, "Feature-enhanced probabilistic models for diffusion network inference," in *Proc. Joint Eur. Conf. Mach. Learn. Knowl. Discovery Databases*. Berlin, Germany: Springer, 2012, pp. 499–514.

[14] P. Netrapalli and S. Sanghavi, "Learning the graph of epidemic cascades," in *ACM SIGMETRICS Perform. Eval. Rev.*, vol. 40, no. 1, pp. 211–222, 2012.

[15] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher, "An analysis of approximations for maximizing submodular set functions-I," *Math. Programm.*, vol. 14, no. 1, pp. 265–294, 1978.

[16] A. D'Aspremont, L. E. Ghaoui, M. I. Jordan, and G. R. Lanckriet, "A direct formulation for sparse PCA using semidefinite programming," in *Proc. Adv. Neural Inf. Process. Syst.*, 2005, pp. 41–48.

[17] Y. Zhang, S. Burer, and W. N. Street, "Ensemble pruning via semidefinite programming," *J. Mach. Learn. Res.*, vol. 7, pp. 1315–1338, Jul. 2006.

[18] G. R. G. Lanckriet, N. Cristianini, P. Bartlett, L. El Ghaoui, and M. I. Jordan, "Learning the kernel matrix with semidefinite programming," *J. Mach. Learn. Res.*, vol. 5, pp. 27–72, Jan. 2004.

[19] P. Jaccard, "The distribution of the flora in the alpine zone. 1," *New Phytol.*, vol. 11, no. 2, pp. 37–50, Feb. 1912.

[20] Q. Han, Y. Ye, and J. Zhang, "An improved rounding method and semidefinite programming relaxation for graph partition," *Math. Program.*, vol. 92, no. 3, pp. 509–535, 2002.

[21] F. Alizadeh, J.-P. A. Haeberly, and M. L. Overton, "Primal-dual interior-point methods for semidefinite programming: Convergence rates, stability and numerical results," *SIAM J. Optim.*, vol. 8, no. 3, pp. 746–768, 1998.

[22] K. Fujisawa, M. Fukuda, M. Kojima, K. Nakata, and M. Yamashita, "SDPA-C (semidefinite programming algorithm-completion method). User's manual-version 6-10," Dept. Math. Comput. Sci., Tokyo Inst. Technol., Tokyo, Japan, Res. Rep. B-409, 2004.

[23] M. X. Goemans and D. P. Williamson, "Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming," *J. Assoc. Comput. Mach.*, vol. 42, no. 6, pp. 1115–1145, 1995.

[24] J. Leskovec and R. Sosič, "SNAP: A general-purpose network analysis and graph-mining library," *ACM Trans. Intell. Syst. Technol.*, vol. 8, no. 1, 2016, Art. no. 1.

[25] J. Leskovec, D. Chakrabarti, J. Kleinberg, C. Faloutsos, and Z. Ghahramani, "Kronecker graphs: An approach to modeling networks," *J. Mach. Learn. Res.*, vol. 11, no. 3, pp. 985–1042, 2010.

[26] J. Leskovec, J. Kleinberg, and C. Faloutsos, "Graphs over time: Densification laws, shrinking diameters and possible explanations," in *Proc. 11th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2005, pp. 177–187.

[27] A. Clauset, C. Moore, and M. E. J. Newman, "Hierarchical structure and the prediction of missing links in networks," *Nature*, vol. 453, no. 7191, p. 98–101, 2008.

[28] J. Leskovec and A. Krevl. (2014). *SNAP Datasets: Stanford Large Network Dataset Collection*. [Online]. Available: http://snap.stanford.edu/data

[29] J. Leskovec, L. Backstrom, and J. Kleinberg, "Meme-tracking and the dynamics of the news cycle," in *Proc. 15th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2009, pp. 497–506.

[30] M. Mathioudakis, F. Bonchi, C. Castillo, A. Gionis, and A. Ukkonen, "Sparsification of influence networks," in *Proc. 17th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2011, pp. 529–537.

[31] K. Nakata, K. Fujisawa, M. Fukuda, M. Kojima, and K. Murota, "Exploiting sparsity in semidefinite programming via matrix completion II: Implementation and numerical results," *Math. Program.*, vol. 95, no. 2, pp. 303–327, 2003.

Naimisha Kolli received the B.Tech. degree in electronics and communications engineering from Jawaharlal Nehru Technological University, Kakinada, India, in 2004, and the M.Sc. (Engg). degree in social network analysis and application from the Indian Institute of Science, Bengaluru, India, in 2008, where she is currently pursuing the Ph.D. degree with the Department of Supercomputer Education and Research Centre.

She is currently a Project Assistant under Prof. N. Balakrishnan with the Information Systems Lab, Indian Institute of Science. Her current research interests include social network analysis for urban computing and the development of algorithms for online social network applications.

Balakrishnan Narayanaswamy received the B.E. degree (Hons.) in electronics and communication from the University of Madras, Chennai, India, in 1972, and the Ph.D. degree from the Indian Institute of Science, Bengaluru, India, in 1979.

He joined the Department of Aerospace Engineering, Indian Institute of Science, as an Assistant Professor, in 1981, where he became a Full Professor in 1991, served as an Associate Director, from 2005 to 2014, and is currently an Honorary Professor with the Supercomputer Education and Research Centre. He has authored over 200 publications in the international journals and international conferences. His current research interests include numerical electromagnetics, high-performance computing and networks, polarimetric radars and aerospace electronic systems, information security, and digital library.

Dr. Narayanaswamy is a fellow of the World Academy of Sciences (TWAS), the Indian Academy of Sciences, the Indian National Academy of Engineering, the National Academy of Sciences, and the Institution of Electronics and Telecommunication Engineers. He is the Vice-President and a fellow of the Indian National Science Academy.